Analysis and Test Development for Parasitic Fails in Deep Sub-Micron Memory Devices

Ijeoma Sandra Irobi

Analysis and Test Development for Parasitic Fails in Deep Sub-Micron Memory Devices

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof.. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen

op donderdag 15 september 2011 om 15:00 uur

door

Ijeoma Sandra IROBI

Master of Science in Computer Science,
Uppsala University, Zweden
Master of Science in Computer Engineering,
Mälardalen University, Zweden

geboren te Aba, Nigeria

Dit proefschrift is goedgekeurd door de promotor:

Prof. dr. ir. H.J. Sips

Copromotor: Dr. ir. Z. Al-Ars

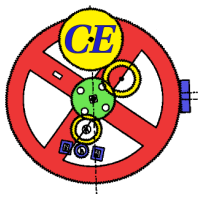
Samenstelling promotiecommissie:

Rector Magnificus	voorzitter
Prof. dr. ir. H.J. Sips	Technische Universiteit Delft, promotor
Dr. ir. Z. Al-Ars	Technische Universiteit Delft, copromotor
Prof. dr. Engr. C. Nebo	University of Nigeria, Nsukka, Nigeria
Prof. dr. P. Prinetto	Politecnico di Torino, Italië
Prof. dr. Z. Peng	Linköping University, Zweden
Prof. dr. ir. C.I.M. Beenakker	Technische Universiteit Delft
Prof. dr. ir. M.A. Gutiérrez	Technische Universiteit Delft
Prof. dr. ir. P. Jonker	Technische Universiteit Delft, reservelid

ISBN: 978-90-72298-22-5
Cover page design: www.Hanike.nl

Typeset by the author with the LaTeX Documentation system.

Author email: i.s.irobi@TUDelft.nl
Alternative email: s.irobi@yefa.nl
Printed in The Netherlands



TU Delft
Delft University of Technology

This dissertation is dedicated to:
The third person in the Trinity, the custodian of my dreams.
To my loving parents for setting my feet early on the solid ground.

Analysis and Test Development for Parasitic Fails in Deep Sub-Micron Memory Devices

Abstract

Emerging technology trends are gravitating towards extremely high levels of integration at the package and chip levels, and use of deeply scaled technology in nanometer, approaching 10nm CMOS. Challenges will arise due to the ability to design complex systems such as robots that encompass sensors, transducers, communications systems and processors, all of which require memory devices, and are required to be fault-free, and exhibit fault-tolerance, reliability and survivability characteristics. A key area of challenge is in memory testing, since deep scaling and smaller dimensions of semiconductor cell area will exacerbate the presence of complex defects and can induce effects, such as parasitic effects, which necessitate fails in memory devices. In this thesis, parasitic effects induced by spot defects in memory devices have been evaluated. The thesis presents the analysis, evaluation, validation and test remedies for parasitic fails in deep sub-micron memories. On the one hand, it presents analysis for parasitic bit line coupling effects, and the impact of bit line coupling effect on the static random access memory (SRAM) faulty behavior. Thereafter, it determines both the necessary and sufficient detection conditions for memory fault models, and demonstrates the limitations of existing industrial memory tests to adequately detect faults in the presence of bit line coupling. In addition, the thesis presents a systematic approach for test development and optimization, and new memory tests - March SSSc an optimal test that detects all single-cell static faults, and March m-MSS and March BLC that detect all two-cell static faults, in the presence and absence of bit line coupling. On the other hand, this thesis also presents the analysis, evaluation, validation and test remedies for parasitic memory effect in SRAMs. The work presents the impact of the parasitic memory effect on the detection of static faults, and clearly shows that fault detection is influenced by the presence of parasitic node components and not the resistive defect alone; something that must be considered in generating effective memory tests. In addition, the thesis presents the detection conditions and a new memory tests, March SME that targets and detects single-cell static faults, in the presence of the parasitic memory effect.

Analysis and Test Development for Parasitic Fails in Deep Sub-Micron Memory Devices

Samenvatting (Abstract in Dutch)

Huidige technologische trends leiden tot uiterst hoge niveaus van integratie in geïntegreerde schakelingen, waarbij gebruik wordt gemaakt van nanometer technologieën met afmetingen die tot 10nm CMOS kunnen gaan. De uitdagingen zijn het ontwerpen van complexe systemen zoals robots welke sensoren, omvormers, communicatie systemen en processors bevatten. Deze componenten vereisen allen een betrouwbaar geheugen met fout tolerantie en overlevingskenmerken. Een belangrijk aspect is het testen van geheugens. Door de compactere transistoren, en de daarbij behorende kleinere afmetingen, kunnen door parasitische effecten complexe defecten in het geheugen worden veroorzaakt. In dit proefschrift, zijn de parasitische effecten in geheugens die door spot defects worden veroorzaakt geëvalueerd. Dit proefschrift presenteert de analyse, evaluatie, validatie en test oplossingen voor parasitaire mankementen in deep sub-micron geheugens. Allereerst wordt de analyse voor parasitische bit-lijn koppeling en de impact van het koppelingseffect van de bit-lijn op het foutieve gedrag van static random access memory (SRAM) beschreven. Vervolgens worden zowel de noodzakelijke als de sufficiënte detectievoorwaarden voor fout modellen van het geheugen gepresenteerd en worden de beperkingen van bestaande industriële geheugentests besproken. Bovendien beveelt dit proefschrift een systematische benadering voor testontwikkeling en optimalisering aan. Nieuwe geheugentests -March SSSc een optimale test die alle eencellige statische fouten ontdekt, en March m-MSS en March BLC die alle tweecellige statische fouten detecteert, welke zowel in aanwezigheid als afwezigheid van bit-lijn koppeling kunnen werken. Anderzijds, presenteert dit proefschrift ook een analyse, evaluatie, validatie en test oplossingen voor parasitische effecten in SRAMs. De impact van parasitische effecten op de detectie van statische fouten wordt aangetoond. Hierbij wordt duidelijk dat foutopsporing wordt beïnvloed door aanwezigheid van parasitische componenten en niet slechts door aanwezigheid van weerstand defecten. Hiermee moet rekening worden gehouden bij het produceren van efficiënte geheugentests. Bovendien presenteert het proefschrift detectievoorwaarden voor nieuwe geheugentests zoals March SME, welke zich richt op statische fouten in enkele cellen en deze detecteert, in de aanwezigheid van parasitische effecten.

Acknowledgments

This doctoral research has been the product of resilience, hard work, determination and conscientious effort over these years. In this journey, several people have played strategic roles at different points that have greatly encouraged me, improved the quality of my work, and made my stay in the Netherlands an enjoyable experience. To these people, words are not enough to represent the depth of gratitude that I wish to express to all of you. Neither the order in which this acknowledgement goes nor the length of the sentences are commensurate to the value I place on your help, kindness and support.

In the first place, I remain very grateful to Prof. Stamatis Vassiliadis, who gave me a 23 minutes convincing pitch that informed part of my decision to follow the PhD program at TU Delft, and encouraged me to make my experiences here count. Even though he is no longer here with us, he is always remembered and the legacy he has left behind speaks in so many ways. I am also ever so grateful to Dr. Koen Bertels, who was my first contact at the CE group, who has also provided much more support and encouragement through these years than I could imagine. Koen *dalu!*.

Importantly, I wish to immensely thank my PhD supervisor, Dr. Zaid Al-Ars who believed in me and my ability to complete this work successfully much more than I did. For his consistent committment and patience that has greatly improved my thought process and the quality of my work I remain so grateful. Zaid, I cannot thank you enough, *Imela!*. I am very thankful to Dr. Said Hamdioui whose comments, reviews and positive chat about Africa encouraged me to think of how much better things could be. I wish to thank Prof. Michel Renovell for his collaboration, insightful reviews and support in the later phase of my research work. I am also grateful to Prof. Claude Thibeault for giving me the opportunity of working with him on my internship at LACIME, École de Technologie Supérieure, Université du Québec, Montreal, Canada in the later part of my PhD.

To the fantastic staff team in CE group that have professionally handled important administrative and technical aspects of my stay, Lidwina, Bert, Eric and Eef *unu emela!* Also, to CICAT staff, especially Veronique van der Varst, who worked tirelessly to get all the paper work and my stay arrangements made, I was indeed lucky to have you, thank you very much. I also wish to thank my colleagues in CE group who have contributed to the lovely atmosphere that made my research enjoyable. I thank my officemate Tariq, who has provided interesting discussions - scientific and non-scientific that made the office lively; my colleagues Motta, Roel, Behnaz, Hamid, Mahmoud, Seyab, Zaidi, Mafalda, Innocent to mention only a few.

I remain grateful to Prof. Henk Sips, my promotor, who is well known for his efficiency and quality inputs. I also thank all my PhD promotion committee members for graciously accepting to serve on the committee. I wish to say thank you to Prof. Chinedu Nebo whose insight and vision as UNN Vice Chancellor inspired me to seek channels of mutual collaboration between TU Delft and UNN, and to Prof. Dan Lenstra and Prof. Kees Beenakker for providing immeasurably support from EEMCS and DIMES to facilitate the collaboration. Many thanks to my teachers who pointed me in the right direction - Dr. Ivan Christoff, Dr. Roland Bol and Prof. Gerhard Fohler.

My appreciation also goes to my Church, friends and acquantancies who have generally made my stay and social life so much fun. I thank Austine, Chinelo, Dubem, Peter, Ijeoma, Solomon, Sam, Obioma, philip, the Adeniyi's, the Nwosu's, the Abarshi's and many more that for the sake of space I cannot mention. I deeply appreciate you all *Ndewo nu!* I wish to also express my gratitude to my colleagues at Young Entrepreneurs for Africa (YEFA), whose constant committment to excellence and development sharpened my focus, *Unu emekalum!*.

Finally, I wish to thank my family, my parents, siblings, nieces and nephews for their unending love, support, committment and the sparkle they have added to my life and have shown beyond measure. To you all, it is impossible to fully express my appreciation, *Chineke gozie unu nile!* To everyone else who I have not mentioned but who have affected my life in special ways, I am grateful to you and wish you well.

Contents

Chapter 1

Introduction

With significant advances in deep sub-micron complementary metal oxide semiconductor (CMOS) technology, more feature-rich integrated silicon devices are being used in consumer electronics, advanced communication and networking systems, computers, servers and virtually all electronic systems. One of such devices is the memory chip.

Memories perform an essential function, which is to store data and provide for its retrieval.

On the one hand, the quest for increase in functionality and smaller dimensions have been the driving force behind the continued scaling of cell area in semiconductor devices as encapsulated by Moore's law [69]. However, smaller dimensions of memory devices can elicit various types of defects such as spot defects - opens, shorts and bridges during the manufacturing process [16, 17, 20, 82, 87]. In addition, it can induce capacitive coupling among signal lines, power and ground lines, thereby resulting in high sensitivity level to defects in the memory.

On the other hand, this quest for smaller dimensions comes with the increased demand for reliability of the manufactured devices. However, for decades it has been obvious that the reliability, availability, and safety of electronics systems cannot be obtained solely by the careful design, quality assurance, or other fault avoidance techniques without proper testing mechanisms [19]. In fact, high defect density as well as new failure mechanisms in the nano-era are expected to be significantly larger in the future [21, 22, 92], and this will create major challenges in designing and testing memories in nanotechnology [38]. Conventional fault models and test approaches are inadequate to realize the required product quality [45, 46, 70, 77]. 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.

Combining the demand for higher densities, which can exacerbate the presence of

complex defects, and the assurance of reliability of the manufactured devices underscores the importance of investigating appropriate and superior methodologies in failure analysis and testing that target the consequences and complexities of the evolving technologies [49, 100].

This thesis presents the work done in one of such studies. The thesis presents the analysis, evaluation and validation of parasitic fails in deep sub-micron memory devices, and the development of suitable memory tests that detect such fails.

In this chapter, an introduction to semiconductor memories is presented. Section 1.1 describes the semiconductor memory technology. Section 1.2 discusses the motivation for memory testing, memory test levels and the concept of test time. The contributions and scope of this thesis are presented in Section 1.3 and finally, Section 1.4 provides the thesis outline and a brief conclusion of the chapter.

1.1 Semiconductor memory technology

The term memory in this thesis refers to data storage that comes in the form of packaged semiconductor memory chips.

A diagrammatic representation of the classification of semiconductor memories is depicted in Figure 1.1. There are two broad categories of memories, which are the *read-only memory (ROM)* and the *random access memory (RAM)*. Both types of memory are often packaged as integrated circuits (ICs). ICs are small electronic circuits that consist mostly of semiconductors.

ROM is a special memory used to store programs for booting and performing diagnostics on computers. It contains data that normally can only be read, but not written to. ROMs are considered non-volatile since they do not loose their information when power is switched off. ROMs can be classified as follows.

1. *Programmable read-only memory (PROM)*. This is the type of ROM that can only be programmed once. Once the PROM has been used, you cannot erase and reuse it.
2. *Erasable programmable read-only memory (EPROM)*. An EPROM is a special type of PROM that can be erased as a result of exposure to ultraviolet rays.
3. *Electrically erasable programmable read-only memory (EEPROM)*. An EEPROM is the type of PROM that can be erased electrically.
4. *Flash memory*. This is a variant of the EEPROM. It has a high density, low cost, fast (to read but not to write) and is electrically reprogrammable. The flash memory can be distinguished from the EEPROM in the sense that flash devices can be erased in chunks, one sector at a time and not on a byte-by-byte basis.

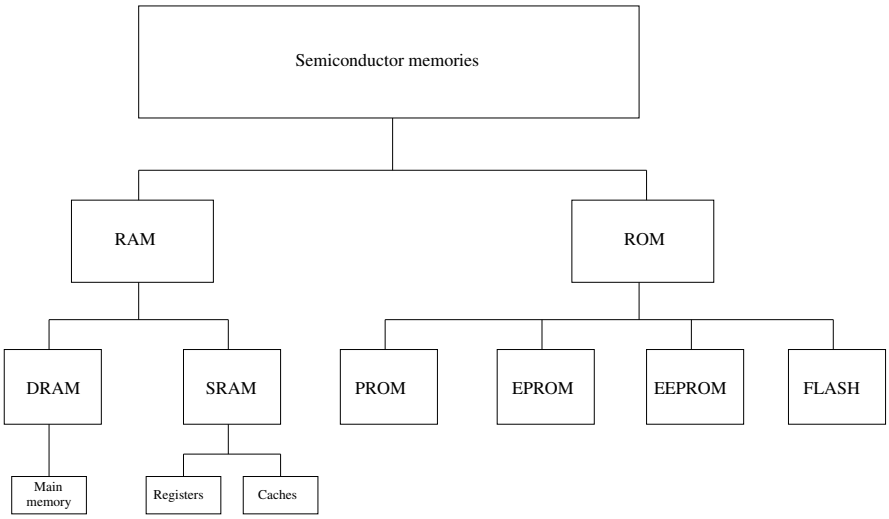


Figure 1.1: Types of semiconductor memories

RAM is the most common memory type found in computers. In contrast to ROMs, it is volatile because it looses its data when power is switched off and therefore requires continuous power supply. More so, data can both be written to and read from RAMs. The word "*random*" in the name "random access memory" refers to the fact that any location in such memory can be addressed directly at any time. This is different from sequential access media, such as magnetic tape, which must be read in sequence irrespective of the desired content. Depending on the technology used in holding the stored data, RAM can be classified into two categories, namely, the static random access memory and the dynamic access random memory. A brief discussion on each of these categories is given below.

1. *Static random access memory (SRAM)*. The SRAM stores its data inside latches and can store it for as long as needed. The term *static* is used because the memory does not need to be refreshed as long as power is supplied to the circuit. Refreshing refers to the process of periodically reading information from an area of the computer memory, and immediately rewriting the read information to the same cells with no alterations. The evolution of hardware systems over the years has led to the manufacture of different types of SRAMs made up of 4 to 6 transistors and is very fast. Examples of memories made with SRAM are registers and cache memories. A main disadvantage of an SRAM is that it consumes more silicon area per bit and relatively more power than DRAMs, which explains why they are equally more expensive.
2. *Dynamic random access memory (DRAM)*. The DRAM stores information in capacitors, which loose their charge over time and therefore needs to be refreshed regularly in order to retain their contents. This refreshing action is the

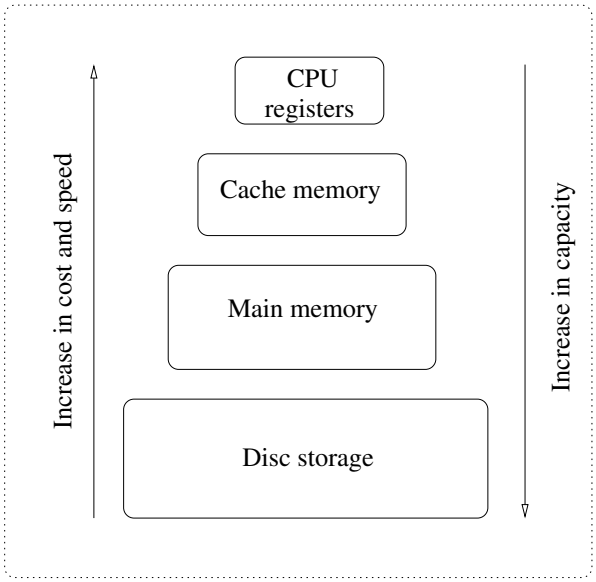


Figure 1.2: Memory hierarchy

reason why the memory is called *dynamic*. The refresh time in DRAMs also explains why SRAMs are faster than DRAMs, since SRAMs do not need to be refreshed. An example application of a DRAM device is the main memory of a computer. In comparison to SRAMs, DRAMs can use one transistor and a capacitor per memory cell. Due to this internal capacitive element, some delay is incurred, with access time usually above $30ns$.

1.1.1 Memory devices hierarchy

Memory devices in most systems can be implemented as a hierarchy. Figure 1.2 depicts the common memory hierarchy in most modern personal computers. Moving from bottom to top of the figure shows the order of increase in speed as well as cost but a decrease in storage capacity.

At the bottom of the hierarchy is the disc storage. It typically has an access time of hundreds of thousands of central processing unit (CPU) cycles. This relatively slow speed is the result of using mechanical parts, particularly electric motors and moving magnetic heads. However, disc storage can have a capacity ranging from tens of gigabytes on small computers to many thousands of gigabytes on large systems.

Above the disc storage is the main memory. Main memory usually has an access time equal to several hundred CPU cycles. The term *main memory* commonly refers to physical memory, which uses DRAM to store its data, and is considerably faster than the disc storage.

The cache memory serves basically to reduce the mismatch in speeds between the CPU and the main memory. There could be different levels of cache memories for example Level 1 (L1), Level 2 (L2) caches. L1 cache consists of high speed SRAM cells. To save space and reduce cost, there is usually only a small amount of L1 cache on a processor. The secondary cache memory, L2, could hold larger sizes of data than the L1. L2 caches are likewise composed of SRAM cells, but they are made up of more such cells than L1 caches. Due to the reduced size of the L1 cache, it is much faster than the L2 cache, thus access time to and from the cache is reduced due to fewer cells.

At the top of the memory hierarchy are processor registers. These comprise an extremely small amount of very fast memory cells that are built into the CPU. The aim is to speed up the execution of the CPU, and thus of programs, by providing quick access to commonly used values. Registers are usually implemented as an array of multi-port SRAM cells.

1.1.2 DRAM architecture and types

The DRAM array consists of cells organized in a number of rows and columns. For example, a 1M bit chip could be externally organized as 1M addresses. Internally, memory cells are arranged as a matrix or number of matrices and cells can be grouped together as words containing multiple bits of data. Figure 1.3 depicts the internal functional block diagram of the DRAM.

During a read operation, first, the row address is presented through the *row address buffer*, and decoded by the *row decoder*. Next, the column address is presented through the *column address buffer*, and decoded by the *column decoder* to determine the exact cell(s) to be accessed. The high-order bits of the address are connected to the row decoder, which selects a row in the memory cell array. The lower-order address bits go to the column decoder which selects the required columns. The contents of the selected row in the memory cell array are amplified by the *sense amplifier*, loaded into the data-out buffer and presented on the data line. For a write operation, the data on the data line is loaded into the data-in buffer and written into the memory cell array through the write driver.

The refresh counter of the memory counts through the memory addresses in order to refresh the data stored in the memory cells. During a refresh operation, the column decoder selects all columns and the row decoder selects the row that is indicated by the address latch. All the bits in the selected row are read and refreshed at the same time. Reading each row and writing it back compensates for the gradual leakage of charge from the capacitors, which store the data. If this is not done regularly, then the DRAM looses its contents, despite continued power supply.

A major drawback in the DRAM technology compared to SRAM is low speed. In or-

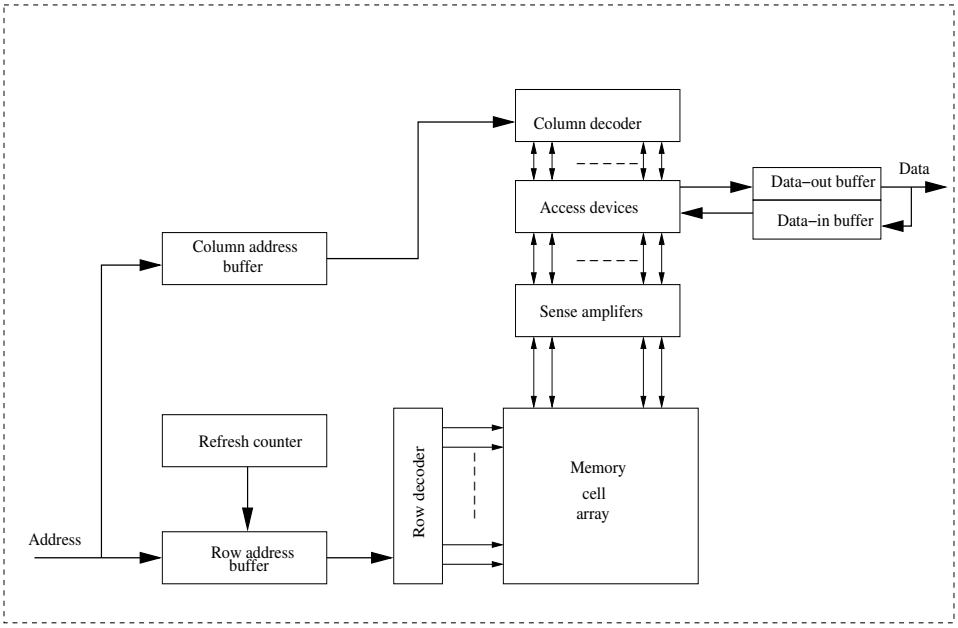


Figure 1.3: An internal functional block diagram of the DRAM

der to overcome this drawback, more efficient types of DRAM for instance, Fast Page Mode DRAM (FPM), Extended Data Out (EDO), Synchronous DRAM (SDRAM), Double Data Rate (DDR) have been developed. A brief explanation of each technology is provided below.

- FPM DRAM: This is the fast page mode dynamic random access memory (FPM DRAM). While the standard DRAM requires that a row and column be sent for each access, FPM DRAM works by sending the row address just once for many accesses to memory in locations near each other, thereby improving access time. The row of bits is selected only once for all columns within the row.
- EDO DRAM: Extended data-out dynamic random access memory is faster than the FPM DRAM. The reason is because unlike conventional DRAM which can only access one block of data at a time, EDO RAM can start fetching the next block of memory at the same time that it sends the previous block to the CPU.
- SDRAM: Synchronous dynamic random access memory is a type of DRAM that has a synchronous interface. This implies that it waits for a clock signal prior to responding to control inputs, thus is synchronized with the computer's system bus. SDRAM chips eliminate wait states by dividing the chip into two cell blocks and interleaving data between them. While a bit in one block is accessed, a bit in the other is prepared for access.

- DDR SDRAM: Double data rate synchronous dynamic RAM is a type of SDRAM that supports data transfers on both edges of each clock cycle (the rising and falling edges), thereby effectively doubling the memory chip's data throughput with less power consumption than the SDRAM.

1.1.3 SRAM architecture and types

The SRAM cell consists of a bi-stable flip-flop connected to the internal circuitry by two access transistors as depicted in Figure 1.4. The access transistors are connected to the word line (WL) and to a bit line (BL) pair comprising the true (BT) and complementary (BC). Each flip-flop has four to six transistors that are cross-coupled to form inverters. When the cell is not addressed, the two access transistors are closed and the data is kept to a stable state, latched within the flip-flop. The flip-flop needs the power supply to keep the information. The data in an SRAM cell is volatile since the cell structures allow data to be stored for an indefinite amount of time in the device as long as powered and is lost when power is removed. Unlike DRAMs, data does not leak away in an SRAM, thus refresh cycles are not required. The flip-flop may be in either of two states interpreted by the support circuitry as a 1 or a 0.

There are several types of SRAM classifications. One type of classification of SRAM is based on the SRAM memory cells. They are classified as the 4T cell which have four NMOS transistors plus two poly load resistors; the 6T cell which have six transistors - four NMOS transistors plus two PMOS transistors; the thin film transistor (TFT) cell which has four NMOS transistors plus two loads called TFTs.

The four-transistor SRAMs are suitable for medium to high performance, but have relatively high leakage current, and consequently high standby current. Four-transistor designs may also be more susceptible to various types of radiation induced soft errors. On the other hand, the six-transistor (6T) memory cell are highly stable, relatively impervious to soft errors, and have low leakage and standby currents. Current SRAM chip architectures use six-transistor memory cells and is the type used throughout this thesis.

SRAMs can be differentiated based on their function or by the transistor type. By their function they can be classified into four main categories, namely, asynchronous SRAMs, synchronous SRAMs, special SRAMs, and non-volatile SRAMs.

1. Asynchronous SRAMs: Asynchronous SRAMs are devices that are not synchronized with an external signal clock, and begin their data operation (read or write) as soon as it receives the instruction to do so. In asynchronous SRAMs the memory is managed by three control signals. One signal is the chip select (CS) or chip enable (CE), the second is the output enable (OE) and the third is the write enable (WE).

   Asynchronous devices can be categorized as low speed, medium speed and

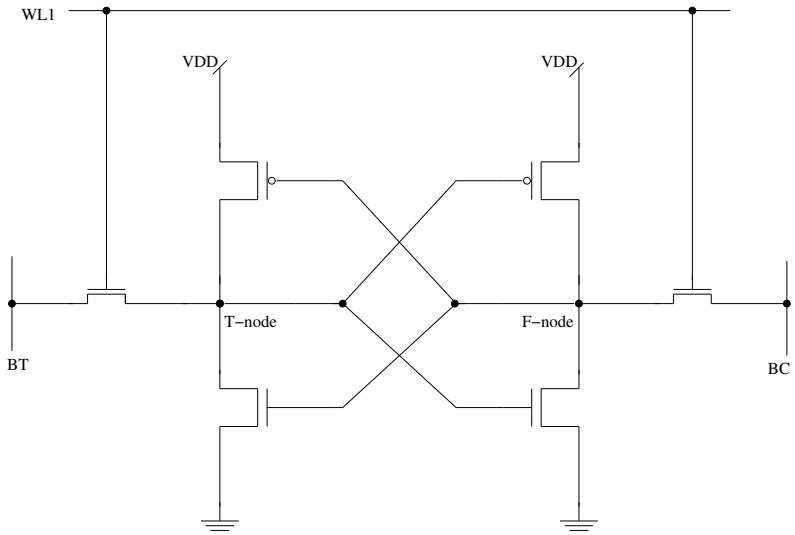


Figure 1.4: 6T SRAM cell

high speed based on their access time. Access time refers to the total time it takes for a device to yield a data output after receiving a read instruction. These categories are:

- Fast asynchronous SRAMs: These devices are often used in buffer memory applications, consumer products, etc. The density range for these types of SRAMs is from the sub 64K to 4Mb and have data words that are mostly configured as x8, x16, which is the size in bits that each memory location can store.
- Low power SRAMs: These can also be referred to as low speed SRAMs. These SRAMs are typically designed to consume very low power and are used in applications where power is a major concern, which include digital signal processors (DSPs), PDAs, consumer electronic products, and so on. The density range for these SRAMs is usually from 64K to 8Mb and are mostly configured with word widths of 8 and 16 bits.

2. Synchronous SRAMs: As computer system clocks increased, the demand for very fast SRAMs necessitated variations on the standard asynchronous fast SRAM. This demand resulted in the development of synchronous SRAMs (SSRAM). In synchronous SRAMs the read or write cycles are synchronized with the microprocessor clock and therefore can be used in very high-speed applications.
3. Special SRAMs: Multi-port SRAMs are specially designed chips using fast SRAM memory cells. Synchronous and asynchronous FIFOs are available.
4. Battery-Back SRAM (BRAM): is also called zero-power SRAMs. Battery-

backed SRAM combine an SRAM and a small lithium battery.

1.1.4 Emerging memory technologies

This discussion will not be complete without examining the memory technologies that are currently being developed. The demand for faster, cheaper, better memory is enormous, and memory manufacturers are responding with several innovative solutions.

- Zero capacitor RAM (Z-RAM): This is a potential replacement for SRAM. It offers the performance of six transistor SRAM using only a single transistor, which therefore can provide much higher densities.
- Ferroelectric RAM (FeRAM): This is another non-volatile memory type, which is similar to flash RAM. However, it provides better power efficiency, write speeds, and write-erase duty cycles.
- Magnetoresistive RAM (MRAM): This memory stores data in magnetic storage elements, not as electrical charge or current flow. MRAM is physically similar to DRAM, however, it does not require refresh cycles.
- Phase-change memory (PRAM or PCM): This is another non-volatile memory type, which is based on the phase change properties of chalcogenide glass, which can be switched from crystalline to amorphous states by the application of heat. PRAM offers high densities and can be useful in harsh environments where radiation disrupts other types of RAM.

1.2 Testing semiconductor memories

The essence of any memory test could be to detect, diagnose or localize faults in a given memory device. A fault is said to have occurred when a wrong output is yielded by a defective system, or a distinguishable difference is observed between a correct version of the given device and the current device under test (DUT). A fault can arise during all phases of a computer system design process specification, design, development, manufacturing, assembly and installation throughout its entire operational life [19, 27]. The process of determining whether a given memory device contains faults or not is referred to as *memory testing.* The algorithms with which the presence or absence of faults are ascertained are called *memory tests.*

Since faults can occur, the need to perform quality tests to ensure that manufactured chips are capable of performing their intended functions at the point of use cannot be over emphasized.

1.2.1 Motivation for memory testing

There are several motivations for memory testing. Memories practically dominate the chip area. Due to the desirability of smaller chip dimensions, there is increase in density on as much reduced chip area as possible [62, 61]. The implication of this increase in density is that the area occupied by each memory cell exponentially decreases and cells are located closer to one another. Because of the increased proximity between memory cells, memories become increasingly sensitive and prone to complex defects.

Failures in the memory device can significantly impact the defect-per-million (DPM) rate, since memories largely dominate the chip area. Therefore, prior to the end of the production chain from the manufacturer to an end user, highly efficient memory tests are required to appropriately test and detect faults in the memory device. Further motivations for memory testing include:

1. **To mitigate the impact of increased chip density**

   Due to high cell density, individual cells become closer to one another and become prone to influences from other cells in their neighborhood. This can adversely affect the cell and impact the expected behavior of the memory device.

2. **To detect disturbances due to use**

   Noise and crosstalk [30] effects, for example, noise on address and data lines, could occur in the cells due to repeated use and could induce disturbances transferable from one part of the memory device to the other.

3. **To detect defects due to the manufacturing process**

   Defects could be introduced into the memory device during the manufacturing process. The continuos shrinking of semiconductor's nodes makes semiconductor memories increasingly prone to electrical defects tightly related to the internal structure of the memory [91]. For example, differences in capacitance or leakage current could occur during the manufacturing since charges are stored in capacitors and is prone to leakage. Such abnormalities could be detected by testing.

4. **To detect faults in the presence of complex defects**

   Due to shrinking memory dimensions relating to increased chip density, complex factors, such as bit line coupling, can affect the memory's behavior thereby causing faults that cannot be detected by already known memory tests. Understanding these behaviors and generating appropriate memory tests that specifically target the resulting faults can help to minimize fails recorded as a result of such complexities.

5. **Ensuring functionality under stressful conditions**

   Testing helps to ascertain the reliability of a device given specific stress conditions such as time, voltage and temperature. Several research work have addressed the effectiveness the effectiveness of stresses [39, 89, 96].

1.2.2 Memory test levels and time

Testing can be carried out at three different levels, namely, the cell array, chip and the memory board levels.

1. Cell array level. At the array level, testing comprises the memory chips, control logic and chip select.
2. Chip level. At this level, testing is focused on the memory chip, which contains the actual data of the memory.
3. Board level. A memory board comprises board selection and memory board controller, data and address registers, support logic as well as the memory array. Tests at this level involve these components.

An objective of memory tests is to basically detect malfunctioning memory cells. With recent memory manufacturing technology, the density of cells per silicon area continues to increase. Since each cell must be tested to ascertain the presence or absence of faults, it follows that the total test time required to investigate a DUT is proportional to the amount of memory being tested. In brief:

$$t_{\text{test}} = n(\#_{\text{op}} \cdot t_{\text{op}}) \tag{1.1}$$

where n is the total number of cells, $\#_{\text{op}}$ is the number of necessary and sufficient test operations performed per cell to detect fault, and t_{op} is the time taken per test operation.

Whereas linear tests have test time complexities of the order $O(n)$, some non-linear tests can have complexity of the order $O(n^y)$, where y is greater than 1. Hence, reduction in both the time required to run a test as well as the number of tests necessary and sufficient for fault detection, while maintaining high *fault coverage* (i.e., the number of detected faults divided by the number of total faults) in a given device is an important challenge and priority for developing memory tests.

1.2.3 Challenges in testing memories

Due to high cell density, the number of bits per chip (example in DRAMs) increases exponentially (estimated as 4 times in less than 3 years), but the price per bit decreases exponentially [60]. Older test algorithms are known to have test run time

complexities in the order of $O(n^2)$, but to cope with the exponential increase in density there is the need to use efficient tests with test times in the order of $O(n)$ that can effectively test the DUT at no significant increase in test cost. As a result of recent complex innovations in semiconductor memory technology, testing memory devices face enormous challenges. The use of embedded memories has compounded this challenge due to lack of *controllability* of the inputs and observability of the outputs, and to ameliorate this problem, built-in-self-test (BIST) was introduced [63, 64, 67, 80, 88]. The challenges in memory testing can be broadly categorized into three main issues namely, test time, test cost and complex fault behaviors.

- Test time: The growing density and capacity of memory chips requires new test methodologies and equipment since the total test time required to investigate a DUT is proportional to the amount of memory being tested. Thus, for tests to be desirable, they must be generated to ensure optimal test times and still yield high fault coverages.
- Test cost: In the deep sub-micron era silicon, packaging and testing costs are substantial parts of the total manufacturing cost [65] depending on production volume. As the volume of production increases, so could the total cost for administering effective tests to ensure reliability. However, if the test cost increases as the increase in density, it could become increasingly difficult to effectively test all manufactured devices.
- Complex faults behavior: Decrease of cell area can result in coupling noise and sensitivity higher to defects in memory devices. In recent times, several companies are not able to explain all electronic failures using existing test approaches [28, 68, 77, 102]. An example is AUDI, which reported that from all its electronic failures, only 35% can be mapped using existing approaches, while about 41% are not yet understood [68]. For example factors such as bit line coupling [10, 11] can occur, which is the development of small coupling voltages on adjacent bit lines that can influence proper sense amplifier operation. In fact, bit coupling and the resulting crosstalk noise is strongly considered as a limiting factor in designing high speed, low power SRAM devices [71]. This has a huge impact on the faulty behavior of the memory, potentially causing readily detectable memory faults to become undetectable with several memory tests.

  Thus, memory tests must continually evolve to investigate, model and develop appropriate detection conditions and test algorithm detecting, diagnosing or locating such resulting complex faulty behavior in memory devices [9, 52, 50]. In addition, such evolving test algorithms are required to run at minimal test times and incur no significant costs compared to existing memory tests.

1.3 Contributions and scope of thesis

In response to the challenges posed to memory testing due to increased memory density as a result of reduced chip size discussed in Section 1.2.3, this thesis presents an investigation of the complex faulty behavior induced by parasitic fails in deep sub-micron memory devices.

The thesis presents the analyzes, modeling, simulation, evaluation and validation of the impact of parasitic component on the faulty behavior of memory devices. It presents new test methodologies developed and memory tests generated to appropriately detect memory fails in the presence of such parasitic effects.

1.3.1 Specific contributions

In a nutshell, the main contributions of this thesis are the following:

1. This thesis presents a theoretical framework that models bit line (BL) parasitic capacitance, and the effect of capacitive BL coupling on the faulty behavior of the memory cell array. The thesis validates this behavior theoretically and through electrical SPICE simulations.
2. Neighboring cells can influence the faulty behavior of defective cells through coupling. This thesis analyzes, simulates and evaluates the impact of parasitic bit line coupling and neighborhood coupling data backgrounds on the faulty behavior of SRAMs. It validates the analysis through defect injection and circuit simulation of all possible spot defects in the memory cell array.
3. The thesis investigates and establishes the worst case coupling backgrounds required to induce worst case coupling effects in deep sub-micron devices. It presents the conditions necessary to ensure proper detection of memory faults, while taking BL capacitive coupling into consideration.
4. The fault coverage of otherwise efficient memory tests can be dramatically reduced due to the influence of bit line coupling. The thesis clearly demonstrates the inadequacies and limitations of several well-known industrial tests in detecting memory faults in the presence of capacitive bit line coupling effect.
5. The thesis presents March SSS, March SSSc and March m-MSS, which target and detect all single and two-cell static faults in the presence and absence of BL coupling for any possible spot defect.
6. Memory test optimization can significantly reduce test complexity and cost, while retaining the quality of the test. This thesis introduces a systematic approach for developing optimized tests for memory faults in the presence of bit line coupling. It shows how to identify the required coupling backgrounds for all static memory faults, and presents March BLC, an optimized test that

detects all static memory faults in the presence of BL coupling.

7. The thesis investigated and analyzed the impact of parasitic node capacitance on defective resistive nodes. The resultant effect referred to as parasitic memory effect can induce dynamic changes in the electrical behavior of the circuit thereby necessitating faults. The thesis presents the modeling of parasitic memory effect in memories. It demonstrates that the faulty behavior in the memory is exacerbated in the presence of parasitic node capacitance; something that reduces the fault coverage of current memory tests, and increases the DPM rate.
8. Parasitic node capacitance and faulty node voltage as components of a defective node can induce serious parasitic effects on the electrical behavior of the memory. This thesis analyzed, evaluated and characterized parasitic memory effect, and the variation of the floating node voltages on the faulty behavior of the memory. In fact, it demonstrates that the detection of memory faults is not determined by the value of the defect resistance alone, but is significantly influenced by the parasitic components of the defective node; something that is often not accounted for during memory testing. It presents the detection conditions for faults in the presence of parasitic memory effect, and finally presents March SME, which detects all targeted faults in the presence of parasitic memory effect.

1.3.2 Scope of work

In this thesis, the main contributions are focused on SRAMs. All spot defects namely, opens, bridges and shorts have been analyzed and evaluated. The work also targeted functional fault models for all single-cell and two cell faults.

1.4 Thesis outline

The work reported in this thesis is structured in seven chapters. This section provides a brief insight into each of the chapters and the general organization of the thesis.

Chapter 2 presents modeling of memory devices. It discusses the different levels of modeling namely, behavioral, functional and electrical modeling.

Chapter 3 introduces the concept of faults, describing the fault models, fault primitives and notations. It also explains the concept of memory testing, march tests and march test notations.

Chapter 4 presents parasitic bit line coupling effects. It extensively provides a theoretical framework for bit line coupling effect, a modeling of parasitic bit line capacitance and the impact of bit line coupling on the faulty behavior of SRAMs.

Chapter 5 introduces the concept of parasitic memory effect. It presents a model for this effect and presents the analysis and evaluation of the impact of parasitic memory effect on the faulty behavior of SRAMs.

Chapter 6 presents memory testing for parasitic fails. It presents a systematic approach used to develop march tests to detect faults in the presence of bit line coupling effect, and an optimization technique that generates an optimized test to detect such faults. Finally, the chapter introduces a testing technique, and test that detects faults in the presence of parasitic memory effect.

Chapter 7 lists conclusions and recommendations of this thesis, and possible future work.

Chapter 2

Modeling memory devices

A **m**emory device is a group of interacting, interrelated, or interdependent parts forming a whole to achieve the storage and retrieval of stored data. Generally, testing a device can be done in different ways. One way is by physically inspecting the device with a view to identifying defects or anomalies between the inspected system and the correct version of the system. However, for semiconductor devices this method of testing will entail inspecting internal structures of the device, which is both cumbersome and error prone, and therefore unrealistic.

Another method of testing involves creating a model, which is an abstraction of the real system. This is done by adequately representing the essential characteristics or functional blocks of the device for which changes can significantly impact the device. Creating an abstraction is a mechanism and practice to reduce and factor out details so that one can focus on a few concepts at a time; or a generalized, condensed, and simplified concept derived from a more complex situation. It is a part representation of some whole. In this way, the behavior of the model can be compared with the actual behavior of the device with the intent of establishing the presence of anomalies or faults. Figure 2.1 shows a simplified general model, where input(s) are introduced and output(s) obtained. In brief,

> *A model is a representation of a system that allows for investigation of the properties of the system and, in some cases, prediction of future outcomes . It is an abstraction of the actual system under consideration.*

Consequently, a memory model abstracts the memory sub-system of the semiconductor device, and contains representations in an appropriate approximation of the device. Using models in semiconductor testing has several advantages. One such advantage is that it provides clarity on the structure and functional behavior of the system. This clarity simplifies the test method, because it allows specific focus on functional blocks of the system, instead of an observation of the system as a whole. Another advantage is that due to its support for a targeted or pre-defined scope of

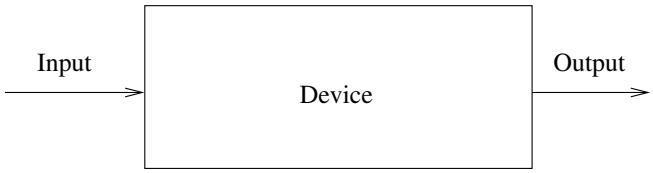


Figure 2.1: A block diagram of a general model

coverage, it saves test time and is invariably less expensive.

A device can be modeled at different levels of abstraction. A level of abstraction refers to the extent of details (for example, functional blocks or detailed components connections) that is revealed by a model. The models that exist categorized by their different levels of abstraction are geometrical, logical, electrical, functional and behavioral models. The lower the level of abstraction of a model, the more representative it is of the device.

Figure 2.2 depicts a typical memory chip model. The inputs to the model include the address, which indicates the cell to be accessed, a read/write switch showing the kind of operation to be performed on the cell and input data lines that provide data in the case of a *write* operation. The outcome is observed through the output data lines. Brief descriptions are given below on the different kinds of models.

The rest of this chapter describes the various types of semiconductor memory models and their characteristics.

2.1 Semiconductor memory models

For semiconductor devices, models are often used to investigate faults. The extent to which a model is put to use can be determined based on the model's level of abstraction, for example whether it can be used for *fault detection* or for *fault localization*.

> *Fault detection* refers to the ability to establish the presence of a fault in a given functional block of a DUT due to an observed anomaly in its behavior.
>
> *Fault localization* is the ability both to establish the presence of a fault and to identify the exact location of the fault (example component). Having a knowledge of the internal structure of the device is essential for fault localization.

Brief descriptions of the different models are given as follows.

- **Geometrical models**. At this level of abstraction, details such as the layout implementation of the system is known, for example, line distances between the electrical components, as well as general system's layout are included. This

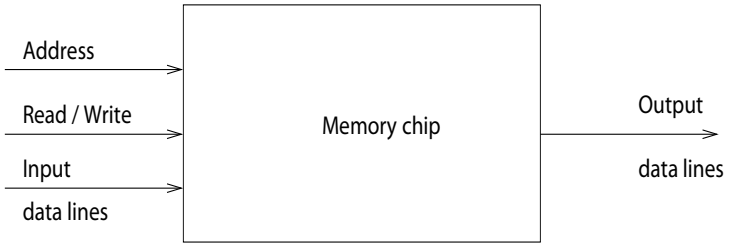


Figure 2.2: A block diagram of a memory chip model

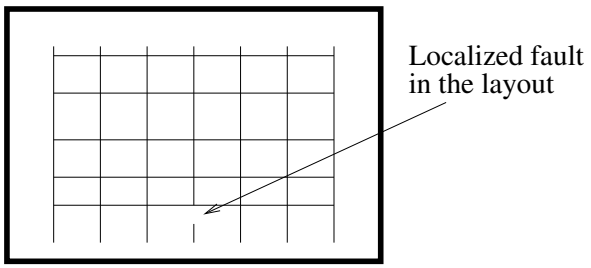


Figure 2.3: Geometrical model

model makes for good identification of faults based on the manufacturing process and provides insight into the aging process of components. A diagrammatic representation of this model is depicted in Figure 2.3.

- **Logical models**. Modeling a system based on the logic gates in which Boolean expressions are used to mimic the targeted system's function yields logic models. The main aim of a logic model is to localize faults resident in logic gates of a device. Because instead of logic gates, transistors and capacitors are used in manufacturing semiconductor memories, modeling faults using the logic gate will not provide a good correspondence to reality. A diagrammatic representation of this model is depicted in Figure 2.4.

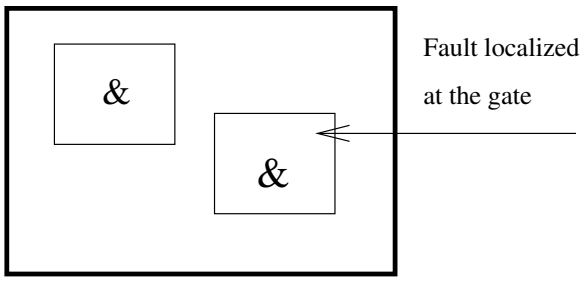


Figure 2.4: Logical model

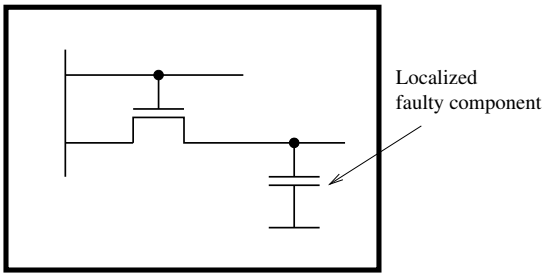


Figure 2.5: Electrical model

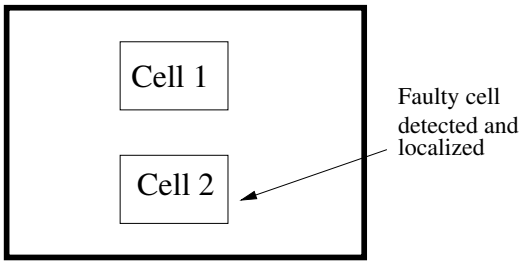


Figure 2.6: Functional model

- **Electrical models**. In this model, a detailed description of the system specifications, based on their electrical components and internal structures' implementation are known and represented. The aim is to target and localize faults at the electrical components level of the system. A diagrammatic representation of this model is depicted in Figure 2.5.
- **Functional models**. This is also known as the gray-box model. Abstraction is based on the functional specifications of the device, with partial assumptions on its internal structure. This means that relevant system parts can be represented as *functional blocks* with definite functions, for instance, the memory cell array or address decoder in the memory. A diagrammatic representation of this model is depicted in Figure 2.6.
- **Behavioral models**. This is also known as a *black-box*, since little or nothing is known of its internal structure. Specifically, the internal structures of the system using this level of abstraction is unknown. The aim of this model is to represent the system only based on its specifications, therefore, only the system's behavior can be verified. For any given system, this model provides the highest level of abstraction of the system. This model can be cited as a special case of the functional model, with the condition that only one function is represented, which is the system's function as a whole. As a result of this, the behavioral and functional models can be referred to as special cases of the structural model, where the structural model describes a system as a number of interconnected functional blocks [2, 4]. A diagrammatic representation of this

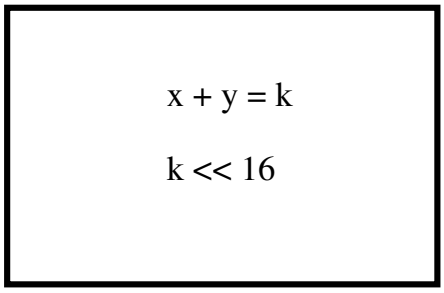


Figure 2.7: Behavioral model

model is depicted in Figure 2.7.

In this thesis, focus will be on functional models. The reason is that at this abstraction level, both the system's functional specifications and partial assumptions on its internal structures (gray-box) can be made, which is sufficient for fault detection. However, some of the assumptions on the device would require some knowledge of the internal structures of the device, for example a cell's electrical structure. Therefore, some details of the electrical behavior have been included. In the rest of this chapter discussions on models of DRAMs and SRAMs are presented.

2.2 DRAM model and characteristics

This section introduces some DRAM characteristics namely, the size of the DRAM cell array, DRAM chip pin layout, the timing parameters and some specific DRAM instructions.

The logical organization of the memory array is made up of words (W), which comprises bits (B). Each bit represents a cell, thus, the total number of cells in the array is given by the number of bits B, multiplied by the number of words W. The internal structure comprises rows and columns.

Addresses are required to specify the right row and column locations. The number of address lines is the maximum number of bits necessary to specify either a row or a column address (a) is given by $\lceil \log_2 max\ \{R, C\} \rceil$. Addressing lines are represented as $A_0, A_1 \ldots A_{\text{a-1}}$. For example, if a memory array has 1024 rows and 2048 columns, then a = $\log_2 2048$ = 11 addressing lines. This means that address $A_0 \ldots A_{10}$ are required. Data lines are required for data input and output. These pins are usually merged in order to save pins. The number of pins required for data input-output is B, where B is the number of bits in a word (W). Control lines are required in order to control the input and output of the device, and to determine when the chip should read a row or column address. The most common of these control pins are the row access strobe ($\overline{RAS}$), column access strobe ($\overline{CAS}$), and the write enable control line

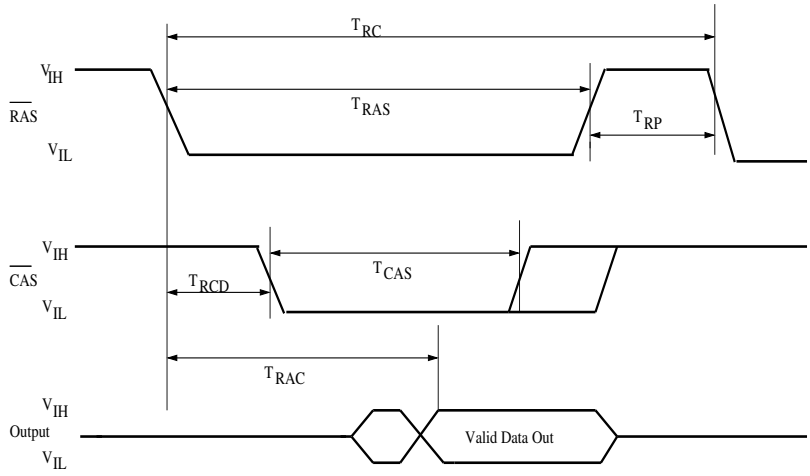


Figure 2.8: DRAM timing characteristics

($\overline{WE}$).

The timing characteristic of the DRAM chip provides the maximum and minimum delays for high and low edges of different signals and their duration to both read and write the needed signals to the output pin.

A typical illustration of the timing characteristics is provided in Figure 2.8. The $\overline{RAS}$ to $\overline{CAS}$ delay time represented as T_{RCD} is defined as the time between the low edges of the $\overline{RAS}$ and $\overline{CAS}$ signals. That is, the delay from the point $\overline{RAS}$ is low to the time $\overline{CAS}$ is low.

The least and highest values of the T_{RCD} provide the least and highest delay, and are directed by the speed of the row and column decoders. The least value produces maximal stress[1] to the row access path, whereas the highest value stresses the column access maximally.

The precharge time is the time necessary before the device can accept a fall of the $\overline{RAS}$ signal, in other words, how earliest the memory can be accessed again after having been accessed from the $\overline{RAS}$ signal.

T_{RC} is the time required to perform a random read/write cycle. T_{RC} is made up of $\overline{RAS}$ precharge time, T_{RP} and the $\overline{RAS}$ pulse width referred to as T_{RAS}.

Similarly, $\overline{CAS}$ precharge, T_{CAS} is defined as how early the memory can be re-accessed after having been accessed from the $\overline{CAS}$ signal. The least delay from the low edge of the $\overline{RAS}$ signal to the time data becomes available on the output pins is known as the access time, T_{RAC}.

Finally, the refresh time, (not shown) T_{REF} refers to the maximum time which can pass before a fresh operation is required to keep the information in the memory cells.

[1] Parameters, for example temperature, needed to test functionality within given specifications

The DRAM uses five main primitives for its commands. These primitives are:

1. *Read - Rd*. This refers to the *read* command. Here, data in one of the sense amplifiers is moved to the data registers and finally to the data bus.
2. *Write - Wr*. This is the *write* command. When given, the data in the data register is conveyed to the sense amplifiers as well as the memory cell array.
3. *Activate - Act*. This is referred to as the *activate command*. In this case, a word line (WL) in the cell array is chosen. It then accesses a row of the memory cells. Coupled with this, an internal read command is done by moving the data from the row of memory cells to the sense amplifiers.
4. *Precharge - Pre*. This is the *precharge* command. At this instance, any selected word line is deselected and the internal voltages are precharged to their already defined voltages.
5. *No Operation - Nop*. This command indicates *no operation*. It does not alter the state of the memory but rather extends the time duration of any already issued command. Thus, its impact depends on the previously issued command and not in itself. The number of *Nop* issued is dependent on the specific timing characteristic of the given memory device.

Nowadays, certain a variety of operational modes that enable DRAMs to provide more functional flexibility or higher performance are available. These include:

1. *Refresh operation* In this case, all cells are rewritten using the same values they contain in order to prevent loss of data as a result of naturally occurring leakage currents. The refresh operation is issued to the memory by providing a special sequence of values on the command bus that the DRAM interprets as a request for refresh.
2. *Fast page mode*. This mode starts by issuing a row address to the memory, which opens a full row of memory cells (also called a memory cell). Any cell from this page can be read or written by providing only the column address of the specific cell to be accessed. Figure 2.9 depicts the timing diagram of the two operations performed in the fast page mode. The first is performed on a cell with *col. add. 1* and thereafter, another write operation is performed on a different cell with *col. add. 2*. This mode of operation can increase the bandwidth of the memory by reducing the access time, since only column address part need to be provided at the inputs [79].

2.2.1 Functional DRAM chip model

The functional DRAM chip model shows the interacting functional blocks in the DRAM. A block diagram of this model is the same as shown in Figure 2.10. At this functional level, the internal structure of the memory that represents it as a collection

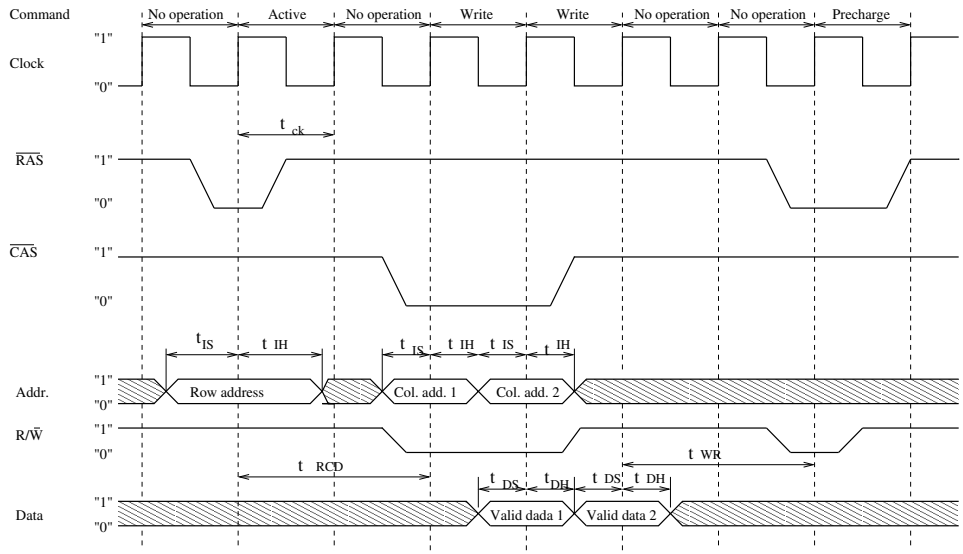


Figure 2.9: Timing diagram corresponding to DRAM fast page mode of operation

of interconnected functional blocks with separate functions are considered. Brief explanation of each block is given below:

- *Memory cell array.* This block occupies up to 60% [4] of the chip area and consists of the memory cells arranged close to one another in an array form in rows and columns. For instance, 1 Mega bits of memory with one million cells can be arranged as an array of 1024 rows and 1024 columns, 2048 rows and 512 columns and vice versa. The external DRAM behavior partly reflects this internal arrangement by requiring to split the address into columns and rows.
- *Control logic*. This is also known as the timing generator. The memory uses the control logic to regulate, activate and deactivate the required functional block when needed.
- *Address decoders*. This is used to decode the row and column addresses of a cell in memory that needed to be referred to (addressed). The inputs are cell addresses and the output is called *word lines* (WL) or the row decoder and *column select* (CS) for the column decoder. Rows and columns are selected by specific lines and a combination of selection of a row and a column results in selecting a single word in the array.
- *Sense amplifiers*. This part of memory is used to identify the data stored within the memory cells. Because the data in the cells are stored in capacitors, which are prone to leakages, data in the cell need amplification so as to drive other circuits in the memory. The interface between the sense amplifiers and the data buffers is called the *access device*. A limited number of columns is connected to the data buffers depending on the column address, while depending on the

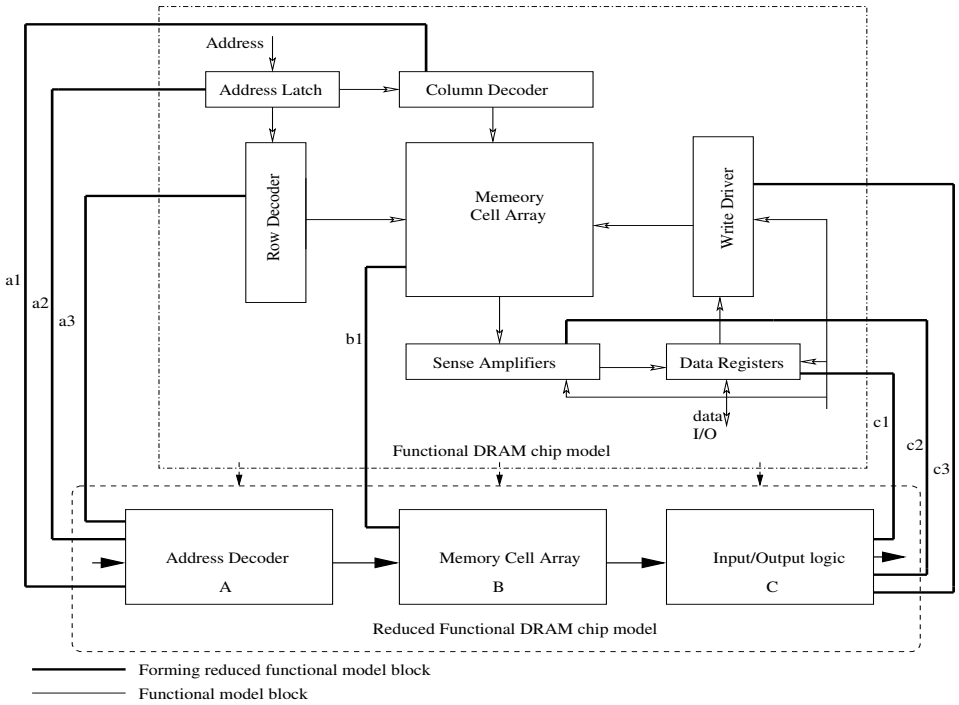


Figure 2.10: Conversion from the functional to the reduced functional DRAM chip model

operation that is performed, either the *read* or *write* buffer is connected to the sense amplifier.

- *Data-in/data-out buffers*. The data-in buffer is required to latch the input data and addresses while the data-out buffer stores the *read* output data and keeps it for the user on the data bus.

2.2.2 Reduced functional DRAM chip model

The reduced functional DRAM chip model is deduced from the functional model. In brief, the reduction is a result of merging functional blocks with similar functions together in order to reduce the overall blocks targeted. Figure 2.10 shows which functional blocks were combined to yield the three main blocks of the reduced functional DRAM chip model.

As shown in Figure 2.10, the row decoder, $a3$, and column $a1$ decoder are merged with the address latch denoted by $a2$ to form the *address decoder*, A, in the reduced functional model. This is because these three entities deal specifically with addressing. Likewise, the sense amplifier, $c2$, write logic, $c3$ as well as the data registers $c1$ are lumped together as the *input/output logic*, C, or read/write logic of the reduced functional model. The reason is that they similarly handle input and output data from the memory cell array. The memory cell array, $b1$, is retained as the only block in the *memory cell array* of the reduced functional model since no other block performs a similar function.

Thus concisely, a typical reduced functional model of a DRAM chip basically contains three main entities namely: The *input and output* or data read/write logic, the *address decoder* logic and the *memory cell array*. The reduced functional fault remains an authentic representation of the entire functional blocks covered by the functional model without any loss of information. Note that the fault coverage achieved by the functional model is the same as that of the reduced model since there is no information loss, i.e., no functional block is discarded.

Because localization of faults is not paramount but fault detection alone suffices; more an equivalent fault coverage obtained from the functional model is achievable using the reduced functional model henceforth, our focus and discussion including testing for the DRAM chip model will remain at the reduced functional chip level.

2.2.3 Electrical DRAM chip model

The DRAM chip is composed of individual memory cells. Each memory cell holds only one bit of information at a time, which could either be a "1" or "0", electrically represented by a different voltage level. Internally, the voltage level corresponding to any of the bits could be high or low. The representative voltage is dependent on

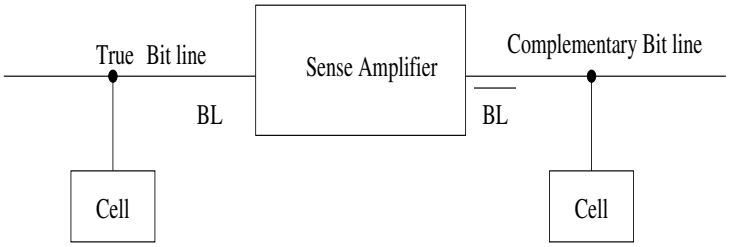


Figure 2.11: Cells connection to a sense amplifier

the location of the bit in the array and the architecture used. However, a high voltage represents the "0" bit whereas a low voltage represents a "1" bit. Basic cell structure is as follows:

Internally, the memory array consists of a number of rows and columns of words. These words can be accessed as a complete row only. Per row there is a WL, which determines if the row is connected to the sense amplifiers. The BL, are needed to connect to the sense amplifiers and get longer when the array increases in size. Raising or lowering the voltage on an entire line takes more time when the difference in voltage gets larger or when the line becomes longer.

Each cell in a row can be connected to one of the BL or $\overline{BL}$ of a sense amplifier as illustrated in Figure 2.11. Which bit line a cell is connected to depends on the location of the cell in the memory array. Sometimes, *odd* rows are connected to BL (also known as true bit line) and *even* rows to $\overline{BL}$ (also known as complementary bit line). In other designs, the sense amplifiers are located midway between the rows, so the left part of the row is connected to BL and the right part to $\overline{BL}$. A sense amplifier compares the voltage difference between its BL and $\overline{BL}$. When connecting a cell to a single bit line, the other bit line is usually connected to a dummy cell, in order to sense a valid voltage difference. Depending on the design, the internal structures of the DRAM cell could differ. These differences culminate in varied cell sizes and power requirements. Here we describe the different cell designs in order of their size and power requirement.

Four types of DRAM cell designs exist [94]. They can be distinguished following their internal structures, size and power requirement. These are:

1. 6-device DRAM cell. This is as shown in Figure 2.12. In this cell structure, there are six clock enhancement mode transistors and two gate capacitors. Qc and Qd have their input gates connected to the read/refresh line. The stored charge in the gate capacitor, Cg, keeps the information dynamically. Note that two gate capacitors are used, that is, one for a logic "1" and the other when charged for a logic "0". During the read or refresh period, the transistive loads are enabled but thereafter disabled in order to avoid power dissipation. The 6-device structure is the most expensive in terms of area and power requirements.

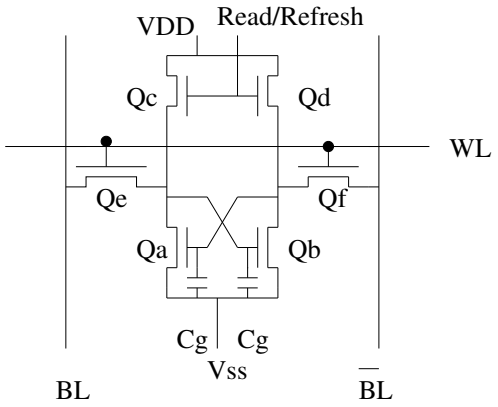


Figure 2.12: Electrical structures of the 6-device DRAM cell

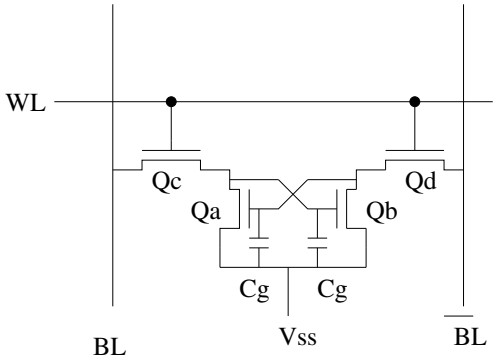


Figure 2.13: Electrical structures of the 4-device DRAM cell

2. 4-device DRAM cell. This is as depicted in Figure 2.13. This cell structure consists of two capacitors and four enhancement mode cross-coupled transistors. As shown, transistors Qa and Qb form a *latch*, with the pass transistors Qc and Qd. This is an improvement over the 6-device cell since the two extra load devices (transistors), V_{DD} and the read/refresh lines are removed in this structure, thus resulting in an area gain of over 50%.

   The charge kept in the gate capacitor Cg determines the device's ability to retain data. Two Cg are used: each for a charge of either logic "1" or "0". During its refresh period, the target row's word line, WL becomes high, while data is read through the already precharged BL. It is then sensed by the sense amplifier and BL is driven to cause the *latch* to be positioned in the required state. This refresh function is carried out by a dummy read cycle, since the read/refresh logic are eliminated.

3. 3-device DRAM cell. This cell structure is pictorially shown in Figure 2.14, being made up of three transistors and one capacitor. It stores its information or data as a charge in a single capacitor, Cg, thereby relegating the need for a

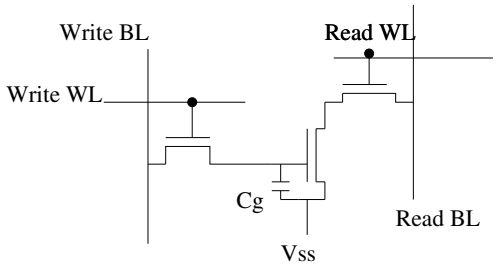


Figure 2.14: Electrical structures of the 3-device DRAM cell

second capacitor as in the case of the 4 and $6-$device cell structures.

In the same way, need for one of the cross-coupled transistor pair is consequently eliminated because the logic stored in the in a cell is kept by the capacitor and a single transistor. However, although a single capacitor and transistor is enough to store its data, it still maintains two extra transistors because it has separate WL and BL. Its data is written by driving the write WL high and forcing the write BL to the desired value. Data can be read by precharging the read BL and then driving the read WL high. A refresh operation consists of a read, followed by a write through the write BL.

4. 1-device DRAM cell. This is the most widely known and deployed DRAM cell structure. It basically comprises of one capacitor, C and an enhancement mode transistor as depicted in Figure 2.15. The drain of the transistor is connected to the bit line, BL while the gate is connected to WL.

   Before reading the cell, the bit lines are precharged with a voltage midway between the low and high voltage. Precharging bit lines decreases the settling time of the bit lines because the voltage difference between the connected bit line and the cell is smaller. After precharging, either BL or $\overline{BL}$ is connected to the cell in order to read it. When reading, the contents of a cell are destroyed as the charge of the capacitor is leaked to the sense amplifier. This operation is called a destructive read. Because there is a single word line for the entire row, the contents of the entire row are destroyed. Therefore, all cells in the row are rewritten to retain the information. When data is written to cells in a row, all the cells in this row are rewritten. This happens because cells can only be accessed as a single row. The unmodified cells are read (which destroys their contents) and rewritten to keep their values.

   A main advantage of the 1-device cell structure is that it has the lowest power requirement and size compared to the other cell structures. It also provides high density and low cost per bit memory cell.

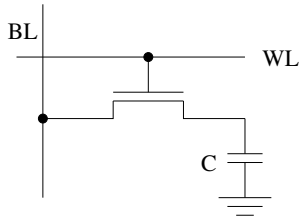


Figure 2.15: Electrical structures of the 1-device DRAM cell

2.3 SRAM model and characteristics

An SRAM functional block diagram is depicted in Figure 2.16. As shown, the functional model consists of sub-functional blocks described in this section.

- **The memory cell array:** The memory cell array consists of several cells, which are addressed by an N-bit address. On the one hand, the row decoder is used for selecting the requested row also referred to as *word line (WL)* among several rows in the memory cell array.
- **Address decoders:** As shown in Figure 2.16, the row decoder receives its input, i.e., specified address from the *address latch* through the address lines, and then selects the required WL in the memory cell array. As the specified WL is selected, all the cells connected to the selected WL become active and thus, put their data on the BLs connected to each cell. On the other hand, the column decoder is used for selecting required BL or BL pair among several BLs or BL pairs that are on the selected WL.
- **Read/write circuitry:** The read/write circuitry consists of the the required devices to read or write a cell. The *data in* line conveys the data to be written to BT and BC, while the *data-out* lines conveys the data that has been read. The read circuitry is more complex than the write circuitry, importantly consisting of data registers and the sense amplifiers.

2.3.1 SRAM memory cell

The SRAM memory array is made up of several individual cells, each connected to BLs and their corresponding WL. These memory cells form the most basic part of the memory. There are different types of SRAM cells based on the type of load used in the elementary inverter of the cell. We identify three types of SRAM memory cells namely, *four-transistor (4T)*, *six-transistor (6T)*, and *thin film transistor (TFT)* SRAM cells. Each of these SRAM cell types are described below as follows.

- **6T cell:** The 6T SRAM cell is depicted in Figure 2.17. The cell consists of six transistors, which are four NMOS transistors and two PMOS transistors The

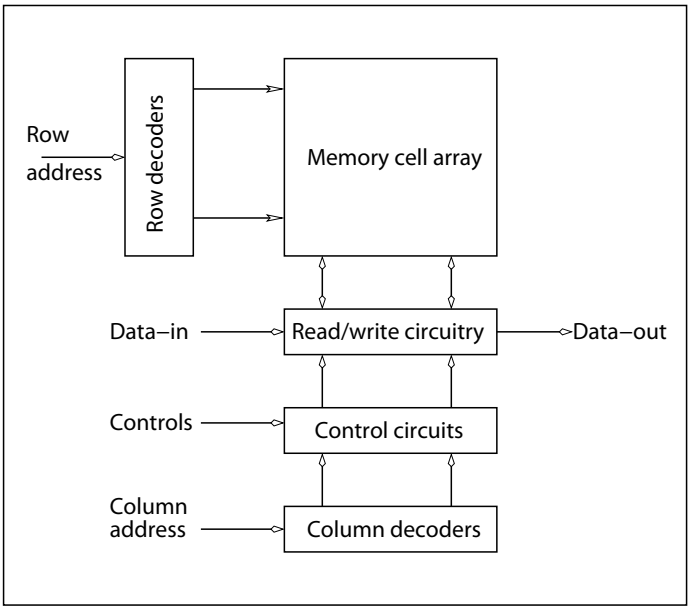


Figure 2.16: SRAM functional block diagram

four transistors form two cross coupled inverters used in SRAM to hold the bits of memory. This cell offers better electrical performances in terms of speed, noise immunity and standby current than the 4T cell structure. 6T cell has better switching performance, a higher impedance, and is relatively insensitive to power supply variations [78, 42]. The work in this thesis has been done using the 6T cell structure.

- **4T cell:** The 4T SRAM cell is depicted in Figure 2.18. The 4T cell SRAM consists of four NMOS transistors and two poly-load resistors [40]. The most common SRAM cell consists of four NMOS transistors plus two poly-load resistors. Two NMOS transistors are pass-transistors. These transistors have their gates tied to the word line and connect the cell to the columns. The two other NMOS transistors are the pull-downs of the flip-flop inverters. The loads of the inverters consist of a very high poly-silicon resistor. This design used to be popular because of its size compared to a 6T cell, since the cell needs only four NMOS transistors. Despite its size advantage, the 4T cells have several limitations. These include the fact that each cell has current flowing in one resistor (i.e., the SRAM has a high standby current), the cell is sensitive to noise and soft errors because the resistance is so high, and the cell is not as fast as the 6T cell.
- **TFT:** This SRAM cell type reduces the current flow through the resistor load of the old 4T cell [55]. This change in electrical characteristics of the resistor

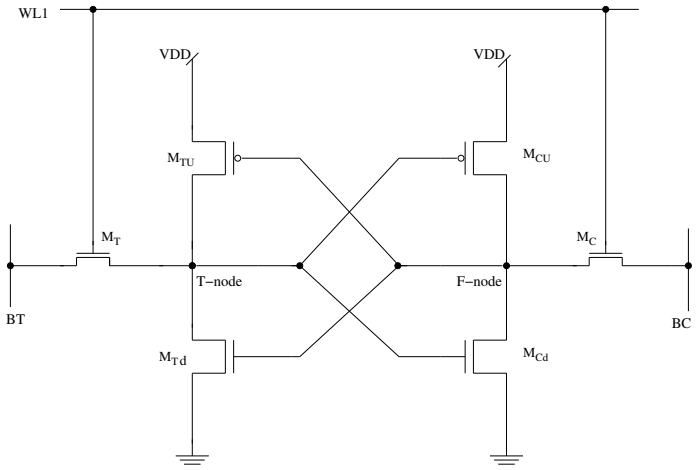


Figure 2.17: 6T SRAM cell

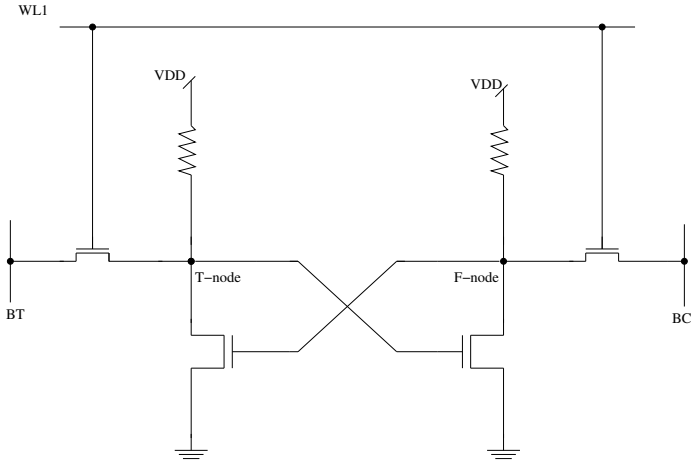


Figure 2.18: 4T SRAM cell

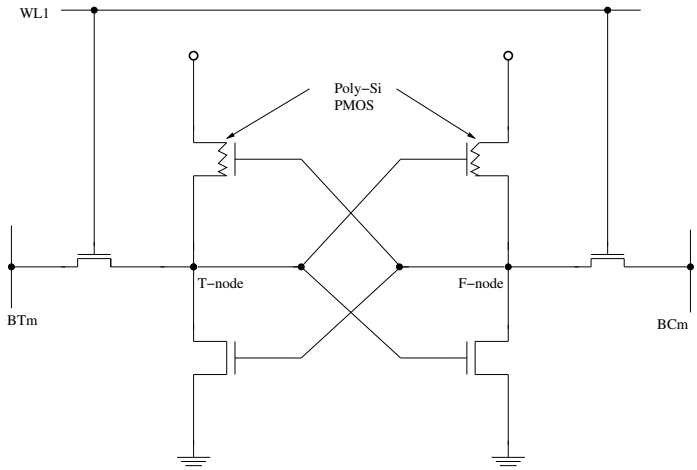


Figure 2.19: TFT SRAM cell

load is done by controlling the channel of a transistor. The resistor is configured as a PMOS transistor and is called a thin film transistor (TFT) [40, 55]. It is formed by depositing several layers of poly-silicon above the silicon surface. The source/channel/ drain is formed in the poly-silicon load. The gate of TFT is poly-silicon and is connected to the gate of the opposite inverter as in the 6T cell architecture. The oxide between this control gate and the TFT poly-silicon channel must be thin enough to ensure the effectiveness of the transistor. Note that the performance of the TFT PMOS transistor is not as good as a standard PMOS silicon transistor used in a 6T cell.

2.3.2 SRAM operations

As depicted in the 6T cell SRAM shown in Figure 2.17, each bit in an SRAM is stored on four transistors that form two cross-coupled inverters. This storage cell has two stable states, which are used to denote a logic 0 and a logic 1. Two additional pass transistors M_T on the true node side (T-node) and M_C on the false node side (F-node) control the access to a storage cell during the read and write operations.

M_T and M_C are connected to WL through which access to the cell is obtained. M_T and M_C also control the cell's connection to the true and complementary BLs (BT and BC), where BT and BC are used for data transfer for both read and write operations. During read accesses, the bit lines are actively driven high and low by the inverters in the SRAM cell. This improves SRAM bandwidth compared to DRAMs. The symmetric structure of SRAMs also allows for differential signaling, which makes small voltage swings more easily detectable. An SRAM cell can operate in three different states, namely:

- **Write:** A write occurs when the content of the cell is updated. Initially, the value to be written is applied to BLs. In order to write a logic 0, BT is set to 0, while BC is set to 1. A logic 1 is written in the cell by setting BT to 1, and BC to 0. WL is then asserted and the value that is to be stored is latched in. Note that the write circuitry (see Figure 2.16) is designed to be much stronger than the relatively smaller transistors in the cell, thus, they can easily override the previous state of the cross-coupled inverters. This underscores the need for proper sizing of the transistors in an SRAM cell to ensure proper operation.
- **Read:** A read occurs when a request has been issued, and the content of a cell assessed. For example, if it is required to read a logic 1 from the cell, the read cycle is initiated by pre-charging BT and BC to a logic 1, and then asserting WL which in turn enables the pass transistors M_T and M_C. The content of the cell, in this case a logic 1 is transferred to the BT and BC. BT remains at its pre-charged value, while BC is discharged through M_C and M_{Cd} to a logic 0.

  In another example, if it is required to read a logic 0 from the cell, then BT is discharged to 0 through M_T and M_{Td}, while BC remains charged and is rather pulled up by transistors M_C and M_{Cu}.
- **No operation:** This occurs when the circuit is in an idle state. An idle state occurs when WL is not asserted, thus, M_T and M_C disconnect the cell from BT and BC. However, the two cross-coupled inverters formed by M_{Tu}, M_{Td}, M_{Cu} and M_{Cd} continue to reinforce each other following this disconnection.

Chapter 3

Functional fault modeling approaches

A fault can be defined as a wrong output or internal state given by a defective device. A *defect* refers to an unintended *physical* difference between an implemented device and its intended design. In order to represent the defective behavior of the memory device and compare it with the expected or correct behavior, an abstraction of the memory device is made. This abstraction is referred to as a model. Models are created at different levels of abstraction, and models that specifically describe faults are called *fault models*.

One of the objectives of fault modeling is to simplify the process of test development by identifying specific types of testing required, which invariably reduces testing cost and time, while retaining the capability to detect the presence of targeted faults. Another objective is to facilitate the analysis of real defects, which have become more rampant due to increased shrinking of devise dimensions, and are often very difficult to analyze.

Fault models used to describe memory faults at the functional level are called *Functional fault models* (FFMs). In this case, the observed memory behavior is compared with the expected behavior at the functional level. FFMs can be defined using *fault primitives* (FPs). FPs are the corresponding deviations between an observed behavior and the expected behavior of the memory [4]. FPs in part comprise of memory operation sequences.

3.1 Memory operation sequence

Memory operation sequences refer to operations performed on memory cells. Such operations include:

- *Read operation*. A read operation is the operation performed on the memory cell, which yields the content of the cell. This operation can be either a read-0

($r0$) or read-1 ($r1$), where the logic values 0 and 1 refer to the content of the cell and is the expected output of the read operation. For example, $r0$ means: read a value from a specified cell and expect this value to be 0. However, the expected value of a read can differ from the actual content of the cell in the presence of a fault.

- *Write operation*. A write operation is the operation performed on the memory cell to input a logic value into the cell. It can be either a write-0 (*w0*) or write-1 ($w1$) operation. For $w0$, 0 is the required value to be written as input into the targeted cell, while for a $w1$, 1 is the required input.
- *No operations*. A no operation (*Nop*) does not change the state of the memory.

Thus, a sequence could be a single or any combination of these memory operations.

An operation sequence is referred to as a *sensitizing operation sequence*, if when performed yields a contrast (fault) between the observed and the expected behavior of the memory device. This contrast or faulty behavior, can be induced by the operation sequences performed on the memory. The components of the sensitizing operation sequence include, listing the *initial* value stored in the cell, and the read or write operation performed on the cell, which will sensitize a fault. We can represent an operation sequence expected to result in a faulty behavior with the notation [4]:

$$D_{\text{ci}} \ldots D_{\text{cm}} O D_{\text{cj}} \ldots O D_{\text{cn}} \tag{3.1}$$

where c = address of cell used; m = number of initialization(s); n = number of operations; D = Data stored in cell c, $D \in \{0, 1\}$; O = operation performed on c, $O \in \{r, w\}$.

An example to illustrate this notation is given as: $1_{\text{c}} w0_{\text{c}} r0_{\text{c}}$. The interpretation of this sequence is as follows. The *initial* data content of the cell (that is, D_{ci}) is 1. $w0$ is performed on cell c, which inputs the value 0. Subsequently, $r0$ is performed on c, to obtain the expected value 0 at the output.

3.2 Fault primitives

An FP is defined as a difference or contrast in behavior of an observed memory device from its correct or expected behavior [95], and is denoted as:

$$FP =< S/F/R > \tag{3.2}$$

In this notation, S represents the sensitizing operation, F denotes the content of of the faulty cell, and $\in \{0, 1\}$, while R represents the logic output value of the read operation and $R \in \{0, 1, -\}$. The values 0 or 1 denotes the value of R when the sensitization of the fault is as a result of a read operation. However, '-' is used to

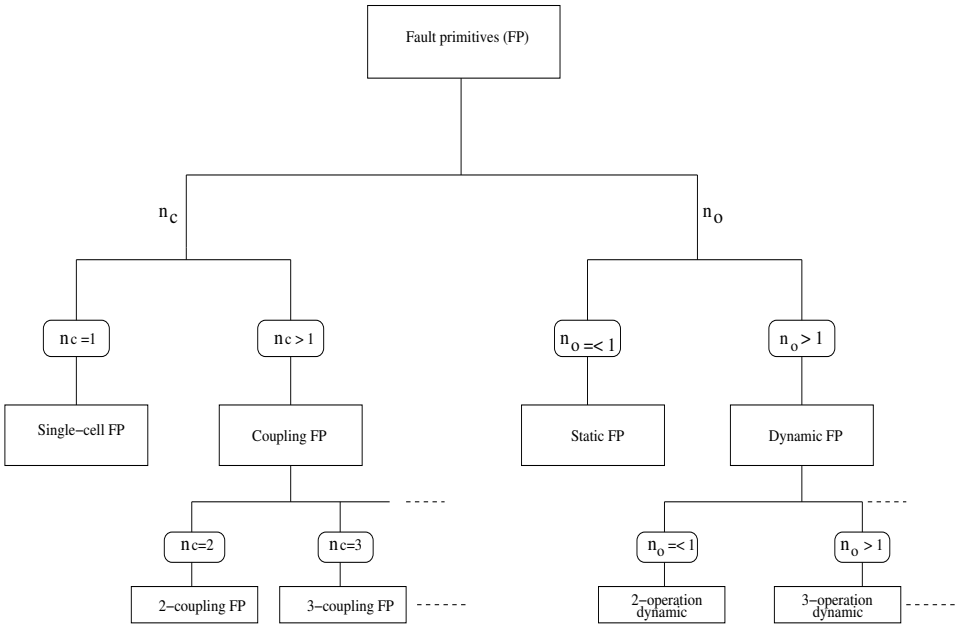


Figure 3.1: Classification of fault primitives

represent the outcome when sensitization is as a result of a write operation, and no output is expected.

For example, consider the FP = $< 0w1/0/- >$. In this example, S is $0w1$. This implies that the cell initially contains a logic 0, after which the value 1 is written into it. Next, the content of the cell, F, is 0, showing that after sensitization the cell still contains a logic 0 and has not made the required transition from 0 to 1. This non-transition indicates the presence of a fault. "$-$" indicates that no output is expected since S comprises no read operation. Using the FP notation, FFMs are defined as a non-empty set of FPs.

As depicted in Figure 2.8, FPs are divided into two categories based on their sensitizing operations such as:

- According to the number of memory cells involved (n_c)
- According to the number of operations performed (n_o)

3.2.1 Classification of FPs by number of cells

Following the number of cells (n_c) involved in a sensitizing operation sequence, this type of FPs can be further classified into two categories, namely,

1. **Single-cell FPs**: For single-cell static FPs, a maximum of $n_c = 1$ is involved. The sensitization operation is performed on, and affects only one cell.

2. **Coupling FPs**: For coupling FPs, $n_c \geq 2$. Since $n_c > 1$, one of the cells in the sensitizing sequence (S) is referred to as the *victim* (v), while the other cell is the aggressor (a). The faulty behavior is related to the victim while the aggressor only contributes to the fault. When n_c = 2, the FP is called a *2-coupling* FP, and when n_c = 3, it is known as a *3-coupling* FP and so on.

3.2.2 Classification of FPs by number of operations

Another classification of FPs is based on the number of memory operations n_o involved. In this classification two types exist:

1. **Static FPs**. In static FPs, fault sensitization is caused by a maximum of one operation, such that $n_o \leq 1$. For example consider the description: S = $< 1_{c1} w0_{c1} >$. Since the number of operations (i.e., $w0$) performed on $c1 = 1$, therefore this is a *static FP*.
2. **Dynamic FPs**. For dynamic FPs, the fault is sensitized by more than one operation, such that $n_o > 1$. Thus, where $n_o = 2$, it is called a *2-operation dynamic* FP, when $n_o = 3$, it is called a *3-operation dynamic* FP, when $n_o = 4$, it is called a *4-operation dynamic* FP and so on.

Figure 3.1 depicts the classification of FPs based on both n_c and n_o.

3.3 Static fault models

This section describes static FFMs and their FPs [95]. FFMs are generally defined as a non-empty set of FPs.

In order to specify a memory fault, one has to represent it in the form of an FP, denoted as $< S/F/R >$. Again, *S* refers to a state or the operation sequence that sensitizes the fault, *F* describes the logic value in the faulty cell ($F \in \{0, 1\}$), while *R* describes the logic output value of a read operation ($R \in \{0, 1, -\}$). This means that *R* has a value of 0 or 1 when the fault is sensitized by a read operation, while '$-$' is used when a write operation sensitizes the fault. For example, consider the FP $< 1w0/1/- >$, which is the Down-Transition Fault. In this FP, $S = 1w0$ implies that a $w0$ operation is applied to a cell initialized to a logic 1. The fault effect, F = 1, indicates that after performing $w0$ the cell remains in state 1. The output of the read operation ($R = -$) indicates that there is no expected output for the memory, since the sensitizing operation is a read.

Static FFMs comprise FPs that are sensitized when the number of memory operations $n_{op} = 1$, despite the number of memory cells involved. There are two important categories of the static FFMs, which are *single-cell* and *two-cell* static FFMs. Thus, for single-cell static FFMs $n_{op} = 1$ and $n_c = 1$, whereas for two-cell static FFMs,

$n_{op} = 1$, but $n_c = 2$,

3.3.1 Single-cell static faults

Single-cell static faults consist of FPs sensitized by performing at most one operation on only one faulty cell. Table 3.1 lists all single-cell static faults and their corresponding FPs. In total, the FFMs are State Fault (SF), Transition Fault (TF), Write Destructive Fault (WDF), Read Destructive Fault (RDF), Deceptive Read Destructive Fault (DRDF) [3], and Incorrect Read Fault (IRF). The description of each of the static single-cell faults are presented as follows:

Table 3.1: Single-cell static FFMs and corresponding FPs

Fault	Fault primitives	Fault	Fault primitives
SF_0	$< 0/1/- >$	RDF_0	$< 0r0/1/1 >$
SF_1	$< 1/0/- >$	RDF_1	$< 1r1/0/0 >$
TF_1	$< 0w1/0/- >$	$DRDF_0$	$< 0r0/1/0 >$
TF_0	$< 1w0/1/- >$	$DRDF_1$	$< 1r1/0/1 >$
WDF_0	$< 0w0/1/- >$	IRF_0	$< 0r0/0/1 >$
WDF_1	$< 1w1/0/- >$	IRF_1	$< 1r1/1/0 >$

1. **State Fault (SF)**. Here, the logic value in the cell flips or changes its value without any operation (read or write) performed on the cell. State faults can manifest either as *state-0* or *state-1* faults.
   - *State-0 Fault (SF_0).* In this fault, the cell initially has a logic value of 0. However, before being accessed (read from or written to) the cell's content flips from a logic 0 to 1.
   - *State-1 fault (SF_1).* In this fault, the initial content of the cell is a logic 1. However, before being accessed it flips from a logic 1 to 0.

   The FPs for State Faults are shown in the second and third rows of Table 3.1.
2. **Transition Fault (TF)**. A transition fault occurs when the value stored in a cell cannot change from one logic value to another when sensitized. The sensitizing operation for this fault is either a transitive $w0$ or $w1$ operation. This fault can manifest as:
   - *Up-Transition Fault (TF_1).* In this case, a non-transition from a logic 0 to 1 $(0 \rightarrow 1)$ occurs in a given cell.
   - *Down-Transition Fault (TF_0).* In this case, a non-transition from logic 1 to 0 $(1 \rightarrow 0)$ occurs in a given cell.

The FPs for transition faults are described in the fourth and fifth rows of Table 3.1.

3. **Write Destructive Fault (WDF)**. This fault occurs when a write operation not intended to cause a transition in the initial logic content of a cell results in a transition. For example, consider the non-transition writes $1w1$ and $0w0$. In both cases, the cell is written with the same logic values as its initial content, thus the logic value after the write should remain the same as their initial values. However, due to this fault the content of the cell flips after the write. This fault manifests as:
   - *Write-0 Destructive Fault (WDF_0).* Here, the logic 0 content of the cell changes to 1 after a non-transition $w0$ has been performed on the cell.
   - *Write-1 Destructive Fault (WDF_1).* Here, the logic 1 content of the cell flips to 0 after a non-transition $w1$ has been performed on the cell.

   These two faults and their corresponding FPs are listed in the sixth and seventh rows of Table 3.1.
4. **Read Destructive Fault (RDF)**. This fault occurs when a read operation performed on a cell changes the stored logic value in a cell, and yields this incorrect value at the output. The fault is sensitized by either a $r0$ or $r1$ and manifests as:
   - *Read-0 Destructive Fault (RDF_0).* For this fault, the sensitizing operation is $0r0$. Performing a $0r0$ causes a flip in the logic content of the cell from a 0 to 1. The incorrect logic value, 1, is returned and read at the output.
   - *Read-1 destructive fault (RDF_1).* For this fault, the sensitizing operation is $1r1$. Performing a $1r1$ causes a change in logic content of the cell from 1 to 0. The incorrect output 0 is returned at the output.

   These two FPs are listed in the second and third rows of right-hand column of Table 3.1.
5. **Deceptive Read Destructive Faults (DRDF)**. This fault is sensitized by performing a read. It occurs when a $r0$ or $r1$ performed on a cell causes a flip in the logic value stored in the cell. However, despite the changed logic value in the cell, a correct value is returned at the output, thereby hiding the faulty value in the cell [3]. This fault has two different type, namely,
   - *Deceptive Read-0 Destructive Fault ($DRDF_0$)* . Here, a $0r0$ sensitizing operation causes the value stored in the cell to change from 0 to 1. However, a correct logic 0 is still obtained at the output.
   - *Deceptive Read-1 Destructive Fault (RDF_1).* Here, a $1r1$ sensitizing operation causes the stored value in the cell to flip from 1 to 0. Despite this wrong value in the cell, a correct logic 1 is read at the output.

The FPs of both faults are shown in the fourth and fourth rows of right-hand column of Table 3.1.

6. **Incorrect Read Fault (IRF)**. This fault occurs when a sensitizing read performed on a cell yields an incorrect logic value at the output, while the correct logic value is still stored in the cell. There are two types of this fault:
   - *Incorrect Read-0 Fault (IRF$_0$)*. In this case, the sensitizing operation $0r0$ is performed on the cell and the correct logic 0 is retained in the cell. However, at the output, an incorrect logic 1 is read.
   - *Incorrect Read-1 Fault (IRF$_1$)*. In this case, the sensitizing operation $1r1$ is performed on the cell. The cell retains this correct logic 1, but an incorrect logic 0 is read at the output.

   The FPs of this fault are as shown in the sixth and seventh rows of the right-hand column of Table 3.1.

3.3.2 Two-cell static faults

In two-cell static fault model two cells and a maximum of one memory operation are involved, i.e., $n_c = 2$ and $O_p \leq 1$. One of the cells is referred to as the *victim* (v), and the other as the aggressor (a). Whereas the faulty behavior is exhibited by the victim, the aggressor only contributes to the fault. The FPs are sensitized by performing an operation on a such that a fault is sensitized on v. In order to describe the two-cell static fault, the notation of the single-cell static FP = $< S/F/R >$ is modified to indicate both v and a as follows.

$$FP =< S/F/R >=< S_a; S_v/F/R >_{a,v} \tag{3.3}$$

where FP = fault primitive S_a represents the sensitizing operation on the aggressor, S_v is the sensitizing operation on the victim, F denotes the logic value stored in the cell, while R indicates the expected output value.

All two-cell static faults are listed in Table 3.2. These faults include, State Coupling Faults (CF_{st}), Disturb Coupling Faults (CF_{ds}), Transition Coupling Faults (CF_{tr}), Write Destructive Coupling Faults (CF_{wd}), Read Destructive Coupling Faults (CF_{rd}), Incorrect Read Coupling Faults (CF_{ir}) and Deceptive Read Destructive Coupling Faults (CF_{dr}).

In the left-hand side of this table, column two shows the state or sensitizing operation performed on a, while the third column lists the state or sensitizing operation performed on v. Also, the forth and fifth columns lists the content of the cell and the output of the cell respectively. Subsequently, the sixth and seventh columns denotes the FP and FFM of each fault.

Table 3.2: A list of two-cell static faults

No	S_a	S_v	F	R	Fault primitive	FFM	No	S_a	S_v	F	R	Fault primitive	FFM
1	0	0	1	-	$<0;0/1/->$	CF_{st}	**19**	0	1w0	1	-	$<0;1w0/1/->$	CF_{tr}
2	0	1	0	-	$<0;1/0/->$	CF_{st}	**20**	1	1w0	1	-	$<1;1w0/1/->$	CF_{tr}
3	1	0	1	-	$<1;0/1/->$	CF_{st}	**21**	0	0w0	1	-	$<0;0w0/1/->$	CF_{wd}
4	1	1	0	-	$<1;1/0/->$	CF_{st}	**22**	1	0w0	1	-	$<1;0w0/1/->$	CF_{wd}
5	$0w0$	0	1	-	$<0w0;0/1/->$	CF_{ds}	**23**	0	1w1	0	-	$<0;1w1/0/->$	CF_{wd}
6	$0w0$	1	0	-	$<0w0;1/0/->$	CF_{ds}	**24**	1	1w1	0	-	$<1;1w1/0/->$	CF_{wd}
7	$0w1$	0	1	-	$<0w1;0/1/->$	CF_{ds}	**25**	0	0r0	1	1	$<0;0r0/1/1>$	CF_{rd}
8	$0w1$	1	0	-	$<0w1;1/0/->$	CF_{ds}	**26**	1	0r0	1	1	$<1;0r0/1/1>$	CF_{rd}
9	$1w0$	0	1	-	$<1w0;0/1/->$	CF_{ds}	**27**	0	1r1	0	0	$<0;1r1/0/0>$	CF_{rd}
10	$1w0$	1	0	-	$<1w0;1/0/->$	CF_{ds}	**28**	1	1r1	0	0	$<1;1r1/0/0>$	CF_{rd}
11	$1w1$	0	1	-	$<1w1;0/1/->$	CF_{ds}	**29**	0	0r0	0	1	$<0;0r0/0/1>$	CF_{ir}
12	$1w0$	1	0	-	$<1w1;1/0/->$	CF_{ds}	**30**	1	0r0	0	1	$<1;0r0/0/1>$	CF_{ir}
13	$0r0$	0	1	-	$<0r0;0/1/->$	CF_{ds}	**31**	0	1r1	0	0	$<0;1r1/1/0>$	CF_{ir}
14	$0r0$	1	0	-	$<0r0;1/0/->$	CF_{ds}	**32**	1	1r1	1	0	$<1;1r1/1/0>$	CF_{ir}
15	$1r1$	0	1	-	$<1r1;0/1/->$	CF_{ds}	**33**	0r0	0	1	-	$<0r0;0/1/->$	CF_{drd}
16	$1r1$	1	0	-	$<1r1;1/0/->$	CF_{ds}	**34**	0r0	1	0	-	$<0r0;1/0/->$	CF_{drd}
17	0	0w1	0	-	$<0;0w1/0/->$	CF_{tr}	**35**	1r1	0	1	-	$<1r1;0/1/->$	CF_{drd}
18	1	0w1	0	-	$<1;0w1/0/->$	CF_{tr}	**36**	1r1	1	0	-	$<1r1;1/0/->$	CF_{drd}

A short description of each FP as listed in the Table 3.2 is given as follows.

1. **State Coupling Fault ($\mathbf{CF_{ST}}$)**. There is no sensitizing operation for this fault, rather it depends on the initial states of the stored data in the cell. It occurs when v is caused to be in a particular logic state as a result of a being in a given state. There are four different manifestations of this fault. FP #1 to 4 of Table 3.2 gives a description of their FPs.
2. **Disturb Coupling Fault ($\mathbf{CF_{DS}}$)**. This fault occurs in two cells when a read or write, which is performed on a causes v to be in a given logic state. In this case, the sensitizing operation is any operation performed on the a. There are twelve different manifestations of this fault. Their FPs are listed in FP #5 to 16 of Table 3.2.
3. **Transition Coupling Fault ($\mathbf{CF_{TR}}$)**. This fault is sensitized by a transition write, that is $(w0, w1)$ or $(w1, w0)$. Two cells have a transition coupling fault when it is observed that due to a given logic state of a, a transition write performed on v fails. There are four types of this fault as described in FP #17 to 20 of Table 3.2.
4. **Write Destructive Coupling Fault ($\mathbf{CF_{WD}}$)**. The sensitizing operations for this fault are non-transition write operations. The fault occurs when v undergoes a non-transition write, which results in an actual transition from an initial logic value to another when a is in a particular logic state. There are four types of this fault and their FPs are described in FP #21 to 24 of Table 3.2.
5. **Read Destructive Coupling Fault ($\mathbf{CF_{RD}}$)**. This fault is sensitized by a read performed on v. It occurs when a $r1$ or $r0$ performed on v changes or destroys the correct logic value stored in the cell, when the aggressor is in a particular logic state. It returns an incorrect logic value at the output. There are four kinds of this fault, and their FPs are described in FP #25 to 28 of Table 3.2.
6. **Incorrect Read Coupling Fault ($\mathbf{CF_{IR}}$)**. This fault is sensitized by a read performed on v. It occurs when a $r1$ or $r0$ performed on v yields an incorrect logic value at the output, while retaining the correct logic value in the cell, with a in a particular logic state. There are four types of this fault, and their FPs are described in FP #29 to 32 of Table 3.2.
7. **Deceptive Read Destructive Coupling Fault (DRD)**. This fault is sensitized by a read performed on v. Two cells are said to have Deceptive Read Destructive Coupling Fault when a $r1$ or $r0$ performed on v results in an actual change in the logic value stored in the cell. However, despite the faulty logic value in the cell, it returns a correct output, when the aggressor is in a given logic state. There are four types of this fault, and their FPs are listed in FP #33 to 36 of Table 3.2.

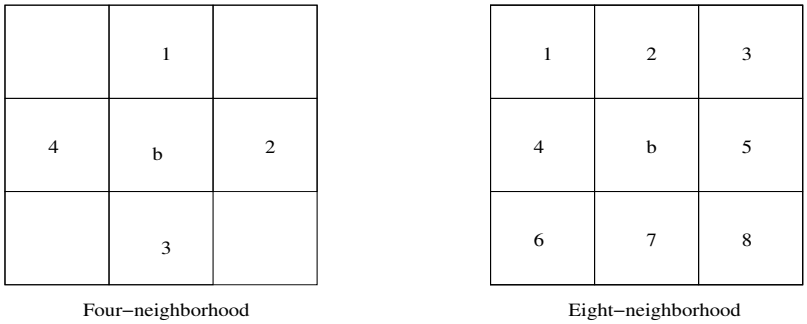


Figure 3.2: Four and eight neighborhood patterns

3.3.3 Multiple cell faults

Multiple cell faults are memory faults involving multiple (more than 2) cells [75]. Now, the total number of cells in a given fault model is referred to as a *neighborhood*. The content of a cell can be influenced by the content or state of other cells in the neighborhood. As depicted in Figure 3.2, the cell currently under test can be referred to as the *base cell (b-cell)*. Other cells in the memory (with the exception of the b-cell), whose influence affects the b-cell are know as the *deleted neighborhood cells*. Faults that occur in the b-cell as a result of the influence of the deleted neighborhood cells are called *neighborhood pattern sensitive faults* (NPSFs).

However, because the b-cell could have numerous neighbors, modeling the influences all possible neighbors becomes difficult and time consuming. Thus, the deleted neighborhood is usually restricted to the physically adjacent cells of the b-cell. The deleted neighborhood could comprise different number of cells as depicted in Figure 3.2. For example,

1. *Four-cell neighborhood*. This is depicted in the left-hand side of Figure 3.2. It comprises the horizontal and vertical neighbors of the b-cell.
2. *Eight-cell neighborhood*. This is shown in the right-hand side of Figure 3.2. It consists of direct adjacent neighbors of the b-cell.

3.4 Fault modeling and memory testing

A semiconductor memory test algorithm is a finite sequence of test elements. Each test element contains a number of memory operations (for example, reads and writes). These memory operations include data pattern backgrounds and addressing sequences for the operations. Among the different types of algorithms proposed for testing SRAMs, march tests have proven to be faster, simpler, and regularly structured [18].

Now, a march test algorithm (or march test) is a finite sequence of march elements. A march element is characterized by an addressing order and a given number of operations. In order to specify a march test, certain notations are used. Some of these notations include:

- r: A read operation, which includes reading a logic 0 ($r0$) or logic 1 ($r1$) from a given cell.
- w: A write operation, which includes writing a logic 0 ($w0$) or logic 1 ($w1$) into a given cell.
- $\Uparrow$: This implies that the addressing sequence of the march operations in the test is in an ascending order.
- $\Downarrow$: This implies that the addressing sequence of the march operations in the test is in a descending order.
- $\Updownarrow$: This implies that the addressing sequence of the march operations in the test is in no particular order, such that it could be either $\Uparrow$ or $\Downarrow$.

Consider this example of a march test:

$$\begin{array}{lll} \text{March Test} = \{ & \Updownarrow(w0); & \text{ME0} \\ & \Updownarrow(w1, r1); & \text{ME1} \\ & \Updownarrow(w1); & \text{ME2} \\ & \Updownarrow(w0, r0)\} & \text{ME3} \end{array}$$

where ME is a march element containing march operation(s) and terminates with a ';'. During testing, all the operations in a march element must be performed before moving to a next cell. The symbols '$\{$' and '$\}$' represent the start and end of the test algorithm.

Many existing fault models are insufficient to represent all important failure in RAMs. Functional fault models are commonly used for memories since the model defines the functional behavior of faulty memories. Thus, new and more representative fault models continue to be researched in order to cover new defects and fails present in modern memories, as a result of new process technologies, manufacturing of new devices, new materials and architectures for cells, circuits and interfaces.

In essence, fault modeling is crucial to memory testing, since the proper testing of any memory devise cannot be adequately done without an understanding and insight into its faulty behavior. Thus, the subsequent chapters of this thesis present some new failure mechanisms, and how to model the occurring new faults. They also present tests developed to properly stress and detect such faults.

Chapter 4

Parasitic bit line coupling effect

Continued scaling for cell area reduction is a key driving force behind the development of modern semiconductor memory devices. Due to this continuous decrease in the cell area, the amount of coupling noise and the sensitivity (in the memory) to defects and failure have continued to increase. Signals in memory devices are carried by lines wired across the memory area such as the word lines (WL). A common attribute due to such connections is a high load and capacitance, which can result in capacitive coupling with other signals and power lines. These parasitic components induce faulty behaviors, which require proper analysis and modeling to facilitate their sensitization and detection.

As a result of the regular structure of memory devices, the optimization of the cell area is highly beneficial. Unfortunately, one of the disadvantages of this trend is the memory's high sensitivity to spot defects. Spot defects in memory devices are geometrical features that emerge during the manufacturing process that were not originally defined by the integrated circuit (e.g., the memory) layout [33, 42, 51]. These defects caused by imperfections in the fabrication process of the devices. Spot defects can cause faults both within and at the peripherals of memory devices and can be classified as opens, bridges or short defects [35, 36, 37, 53, 98]. In order to analyze the faulty effect, simulations are performed on an electrical model of the memory in which the spot defects are injected, a single defect at a time [31, 48]. This chapter will evaluate the impact of parasitic effects on the faulty behavior when spot defects are injected into the memory.

The chapter focuses on the analysis and evaluation of bit line (BL) coupling effect in SRAMs. It presents:

- Modeling of BL coupling effect
- Detailed simulations and evaluation of BL coupling impact on SRAM's faulty behavior in the presence of opens, shorts and bridge defects

- Detection conditions for the faulty behaviors induced by BL coupling

4.1 Modeling BL coupling

Parasitic BL coupling results in the development of small coupling voltages on adjacent BLs, which for example, can influence proper sense amplifier operation. BL coupling has a substantial impact on the faulty behavior of the memory, potentially causing readily detectable memory faults to become undetectable. Such coupling and the resulting cross talk noise is strongly considered as a limiting factor in designing high speed, low power SRAM devices [71]. More so, capacitive coupling was identified as the cause of about 67% field returns in Intel microprocessors in 2002 [41], which buttresses the increasing challenge posed by this effect.

Research on the impact of parasitic capacitance on the faulty behavior of SRAMs has up till now addressed only faults in peripheral memory circuits as well as address decoders [5, 6, 14, 25, 44, 97], and evaluating the way BL coupling and BL twisting maps a given memory fault into a different fault [90]. In other studies to solve the BL coupling challenge, BL twisting as well as BL segmentation (global and local bit lines) have been proposed to prevent cross talk noise and increase the *signal-to-noise ratio* [73, 74]. However, such solutions focus mainly on overcoming BL coupling effect from a design perspective, in order to prevent data destruction during a write operation [93], or in order to reduce the BL delay time during the precharge cycle, thereby increasing overall memory performance [72].

Despite these solutions, BL coupling effect remains a problem due to the expensive cost of implementation of such solutions, thereby making them infeasible for many applications. It becomes imperative therefore to develop other methods to properly detect BL coupling effects in memory devices.

4.1.1 Quantifying BL coupling effect

The electrical Spice SRAM model used in the evaluation of BL coupling effects in this work is presented in Figure 4.1. The model transistor parameters are based on the 65nm BSIM4 model card as described by the Predictive Technology Model [101].

The memory has an array of memory cells to enable simulation of all neighboring coupling effects. For a 3x3 cell array, the cells are connected to three BL pairs: *left BL* (BLl), which has the *left true* (BTl) and *left complementary* (BCl) BLs, *middle BL* (BLm), which has the *middle true* (BTm) and *complementary* BLs (BCm), and the *right BL* (BLr), which has BTr and BCr BLs.

Each WL or cell array row in the model has 3 cells: left (l), middle (m) and right (r); while each BL or cell array column has 3 cells numbered as 0, 1 and 2. The cell

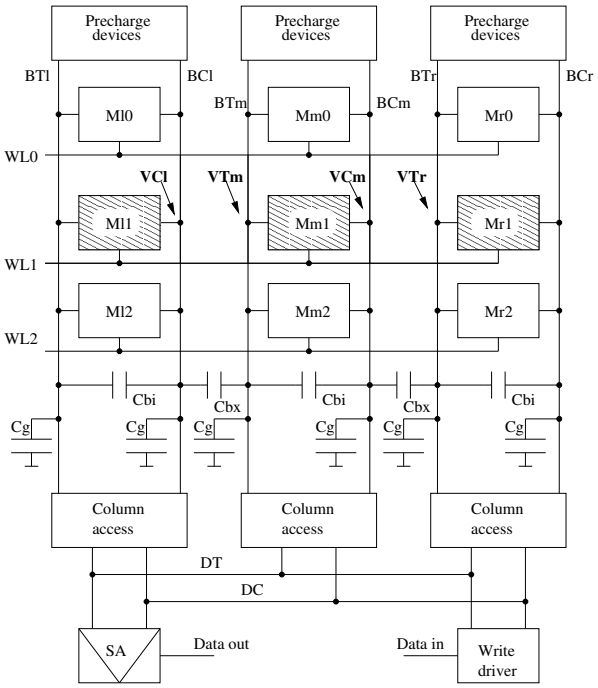


Figure 4.1: SRAM electrical Spice model

in the center of the array (i.e., memory cell Mm1) is the faulty cell under analysis. Each BL is also connected to precharge devices to ensure proper initial BL voltages. The read/write access to different BLs is controlled by the column access devices, which ensure that only one BT gets connected to the *true data line (DT)* and only one BC gets connected to the *complementary data line (DC)* during each memory operation. The model also contains a *sense amplifier (SA)* to inspect the read output (data out), as well as a write driver to drive input data (data in). Note that since WLs are driven by strong voltage signals (V_{DD}), there is no significant impact on a selected WL imposed by adjacent unselected WLs in terms of coupling. Thus, WL coupling has not been considered in this work.

The total BL capacitance (C_t) is made up of three components: internal coupling to the complementary BL (C_{bi}), external coupling to a neighboring BL (C_{bx}) and an inherent BL capacitance to ground (C_g) composed of coupling to all other parts of the memory (cells, WLs, substrate, etc). This is expressed as:

$$C_t = C_{bi} + C_{bx} + C_g \tag{4.1}$$

The exact values of these capacitances depend on the layout of the memory and its manufacturing technology. In general, the value of C_g accounts for a large portion

of C_t. In literature, reported C_g/C_t ratios range from 40% to over 90% [93]. On the other hand, due to the symmetry of the layout implementation of the BLs, the values of C_{bi} and C_{bx} are rather close to each other, and therefore are considered to be equal (i.e., $C_{bi} = C_{bx} = C_b$) such that:

$$C_t \approx 2C_b + C_g \tag{4.2}$$

We focus on read operations. The reason is that read operations are more sensitive to the impact of coupling than write operations. During a read operation, the WL accesses the cell and connects it to the precharged BLs. Based on the value stored in the cell, a voltage differential develops on the BLs that the sense amplifier subsequently attempts to detect. The presence of C_b causes neighboring BLs to influence the voltage development during a read. If we assume that a defective BL is totally floating, while the neighboring BL develops a voltage *V*, then the amount of coupling voltage (ΔV) induced on the floating BL can be expressed as:

$$\frac{\Delta V}{V} \approx \frac{1}{(C_g/C_b) + 1} \tag{4.3}$$

It is essential to understand how a specific initialization of a neighborhood of cells affects the sensing of a given faulty cell, in order to write such worst-case values in the neighboring cells (worst stress condition) during testing. From a testing perspective, it is possible to use BL coupling to introduce extra stress on specific faults, thereby making them easier to detect by a given test All of these will increase fault/defect coverage.

4.1.2 Impact on faulty behavior

In this section, we present the theoretical analysis of the impact of BL coupling.

In the memory cell array, when a specific victim cell is accessed, the only neighboring cells also being accessed at the same time are those that belong to the same row as the victim, that is, those cells connected to the same WL as the victim. As shown in the shaded cells of the model in Figure 4.1, when the middle memory cell (Mm1) is accessed, the only other influential cells are the left memory cell (Ml1) and the right memory cell (Mr1) connected to the same WL1.

For this reason, it is interesting to focus on the effect of coupling, and varied data in both Ml1 and Mr1 on the faulty behavior of the victim (Mm1) in the presence of different defects. Now, we explain the impact of the data contents of the neighboring cells, Ml1 and Mr1, referred to as *coupling backgrounds (CBs)* on the sensing of Mm1.

- **Impact of Ml1 on Mm1**

  If cell Ml1 contains a 1, then when it is accessed, it pulls BCl down by some voltage VCl. Due to BL coupling, this in turn pulls the voltage on BTm down by VTm (Figure 4.1). Thus, the presence of a logic 1 in Ml1 makes the detection of logic 1 in Mm1 more difficult while it makes the detection of logic 0 easier. On the other hand, having a 0 in cell Ml1 does not modify the voltage on BCl, which in turn does not modify the voltage on BTm. In brief,

  – In order to maximally stress logic 1 in Mm1, Ml1 must contain a logic 1.

  – In order to stress a logic 0 in Mm1, Ml1 must *not* contain a logic 1, thereby requiring a stored logic 0 instead.

- **Impact of Ml1 on Mm1**

  If cell Mr1 contains a 0, then when it is accessed, it pulls BTr down by some voltage VTr. Due to BL coupling, this in turn pulls the voltage on BCm down by VCm (Figure 4.1). Thus, the presence of a logic 0 in Mr1 makes the detection of logic 0 in Mm1 more difficult while it makes the detection of logic 1 easier. On the other hand, having a 1 in cell Mr1 does not modify the voltage on BTr, which in turn does not modify the voltage on BCm. In brief,

  – In order to maximally stress logic 0 in Mm1, Mr1 must contain a logic 0.

  – In order to stress a logic 1 in Mm1, Mr1 must *not* contain a logic 0, thereby requiring a stored logic 1 instead.

In conclusion, the most stressful background to detect parasitic BL coupling in an SRAM cell containing a logic 1 is 11 in both neighboring cells connected to the same WL (we refer to this as CB11). In contrast, the most stressful background to detect a logic 0 is CB00. Thus, CB11 and CB00 are referred to as *worst-case coupling backgrounds (WCB)*.

4.2 BL coupling impact on opens

At the layout level, open defects (ODs) are usually caused by broken lines or particle contamination that results in increasing line resistivity at the open position. ODs in the memory cell array can be either opens within the cell, opens on BLs or opens on WLs. Figure 4.2 shows all possible ODs within the SRAM cell, on which our analysis is based. The opens within the cell [57] and their complements are listed in Table 4.1. The opens on the true node side (T-node) have an additional subscript 't' and are listed on the upper part of the table, while the opens on the false node side (F-node) are their corresponding complements with the subscript 'c' and are listed on the lower part of the table. Two defects are said to be *complementary (c)*, when their locations are symmetrical to each other within the cell.

Table 4.1: Open defects on the true & false nodes

OD	**Position on T-node side**
$R1_t$	Pass transistor connection to BL broken (drain)
$R2_t$	Pass transistor connection to WL broken (gate)
$R3_t$	Pass transistor connection to T-node broken (source)
$R4_t$	NMOS down transistor connection to T-node broken (drain)
$R5_t$	NMOS down transistor connection to ground broken (source)
$R6_t$	NMOS down transistor connection to F-node broken (gate)
$R7_t$	PMOS up transistor connection to T-node broken (drain)
$R8_t$	PMOS up transistor connection to F-node broken (gate)
$R9_t$	PMOS up transistor connection to V_{DD} broken (source)
OD	**Position on F-node side**
$R1_c$	Pass transistor connection to BL broken (drain)
$R2_c$	Pass transistor connection to WL broken (gate)
$R3_c$	Pass transistor connection to F-node broken (source)
$R4_c$	NMOS down transistor connection to F-node broken (drain)
$R5_c$	NMOS down transistor connection to ground broken (source)
$R6_c$	NMOS down transistor connection to T-node broken (gate)
$R7_c$	PMOS up transistor connection to F-node broken (drain)
$R8_c$	PMOS up transistor connection to T-node broken (gate)
$R9_c$	PMOS up transistor connection to V_{DD} broken (source)

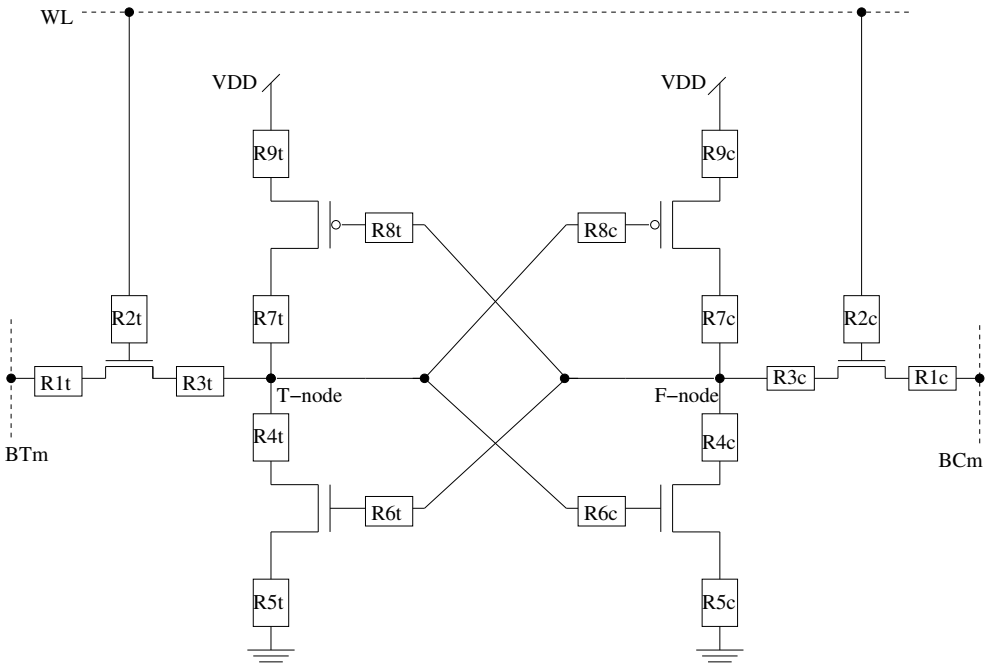


Figure 4.2: Open defect positions in SRAM cell

4.2.1 Simulation of open defects

The analysis in this section focuses on ODs within the SRAM cell. As shown in Figure 4.2, an open resistor (R_{def}) is injected on the defective signal line denoted in the figure as $R1_t$, $R2_t$,..., $R1_c$, $R2_c$,..., where *t* and *c* represent the T-node and F-node sides of the cell. R_{def} can vary from 0Ω to $\infty\Omega$. Despite the symmetry that exists between the T and F-nodes in SRAM cells, defects on both sides can exhibit different behaviors, therefore full simulations for each OD have been performed and analyzed.

For each OD evaluated, all scenarios are considered, namely, $r0$ and $r1$ operations performed for each CB (i.e., Ml1 and Mr1 containing all possible logic values when Mm1 is 0 or 1) for a number of $\frac{C_g}{C_b}$ values. The value of C_g is considered to be a typical 500fF [32], while $\frac{C_g}{C_b}$ values are modified for each simulation in the range $1 \leq C_g/C_b \leq 20$ [7], with used C_b values as 500fF, 100fF, 50fF, 30fF and 25fF.

In general, both the value of R_{def} as well as the amount of the coupling capacitance influence the BL voltage differential and therefore decide the eventual output value of the memory. This creates a space of possible ($\frac{C_g}{C_b}$, R_{def}) values, where the defective cell can either function properly or fail. The specific resistive value in the R_{def} range, beyond which a fail occurs is called the *critical resistance (R_{cr})*. Our analysis is based

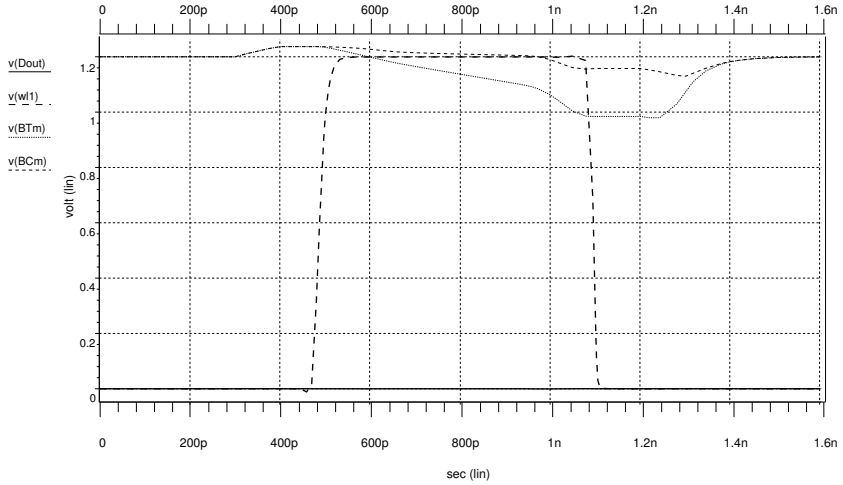


Figure 4.3: Defect-free read 0 of OD-R1$_t$ with CB00, $\frac{C_g}{C_b}$ = 10

on detecting the differences in behavior between a properly functional circuit and its behavior after an OD and the parasitic BL components have been injected.

4.2.2 Simulations of OD-R1$_t$ and OD-R1$_c$

In this section, we analyze the simulation and results for each read operation when OD-R1$_t$ and OD-R1$_c$ are injected using all possible CBs.

Read at Mm1 with OD-R1$_t$.

OD-R1$_t$ is the defect injected between the access transistor and BTm. The presence of this defect limits the ability of the cell to discharge BTm, thereby reducing the voltage differential between BTm and BCm, and making the sense amplifier more prone to crosstalk errors.

Figure 4.3 shows the simulation result of a defect-free $r0$ in cell Mm1, at $\frac{C_g}{C_b}$ = 10 and CB00. Once WL1 is accessed, a differential voltage starts to develop between BTm and BCm, which is then detected by the sense amplifier and amplified as a full 0, thereby leaving the data out (Dout) line at 0.

On the other hand, Figure 4.4 shows the simulation results of a defective $r0$ performed on Mm1, with R_{def} = 110KΩ and $\frac{C_g}{C_b}$ = 10 using CB00. Comparing these with the defect-free simulation results in Figure 4.3, we identify a number of differences. First, the differential voltage developing on the BLs is significantly reduced

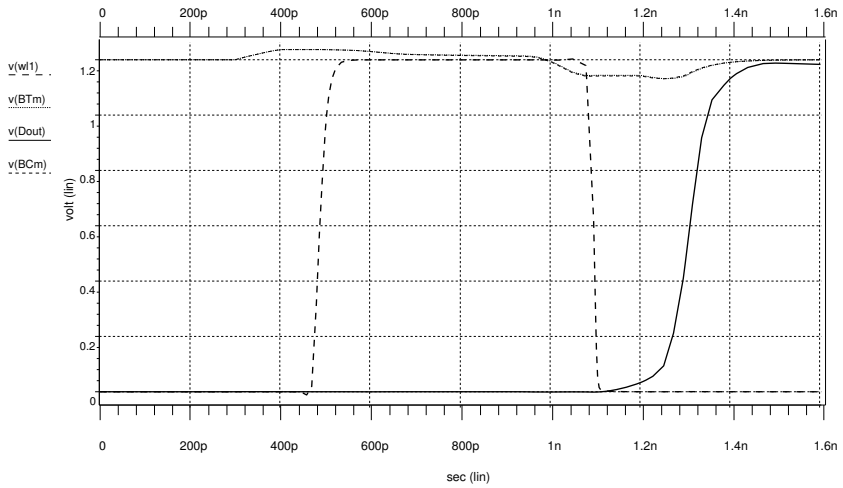


Figure 4.4: Read 0 of OD-R1$_t$, with CB00, $\frac{C_g}{C_b}$ = 10, R$_{def}$ = 110KΩ

in the defective case between $t = 0.4$ ns and $t = 1.4$ ns, making it extremely difficult for the sense amplifier to identify the correct stored value in the cell. Adding to that the BL coupling voltage from neighboring cells causes the sense amplifier to detect an incorrect logic 1 in the cell rather than a logic 0, as indicated by the Dout signal in the figure.

For all simulated $\frac{C_g}{C_b}$ values, OD-R1$_t$ behavior is plotted and depicted as curve CB000t in Figure 4.6. In the plot, the $x-$axis denotes $\frac{C_g}{C_b}$, while the $y-$axis represents R_{def} values. Each curve in the figure divides the ($\frac{C_g}{C_b}$, R_{def}) plane into two regions. The region above the curve is the *fail* region while the region below is the *pass* region. Note that only CBs for which fails have been observed are included in the plot.

As curve CB000t in Figure 4.6 indicates, the fail region expands gradually as the amount of coupling capacitance increases (i.e., decreasing $\frac{C_g}{C_b}$ values). This buttresses the importance of keeping the amount of BL coupling capacitance small relative to the total capacitance of BL. It also indicates that with continued technology scaling, the importance of testing for coupling effects will increase.

Result of a $r0$ using CB11 is as shown in Figure 4.5. Again, due to the defect in the memory cell, the differential voltage on BLs is very limited. However, as a result of the CBs in the neighboring cells and BL coupling effects, the differential voltage is biased towards detecting a logic 0 in the cell. This bias corrects the faulty behavior and prevents detecting a fail. In the same way, using other CBs (CB10 and CB01)

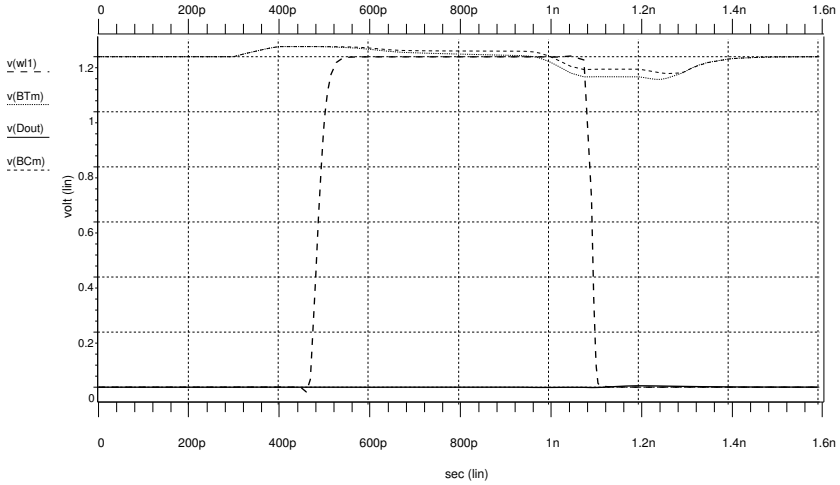


Figure 4.5: Read 0 of OD-R1$_t$, with CB11, $\frac{C_g}{C_b}$ = 10, R_{def} = 110KΩ

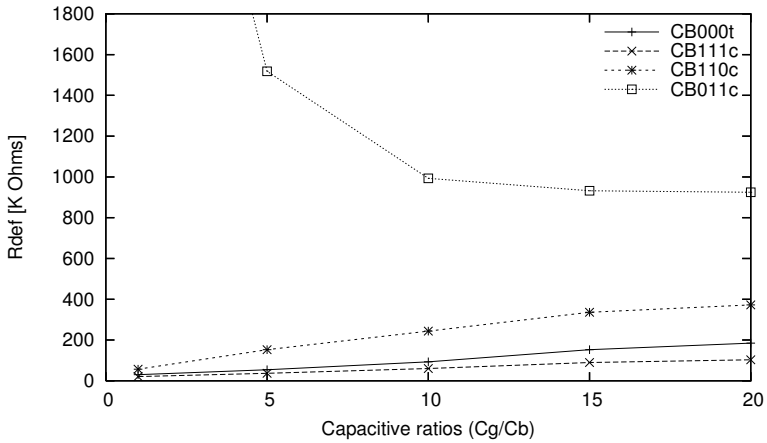


Figure 4.6: Pass and fail regions for OD-R1$_t$ and OD-R1$_c$

corrects the faulty behavior and prevents any fault from being detected.

A $r1$ in the presence of OD-R1$_t$ will produce a correct logic 1 at the output irrespective of CBs used. The reason is that for a read operation, BL voltages are influenced by the cell node voltages. Since BLs are precharged to V_{DD}, only one BL is discharged during the operation, in this case BCm. Since BTm is not discharged, a read 1 operation will yield a correct logic 1 output.

Read at Mm1 with OD-R1$_c$.

OD-R1$_c$ lies between BCm and the pass transistor at the F-node side (symmetric counterpart of OD-R1$_t$) of the cell. For $r1$ using CB11, the differential voltage devel-

oping on BLs is significantly reduced in the defective case between $t = 0.4$ ns and $t = 1.4$ ns. An increase in coupling capacitance results in making the faulty behavior more dominant causing the sense amplifier to record an incorrect logic 0 instead of a logic 1. Plots of R_{cr} at varying $\frac{C_g}{C_b}$ for OD-$R1_c$ for CB 11 is shown in Figure 4.6 depicted by curve CB111c. Coupling due to both CB01 and CB10 also yielded incorrect read outputs as depicted by curves CB011c and CB110c in the figure.

Thus, CB01, CB10 and CB11 are stressful, while the use of CB00 corrects the faulty behavior. A $r0$ in the presence of OD-$R1_c$ succeeds irrespective of the CB used.

4.2.3 Analysis of other ODs

In this section, we present a behavioral summary for the rest of the open defects in the memory cell. The simulation analysis and results for OD-$R1_t$ ... OD-$R9_t$ (all on the T-node side) are listed in the upper part of Table 4.2, while those for OD-$R1_c$ ... OD-$R9_c$ (all on the F-node side) are listed in the lower part of Table 4.2. The first column of Table 4.2 lists the ODs, while the second column lists the corresponding worst-case CB. The third column gives the values of R_{cr} at $\frac{C_g}{C_b}$=10 for the worst-case CB. The fourth column lists whether other CBs are also stressful ($+$) or not ($-$) for each given OD.

As listed in Table 4.2, for OD-$R4_t$ and OD-$R5_t$ very low R_{cr} values were recorded. This underscores the high sensitivity to a resistive open on the pull-down transistor, which is on the current path for a $r0$. This is also the case for OD-$R4_c$ and OD-$R5_c$ at the F-node side of the cell while performing $r1$ where BCm is discharged.

Furthermore, In the presence of OD-$R7_t$ ... OD-$R9_t$ the cell exhibits a defect-free behavior irrespective of the CB used. These three ODs represent broken connections on the source, gate and drain of the pull-up transistor. For a $r0$ in SRAM, when WL1 is activated, current flows through the NMOS pass transistor on the BT side, through the pull-down NMOS transistor to ground. Since this necessary current path for a $r0$ does not pass through OD-$R7_t$ ... OD-$R9_t$, the cell exhibits a defect-free behavior such that the sense amplifier gives a correct logic 1 output for all performed simulations. Here, a delay fault can occur, which takes place a while after the operation is performed. Special tests are used to detect such faults [29]. In the same way, OD-$R7_c$ ... OD-$R9_c$ on the F Node side exhibit a complementary behavior for $r1$ operation.

4.3 BL coupling impact on shorts

A *short* can be defined as a connection between one memory node and V_{DD} or GND. Short defects (SDs) in the memory cell array can be classified generally as shorts within the cell, shorts at BLs and shorts at WLs. A list of SDs is given in Table 4.3.

Table 4.2: Simulation results of open defects

Defects Read 0	Worst CB	R_{cr} of CB00 at $\frac{C_g}{C_b}$=10	Stressful CBs		
			01	10	11
OD-R1$_t$	00	93KΩ	−	−	−
OD-R2$_t$	00	2.43MΩ	−	+	−
OD-R3$_t$	00	45KΩ	−	+	−
OD-R4$_t$	00	2.28KΩ	+	+	+
OD-R5$_t$	00	1.54KΩ	+	+	+
OD-R6$_t$	00	38GΩ	+	+	+
OD-R7$_t$	−	−	−	−	−
OD-R8$_t$	−	−	−	−	−
OD-R9$_t$	−	−	−	−	−
Read 1	Worst CB	R_{cr} of CB 11	00	01	10
OD-R1$_c$	11	61KΩ	−	+	+
OD-R2$_c$	11	2.08MΩ	−	+	+
OD-R3$_c$	11	29KΩ	−	+	+
OD-R4$_c$	11	8KΩ	+	+	+
OD-R5$_c$	11	3.86KΩ	+	+	+
OD-R6$_c$	11	207GΩ	+	+	+
OD-R7$_c$	−	−	−	−	−
OD-R8$_c$	−	−	−	−	−
OD-R9$_c$	−	−	−	−	−

Table 4.3: Position of short defects

Shorts within the cell	**Position**	**Complement**
SDC-R1	T_m - V_{DD}	SDC-R1$_c$: F_m - V_{DD}
SDC-R2	T_m - GND	SDC-R2$_c$: F_m - GND
Shorts at BL	**Position**	**Complement**
SDB-R1	BT_m - V_{DD}	SDB-R1$_c$: BC_m - V_{DD}
SDB-R2	BT_m - GND	SDB-R2$_c$: BC_m - GND
Shorts at WL	**Position**	**Complement**
SDW-R1	WL - V_{DD}	-
SDW-R2	WL - GND	-

Table 4.4: Simulation results for shorts

Shorts Read 0	WCB	R_{cr} of WCB at $\frac{C_g}{C_b}$=10	Stressful CBs 01	10	11	00
SDC-R1	00	5.017KΩ	+	+	+	+
SDC-$R1_c$	−	−	−	−	−	−
SDC-R2	−	−	−	−	−	−
SDC-$R2_c$	00	23.183KΩ	+	+	+	+
SDB-R1	00	0.047KΩ	+	+	+	+
SDB-$R1_c$	−	−	−	−	−	−
SDB-R2	−	−	−	−	−	−
SDB-$R2_c$	00	7.738KΩ	+	+	+	+
SDW-R1	−	−	−	−	−	−
SDW-R2	−	−	−	−	−	−
Read 1	WCB	R_{cr} of WCB	01	10	11	00
SDC-R1	−	−	−	−	−	−
SDC-$R1_c$	11	0.915KΩ	+	+	+	+
SDC-R2	11	7.272KΩ	+	+	+	+
SDC-$R2_c$	−	−	−	−	−	−
SDB-R1	−	−	−	−	−	−
SDB-$R1_c$	11	0.052KΩ	+	+	+	+
SDB-R2	11	6.488KΩ	+	+	+	+
SDB-$R2_c$	−	−	−	−	−	−
SDW-R1	−	−	−	−	−	−
SDW-R2	11	2.101KΩ	+	+	+	+

For each evaluated SD, all scenarios are considered namely, $r0$ and $r1$ operations performed using all possible CBs for various $\frac{C_g}{C_b}$ values. Table 4.4 gives a summary of the results for both $r0$ and $r1$ operations.

In the table, SDC represents shorts within the cell, SDB denotes shorts at BLs, while SDW refers to shorts at WLs. The first column of the table lists the SDs, while the second column lists their corresponding WCBs. The third column gives the values of R_{cr} at $\frac{C_g}{C_b}$=10 for each WCB, while the fourth column lists whether other CBs cause a fail $(+)$ or not $(-)$ for each given SD. As shown, CB00 and CB11 are the WCBs for $r0$ and $r1$ operations respectively. These WCBs represent the worst-case coupling backgrounds required for stressing these operations; something that is important to consider while deriving tests that would detect faults in the presence of BL coupling.

4.4 BL coupling impact on bridges

A Bridge is a connection between any arbitrary pair of nodes. We identify two categories of bridge defects (BDs), namely, bridges within the cell and bridges between cells. Note that nodes need to be located close to each other to make BDs between them possible [47, 90].

4.4.1 Location of bridges

1. Bridges within the cell

BDs within the cell connect two nodes of the same cell, and this includes the pair of BLs (i.e., BT and BC) and WL to which the cell is connected. Each cell consists of five nodes, {T-node, F-node, BT, BC, WL}. Thus, the number of bridges resulting from pairing the listed nodes is 10 as enumerated in Table 4.5, denoted as *BDC*. BDCs with complementary behavior have been listed on the same row.

However, despite the symmetry that exists between the T and F-nodes in SRAM cells, complementary defects can exhibit different behaviors [56], therefore full simulations for each BDC and the corresponding complement have been performed and analyzed.

2. Bridges between cells

BDs between cells connect nodes of adjacent cells, which include BL pairs and WL to which they are connected. BDs between the cells include BrDs between cells on the *same row*, BDs between cells on the *same column*, and BDs between cells on the *same diagonal*.

- **Cells on the same row**

  For adjacent cells on the same row, consider all possible BDs between Ml1 and Mm1 and refer to them as BDLs, while all possible BDs between Mm1 and Mr1, are referred to as BDR. Table 4.5 lists all BDs. It lists the BDs on the first column, and indicates the BD position on the second column. The third column lists the complementary defects, the fourth column lists the interchange defects *(i)*, while the last column lists the interchange complement *(ic)*. An interchange behavior (involving two cells) occurs if the faulty behavior of one of the cells is similar to that of the other cell, such that the difference is an interchange of the aggressor and the victim.

  To determine the full space of BDL, each cell consists of five nodes, where the nodes of cell Ml1 are $\{T_l, F_l, BT_l, BC_l, WL1\}$, and nodes of cell Mm1 are $\{T_m, F_m, BT_m, BC_m, WL1\}$. However, because WL1 is common to both Ml1 and Mm1, only $\{T_l, F_l, BT_l, BC_l\}$ and $\{T_m, F_m, BT_m, BC_m\}$ can form bridges. Thus, the possible number of BDLs is 16. In the same way, the possible number of BDRs between Mm1 and Mr1 is also 16 as shown in Table 4.5.

Table 4.5: Position of bridges

Bridges within the cell				
BDC	position	complement	interchange	int. comp
BDC-R_1	T_m - F_m			
BDC-R_2	T_m - BT_m	F_m - BC_m		
BDC-R_3	T_m - BC_m	F_m - BT_m		
BDC-R_4	T_m - WL1	F_m - WL1		
BDC-R_5	BT_m - BC_m			
BDC-R_6	BT_m - WL1	BC_m - WL1		
Bridges between cells on the same row (left side)				
BDL	BDL	complement	interchange	int. comp
BDL-R_1	T_l - T_m	F_l - F_m		
BDL-R_2	T_l - F_m	F_l - T_m		
BDL-R_3	T_l - BT_m	F_l - BC_m	BT_m - T_r	BC_l - F_m
BDL-R_4	T_l - BC_m	F_l - BT_m	BC_1 - T_m	BT_l - F_m
BDL-R_5	BT_l - BT_m	BC_l - BC_m		
BDL-R_6	BT_l - BC_m		BC_l - BT_m	
Bridges between cells on the same row (right side)				
BDR	BDR	complement	interchange	int. comp
BDR-R_1	T_m - T_r	F_m - F_r		
BDR-R_2	T_m - F_r	F_m - T_r		
BDR-R_3	T_m - BT_r	F_m - BC_r	BT_m - T_r	BC_m - F_r
BDR-R_4	T_m - BC_r	F_m - BT_r	BC_m - T_r	BT_m - F_r
BDR-R_5	BT_m - BT_r	BC_m - BC_r		
BDR-R_6	BT_m - BC_r		BC_m - BT_r	
Bridges between cells on the same column				
BDU	BDU	complement	interchange	int. comp
BDU-R_1	T_m - T_{m0}	F_m - F_{m0}		
BDU-R_2	T_m - F_{m0}	F_m - T_{m0}		
BDU-R_3	T_m - WL_{m0}	F_m - WL_{m0}	WL_m - T_{m0}	WL_m - F_{m0}
BDU-R_4	WL_m - WL_{m0}			
Bridges between cells on the same diagonal				
BDG	BDG	complement	interchange	int. comp
BDG-R_1	T_m - T_{r0}	F_m - F_{r0}		
BDG-R_2	T_m - F_{r0}	F_m - T_{r0}		

- **Cells on the same column**

  These are denoted as BDUs. To determine all possible BDUs we consider BDs between Mm0, Mm1. Both Mm0 and Mm1 contain five nodes each. The nodes of cell Mm0 are $\{T_m, F_m, BT_m, BC_m, WL0\}$, while the nodes of cell Mm1 are $\{T_m, F_m, BT_m, BC_m, WL1\}$. Since Mm0 and Mm1 share the same BL, only 9 BDUs exist between Mm0 and Mm1. Note that BDs between the nodes T_m and F_m have not been included. The reason is that they have been considered earlier and included while determining BDCs. The lower part of Table 4.5 lists all BDUs.

- **Cells on the same diagonal**

  Adjacent cells on the same diagonal are denoted as BDGs. Here, we consider all possible BrDGs between diagonal cells such as Mm1 and Mr0. Both Mm1 and Mr0 consists of five nodes each, where the nodes of cell Mm1 are $\{T_{m0}, F_{m0}, BT_m, BC_m, WL1\}$, and nodes of cell Mr0 are $\{T_{r0}, F_{r0}, BT_r, BC_r, WL0\}$. Since BDs connecting WL and BLs of Mm1 have been already considered, only four bridges are derived for BDGs as listed also in the lower part of Table 4.5.

4.4.2 Simulation analysis of bridges

In this section, simulation analysis and results for bridges are discussed. For each evaluated BD, all scenarios are considered namely, $r0$ and $r1$ operations performed using all CBs for $\frac{C_g}{C_b}$ values. The value of C_g is also considered to be a typical 500fF [32], while $\frac{C_g}{C_b}$ values are modified for each simulation in the range $1 \leq C_g/C_b \leq 20$ [7], with C_b values as 500fF, 100fF, 50fF, 30fF and 25fF.

Our analysis is also based on detecting the differences in behavior between a properly functional circuit and its behavior after each BD and parasitic BL coupling components have been injected. The injected resistances for each bridge (R_{br}) can vary within the range of $0 \leq R_{br} \leq \infty$.

Simulation analysis of BDC-R1

Here, we analyze the simulated results for read operations for BDC-R1 using all CBs and $\frac{C_g}{C_b}$ values.

1. BDC-R1: Read 0 at Mm1

The defect BDC-R1 is injected between the T and F-nodes of cell Mm1. In order to clearly demonstrate the impact of BL coupling and influence of CBs, we simulate three scenarios as follows. First, a defect-free $r0$ on Mm1 as depicted in Figure 4.7, then a defective $r0$ on Mm1 with CB00 as shown in Figure 4.8 and a defective $r0$ on

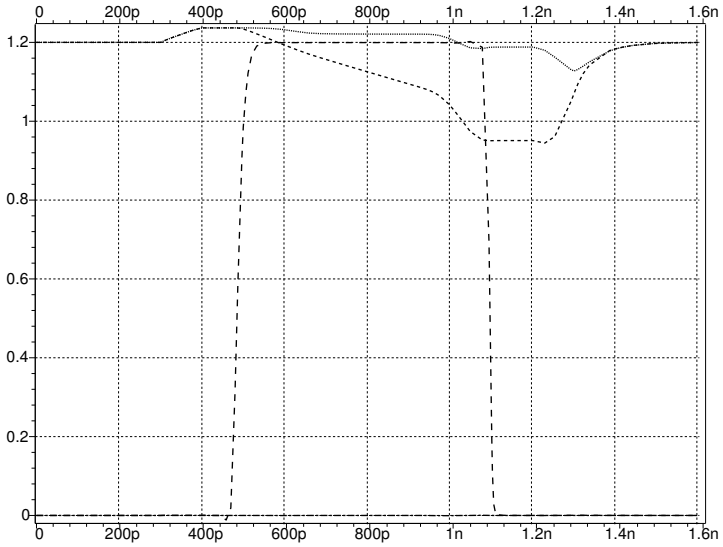


Figure 4.7: Defect-free read 0 with BL coupling

Mm1 with CB10 as depicted in Figure 4.10. Once WL1 is accessed, a differential voltage starts to develop between BTm and BCm, which is detected by the sense amplifier. This voltage is amplified as a full 0, thereby leaving the data out (Dout) line at 0.

Figure 4.8 shows a $r0$ performed on Mm1, when the defect BDC-R1 is injected with BL coupling ($\frac{C_g}{C_b}$ = 10), at CB00. The value of the injected bridge is R_{br}=18.40KΩ. At this particular R_{br} value, comparing Figure 4.8 with Figure 4.7, readily shows a number of differences. First, the differential voltage developing on the BLs is significantly reduced in this case as shown in Figure 4.8 between $t = 0.4$ ns and $t = 1.4$ ns, thereby making it extremely difficult for the sense amplifier to identify the correct stored value in the cell. Thus, the BL coupling voltage from neighboring cells causes the sense amplifier to detect an incorrect logic 1 in the cell rather than a logic 0, as shown by the Dout signal in the figure.

For all simulated $\frac{C_g}{C_b}$, R_{cr} of BDC-R1 is plotted and depicted as curve *CB00* in Figure 4.9. In the plot, the $x-$axis indicates $\frac{C_g}{C_b}$, while the $y-$axis represents R_{br} values. The curve in the figure divides the ($\frac{C_g}{C_b}$, R_{br}) plane into two regions. The region above the curve is the *pass* region while the region below is the *fail* region. Note that only CBs for which fails have been recorded are included in the plot.

As curve CB00 in Figure 4.9 indicates, the fail region expands gradually as the amount of coupling capacitance increases. Thus, at resistances above R_{cr}, the cell exhibits a defect-free behavior, whereas at resistances below R_{cr}, the faulty behavior

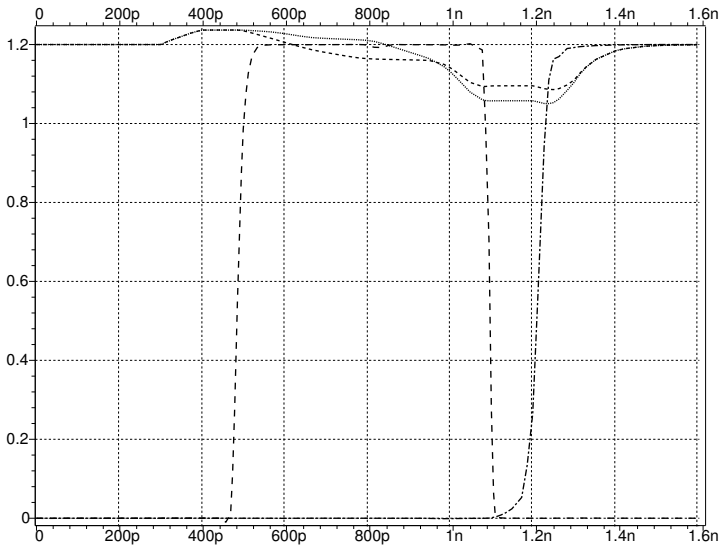


Figure 4.8: Read 0 with BDC-R1$_t$ and BL coupling, at CB00, $\frac{C_g}{C_b}$ = 10 and R_{br}=18.40KΩ

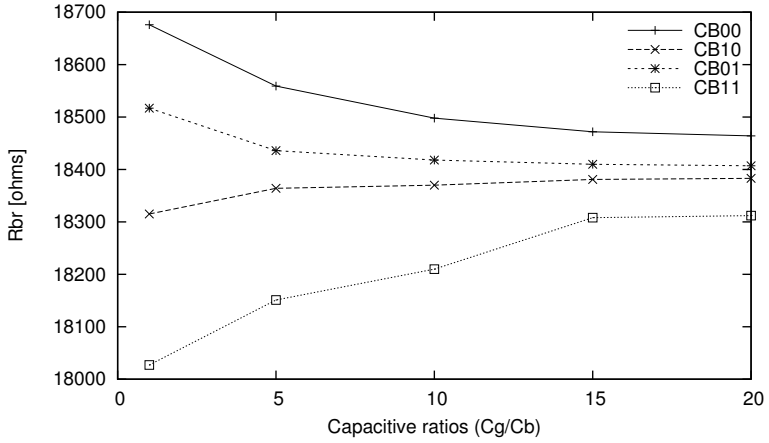


Figure 4.9: Pass and fail regions for read 0 with BDC-R1$_t$

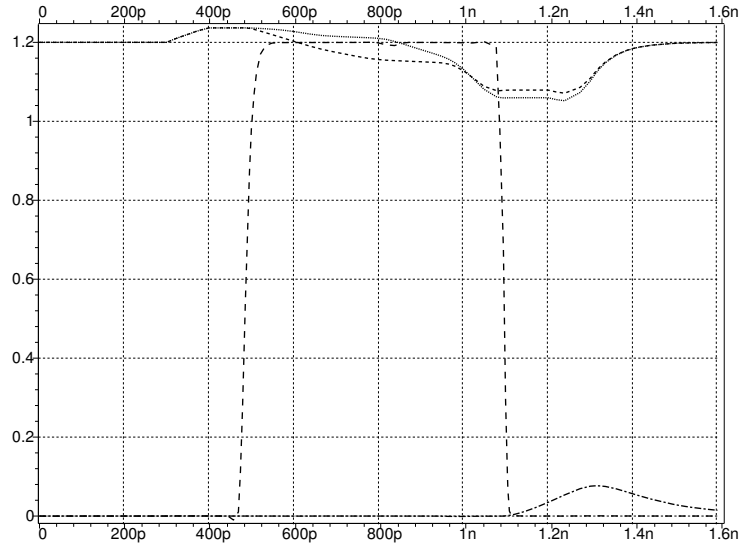


Figure 4.10: Read 0 with BDC-R1$_t$ and BL coupling, at CB10, $\frac{C_g}{C_b}$ = 10 and R_{br}=18.40KΩ

manifests. Likewise, using CB01, due to BD at 18.40KΩ in the memory cell, the differential voltage on BLs is very limited and the differential voltage is biased towards detecting an incorrect logic 1 in the cell. However, using other CBs (10 and 11) corrects the faulty behavior and prevents a fail from being detected, as indicted by the Dout signal in Figure 4.10 for CB10. Thus, WCB is CB00 due to the high R_{br} value at which CB00 necessitated a fail.

2. BDC-R1: Read 1 at Mm1

For a $r1$ using CB11, the differential voltage developing on BLs is significantly reduced in the defective case between $t = 0.4$ ns and $t = 1.4$ ns. An increase in coupling capacitance causes the sense amplifier to record an incorrect logic 0 instead of a logic 1. Plots of R_{cr} at varying $\frac{C_g}{C_b}$ for BDC-R1 for CB11 is shown in Figure 4.11 depicted by curve *CB11*.

Coupling due to both CBs 01 and 10 also yielded incorrect read outputs as depicted by curves CB01 and CB10. Thus, CBs 01, 10 and 11 caused a fail, while CB00 rather corrects the faulty behavior. CB00 is not included in Figure 4.11 since it did not necessitate a fail. Thus, CB11 is the WCB because it returned R_{cr} values higher than the rest of the CBs.

3. Analysis of other BDCs, BDLs and BDRs

In this section, we present a behavioral summary for the rest of the bridge defects

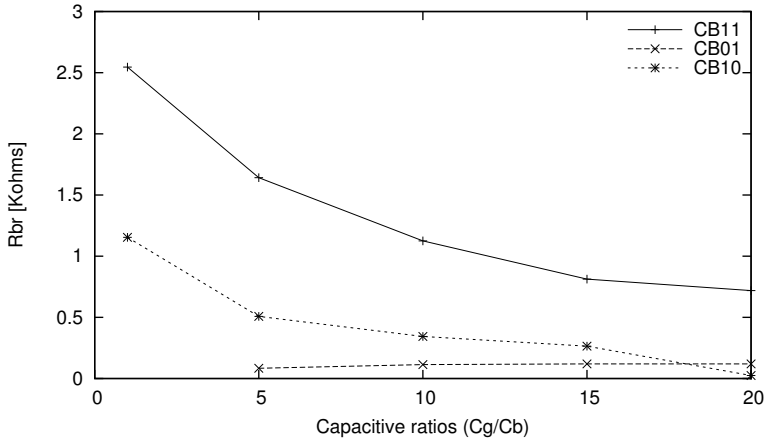


Figure 4.11: Pass & fail regions for read 1 with BDC-R1

Table 4.6: Simulation results for bridges - BDCs

Results for BDs within the cell						
Defects BD	WCB	R_{cr} of WCB at $\frac{C_g}{C_b}$=10	Stressful CBs			
			01	10	11	00
BDC-$R1_t$	00	18.498KΩ	+	+	+	+
BDC-$R2_t$	00	4.18KΩ	+	+	+	+
BDC-$R2_c$	−	−	−	−	−	−
BDC-$R3_t$	00	8.046KΩ	+	+	+	+
BDC-$R3_c$	−	−	−	−	−	−
BDC-$R4_t$	−	−	−	−	−	−
BDC-$R4_c$	00	7.946KΩ	+	+	+	+
BDC-$R5_t$	−	−	−	−	−	−
BDC-$R6_t$	−	−	−	−	−	−
BDC-$R6_c$	−	−	−	−	−	−

within the cell (i.e., BDC), and between adjacent cells. The analysis and results for BDC-$R1_t$ ... BDC-$R6_t$ with their complements, interchange and comp. interchange behaviors are listed in Table 4.6. Likewise, results for BDL-$R1_t$ ... BDL-$R6_t$, their complement, interchanged and interchanged complement behaviors are listed in Table 4.7, and for BDR-$R1_t$ ... BDR-$R6_t$, with their complement, interchange and interchanged complement behaviors are also listed in Table 4.7.

The first column of Table 4.6 and Table 4.7 list the BDs, while the second column lists their corresponding WCBs. The third column gives the values of R_{cr} at $\frac{C_g}{C_b}$=10 for the WCBs. The fourth column lists whether other CBs necessitated a fail (+) or not (−) for each given BD.

Table 4.7: Simulation results for bridges - BDLs

Results for BDs of two cells on the same row (L)						
BD	WCB	R_{cr} of WCB	11	01	00	10
BDL-$R1_t$	−	−	−	−	−	−
BDL-$R1_c$	10	21.212KΩ	+	−	−	+
BDL-$R2_t$	00	18.198KΩ	−	+	+	−
BDL-$R2_c$	−	−	−	−	−	−
BDL-$R3_t$	−	−	−	−	−	−
BDL-$R3_c$	10	6.621KΩ	+	−	−	+
BDL-$R3_i$	00	4.478KΩ	+	+	+	+
BDL-$R3_{ic}$	−	−	−	−	−	−
BDL-$R4_t$	00	6.286KΩ	−	+	+	−
BDL-$R4_c$	−	−	−	−	−	−
BDL-$R4_i$	10	4.440KΩ	+	+	+	+
BDL-$R4_{ic}$	−	−	−	−	−	−
BDL-$R5_t$	−	−	−	−	−	−
BDL-$R5_c$	−	−	−	−	−	−
BDL-$R6_t$	−	−	−	−	−	−
BDL-$R6_i$	−	−	−	−	−	−
Results for BDs of two cells on the same row (R)						
BD	WCB	R_{cr} of WCB	11	00	10	01
BDR-$R1_t$	−	−	−	−	−	−
BDR-$R1_c$	01	21.316KΩ	+	−	−	+
BDR-$R2_t$	−	−	−	−	−	−
BDR-$R2_c$	00	18.194KΩ	−	+	+	−
BDR-$R3_t$	01	4.488KΩ	+	+	+	+
BDR-$R3_c$	−	−	−	−	−	−
BDR-$R3_i$	−	−	−	−	−	−
BDR-$R3_{ic}$	01	6.716KΩ	+	−	−	+
BDR-$R4_t$	00	4.457KΩ	+	+	+	+
BDR-$R4_c$	−	−	−	−	−	−
BDR-$R4_i$	00	6.133KΩ	−	+	−	−
BDR-$R4_{ic}$	−	−	−	−	−	−
BDR-$R5_t$	−	−	−	−	−	−
BDR-$R5_c$	−	−	−	−	−	−
BDR-$R6_t$	−	−	−	−	−	−
BDR-$R6_i$	−	−	−	−	−	−

For BDC, as listed in Table 4.6, WCB for BDC-R1$_t$, BDC-R2$_t$, BDC-R3$_t$ and BDC-R4$_c$ is CB00. However, for the rest of the BDCs, correct logic outputs were recorded and no fails occurred. These results can be explained as follows.

For example, for BDC-R2$_c$, where the bridge defect lies between the F-node (F$_m$) and BCm, since the content of Mm1 is 0, irrespective of the content of Ml1 and Mr1, the cell will not be modified to yield an incorrect logic 1. The reason is that during a $r0$, BCm remains unchanged whereas only BTm is discharged. Since this BD is located along this unmodified path, influence on the content of Mm1 is very minimal and does not modify the content of the cell. More so, the impact of VCl, (which develops on BCl) will also not have any modifying effect on Mm1 due to the position of the BD thus, the $r0$ operation succeeds.

Also as listed in Table 4.7, the BDLs: BDL-R1$_c$, BDL-R2$_t$, BDL-R3$_c$, BDL-R3$_i$, BDL-R4$_t$ and BDL-R4$_i$ necessitate incorrect logic 1 outputs, whereas the rest of the BDLs all yielded correct logic 0 outputs. However, For BDL-R1$_c$ and BDL-R3$_c$ WCB is CB10. The reason for this behavior is that VTr has no obvious impact on the content of Mm1 due to the location of the bridge, since both cells contain a logic 0, and BCm is not discharged. A logic 0 in Ml1 would as well not impact changing the content of Mm1. However, a logic 1 in Ml1 will cause the content of Mm1 to flip due to the coupling effect of some voltage VCl, which pulls BTm up by some voltage VTm. This therefore explains why CBs 00 and 01 would not necessitate a fail for this BD. Note that the position of BDRs are symmetric to those of BDLs relative to Mm1, thus they have exhibited complementary behaviors as shown by our results.

Chapter 5

Parasitic memory effect

Many types of defects such as spot defects can occur as a result of smaller dimensions of memory devices using today's manufacturing processes. Spot defects include opens, shorts and bridges and can occur at different metal layers of the devices. Test generation for failures caused by such defects depends on fault models that are supposed to represent the defective behavior and allow easy generation of test operations through fault simulation [83]. However, certain real failures in today's integrated circuits may exhibit complex behavior that is not accurately modeled by existing approaches due to factors such as the presence of parasitic capacitance on defective nodes [66].

The presence of parasitic node capacitance on a defective resistive node can induce dynamic changes in the electrical behavior of the circuit in SRAM devices. Such behavior is referred to as the *parasitic memory effect*. This chapter presents an analysis of the parasitic memory effect in SRAMs. The chapter demonstrates that the faulty behavior in SRAMs is exacerbated by the presence of parasitic node capacitances; something that reduces the fault coverage of industrial memory tests, and increases the defect-per-million rates of the out-going devices.

Some research work has been done on the electrical characterization and modeling of resistive opens [8, 12, 13, 23, 48, 52, 81, 83, 84, 86]. However, no work has been done on evaluating the impact of parasitic node capacitance on the faulty behavior of SRAMs. In addition, no memory tests exist nor detection mechanism been developed that account for this faulty behavior in SRAMs.

In this chapter, an analysis of the scenario where a defective node (N), in the SRAM cell array is characterized both by a resistive component, (R_{def}), which may have an arbitrary value, and a parasitic capacitive component (C_{node}) is considered. Taking the parasitic node capacitance into account is important in order to guarantee the proper detection of defects using appropriate detection conditions in the presence of the parasitic memory effect. Therefore this chapter presents:

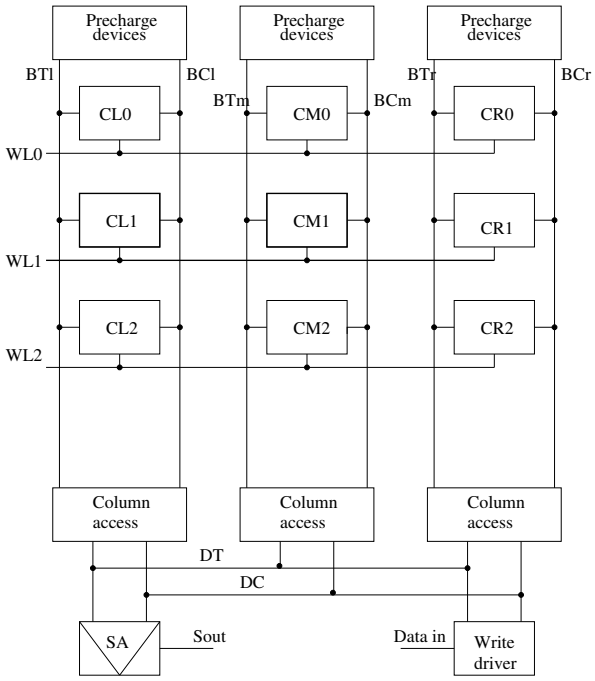


Figure 5.1: Electrical model of SRAM cell array

- The modeling of the parasitic memory effect.
- The impact of the parasitic memory effect on SRAM faulty behavior, showing degradation as a result of the presence of parasitic node capacitance of a defective node.

5.1 Modeling the failure mechanism

In order to model the SRAM failure mechanism, an electrical Spice model of a 3x3 SRAM cell array presented in Figure 5.1 is used for the evaluation. The transistor parameters are based on the 65nm BSIM4 model card as described by the Predictive Technology Model (PTM) [101]. The cells are connected to three BL pairs: *left BL (BLl)*, which has the *left true (BTl)* and *left complementary (BCl)* BLs, *middle BL (BLm)*, which has the *middle true (BTm)* and *complementary (BCm)* BLs, and the *right BL (BLr)*, which has *BTr* and *BCr* BLs.

Each *word line (WL)* or cell array row in the model has 3 cells: left (l), middle (m) and right (r); while each BL or cell array column has 3 cells numbered as 0, 1 and 2. Each BL is also connected to precharge devices to ensure proper initial BL voltages. Read/write access to different BLs is controlled by the column access devices, which ensure that only one BT gets connected to the *true data line (DT)* and only one BC

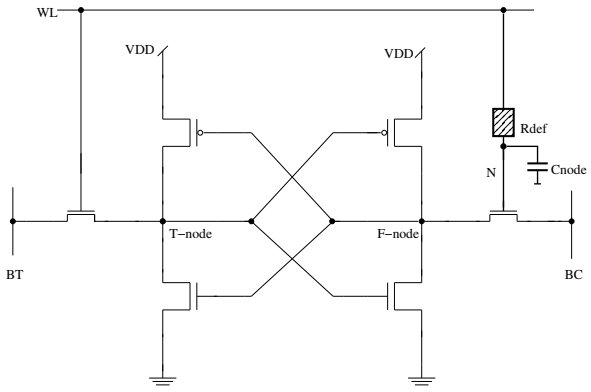


Figure 5.2: SRAM cell showing R_{def} and C_{node}

gets connected to the *complementary data line (DC)* during each memory operation. The model also contains a *sense amplifier (SA)* to inspect the read output (Sout), and a write driver to drive input data (data in).

As shown in Figure 5.2 the defective node (N) in the SRAM device is considered to be characterized by two important components, namely,

- The resistive defect (R_{def}) of the defective node
- The parasitic capacitance (C_{node}) of the defective node

5.1.1 Resistive open defects

As depicted in Figure 5.2, we consider the model where a resistive open defect is injected into an SRAM. Open defects are usually caused by broken lines or particle contamination that result in increasing line resistivity at the open position. In the SRAM cell array, opens can be present within the cell, at BLs, or at WLs.

The analysis of a resistive open injected between WL and the gate of the pass transistor on the false node side (F-node) as shown in Figure 5.2 is presented. An open between WL and the gate of the pass transistor on F-node side will limit connectivity to the gate such that the pass transistor will not function properly.

However, resistive opens combined with parasitic capacitance of a defective node, can modify the timing behavior of the circuit, which can necessitate faults. In this case, such faults can be detected only by tests that expose the incorrect timing behavior due to these parasitic effects.

5.1.2 Modeling parasitic node capacitance

In order to model the total parasitic capacitance (C_{node}) of a defective node, the actual position of the defect is considered. In this analysis, the defect is injected between the gate of the pass transistor and WL as shown in Figure 5.2. Therefore, the total parasitic capacitance of the defective node will comprise the *gate capacitance (C_G)* and the wire or *line capacitance (C_L)* connecting the defect to the gate of the pass transistor. Thus, the total capacitance of the defective node is expressed as:

$$C_{\text{node}} = C_{\text{G}} + C_{\text{L}} \tag{5.1}$$

The CMOS gate capacitance consists of the *gate-source capacitance (C_{gs})*, the *gate-drain capacitance (C_{gd})*, and the *gate-bulk capacitance (C_{gb})*, such that:

$$C_{\text{G}} = C_{\text{gs}} + C_{\text{gd}} + C_{\text{gb}} \tag{5.2}$$

On the other hand, the parasitic line (wire) capacitance (C_L) consists of three main components [76, 15], as depicted in Figure 5.3. These components are:

- *line-to-ground* capacitance (C_{lg}): This is the capacitance between a wire and ground (substrate).
- *line-to-line* capacitance (C_{ll}): This is the coupling capacitance between different wires on the same metal layer.
- *crossover* capacitance (C_{co}): This is the coupling capacitance between wires on different metal layers.

So, the total line capacitance across metal planes is the sum of these three components, that is,

$$C_{\text{L}} = C_{\text{ll}} + C_{\text{lg}} + C_{\text{co}} \tag{5.3}$$

C_L is a function of:

$$C_{\text{L}} = f(H, W, T, S) \tag{5.4}$$

where W is the width of the metal line, T is the thickness of the metal line, H is the thickness of the dielectric layer between metal layers, and S is the spacing between the parallel lines [26, 34, 76].

In advanced MOS technologies, the line-to-line capacitance is comparable to, or larger than the line-to-ground capacitance [15]. The exact values of the capacitances depend on the connecting wire and the manufacturing technology. Note that when the length of the connecting wire is 0, then the only active parasitic capacitance is

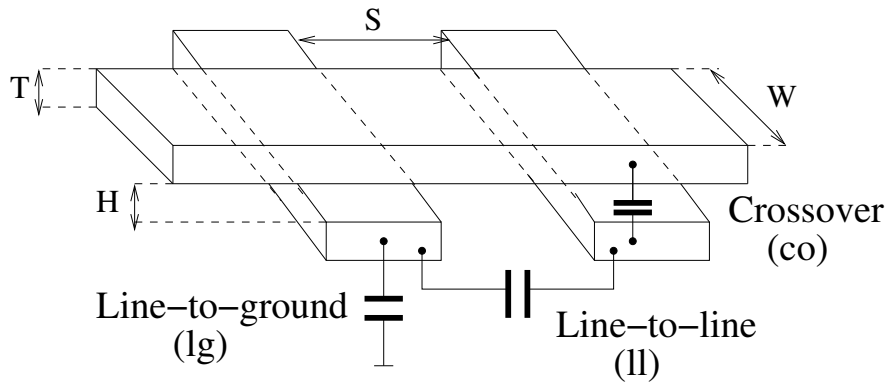


Figure 5.3: Three components of line capacitance

C_{G}. C_{G} is incorporated into the simulation model, while estimated C_{L} values used in this chapter are in the range $0 < C_{\mathrm{L}} < 10$fF [99, 101].

5.2 Impact of parasitic node capacitance

The timing behavior of the SRAM cell can be modified in the presence of parasitic capacitance.

Figure 5.4 depicts simulation results of WL for read operations performed on an SRAM cell using different $\mathrm{R_{def}}$ values in the range $0k\Omega \leq R_{\mathrm{def}} \leq 200k\Omega$ at $\mathrm{C_L} = 4.5$fF. Note that $\mathrm{C_L} = 4.5$fF value is used as an estimate. Other close values of $\mathrm{C_L}$ yield the same behavior shown in this section. In the figure, WL is the defect-free signal, while $\mathrm{WL_{def}}$ represented by 5 defective signals is the defective WL signal each when a different defect resistance $\mathrm{R_{def}}$ has been injected.

As shown in Figure 5.4, after propagation WL signal quickly rises to $\mathrm{V_{DD}}$ at 400ns and maintains this voltage until the fall of the signal to GND at 1.2ns. However, observing the defective $\mathrm{WL_{def}}$ signals, it is clear that each signal is delayed after propagation, such that with an increase in defect values, $\mathrm{WL_{def}}$ subsequently is increasingly delayed from reaching $\mathrm{V_{DD}}$.

Now, in order to show the impact of $\mathrm{C_{node}}$, Figure 5.5 shows two simulation results for another operation sequence. An *operation sequence* (seq) refers to a set of sequential operations performed on a cell, where the operation $\in \{r, w\}$. Here, seq = $\{w1, r1\}$ is performed on the cell in the presence of $\mathrm{R_{def}}$, *with* $\mathrm{C_L}$ and *without* $\mathrm{C_L}$ as a component of the defective node.

In Figure 5.5 the curve denoted as *"$\mathrm{WL_{def}}$ without C_L"*, shows that the defective $\mathrm{WL_{def}}$ signal is slightly delayed, but still reaches $\mathrm{V_{DD}}$ after propagation, and reaches GND before the start of the next operation at 1.5ns. As can also be observed, this

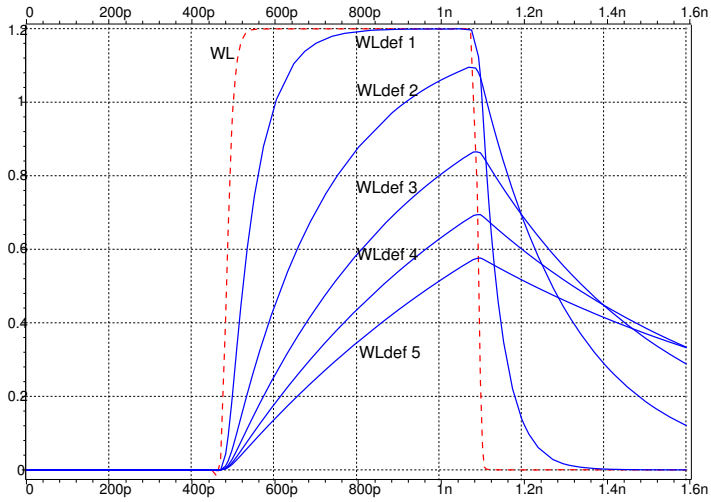


Figure 5.4: R_{def} of $0k\Omega \leq R_{def} \leq 200k\Omega$ at C_L = 4.5fF

signal is slightly distorted between 700ps and 1.1ns. The reason is due to coupling to BLs in the cell array, which is magnified as a result of the resistive open.

In the second simulation in the same figure denoted as *"WL_{def} with C_L"*, where R_{def} = 150kΩ and C_L = 4.5fF, it is observed that a longer delay is created after propagation of the signal, such that WL_{def} neither reaches V_{DD} after propagation, nor GND before the start of the next operation. Consequently, for the next operation, the propagated signal starts with a voltage (V_{def}) higher than the expected initial voltage, and so on. V_{def} is the defective node voltage at the beginning of a clock period. It is important to note that with a sufficiently high V_{def}, WL will be turned on, hence making the cells connected to BLs.

Thus, it is important to remark that at a given R_{def} value, the dynamic faulty behavior of the circuit can be exacerbated by the presence of C_{node}.

5.3 Analysis of the faulty behavior

This section presents a demonstration of how the timing behavior of a defective node can affect the output of an SRAM cell. The analysis of this behavior requires to create some signal transitions, and to propagate them through the circuit, while observing the impact at the defective node and at the output.

The analysis is based on observing the differences in the electrical behavior of the defective (and output) node at different R_{def} and C_L values before and after the defect is injected.

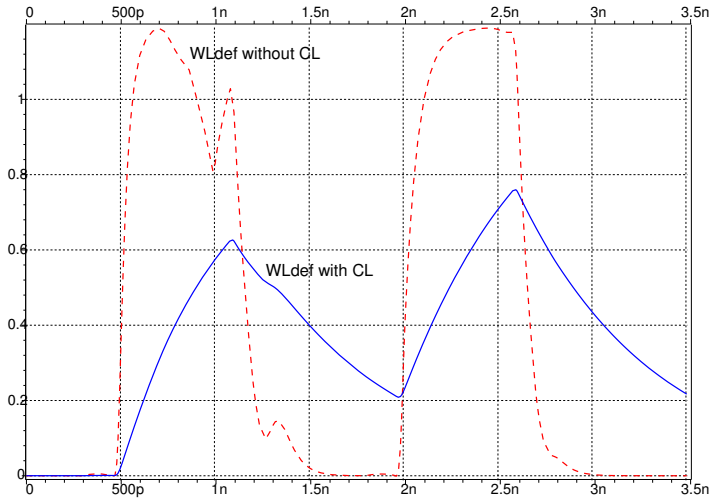


Figure 5.5: Comparison of behavior with/without C_L at R_{def}=150kΩ

5.3.1 Undetectable faulty behavior

Figure 5.6 shows the simulation results of WL for the defect-free signal (WL) and defective signal, WL_{def}. Likewise, Figure 5.7 shows results for the logic output signals of the defect-free (*Sout*) and defective (*Sout-def*) operations.

As shown in Figure 5.6, a relatively small resistance $R_{def} = 50k\Omega$ is injected at $C_L = 4.5$fF. As shown, it is observed that WL_{def} is slowed down, gradually moving from GND to V_{DD}. This observed delay is propagated and can be seen in the Sout-def signal, which is also slowed down compared to Sout. Thus, an additional delay (T_{def}) of 0.04ns occurs as shown in Figure 5.7 between Sout and Sout-def. The propagation of these rising edges therefore allows to make the following definitions:

Let the input signal switch to a given logic value at time $t = 0$. Then, let time t + T_{pb} be the propagation time needed by the signal transition to reach the defect, i.e., the rising transition reaches the defective node, and thus mark the start of the *node voltage* switching from one logic value to another logic value.

In the same way, let T_{def} be the delay between Sout, and the transition in Sout-def. Also, let V_{def} represent the actual voltage $V(WL_{def})$ of the defective signal at the *clock period* $T_{period} = 1.5$ns for a given operation. Note that at T_{period} for a defect-free cell, the corresponding voltage V(WL) is GND, since a previous cycle has been completed, and a new cycle begins.

Thus, this analysis evaluates and considers V_{def}, since its impact will influence the next propagated signal of a subsequent clock period. Finally, the *slack time* T_{slack} is defined as the time at which the output signal after the propagation stays stable till the

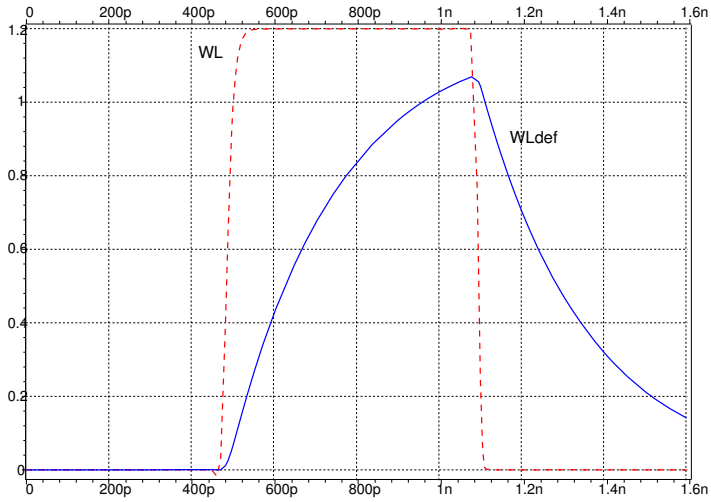


Figure 5.6: WL at R_{def} = 50kΩ with C_L = 4.5fF

end of the clock period for that operation. Thus, from the simulation, the total time t = 0 of one operation, to the start time of the next operation for a properly functional circuit must be less or equal to the clock period T_{period} = 1.5ns.

Now, as shown in the simulation results in Figure 5.6 the WL_{def} signal at time t + T_{pb} = 450ps, gradually rises from GND, but does not reach V_{DD}. A delay is clearly observed. Consequently, the effect of this delay is also seen at the end of the cycle, such that WL_{def} does not reach GND prior to, or at T_{period} = 1.5ns, at which the next period is expected to begin. Here, V_{def} only reached 0.2V, at time = 1.5ns instead of GND.

Likewise in Figure 5.7 a delay (T_{def}) is observed in Sout-def propagated as a result of the defective node. Because the total delay (T_{pb}, T_{def}, T_{slack}) caused by the defect does not exceed T_{period}, the memory still yields the correct output.

$$t + T_{pb} + T_{def} + T_{slack} < T_{period} \tag{5.5}$$

5.3.2 Detectable faulty behavior

Now, consider the simulation results in Figure 5.8, for R_{def} = 320kΩ and C_L = 4.5fF. As shown in the figure, WL_{def} signal is significantly slowed down. Consequently, V_{def} rises to about 0.3V. The implication is that a subsequent operation will be initiated at this voltage, which will influence V_{def} for that operation, and so on.

The output signals Sout and Sout-def are depicted in Figure 5.9. As can be seen, the size of the timing defect T_{def} is observably large, and so the rising transition of the

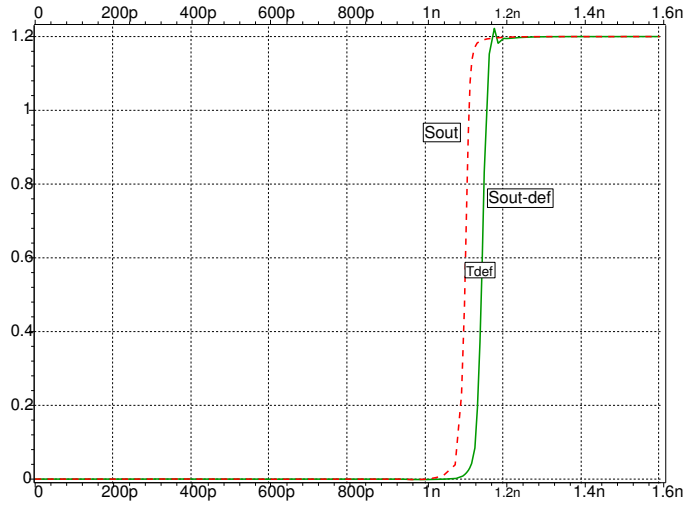


Figure 5.7: Output at R_{def} = 50kΩ with C_L = 4.5fF

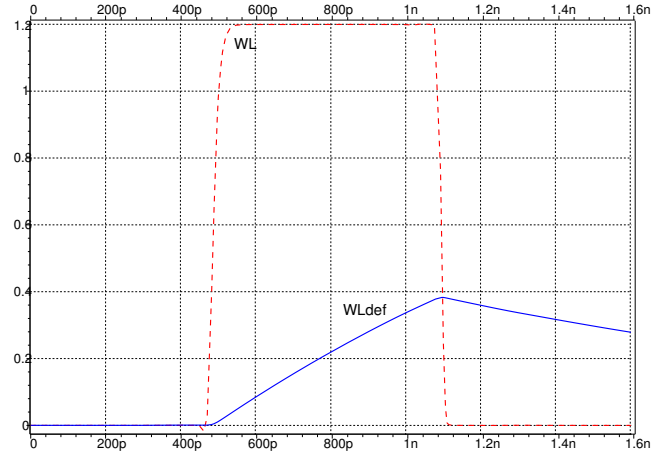


Figure 5.8: WL at R_{def} = 320kΩ with C_L = 4.5fF

signal occurs after T_{slack} begins. Therefore, an incorrect logic value is captured at the output, and the resistive open is detected.

In brief, the rising transition of the defective node, and consequently the additional delay time, T_{def} experienced by the cell is a function of:

- the faulty node's parasitic capacitance C_{node}
- the defect resistance R_{def}
- the initial voltage $V_0 = V_{def}$

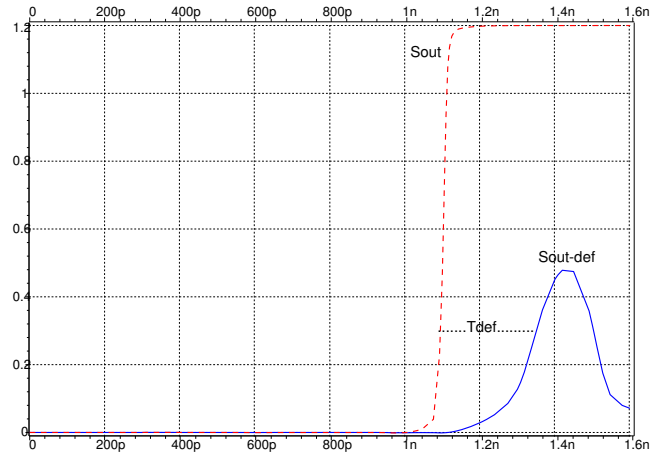


Figure 5.9: Output at R_{def} = 320kΩ with C_L = 4.5fF

$$T_{\text{def}} = f(C_{\text{node}}, R_{\text{def}}, V_0) \tag{5.6}$$

From this analysis, it is clear that an open with a relatively small parasitic node parameters, which does not cause a fault cannot be detected since the resulting delay, $t + T_{\text{pb}} + T_{\text{def}} + T_{\text{slack}} < T_{\text{period}}$, while the open with relatively large parasitic node parameters can be detected since $t + T_{\text{pb}} + T_{\text{def}} + T_{\text{slack}} > T_{\text{period}}$.

5.3.3 Detection using n-sequence

Here, another scenario is considered where a seq of m operations is applied to the circuit targeting a defect/fault, such that:

$$seq = \{op_1, \ldots, op_{\text{m-1}}, op_{\text{m}}\} \tag{5.7}$$

For this analysis, seq = $\{w0, r0, w0, r0, w1, r1\}$ is used, which has been specifically generated to detect the resistive open on the defective node, such that the open is not detected before $\{\text{op}_\text{m}\}$. This enables the separation of the test sequence into two stages, namely, the *sensitization* stage where, with the application of $\{\text{op}_1, \ldots, \text{op}_{\text{m-1}}\}$ a correct logic value is still observable at the output; and *the detection* stage where, with the application of $\{\text{op}_\text{m}\}$ an incorrect logic value is observed at the output. These stages and seq operations are listed in the first three columns of Table 5.1.

During the sensitization stage (first 5 operations), the faulty condition is not observable at the output, but the voltage on the defective node V_{def} may switch from GND

Table 5.1: Test seq and intermediate voltages

Stage	**Operation**	**seq op**	Intermediate voltages
			$V(WL_{def})$
Sensitization	op_1	w0	$V_1 = 0.38V$
Sensitization	op_2	r0	$V_2 = 0.44V$
Sensitization	op_3	w0	$V_3 = 0.47V$
Sensitization	op_4	r0	$V_4 = 0.46V$
Sensitization	op_5	w1	$V_5 = 0.48V$
Detection	op_6	r1	-

towards V_{DD} according to the activities induced by the operations.

Now, the initial state of the cell is 0. Within the first clock period ($T_{period} = 1.5$ns), at time ($t + T_{pb}$), the rising transition is propagated, and WL signal starts switching from GND to V_{DD}. However, due to the defective node parameters, WL_{def} is slowed down, and consequently the time required by the signal to reach V_{DD} is much higher. In addition, the time required by WL_{def} to reach GND at the clock period is also delayed. As a result of this delay, a voltage V_{def} develops and is observed at time T_{period}. It is observed that during this first operation, WL_{def} rises from $V_0 = V(t)$ to $V_1 = V(t + T_{period}) = 0.38$V.

At this point, a new operation interrupts the previous one despite the fact that WL_{def} is yet to reach GND. Subsequently for the second cycle, WL_{def} goes from V_1 to $V_2 = V(t + 2T_{period})$, and so on.

Therefore, $V(WL_{def})$ does not simply switch between GND and V_{DD}, but varies according to the operations, and goes through successive intermediate voltages V_0, V_1,... $V_{m\text{-}1}$ as listed in the fourth column of Table 5.1. In addition, Sout-def voltages for each seq operation indicates variations according to the input operations. Each corresponding intermediate voltage depends on R_{def}, C_{node} and the intermediate voltage V_{def} of the preceding operation, in a form of inter-dependence chain referred to as the *memory effect*.

At the detection stage, seq $\{op_6\}$ is applied to the cell and the transition induces delay, which is propagated to and observed at the output. The input transition is initiated by op_5 and detected by op_6. For the operation sequence (seq), the critical resistance (R_{cr}) is 248kΩ. We associate to this defect the Detection Interval DI [85] defined below as:

$$DI(seq, C_L) = [R_{cr}^{op_1,...op_6}, \infty] \tag{5.8}$$

In general, both the values of R_{def} as well as the parasitic capacitance C_L influence the timing behavior of the circuit, and therefore decide the eventual output at the sense amplifier from the memory cell. These two parameters create a space of possible

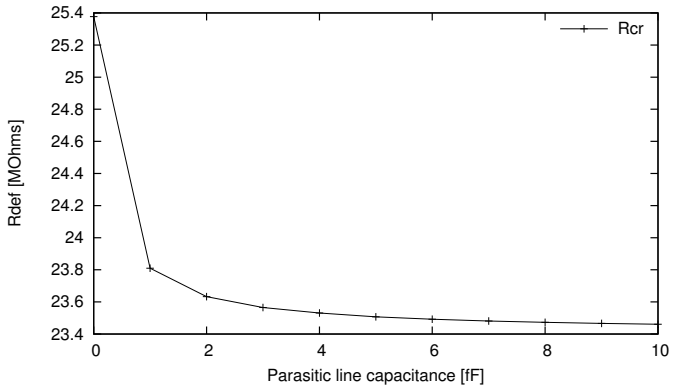


Figure 5.10: Plot of R_{cr}: R_{def} against C_L values

(C_L, R_{def}) values, as shown by the demarcating curve in Figure 5.10 for the plot of seq = $\{w1, r1\}$. In this plot, the $x-$axis denotes C_L, while the $y-$axis represents R_{def} values. The R_{cr} curve in the figure divides the (C_L, R_{def}) plane into two regions. R_{cr} is the resistive value (*critical resistance*) for a specific operation in the R_{def} range, below which a cell functions properly despite the presence of a defect (the *pass* region), and above which the cell fails to yield the correct output (the *fail* region).

Figure 5.10 shows that as C_L increases, the fail region expands, while the pass region decreases. This underscores the importance of C_L, and the need to account for it as an important component of the defective node.

5.4 Impact on static faults

In the previous section, the presence of parasitic node capacitance (C_n) in the defective node has been shown to exacerbate the faulty behavior in SRAMs [59]. It can also induce the dependence of a successive faulty node's voltage on the voltage of a previous operation; an effect that is known as the parasitic memory effect. In fact, resistive opens combined with parasitic capacitance on a defective node, can modify the timing behavior of the circuit, which can cause faults. Such modified behavior can result in different faults depending on the operating faulty node's voltage, which could be in the range GND $< V_n < V_{DD}$.

This section presents the analysis and evaluation the the impact of the parasitic memory effect on the detection of single-cell faults in SRAMs, and shows that the detection of single-cell faults is not determined by the value of the defect resistance alone, but is significantly influenced by the parasitic components of the defective node; something that is often not accounted for during memory testing. The section thus identifies the impact of parasitic node components on all open defects in the SRAM memory cell array, for varied node voltages ranging from GND to V_{DD}. In

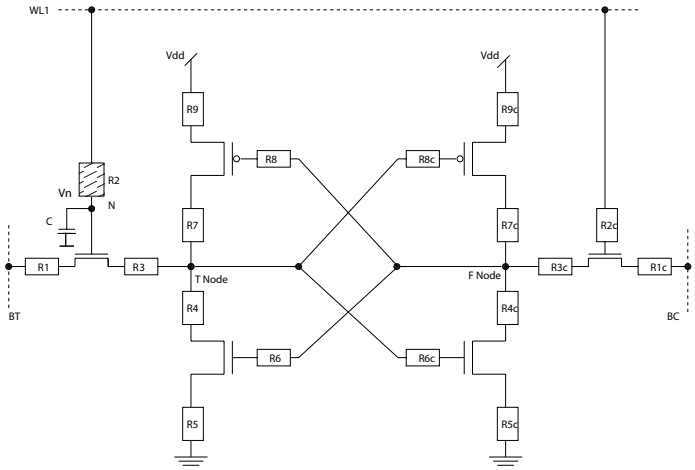


Figure 5.11: SRAM cell showing R_{def} and C_n

brief, the section presents the following.

- An analysis and evaluation of the impact of parasitic memory effect on single-cell static faults detection.
- A characterization of the impact of variation of the floating node voltages (from GND to V_{DD}) on the SRAM faulty behavior.

In this section, single-cell static faults are targeted. These are faults that are sensitized by at most one operation. These FFMs and their corresponding *fault primitives (FPs)* have been earlier discussed and presented in Section 3.3 and listed in Table 3.1.

5.4.1 Simulation parameters

Figure 5.11 shows an example of the model of an SRAM cell within the memory cell array used in this evaluation. The parasitic components of the defective node (N) that influence the fault space are the node voltage (V_n), the resistive open (R_{def}) and the parasitic line capacitance (C_n). In this model, for each defective node the parasitic components are taken into consideration. Each injected open creates a *floating node*, whose voltage varies between GND and V_{DD}, in this case 0V to 1.2V. A floating node is a memory node that is not properly controlled by a memory operation due to a defect, which leads to an improper voltage on the floating node at the end of the operation.

All open defects in the SRAM cell as depicted in Figure 5.11 are simulated and analyzed. The simulations are performed using R_{def} values in the range $0 < R_{def} < 10G\Omega$. Logarithmic incremental steps (for example, 1, 10, 100, 1000, etc,.) have been used while simulating values in the R_{def} range. Also, C_L within the range $0 <$

Table 5.2: Static faults for R1c F-node side using seq = $\{1r1\}$ and seq = $\{0w1\}$

V_n	1k	10k	100k	1M	10M	100M	1G	10G
0.0	–	–	TF_1	– TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
0.1	–	–	TF_1	– TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
0.2	–	–	TF_1	– TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
0.3	–	–	TF_1	– TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
0.4	–	–	TF_1	– TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
0.5	–	–	TF_1	– TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
0.6	–	–	TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
0.7	–	–	TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
0.8	–	–	TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
0.9	–	–	TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
1.0	–	–	TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
1.1	–	–	TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1
1.2	–	–	TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1	IRF_1 TF_1

$C_{\mathrm{L}} < 10$fF has been used. For each R_{def} value, since V_n is floating, the simulation is performed using 13 different possible values of V_n in the range $0.0V < V_{\mathrm{n}} < 1.2$V, where the range corresponds to GND to V_{DD} with incremental steps of 0.1V. After injecting a defect, analysis is done while performing six operation sequences: seq = $\{1r1$ (i.e., apply read 1 to the defective cell initialized to 1), $0r0$, $1w1$, $1w0$, $0w1$ and $0w0\}$. Note that at most one operation is applied at a time; therefore the analysis is static. Only static faults can be sensitized with such analysis. Irrespective of the symmetry that exists between the T and F-nodes in SRAM cells, defects on both sides can exhibit different behaviors, and therefore full simulations for all 18 defects have been performed and analyzed. The analysis is based on observing the differences in the electrical behavior at the output and the internal content of the cell through the true node.

Table 5.3: Static faults for R2c F-node side using seq = $\{1r1\}$ and seq = $\{0w1\}$

R2c	Defective node voltage (V_n) in Volts												
	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2
10G	IRF_1	IRF_1	IRF_1	–	–	–	–	–	–	–	–	–	–
	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	–	–	–	–	–	–
1G	IRF_1	IRF_1	IRF_1	–	–	–	–	–	–	–	–	–	–
	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	–	–	–	–	–	–
100M	IRF_1	IRF_1	IRF_1	–	–	–	–	–	–	–	–	–	–
	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	–	–	–	–	–	–
10M	IRF_1	IRF_1	IRF_1	–	–	–	–	–	–	–	–	–	–
	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	–	–	–	–	–	–
1M	IRF_1	IRF_1	IRF_1	–	–	–	–	–	–	–	–	–	–
	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	TF_1	–	–	–	–	–	–
100k	–	–	–	–	–	–	–	–	–	–	–	–	–
10k	–	–	–	–	–	–	–	–	–	–	–	–	–
1k	–	–	–	–	–	–	–	–	–	–	–	–	–

5.4.2 Analysis for defect R1c

This defect is located between the drain of the pass transistor and the complementary BL (BC) of the cell on the F-node side. The floating node is located between BC and R1c. Results for the analysis of R1c are listed in Table 5.2. In the table, the first row lists the simulated R_{def} values, while the first column lists the values of V_n used. The entries show the faults detected at the listed R_{def} at the corresponding V_n value, while the entry '-' indicates the absence of a fault.

Now, using seq = $\{1r1\}$ for defect R1c, our results show that for C_n = 4.5fF and V_n values $0.0V < V_n < 0.5$V the correct logic value is yielded at the output when R_{def} is in the range $1K\Omega < R_{def} < 1M\Omega$. However, beyond this R_{def} range, incorrect logic values are yielded at the output. Inspection of the BLs indicates a distortion such that the difference in potential between the true BL and the complementary BL is greatly reduced, thereby making it hard for the sense amplifier to read the correct value from the cell. Furthermore, an inspection of the true node shows that the cell contains the correct logic values for all simulated R_{def} values from $1K\Omega$ to $10G\Omega$. Thus, at $0.0V < V_n < 1.2$V and for $10M\Omega < R_{def} < 10G\Omega$ at C_n = 4.5fF the cell exhibits an Incorrect Read Fault, IRF_1 ($< 1r1/1/0 >$).

However, at an increased faulty node voltage of $0.6V < V_n < 1.2$V, we observe that IRF is detected for $1M\Omega < R_{def} < 10G\Omega$. This shows the impact of the floating node voltage on the defective node, and that R_{cr} of a specific defect can vary based on variation of the parasitic node components.

Now, using seq = $\{0r0\}$ for R1c, we observe that this operation passed such that the output and the content of the true node both recorded the correct logic 0 for all

values of V_n. This result is expected since for a $r0$, only true BL is discharged while complementary BL is expected to remain approximately at V_{DD}. Thus the parasitic components of the faulty R1c node will have no impact on $r0$ operation.

In the same way, using seq = $\{0w1\}$ in the presence of R1c, we evaluate the impact on the cell by observing the content of the true node. Results show that for all simulated V_n values in the range $0.0V < V_n < 1.2$V when $1K\Omega < R_{def} < 10K\Omega$ the true node shows that the cell contains the correct logic 1 value as expected. However, for all V_n values, when R_{def} is in the range $100K\Omega < R_{def} < 10G\Omega$, the true node shows that the cell contains an incorrect logic 0. Thus, at $0.0V < V_n < 1.2$V and for $100K\Omega < R_{def} < 10G\Omega$ the cell exhibits the Transition Fault TF_1 ($< 0w1/0/- >$).

Using seq = $\{1w1\}$, the performed operation passed and the true node shows that the cell contains the correct logic 1 value as expected.

Also, using the operation sequences seq = $\{0w0\}$ and seq = $\{1w0\}$, both operations passed irrespective of the values of the parasitic node components used. The reason is that for a $w0$ despite the initial content of the cell, the true BL is expected to be discharged to GND, while BC is expected to remain at V_{DD}. Since R1c is positioned on the non-discharged path, its presence does not significantly influence the content of the cell to necessitate a fail.

5.4.3 Analysis for defect position R2c

The defect R2c is located between the WL and the gate of the pass transistor on the F-node side. The floating node is located between the defect and WL. An open between WL and the gate of the pass transistor on F-node side will limit connectivity to the gate such that the pass transistor will not function properly. Results for this analysis are presented in Table 5.3.

Performing seq = $\{1r1\}$ in the presence of defect R2c, it is observed that for $C_n = 4.5$fF and V_n values in the range of $0.0V < V_n < 0.2$V, the correct logic value is yielded by the sense amplifier at the output when R_{def} is in the range $1K\Omega < R_{def} < 100K\Omega$. but, for $R_{def} = 1M\Omega$ and above for the same V_n values, incorrect logic values are recorded at the output. However, the content of the true node shows correct logic values for all simulated R_{def} values from $1K\Omega$ to $10G\Omega$. Thus, at $0.0V < V_n < 0.2$V and for $1M\Omega < R_{def} < 10G\Omega$ at $C_n = 4.5$fF the cell exhibits an Incorrect Read fault IRF_1 ($< 1r1/1/0 >$). Consequently, an inspection of the BLs indicated a distortion such that the difference in potential between the true BL and the complementary BL is greatly reduced for the values where the fails occurred, thereby making it hard for the sense amplifier to read the correct value from the cell.

Furthermore, at increased faulty node voltages of $0.3V < V_n < 1.2$V it is observed that no fail occurs. Again, this underscores the importance of taking into consideration the parasitic effects of the faulty node during fault detection.

Table 5.4: Static faults for defects on T-node side

Vn	Defects								
	R1	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**
0.0	–	–	–	RDF_0	RDF_0	RDF_0	–	–	–
0.1	–	–	–	RDF_0	RDF_0	RDF_0	–	–	–
0.2	–	–	–	RDF_0	RDF_0	RDF_0	–	–	–
0.3	–	–	–	RDF_0	RDF_0	RDF_0	–	–	–
0.4	–	–	–	RDF_0	RDF_0	RDF_0	–	–	–
0.5	–	–	–	RDF_0	RDF_0	RDF_0	–	–	–
0.6	–	–	–	RDF_0	RDF_0	$DRDF_1$	–	–	–
0.7	–	–	–	RDF_0	RDF_0	RDF_1	–	–	–
0.8	–	–	–	RDF_0	RDF_0	RDF_1	–	–	–
0.9	–	–	–	RDF_0	RDF_0	RDF_1	–	–	–
1.0	–	–	–	RDF_0	RDF_0	RDF_1	–	–	–
1.1	–	–	–	RDF_0	RDF_0	RDF_1	–	–	–
1.2	–	–	–	RDF_0	RDF_0	RDF_1	–	–	–

Using seq = $\{0r0\}$ in the presence of R2c, it is observed that this operation passes such that the sense amplifier output and the content of the true node both record correct logic 0 values for all simulated V_n values. The is because for a $r0$, only the true BL is discharged, while the complementary BL is expected to remain approximately at V_{DD}. Thus the parasitic components of the faulty node will have no significant impact on this operation.

In the same way, using seq = $\{0w1\}$, the impact on the faulty behavior of the cell by observing the content of the true node is evaluated. The results show that for V_n values in the range $0.0V < V_n < 0.6$V when $1K\Omega < R_{def} < 100K\Omega$, the true node shows that the cell contains the expected correct logic 1 value thereby indicating a successful write transition. But, when R_{def} is in the range $1M\Omega < R_{def} < 10G\Omega$, incorrect logic 0 is observed at the true node. Thus, at $0.0V < V_n < 0.6$V, for $1M\Omega < R_{def} < 10G\Omega$ the cell exhibits the Transition Fault TF_1 ($< 0w1/0/- >$). However, at $0.7V < V_n < 1.2$V for all simulated values of R_{def} at $1K\Omega < R_{def} < 10G\Omega$, the true node shows that the cell contains correct logic 1 and did not fail.

Now, using seq = $\{1w1\}$, the performed operation also passed successfully and the true node shows that the cell contains the correct logic 1 value as expected. Likewise, using the operation sequences seq = $\{0w0\}$ and seq = $\{1w0\}$, both operations passed irrespective of the value of the parasitic components used. The reason is that for a $w0$ despite the initial content of the cell, BT is expected to be discharged to GND, while BC is expected to remain at V_{DD}. Since R2c is positioned on the non-discharged path, its presence does not significantly influence the content of the cell to necessitate a fail.

Table 5.5: Static faults for defects on F-node side

V_n	$R1_c$	$R2_c$	$R3_c$	$R4_c$	$R5_c$	$R6_c$	$R7_c$	$R8_c$	$R9_c$
0.0	RDF_1 TF_1	RDF_1 TF_1	–	RDF_1	RDF_1	– RDF_1 TF_0	–	–	RDF_0
0.1	RDF_1 TF_1	RDF_1 TF_1	–	RDF_1	RDF_1	– RDF_1 TF_0	–	–	RDF_0
0.2	RDF_1 TF_1	RDF_1 TF_1	–	RDF_1	RDF_1	– RDF_1 TF_0	–	–	RDF_0
0.3	RDF_1 TF_1	– TF_1	–	RDF_1	RDF_1	– RDF_1 TF_0	–	–	RDF_0
0.4	RDF_1 TF_1	– TF_1	–	RDF_1	RDF_1	– – TF_0	–	–	RDF_0
0.5	RDF_1 TF_1	– TF_1	–	RDF_1	RDF_1	RDF_0 WDF_1 TF_0	–	–	RDF_0
0.6	RDF_1	–	RDF_1	RDF_1	RDF_1	RDF_0 WDF_1 TF_0	–	–	RDF_0
0.7	RDF_1 TF_1	–	RDF_1 TF_1	RDF_1	RDF_1	RDF_0 WDF_1 TF_0	–	–	–
0.8	RDF_1 TF_1	–	RDF_1 TF_1	RDF_1	RDF_1	RDF_0 WDF_1 TF_0	–	–	–
0.9	RDF_1 TF_1	–	RDF_1 TF_1	RDF_1	RDF_1	RDF_0 WDF_1 TF_0	–	–	–
1.0	RDF_1 TF_1	–	RDF_1 TF_1	RDF_1	RDF_1	RDF_0 WDF_1 TF_0	–	–	–
1.1	RDF_1 TF_1	–	RDF_1 TF_1	RDF_1	RDF_1	RDF_0 WDF_1 TF_0	–	–	–
1.2	RDF_1 TF_1	–	RDF_1 TF_1	RDF_1	RDF_1	RDF_0 WDF_1 TF_0	–	–	–

5.4.4 Analysis for the other defect positions

A summary for the rest of the results for the open defects are presented in Table 5.4 and 5.5. Table 5.4 lists the results for defects at the T-node side, while Table 5.5 lists the results for defects at the F-node side. In both tables, the first row indicates the defect considered, while the first column lists the defective node voltages simulated. For all operation sequences performed, only the detected faults are listed against the corresponding defective node voltage value at which the fault is detected. The entry '-' indicates the absence of a fault for the corresponding defect and/or defective node voltage listed.

As shown in the tables, some faults are only detected at specific V_n and R_{def} values and not throughout the whole range of the parasitic node components expressed. This clearly buttresses the fact that the faulty behavior is influenced by parasitic node components, and thus underscores the importance of taking these components into consideration during fault detection.

Chapter 6

Memory testing for parasitic fails

This chapter presents the approaches and techniques developed in this thesis, for parasitic fails in SRAM memory devices. It describes the conditions necessary and sufficient to test for, and detect such fails. These detection conditions provide insight and demonstrate the limitations of already existing industrial tests in the presence of parasitic effects. The first two sections of the chapter present discussions on testing for parasitic BL coupling. They present novel march tests that detect static faults in the presence of BL coupling. The third section presents a novel optimization technique developed in this work, which utilizes only limited coupling backgrounds to ensure proper detection of all targeted faults. The last section provides our testing approach for faults detection in the presence of parasitic memory effects, and presents a novel test that detects static faults in the presence of parasitic memory effect.

6.1 Data backgrounds for BL coupling

In order to ensure the detection of a given type of faulty behavior in the presence of BL coupling, a test needs to ensure the use of the most stressful coupling backgrounds. An important consequence for determining WCBs for each category of spot defects (opens, bridges and shorts) is that it provides insight for determining the detection conditions for failures occurring as a result of BL coupling effect. As provided by the analysis and results, certain coupling backgrounds exacerbate the failure mechanism, whereas some CBs rather support the faulty behavior. In summary, the necessary detection backgrounds for detecting opens, bridges and shorts in the presence of BL coupling are as follows:

- Opens - CB00 and CB11
- Shorts - CB00 and CB11

0	0	0
0	0	0
0	0	0

1	1	1
1	1	1
1	1	1

Figure 6.1: Solid-0, Solid-1 backgrounds

0	1	0
0	1	0
0	1	0

1	0	1
1	0	1
1	0	1

Figure 6.2: Column stripes backgrounds

- Bridges - CB00, CB01, CB10 and CB11. This is the case since certain bridges result in coupling faults with more than one cell involved.

During testing, to ensure the correct WCBs in the memory cells while the test is being applied, the memory cell array is initialized using specific *data backgrounds*. Now, data backgrounds refer to the specific arrangements of cells in the memory cell array based on the logic content of each cell. There are different data backgrounds. The solid, column stripes, row stripes and checkerboard backgrounds are depicted in Figures 6.1, 6.2, 6.3 and 6.4 respectively. As shown, a solid-0 or solid-1 data background has all the cells in the memory initialized to a logic 0 or logic 1, while a column stripe background has alternate 0s and 1s written on each column, and so on.

For example, if a test is designed to detect a logic 1 from a given cell, and both neighboring cells (on the left and on the right) on the same WL should each contain a logic 1, then the memory cell array can be initialized to a solid-1 data background, since the background will consistently ensure that each neighboring cell to the victim contains a logic 1 irrespective of the position of the cell being tested. The same is true when the test is supposed to detect a logic 0 from a given cell and both neighbors are required to be logic 0, then a solid-0 background could be used, and so on.

0	0	0
1	1	1
0	0	0

1	1	1
0	0	0
1	1	1

Figure 6.3: Row stripes backgrounds

0	1	0
1	0	1
0	1	0

1	0	1
0	1	0
1	0	1

Figure 6.4: Checkerboard backgrounds

6.2 Tests for BL coupling

This section presents the new testing methodologies developed in this research work, and memory tests generated for testing static faults in the presence of BL coupling effect. Section 6.2.1 discusses BL coupling tests for single-cell faults, and clearly demonstrates the limitation of existing tests in detecting such faults in the presence of BL coupling. Section 6.2.2 also presents March SSSc, an optimal test that detects all single-cell static faults in the presence and absence of BL coupling. Section 6.2.3 discusses testing for two-cell static faults, and shows the limitations of existing tests to properly detect them, while Section 6.2.4 presents March m-MSS, a test that detects all two-cell static faults in the presence and absence of BL coupling.

6.2.1 Single-cell faults: limitations of existing tests

For detecting single-cell static faults listed in Table 6.1, the WCB required is CB=xxx. Therefore, for initializing the memory cell array, a solid (i.e., /0000...or /1111...) and a row stripe (i.e., /0000...1111 or /1111...0000) data background could be used. This means that each cell should retain the same logic value at the beginning and at the end of the march element in order to ensure worst-case BL coupling.

Table 6.1: Single-cell static FFMs and their corresponding FPs

Fault	FP	Fault	FP
SF_0	$< 0/1/- >$	RDF_0	$< 0r0/1/1 >$
SF_1	$< 1/0/- >$	RDF_1	$< 1r1/0/0 >$
TF_0	$< 1w0/0/- >$	$DRDF_0$	$< 0r0/1/0 >$
TF_1	$< 0w1/1/- >$	$DRDF_1$	$< 1r1/0/1 >$
WDF_0	$< 0w0/1/- >$	IRF_0	$< 0r0/0/1 >$
WDF_1	$< 1w1/0/- >$	IRF_1	$< 1r1/1/0 >$

Table 6.2: Comparison of tests fault coverage for single-cell FFMs

FFM	Scan	March SSS	March SR	March MSS	March SSSc
SF_0	$+/-$	$+/-$	$+/+$	$+/-$	$+/+$
SF_1	$+/-$	$+/-$	$+/+$	$+/-$	$+/+$
TF_0	$-/-$	$+/-$	$+/+$	$+/-$	$+/+$
TF_1	$+/-$	$+/-$	$+/+$	$+/-$	$+/+$
WDF_0	$-/-$	$+/-$	$-/-$	$+/-$	$+/+$
WDF_1	$-/-$	$+/-$	$-/-$	$+/-$	$+/+$
RDF_0	$+/-$	$+/-$	$+/+$	$+/-$	$+/+$
RDF_1	$+/-$	$+/-$	$+/+$	$+/-$	$+/+$
$DRDF_0$	$-/-$	$+/-$	$+/+$	$+/-$	$+/+$
$DRDF_1$	$-/-$	$+/-$	$+/+$	$+/-$	$+/+$
IRF_0	$+/-$	$+/-$	$+/+$	$+/-$	$+/+$
IRF_1	$+/-$	$+/-$	$+/+$	$+/-$	$+/+$
FC	$7/0$	$12/0$	$10/10$	$12/0$	$12/12$
TL	$4n$	$9n$	$14n$	$22n$	$12n$

Table 6.2 compares the *fault coverage (FC)* of a number of memory tests for single-cell static FFMs. The first row lists the tests, while the first column lists the FFMs. Under each FFM, the notation x/y is used. x shows whether the fault is detected $(+)$ or not $(-)$ by the listed test in the absence of BL coupling, while y shows if the fault is detected $(+)$ or not $(-)$ in the presence of BL coupling. FC lists the fault coverage of the tests without/with BL coupling for the FFMs (12 in all). TL is the test length.

A test that can detect all single-cell static faults is March MSS [54]. However, in the presence of BL coupling, the test fails to detect these faults. This is because March MSS does not fulfill the necessary and sufficient requirement that each cell must retain the same logic value at the beginning and end of the march element.

Now, one test that satisfies the BL coupling detection requirement is March SR = $\{\Downarrow(w0);\ \Uparrow(r0, w1, r1, w0);\ \Uparrow(r0, r0);\ \Uparrow(w1);\ \Downarrow(r1, w0, r0, w1);\ \Downarrow(r1, r1)\}$ [43]. In this test, each accessed cell retains the same logic value at the beginning and at the end of each march element, thereby generating the background xx. However, this

test fails to detect the entire space of all targeted single-cell faults, thus falling short of extensive use in detecting faults in the presence of BL coupling.

Another well-known test that satisfies the BL coupling detection requirement is Scan = $\{\Updownarrow(w0); \Updownarrow(r0); \Updownarrow(w1); \Updownarrow(r1)\}$ [1]. But, in the same way as March SR, Scan can only detect a limited number of single-cell static faults.

6.2.2 March SSS and March SSSc

Here, we present a march test that detects all single-cell static faults in the *presence* and *absence* of BL coupling.

First, the test March SSS is proposed, which is an optimal test for detecting all single-cell static faults. However, as can be seen from this test, March SSS detects no faults when worst-case BL coupling is needed. This is true since each march element in the test inverts the value of the cell, thereby preventing the necessary xxx background pattern from taking place.

$$\begin{array}{lll} \text{March SSS} = \{ & \Updownarrow(w0); & \text{ME0} \\ & \Updownarrow(w1, w1, r1, r1); & \text{ME1} \\ & \Updownarrow(w0, w0, r0, r0)\} & \text{ME2} \end{array}$$

Therefore, we present an efficient test, March SSSc that detects all single-cell static faults both in the absence and in the presence of the influence of BL coupling. This is done by modifying March SSS to March SSSc in the following way:

$$\begin{array}{lll} \text{March SSSc} = \{ & \Updownarrow(w0); & \text{ME0} \\ & \Updownarrow(w1, w1, r1, r1, w0); & \text{ME1} \\ & \Updownarrow(w1); & \text{ME2} \\ & \Updownarrow(w0, w0, r0, r0, w1)\} & \text{ME3} \end{array}$$

March SSSc should be performed using the checkerboard data background (i.e., /0101.../1010...) or the column stripes (i.e., /0101.../0101...) data background. March SSSc has a time complexity of $12n$, which is $3n$ higher than March SSS due to the addition of 3 march operations. This test ensures that each read operation is performed on a given cell when the neighboring cells contain exactly the same value, by resetting the value of the tested cell at the end of ME1 and ME3. This ensures the worst-case conditions necessary for detecting single-cell static faults in the presence of BL coupling as follows.

March element ME0 initializes the memory to 0. ME1 starts by sensitizing TF_1 during the first $w1$ operation, then WDF_1 during the second $w1$ operation. These two faults are detected during the first $r1$ of ME1, which also sensitizes and detects SF_1, RDF_1 and IRF_1. The second $r1$ operation of ME1 sensitizes and detects $DRDF_1$.

Thereafter, the content of the cell is reset to 0, thereby maintaining the worst-case detecting condition for the faults.

In the same way, the complementary counterparts of these faults are sensitized and detected as follows. At ME2 the memory is initialized to 1. ME3 starts by sensitizing TF_0 during the first $w0$ operation, then WDF_0 during the second $w0$ operation. These two faults are detected during the first $r0$ of ME3, which also sensitizes and detects SF_0, RDF_0 and IRF_0. The second $r0$ operation of ME3 sensitizes and detects $DRDF_0$. Thereafter, the content of the cell is reset to 1, thereby maintaining the worst-case detecting condition for the faults.

6.2.3 Two-cell faults: limitations of existing tests

Table 6.3 and Table 6.4 compare the *fault coverage (FC)* of a number of memory tests for single-cell and two-cell static faults respectively. The tables demonstrate that the presence of BL coupling can reduce the fault coverage of memory tests. In Table 6.3, the first column lists the tests, while the subsequent columns list the single-cell static FFMs. In Table 6.4 the first column lists the FFMs and the subsequent columns list the tests. Under each FFM, the notation $x/y : a/b$ is used. x shows how many faults out of the y specified FPs are detected by the listed test in the *absence* of BL coupling, while a shows if all such faults out of b specified FPs are detected in the *presence* of BL coupling. In the last column of Table 6.3 FC is listed, while in the last row of Table 6.4 FC is listed. Again, the notation x/y is used. Here, x indicates the faults detected in the presence of BL coupling, while y denotes the total number of FFMs listed.

An industrial test that can effectively detect all single-cell and two-cell static faults is March MSS [54]. However, in the presence of BL coupling, this optimal memory test fails to detect all such faults. The reason is due to the absence of the necessary WCBs required to detect the faults during testing under the influence of BL coupling.

Another well-known industrial test, which is used for detecting unique faults that are not detected by other tests is the Galloping Pattern (GalPat) test [24]. This tests has long been used in the industry, but is vaguely understood. The test effectively detects most (but not all) single and two-cell faults in the presence of BL coupling. The reason why GalPat can detect certain unique faults in the presence of BL coupling is that it performs the test for each cell, using all possible neighborhood combinations (thereby generating the worst case coupling backgrounds xx, xy and yx). However, GalPat is known to be expensive both in test time and complexity. Therefore, these factors prompt the necessity to develop cheaper and more efficient tests for BL coupling.

Table 6.3: Comparison of fault coverage for single-cell FFMs

Tests	**SF**	**TF**	**WDF**	**RDF**	**DRDF**	**IRF**	FC
GalPat	$2/2:2/2$	$2/2:2/2$	$0/2:0/2$	$2/2:2/2$	$0/2:0/2$	$2/2:2/2$	8/12
MATS+	$2/2:0/2$	$1/2:0/2$	$0/2:0/2$	$2/2:0/2$	$0/2:0/2$	$2/2:0/2$	0/12
March SR	$2/2:0/2$	$2/2:0/2$	$0/2:0/2$	$2/2:0/2$	$2/2:0/2$	$2/2:0/2$	0/12
March C-	$2/2:0/2$	$2/2:0/2$	$0/2:0/2$	$2/2:0/2$	$0/2:0/2$	$2/2:0/2$	0/12
March B	$2/2:0/2$	$2/2:0/2$	$0/2:0/2$	$2/2:0/2$	$0/2:0/2$	$2/2:0/2$	0/12
PMOVI	$2/2:0/2$	$2/2:0/2$	$0/2:0/2$	$2/2:0/2$	$2/2:0/2$	$2/2:0/2$	0/12
March MSS	$2/2:0/2$	$2/2:0/2$	$2/2:0/2$	$2/2:0/2$	$2/2:0/2$	$2/2:0/2$	0/12
March m-MSS	$2/2:2/2$	$2/2:2/2$	$2/2:2/2$	$2/2:2/2$	$2/2:2/2$	$2/2:2/2$	12/12

Table 6.4: Comparison of tests fault coverage for two-cell FFMs

FFM	**Gal Pat**	**MATS+**	**March SR**	**March C-**	**March B**	**PMOVI**	**March MSS**	**March m-MSS**
$\mathbf{CF_{st}}$	$8/8:8/8$	$4/8:0/8$	$8/8:0/8$	$8/8:0/8$	$6/8:0/8$	$8/8:0/8$	$8/8:0/8$	$8/8:8/8$
$\mathbf{CF_{ds_{rx}}}$	$8/8:8/8$	$3/8:0/8$	$8/8:0/8$	$8/8:0/8$	$7/8:0/8$	$8/8:0/8$	$8/8:0/8$	$8/8:8/8$
$\mathbf{CF_{ds_{wx}}}$	$8/8:8/8$	$0/8:0/8$	$0/8:0/8$	$0/8:0/8$	$0/8:0/8$	$0/8:0/8$	$8/8:0/8$	$8/8:8/8$
$\mathbf{CF_{id}}$	$8/8:8/8$	$3/8:0/8$	$8/8:0/8$	$8/8:0/8$	$8/8:0/8$	$7/8:0/8$	$8/8:0/8$	$8/8:8/8$
$\mathbf{CF_{wd}}$	$0/8:0/8$	$0/8:0/8$	$0/8:0/8$	$0/8:0/8$	$0/8:0/8$	$0/8:0/8$	$8/8:0/8$	$8/8:8/8$
$\mathbf{CF_{rd}}$	$8/8:8/8$	$4/8:0/8$	$8/8:0/8$	$8/8:0/8$	$4/8:0/8$	$8/8:0/8$	$8/8:0/8$	$8/8:8/8$
$\mathbf{CF_{drd}}$	$0/8:0/8$	$0/8:0/8$	$8/8:0/8$	$0/8:0/8$	$0/8:0/8$	$0/8:0/8$	$8/8:0/8$	$8/8:8/8$
$\mathbf{CF_{ir}}$	$8/8:8/8$	$4/8:0/8$	$6/8:0/8$	$8/8:0/8$	$4/8:0/8$	$8/8:0/8$	$8/8:0/8$	$8/8:8/8$
$\mathbf{CF_{tr}}$	$8/8:8/8$	$2/8:0/8$	$8/8:0/8$	$8/8:0/8$	$4/8:0/8$	$8/8:0/8$	$8/8:0/8$	$8/8:8/8$
FC	$7/9$	$0/9$	$0/9$	$0/9$	$0/9$	$0/9$	$0/9$	$9/9$

0	0	1	1
0	0	1	1
0	0	1	1

1	1	0	0
1	1	0	0
1	1	0	0

Figure 6.5: Double-column stripes backgrounds

6.2.4 March m-MSS

In this thesis, March m-MSS is presented for detecting all single-cell and two-cell static faults in the presence of BL coupling. March m-MSS is derived by modifying March MSS [54]. March MSS is modified by implementing a number of different data backgrounds used with the test, such that all single-cell and two-cell static faults, in the absence or presence of BL coupling are detected. The use of these data backgrounds will ensure the generation of the WCBs xxx, yxx and xxy, which are necessary for detecting all FFMs. This modification is achieved by using the following data backgrounds in combination with the test:

1. Solid-0 data background (00000000...) in Figure 6.1
2. Solid-1 data background (11111111...) in Figure 6.1
3. Double-column stripes data background (00110011...) in Figure 6.5
4. Double-column stripes data background (11001100...) in Figure 6.5
5. Shifted double-column stripes data background (01100110...) in Figure 6.6
6. Shifted double-column stripes data background (10011001...) in Figure 6.6

$$
\begin{aligned}
\text{March m-MSS} = \{\ & \Updownarrow(w0); && \text{ME0}\\
& \Uparrow(r0, r0, w1, w1); && \text{ME1}\\
& \Uparrow(r1, r1, w0, w0); && \text{ME2}\\
& \Downarrow(r0, r0, w1, w1); && \text{ME3}\\
& \Downarrow(r1, r1, w0, w0); && \text{ME4}\\
& \Updownarrow(r0)\} && \text{ME5}
\end{aligned}
$$

Thus, March m-MSS detects all single-cell faults in the presence of BL coupling as follows. March element ME0 initializes the memory to 0. ME1 starts by sensitizing and detecting SF_0, RDF_0 and IRF_0, while the second $r0$ sensitizes and detects $DRDF_0$. ME1 also sensitizes TF_0 during the first $w1$ operation, and then WDF_1 during the second $w1$ operation. These two faults are detected during the first $r1$ of ME2, as well as SF_1, RDF_1 and IRF_1, while the second $r1$ detects $DRDF_1$ and so on. The complementary counterparts of the faults are sensitized and detected in the same

0	1	1	0
0	1	1	0
0	1	1	0

1	0	0	1
1	0	0	1
1	0	0	1

Figure 6.6: Shifted double-column stripes backgrounds

way by ME3, ME4 and ME5.

This test is performed using a different data background each time. This ensures that the worst case conditions necessary for detecting faults in the presence of BL coupling are applied. Note that the data background pattern can be repeated for any number of memory cells. The time complexity of March m-MSS is $108n$ because the test is performed six times using each of the six different data backgrounds.

6.3 Test optimization for BL coupling

Optimizing a memory test can significantly reduce the test complexity and run time, while retaining the quality of the test. This section introduces a systematic approach for developing optimized tests for memory faults in the presence of BL coupling. The section will show how to identify the required CBs for all static faults. Furthermore, March BLC is presented, an optimized memory test that detects all static faults in the presence of BL coupling.

In the presence of parasitic BL coupling, faults may be only detected by writing appropriate CBs in the neighboring cells of the victim. Understanding how specific initializations of a neighborhood of cells affects the sensing of a given faulty cell, and in addition, identifying such neighborhood data for each fault for testing purposes [57, 58], can increase fault/defect coverage and significantly reduce the test time required for detecting such faults.

In order to ensure proper detection each fault is tested for, using *all* possible CBs. However, using *all* possible CBs while testing for every fault consumes enormous test time and hence increases cost. But the test time complexity can be significantly reduced if the specifically required CBs (instead of all possible CBs) are identified for each fault model, and written in the neighboring cells in order to maximally stress such faults. Therefore, this section presents:

- A systematic approach demonstrating how to identify the limited required CBs necessary for the testing of each static fault, instead of using all possible CBs; something that significantly reduces the test time complexity.

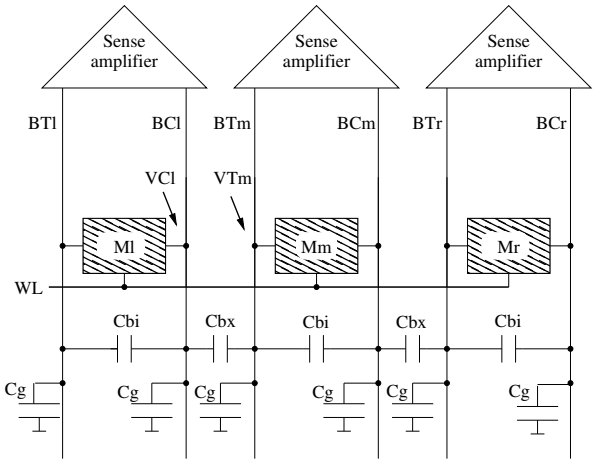


Figure 6.7: CB model for BL coupling

- An optimized test, March BLC, which detects all static faults in the presence of BL coupling.

6.3.1 Impact of BL coupling on static FFMs

This section presents the impact of BL coupling on both single-cell and two-cell static faults. Consider any three cells on the same WL: left (l), middle (m) and right (r) cells as shown in Figure 6.7. The middle cell (i.e., memory cell Mm) is the faulty cell under analysis. The BLs are connected to the *sense amplifier (SA)* to inspect the read output. As already explained in Chapter 4 using detailed validation with electrical Spice simulations by defect injection, in brief: the necessary detection backgrounds for detecting opens, bridges and shorts in the presence of BL coupling are:

- Opens - CB00 and CB11
- Shorts - CB00 and CB11
- Bridges - CB00, CB01, CB10 and CB11. This is the case since certain bridges result in coupling faults with more than one cell involved.

Impact on single-cell static FFMs

Single-cell static faults occur within the faulty cell, while BL coupling effect takes place in the neighborhood of the faulty cell. Thus, these two effects are independent. As a result, one can only maximally stress single-cell faults by ensuring the worst-case BL CBs (CB00 for a logic 0, and CB11 for a logic 1) for such faults.

Table 6.5: Detection conditions of two-cell static FFMs ($x, y \in \{0, 1\}$)

FFM	FP	Detection condition ($a < v$)	Detection condition ($a > v$)
1. $CF_{st_{0,0}}$	$<0;0/1/->$	$\Updownarrow(\ldots,0);\Updownarrow(\ldots,r0,\ldots)$	$\Updownarrow(\ldots,0);\Updownarrow(\ldots,r0,\ldots)$
2. $CF_{st_{0,1}}$	$<0;1/0/->$	$\Uparrow(\ldots,r1,\ldots,w0,\ldots)$	$\Downarrow(\ldots,r1,\ldots,w0,\ldots)$
3. $CF_{st_{1,0}}$	$<1;0/1/->$	$\Uparrow(\ldots,r0,\ldots,w1,\ldots)$	$\Downarrow(\ldots,r0,\ldots,w1,\ldots)$
4. $CF_{st_{1,1}}$	$<1;1/0/->$	$\Updownarrow(\ldots,1);\Updownarrow(\ldots,r1,\ldots)$	$\Updownarrow(\ldots,1);\Updownarrow(\ldots,r1,\ldots)$
5. CF_{ds}	$<0w0;0/1/->$	$\Updownarrow(\ldots,w0);\Updownarrow(r0,\ldots)$	$\Updownarrow(\ldots,w0);\Updownarrow(r0,\ldots)$
6. CF_{ds}	$<0w0;1/0/->$	$\Uparrow(r1,\ldots,w0,\ldots)$	$\Downarrow(r1,\ldots,w0,\ldots)$
7. CF_{ds}	$<1w1;0/1/->$	$\Uparrow(r0,\ldots,w1,\ldots)$	$\Downarrow(r0,\ldots,w1,\ldots)$
8. CF_{ds}	$<1w1;1/0/->$	$\Updownarrow(\ldots,w1);\Updownarrow(r1,\ldots)$	$\Updownarrow(\ldots,w1);\Updownarrow(r1,\ldots)$
9. CF_{ds}	$<0w1;0/1/->$	$\Uparrow(r0,\ldots,w1,\ldots)$	$\Downarrow(r0,\ldots,w1,\ldots)$
10. CF_{ds}	$<0w1;1/0/->$	$\Downarrow(\ldots,0,w1);\Updownarrow(r1,\ldots)$	$\Uparrow(\ldots,0,w1);\Updownarrow(r1,\ldots)$
11. CF_{ds}	$<1w0;0/1/->$	$\Downarrow(\ldots,1,w0);\Updownarrow(r0,\ldots)$	$\Uparrow(\ldots,1,w0);\Updownarrow(r0,\ldots)$
12. CF_{ds}	$<1w0;1/0/->$	$\Uparrow(r1,\ldots,w0,\ldots)$	$\Downarrow(r1,\ldots,w0,\ldots)$
13. CF_{ds}	$<0r0;0/1/->$	$\Uparrow(r0,\ldots)$	$\Downarrow(r0,\ldots)$
14. CF_{ds}	$<0r0;1/0/->$	$\Downarrow(r0,\ldots,1);\Updownarrow(r1,\ldots)$	$\Uparrow(r0,\ldots,1);\Updownarrow(r1,\ldots)$
15. CF_{ds}	$<1r1;0/1/->$	$\Downarrow(r1,\ldots,0);\Updownarrow(r0,\ldots)$	$\Uparrow(r1,\ldots,0);\Updownarrow(r0,\ldots)$
16. CF_{ds}	$<1r1;1/0/->$	$\Uparrow(r1,\ldots)$	$\Downarrow(r1,\ldots)$
17. CF_{wd}	$<0;0w0/1/->$	$\Uparrow(\ldots,w0);\Updownarrow(r0,\ldots)$	$\Downarrow(\ldots,w0);\Updownarrow(r0,\ldots)$
18. CF_{wd}	$<1;0w0/1/->$	$\Updownarrow(\ldots,w1);\Downarrow(\ldots,0,w0);$ $\Updownarrow(r0,\ldots)$	$\Updownarrow(\ldots,w1);\Uparrow(\ldots,0,w0);$ $\Updownarrow(r0,\ldots)$
19. CF_{wd}	$<0;1w1/0/->$	$\Updownarrow(\ldots,w0);\Downarrow(\ldots,1,w1);$ $\Updownarrow(r1,\ldots)$	$\Updownarrow(\ldots,w0);\Uparrow(\ldots,1,w1);$ $\Updownarrow(r1,\ldots)$
20. CF_{wd}	$<1;1w1/0/->$	$\Uparrow(\ldots,w1);\Updownarrow(r1,\ldots)$	$\Downarrow(\ldots,w1);\Updownarrow(r1,\ldots)$
21. CF_{tr}	$<0;0w1/0/->$	$\Updownarrow(\ldots,w0);\Downarrow(\ldots,w1);$ $\Updownarrow(r1,\ldots)$	$\Updownarrow(\ldots,w0);\Uparrow(\ldots,w1);$ $\Updownarrow(r1,\ldots)$
22. CF_{tr}	$<1;0w1/0/->$	$\Updownarrow(\ldots,0);\Downarrow(\ldots,w1);\Updownarrow(r1,\ldots)$	$\Updownarrow(\ldots,0);\Uparrow(\ldots,w1);\Updownarrow(r1,\ldots)$
23. CF_{tr}	$<0;1w0/1/->$	$\Updownarrow(\ldots,1);\Downarrow(\ldots,w0);\Updownarrow(r0,\ldots)$	$\Updownarrow(\ldots,1);\Uparrow(\ldots,w0);\Updownarrow(r0,\ldots)$
24. CF_{tr}	$<1;1w0/1/->$	$\Updownarrow(\ldots,w1);\Downarrow(\ldots,w0);$ $\Updownarrow(r0\ldots)$	$\Updownarrow(\ldots,w1);\Uparrow(\ldots,w0);$ $\Updownarrow(r0\ldots)$
25. CF_{drd}	$<0;0r0/1/0>$	$\Updownarrow(\ldots,w0);\Updownarrow(r0,r0,\ldots)$	$\Updownarrow(\ldots,w0);\Updownarrow(r0,r0,\ldots)$
26. CF_{drd}	$<1;0r0/1/0>$	$\Uparrow(r0,r0,\ldots,w1)$	$\Downarrow(r0,r0,\ldots,w1)$
27. CF_{drd}	$<0;1r1/0/1>$	$\Uparrow(r1,r1\ldots,w0)$	$\Downarrow(r1,r1,\ldots,w0)$
28. CF_{drd}	$<1;1r1/0/1>$	$\Updownarrow(\ldots,w1);\Updownarrow(r1,r1,\ldots)$	$\Updownarrow(\ldots,w1);\Updownarrow(r1,r1,\ldots)$
29. CF_{ir}	$<0;0r0/0/1>$	$\Updownarrow(\ldots,w0);\Updownarrow(r0,\ldots)$	$\Updownarrow(\ldots,w0);\Updownarrow(r0,\ldots)$
30. CF_{ir}	$<1;0r0/0/1>$	$\Uparrow(r0,\ldots,w1)$	$\Downarrow(r0,\ldots,w1)$
31. CF_{ir}	$<0;1r1/1/0>$	$\Uparrow(r1,\ldots,w0)$	$\Downarrow(r1,\ldots,w0)$
32. CF_{ir}	$<1;1r1/1/0>$	$\Updownarrow(\ldots,w1);\Updownarrow(r1,\ldots)$	$\Updownarrow(\ldots,w1);\Updownarrow(r1,\ldots)$
33. CF_{rd}	$<0;0r0/1/1>$	$\Updownarrow(\ldots,w0);\Updownarrow(r0,\ldots)$	$\Updownarrow(\ldots,w0);\Updownarrow(r0,\ldots)$
34. CF_{rd}	$<1;0r0/1/1>$	$\Uparrow(r0,\ldots,w1)$	$\Downarrow(r0,\ldots,w1)$
35. CF_{rd}	$<0;1r1/0/0>$	$\Uparrow(r1,\ldots,w0)$	$\Downarrow(r1,\ldots,w0)$
36. CF_{rd}	$<1;1r1/0/0>$	$\Updownarrow(\ldots,w1);\Updownarrow(r1,\ldots)$	$\Updownarrow(\ldots,w1);\Updownarrow(r1,\ldots)$

Impact on two-cell static FFMs

Two-cell static faults occur between two cells, and require a specific state or operation in an aggressor (a) as a necessary condition in order to sensitize a given fault. Table 6.5 shows all two-cell static faults, their corresponding FPs and detecting conditions needed to detect them. To effectively analyze the impact of BL coupling on two-cell FFMs, differentiation is made between two scenarios, namely, when:

a. Aggressor is not one of the neighboring cells

In this case, the aggressor is neither Ml nor Mr, but is at another location within the memory cell array. Thus, the influence of BL coupling is independent of the given fault. In order to properly test and detect such faults, they are stressed by ensuring the worst-case BL coupling backgrounds during testing.

b. Aggressor is one of the neighboring cells

In this case, the aggressor is either Ml or Mr. Thus, the influence of BL coupling depends on the fault model. Since the fault model requires a specific state or logic value (fault definition) in one of the neighboring cells in order to sensitize the faulty behavior, only the other coupling cell can be used to stress such faulty behavior. We distinguish between two types of these fault models:

1. *Homogeneous fault models*: In this case, the required logic value in the aggressor is the same as the expected logic value in the victim after sensitization. Therefore one only needs to identify the worst case coupling data in the other background cell, which has not been defined by the fault.

   For example, consider the Disturb Coupling Fault (CF_{ds}) = $<1w0; 0/1/->$. After the $1w0$ sensitizing operation, the content of the aggressor becomes a logic 0, while the content of the victim is also a logic 0. A subsequent $r0$ is required to detect the fault.

2. *Non-homogeneous fault models*: In this case, the required logic value in the aggressor is not the same as the expected logic value in the victim after sensitization. However, BL coupling requires the aggressor to have the same value as the victim during detection. Selecting the aggressor value depends on which of the two effects is dominant, which cannot be theoretically derived. Therefore, for such fault models, both backgrounds should be tested for, in order to ensure proper detection of the fault.

   For example, the Transition Coupling Fault (CF_{tr}) = $<0; 0w1/0/->$. Here, the expected content of the victim after the $0w1$ sensitizing operation is a logic 1, while the aggressor remains at a logic 0. To properly test and detect this fault, more than one CB is required in order to detect the effects of the faulty condition, as well as BL coupling. The reason is that whereas CB01 is required as a result of the FP's fault definition when the faulty condition is dominant,

Table 6.6: Required CBs for single-cell faults

Fault	FP	CBs	Fault	FP	CBs
SF_0	$< 0/1/- >$	00	RDF_0	$< 0r0/1/1 >$	00
SF_1	$< 1/0/- >$	11	RDF_1	$< 1r1/0/0 >$	11
TF_0	$< 1w0/0/- >$	11	$DRDF_0$	$< 0r0/1/0 >$	00
TF_1	$< 0w1/1/- >$	00	$DRDF_1$	$< 1r1/0/1 >$	11
WDF_0	$< 0w0/1/- >$	00	IRF_0	$< 0r0/0/1 >$	00
WDF_1	$< 1w1/0/- >$	11	IRF_1	$< 1r1/1/0 >$	11

CB11 is the WCB required for detecting a logic 1 when BL coupling impact is dominant. Thus CB01 and CB11 are used during testing.

6.3.2 CBs identification for static FFMs

This section shows how to identify the limited required CBs needed to properly detect a fault given a set of all possible CBs $\{00, 01, 10, 11\}$.

CBs identification for single-cell FPs

For single-cell faults, the faulty behavior and BL coupling effects are independent of each other since the fault occurs within the cell. Therefore, WCB due to BL coupling is applied in detecting the faults. For example, consider the single-cell Up-Transition Fault (TF_0) with FP = $<0w1/0/->$. Here, the memory is initialized to 0, and the fault is sensitized by a $w1$ operation. Thereafter, a $r1$ is required to detect the fault.

In this case, because the faulty behavior and BL coupling effects are independent, since both sensitizing and detecting occur on a single cell, the required CB is the WCB for $r1$, which is CB11. In brief, the required CBs to detect any static single-cell fault is the same as the WCB for the required read operation (i.e., CB00 for a $r0$ and CB11 for a $r1$) necessary to detect the fault. Table 6.6 lists all single-cell static faults, their FPs and the required limited CBs for their detection.

CBs Identification for two-cell FPs

In this section, all two-cell FPs are considered. Because two-cell FPs require specific state or value in the aggressor as a necessary condition for sensitizing a given fault, it is important to consider such specific conditions while identifying the required CBs for each given fault. Now, let the required CBs due to the fault definition necessary for sensitization be referred to as *Fault Coupling Background (FCB)*. Also, let the WCB due to BL coupling effect be *WCB*, thus coupling background,

$$CB = \{FCB, WCB\} \tag{6.1}$$

where {FCB, WCB} $\subset \{00, 01, 10, 11\}$. For example, the FCB for FP = $<0; 0w1/0/->$ is 01 and is obtained as follows. At first, the memory is initialized to a logic 0. $w1$ is performed on the victim in order to sensitize the fault, thus making the expected content of the victim a logic 1 after sensitization. Thereafter, a $r1$ is required to detect this fault. As defined by the FP, the content of the aggressor is 0 and the expected content of the victim is 1 after sensitization. Therefore, we choose the content of the other neighboring cell such that would ensure stressing a logic 1 in the victim. Therefore, FCB in this case is 01.

CBs identification for homogeneous two-cell FPs

For homogeneous two-cell fault models, the logic value in the aggressor (x) and the expected content of the victim (y) are the same after sensitization, such that: FP = $<\ldots x; \ldots y/\bar{y}/->$, where $x = y$.

For example, consider the disturb coupling fault, CF_{ds}, with FP = $<0; 0w0/1/->$. Here, the memory is initialized to 0. Then, this fault is sensitized by a non-transition $w0$ performed on the victim. After sensitization, the expected content of the victim remains 0, which is the same as the logic content of the aggressor, thus, $x = y$. Thereafter, a detecting $r0$ is performed to detect the fault.

However, the content of the aggressor is already determined by the FP's definition, i.e., a logic 0. Thus, we may only choose the content of the other coupling cell, such that the expected content of the victim is maximally stressed during the detecting $r0$. In this case, the other coupling cell should also contain a logic 0 because to stress a logic 0 in the victim, both neighboring cells must not contain a logic 1. Therefore, since the aggressor contains a logic 0, and the remaining coupling cell also contains logic 0, FCB is 00.

Table 6.7 presents a list of all two-cell homogeneous fault models. In this table, the first column lists the homogeneous FFMs, while the second column lists their corresponding FPs. Their FCBs when $a < v$ are listed in the third column, and FCBs when $a > v$ are listed in the fourth column. Finally, the fifth column shows the required CBs necessary to detect the faults. The table shows that homogeneous two-cell faults can be detected using only CB 11 or CB 00, instead of testing with all possible CBs.

CBs identification for non-homogeneous two-cell FPs

For non-homogeneous fault models, the logic value in the aggressor and in the victim cell after sensitization are not the same. That is, for non-homogeneous fault model,

Table 6.7: Required CBs for homogeneous FPs

FFM	FP = $<S_a; S_v/F/R>$	FCBs		Required CBs
		when $a < v$	when $a > v$	(FCB, WCB)
CF_{st}	$<0; 0/1/->$	00	00	00
CF_{st}	$<1; 1/0/->$	11	11	11
CF_{ds}	$<0w0; 0/1/->$	00	00	00
CF_{ds}	$<1w1; 1/0/->$	11	11	11
CF_{ds}	$<0w1; 1/0/->$	11	11	11
CF_{ds}	$<1w0; 0/1/->$	00	00	00
CF_{ds}	$<0r0; 0/1/->$	00	00	00
CF_{ds}	$<1r1; 1/0/->$	11	11	11
CF_{wd}	$<0; 0w0/1/->$	00	00	00
CF_{wd}	$<1; 1w1/0/->$	11	11	11
CF_{tr}	$<1; 0w1/0/->$	11	11	11
CF_{tr}	$<0; 1w0/1/->$	00	00	00
CF_{drd}	$<0; 0r0/1/0>$	00	00	00
CF_{drd}	$<1; 1r1/0/1>$	11	11	11
CF_{ir}	$<0; 0r0/0/1>$	00	00	00
CF_{ir}	$<1; 1r1/1/0>$	11	11	11
CF_{rd}	$<0; 0r0/1/1>$	00	00	00
CF_{rd}	$<1; 1r1/0/0>$	11	11	11

Table 6.8: Required CBs for non-homogeneous FPs

FFM	FP = $<S_a; S_v/F/R>$	FCBs		Required CBs (FCB, WCB)	
		when $a < v$	when $a > v$	when $a < v$	when $a > v$
CF_{st}	$<0; 1/0/->$	01	10	01, 11	10, 11
CF_{st}	$<1; 0/1/->$	10	01	10, 00	01, 00
CF_{ds}	$<0w0; 1/0/->$	01	10	01, 11	10, 11
CF_{ds}	$<1w1; 0/1/->$	10	01	10, 00	01, 00
CF_{ds}	$<0w1; 0/1/->$	10	01	10, 00	01, 00
CF_{ds}	$<1w0; 1/0/->$	01	10	01, 11	10, 11
CF_{ds}	$<0r0; 1/0/->$	01	10	01, 11	10, 11
CF_{ds}	$<1r1; 0/1/->$	10	01	10, 00	01, 00
CF_{wd}	$<1; 0w0/1/->$	10	01	10, 00	01, 00
CF_{wd}	$<0; 1w1/0/->$	01	10	01, 11	10, 11
CF_{tr}	$<0; 0w1/0/->$	01	10	01, 11	10, 11
CF_{tr}	$<1; 1w0/1/->$	10	01	10, 00	01, 00
CF_{drd}	$<1; 0r0/1/0>$	10	01	10, 00	01, 00
CF_{drd}	$<0; 1r1/0/1>$	01	10	01, 11	10, 11
CF_{ir}	$<1; 0r0/0/1>$	10	01	10, 00	01, 00
CF_{ir}	$<0; 1r1/1/0>$	01	10	01, 11	10, 11
CF_{rd}	$<1; 0r0/1/1>$	10	01	10, 00	01, 00
CF_{rd}	$<0; 1r1/0/0>$	01	10	01, 11	10, 11

FP = $< \ldots x; \ldots y/\bar{y}/- >$, where $x \neq y$. In addition, the coupling effect has its own requirement during the fault's detection.

For example, consider the Disturb Coupling Fault CF_{ds} with FP = $<1w0; 1/0/->$. The fault is sensitized by a transition $w0$ operation performed on an aggressor, and thereafter detected by a $r1$ performed on the victim. Thus, when $a < v$, FCB is 01.

The reason is that after the sensitizing $w0$, the content of the aggressor is expected to change from a logic 1 to 0, while the victim should still be 1. Since $x \neq y$, both FCB and WCB are required to properly test for, and detect the fault. FCB is necessary to ensure that the FP's non-homogeneous requirement is maintained, whereas WCB is required to invoke the worst-case stressful condition due to BL coupling. All non-homogeneous FFMs, their corresponding FPs, FCBs, and the required CBs for proper detection are listed in Table 6.8.

Table 6.7 and Table 6.8 show that only a limited number of CBs are required to properly test and detect static faults.

6.3.3 Optimized test for BL coupling: March BLC

March SSSc, an optimal test for detecting all single-cell static faults in the presence of BL coupling has been proposed in this work. This test ensures that each read operation is performed on a given cell when the neighboring cells contain exactly the same value. This is achieved by retaining the value of the tested cell at the end of specific march elements.

Also, March MSS a test that detects all single-cell and two-cell static faults in the presence of BL coupling has been proposed with a test complexity of $108n$. But, in order to detect all such faults, March m-MSS applies all possible CB combinations while testing (i.e., CBs 00, 01, 10 and 11), thereby consuming much test time. However, the required test time can be reduced by optimizing the test such that only limited or required CBs are applied.

$$
\begin{aligned}
\text{March BLC} = \{ \quad & \Updownarrow(w0); && \text{ME0} \\
& \Uparrow(r0, r0, w0, r0, w1, w1, r1); && \text{ME1} \\
& \Uparrow(r1, r1, w1, r1, w0, w1); && \text{ME2} \\
& \Uparrow(r1, r1, w0, w0, r0); && \text{ME3} \\
& \Uparrow(r0, r0, w0, r0, w1, w1, w0); && \text{ME4} \\
& \Downarrow(r0, r0, w0, w1, w1, r1); && \text{ME5} \\
& \Downarrow(r1, r1, w0, w1); && \text{ME6} \\
& \Downarrow(r1, r1, w0, w0, r0); && \text{ME7} \\
& \Downarrow(r0, r0, w1, w1, w0)\} && \text{ME8}
\end{aligned}
$$

Therefore, March BLC, an optimized test that detects all single-cell and two-cell static faults in the presence of BL coupling with limited CBs is presented in this sec-

Table 6.9: Comparison of tests fault coverage and test time

FFM	**March SSS**	**March SSSc**	**March MSS**	**March m-MSS**	**March BLC**
Single-cell					
SF	$+/-$	$+/+$	$+/-$	$+/+$	$+/+$
TF	$+/-$	$+/+$	$+/-$	$+/+$	$+/+$
WDF	$+/-$	$+/+$	$+/-$	$+/+$	$+/+$
RDF	$+/-$	$+/+$	$+/-$	$+/+$	$+/+$
DRDF	$+/-$	$+/+$	$+/-$	$+/+$	$+/+$
IRF	$+/-$	$+/+$	$+/-$	$+/+$	$+/+$
FC	$0:6$	$6:6$	$0:6$	$6:6$	$6:6$
Two-cells					
CF_{st}	$-/-$	$-/-$	$+/-$	$+/+$	$+/+$
CF_{ds}	$-/-$	$-/-$	$+/-$	$+/+$	$+/+$
CF_{wd}	$-/-$	$-/-$	$+/-$	$+/+$	$+/+$
CF_{tr}	$-/-$	$-/-$	$+/-$	$+/+$	$+/+$
CF_{drd}	$-/-$	$-/-$	$+/-$	$+/+$	$+/+$
CF_{ir}	$-/-$	$-/-$	$+/-$	$+/+$	$+/+$
CF_{rd}	$-/-$	$-/-$	$+/-$	$+/+$	$+/+$
FC	$0:7$	$0:7$	$0:7$	$7:7$	$7:7$
TL	$9n$	$12n$	$18n$	$108n$	$46n$

tion. March BLC has a test time complexity of $46n$. The test ensures that while detecting single-cell and two-cell homogeneous faults the required WCBs are applied, and while detecting two-cell non-homogeneous faults *only* the required WCBs and FCBs are applied. This approach reduces the number of CBs used. Consequently, Compared to March m-MSS ($108n$) that applies all possible CBs, the test complexity of March BLC is reduced by over 50% since only required CBs are applied.

Table 6.9 compares the fault coverage (FC) and test length (TL) of March BLC and some tests for single-cell and two-cell faults. The first row lists the tests, and subsequent rows list the FFMs. On each row, the notation x/y is used. x shows if the listed test detects the FFM ($+$), or not ($-$) in the *absence* of BL coupling, and y shows if it is detected, ($+$), or not ($-$) in the *presence* of BL coupling.

For FC, the notation $x : y$ is used. For each test in the single-cell part of the table, x indicates the number of faults out of *all* y single-cell FFMs that are detected by the listed test in the *presence* of BL coupling, while for two-cells part of the table, x shows the number of faults out of *all* y two-cell FFMs are detected in the *presence* of BL coupling. The table shows that March BLC gains the most in terms of FC and cost effective test time.

6.4 Testing for parasitic memory effect

Parasitic memory effect can occur due to the impact of parasitic node capacitances and faulty node voltages on the electrical behavior of SRAMs. This memory effect can cause detectable faults to become undetectable using existing industrial tests.

However, no fault detection mechanism nor memory tests have been developed that account for the faulty behavior induced by parasitic memory effects in SRAMs. Therefore, this work aims at determining the detection conditions for all single-cell FFMs, and developing a test that detects all such faults irrespective of the prevailing faulty node conditions (voltages and parasitic capacitance).

The work specifically identifies and describes the detection conditions for all single-cell static faults taking into consideration the impact of parasitic node components, thereby demonstrating the limitation of existing industrial tests. Finally, March SME is presented, which is a memory test that detects all targeted single-cell static faults in the presence of parasitic memory effect.

Thus this section presents the following contributions:

- Simulation, analysis and evaluation of memory effect on single-cell static faults detection.
- Identification and description of the detection conditions (i.e., for both initialization and fault sensitization for targeted static faults.
- March SME, which ensures the detection of targeted static faults in the presence and absence of parasitic memory effect.

Resistive opens combined with parasitic capacitance on a defective node, can modify the timing behavior of the circuit, which can cause faults. Such modified behavior can result in faults being manifested depending on the operating faulty voltage on the defective node. Such faults can only be detected using tests that expose the incorrect timing behavior due to these parasitic effects. This paper focuses on the generation of such tests.

Implications for static faults detection

As can be seen from Table 5.4 and Table 5.5, it is clear that certain single-cell static faults are only observed at specific V_n values and not throughout the whole range of the faulty node's voltage. The fact that the faulty behavior depends on the different parameters of the parasitic node components underscores the importance of taking into consideration the presence of parasitic memory effects on the faulty node during fault detection. Whereas the value of the defective resistance cannot be readily pre-determined or influenced by external operations, it is possible to influence and determine the value of the parasitic node voltage using memory operations, such that

the appropriate V_n values are initialized in the faulty nodes prior to sensitization and detection. In this way, one can ensure proper detection of each possible faults in the presence of parasitic memory effect.

Having observed these challenges posed by parasitic memory effect in detecting memory faults, it is clear that most industrial tests would not properly detect static faults in the presence of parasitic memory effect. Thus, it is important to specifically develop suitable detection conditions and therefore tests that would detect such faults.

6.4.1 Detection conditions for parasitic memory effect

Now, for the simulation an electrical Spice model of the SRAM cell has been used. In this model, the transistor parameters are based on the 65nm BSIM4 model card as described by the Predictive Technology Model [101]. The defect resistance R_{def} of $0 < R_{def} < 10\text{G}\Omega$ with logarithmic incremental steps of 1, 10, 100, 1000, etc., and parasitic node capacitance C_n of 4.5fF are used. Note that other close values of C_n will yield the same behavior shown in this thesis.

Each injected resistive open within the cell creates a floating node (V_n), whose voltage varies between GND and V_{DD}. A floating node is a memory node that is not properly controlled by a memory operation due to a defect, which leads to an improper voltage on the floating node at the end of the operation. The analysis require performing memory operations, while observing the impact on V_n. For all injected defects, the performed memory operations are write-0 ($w0$), read-0 ($r0$), write-1 ($w1$), and read-1 ($r1$). For each fault, the failing V_n range is simulated and determined.

Our aim is to determine what specific memory operations are needed to ensure that the required V_n range is induced prior to sensitization and detection of each fault.

Now, three important phases are considered during test development and generation, namely, the *initialization*, *sensitization* and *detection* phases. This work focuses on the initialization phase. The reason is that the sensitization and detection requirements for the faults remain the same, whereas the initialization conditions (appropriate V_n range) required for proper sensitization must be induced. For detection of a given type of faulty behavior in the presence of parasitic memory effect, a test must ensure that the required V_n range is initialized prior to sensitization and detection.

Note that on the one hand, different faults require different V_n ranges to be detected. For example, Table 5.5 shows that to detect an Incorrect Read Fault (IRF_1) caused as a result of defect R3c, V_n should be in the range $0.6\text{V} < V_n < 1.2\text{V}$, below which the fault could be undetectable. On the other hand, the same fault model induced when a specific defect has been injected could have different V_n requirement when induced by another injected defect. For example, the Incorrect Read Fault (IRF_1) observed when R2c is injected requires a lower V_n range, while the same Incorrect Read Fault (IRF_1) induced in the presence of R3c requires higher V_n range. These

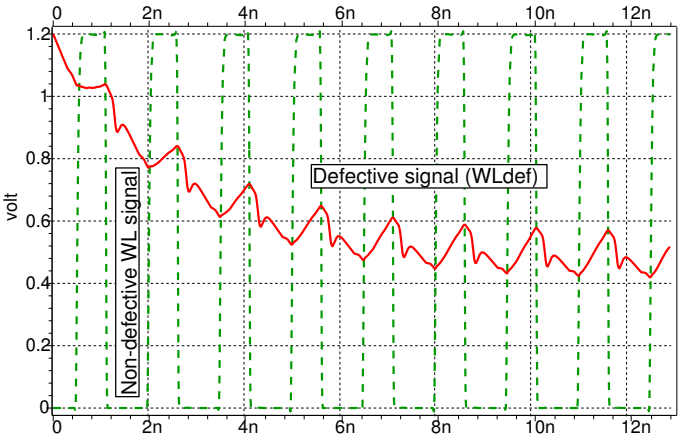


Figure 6.8: Impact on V_n when R2c is injected at V_n = 1.2V

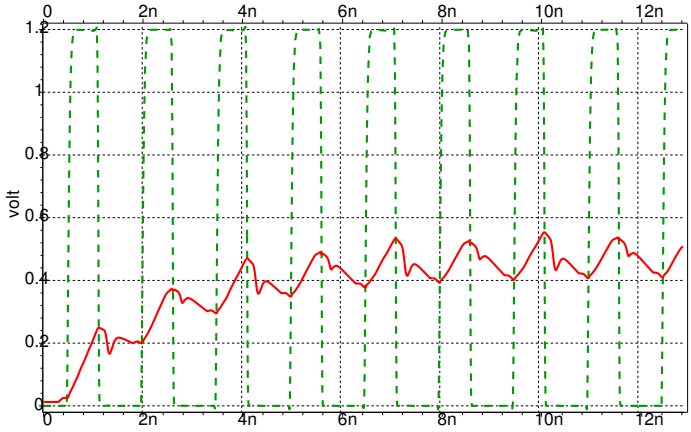


Figure 6.9: Impact on V_n when R2c is injected at V_n = 0.0V

unique requirements have been accounted for in our analysis.

Figure 6.8 shows simulation results for multiple $w0$ operation performed when defect R2c is injected for the detection of the Transition Fault (TF_1). Assuming a scenario, where $V_n = V_{DD} = 1.2$V, the figure shows that when the operation is performed, V_n significantly decreases from 1.2V to between 0.4V and 0.5V. The figure also shows that a single $w0$ will not be sufficient to appropriately initialize V_n to any value lower than 0.6V at which this fault can be detected.

Likewise, assuming a scenario, where V_n = GND = 0.0V, Figure 6.9 shows that the performed multiple $w0$ operations causes V_n to increase from 0.0V to about 0.5V at which the fault can be detected. In addition, a single $w0$ operation could be sufficient to initialize V_n of lower than 0.6V such that this fault can also be detected.

Furthermore, Figure 6.10 shows the simulation result for multiple $w0$ operations when defect R6 on the T-node side is injected. The figure shows that for multiple $w0$ operations, V_n increases from 0.0V to about 1.0V at which the Read Destructive Fault (RDF_1) is detected. It also shows that a single $w0$ operation will initiate V_n at a value less than the 0.5V value necessary for detecting Read Destructive Fault (IRF_0).

Table 6.10 presents a summary of the initializing operations that yield proper conditions for the sensitization and detection of the occurring fault models. In the table, the first column lists the defects, while the second column states the fault model. The third column lists the R_{def} range where the corresponding fault occurs, while the fourth column gives the required V_n range to enable sensitizing the fault. The entries in this column are *high* indicating that a V_n is needed between 0.6V to 1.2V, and *low* which implies values between 0.0V to 0.6V. The fifth column lists the initializing operations needed to achieve the required V_n values. An entry $'-'$ indicates that no operation is able to initialize V_n to the required voltage range needed to sensitize the fault. This means that we only need to test for those faults that can be initialized to the required V_n. The sixth column states the sensitizing operations for the fault model.

Table 6.10: Detection conditions for single-cell FFMs

Defect	**Fault**	**R_{def}** (KΩ)	**Required V_n**	**Initializing op**	**Sensitizing op**
R2c	TF_1	> 100	Low	$w0, w1, r0, r1$	$0w1$
	IRF_1	> 1000	Low	—	$1r1$
R3c	IRF_1	> 1000	High	$w0, r1, r0$	$1r1$
R6c	WDF_0	> 100	High	$w0, w1, r1$	$1w1$
	RDF_0	> 1000	High	$w0, w1, r1$	$0r0$
	RDF_1	> 1000	Low	—	$1r1$
R9c	RDF_0	> 1000	Low	—	$0r0$
R6	RDF_0	> 1000	Low	—	$0r0$
	RDF_1	> 100	High	$w0$	$1r1$

6.4.2 March SME

Several tests exist that are generated to detect all single-cell static faults, but from the analysis in this thesis, several such tests will not detect these faults in the presence of parasitic memory effect. For example, March SSS shown below is an optimal test for detecting all single-cell static faults. March SSS detects these faults as follows.

March element ME0 initializes the memory to 0. ME1 starts by sensitizing TF_1 during the first $w1$ operation, then WDF_1 during the second $w1$ operation. These two faults are detected during the first $r1$ of ME1, which also sensitizes and detects

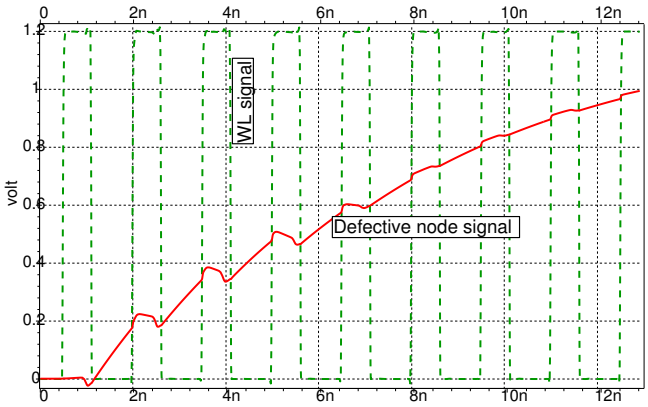


Figure 6.10: Impact on V_n when R6 is injected

SF_1, RDF_1 and IRF_1. Finally, the second $r1$ operation of ME1 sensitizes and detects $DRDF_1$. The complementary counterparts of these faults are sensitized and detected in the same way by ME2.

However, March SSS will not detect all faults in the presence of parasitic memory effect. This is true since all march elements in the test do not ensure the required initialization to sensitize and detect all faults.

$$\begin{array}{lll} \text{March SSS} = \{ & \Updownarrow(w0); & \text{ME0} \\ & \Updownarrow(w1, w1, r1, r1); & \text{ME1} \\ & \Updownarrow(w0, w0, r0, r0)\} & \text{ME2} \end{array}$$

Now, we present March SME, a test that detects single-cell static faults in the absence of parasitic memory effect, and the faults shown in Table 6.10 in the presence of parasitic memory effect. March SME is presented below.

$$\begin{array}{lll} \text{March SME} = \{ & \Updownarrow(w0, (r0)^i); & \text{ME0} \\ & \Updownarrow(w1, w1); & \text{ME1} \\ & \Updownarrow(r1)^i; & \text{ME2} \\ & \Updownarrow(w0)^i; & \text{ME3} \\ & \Updownarrow(r0, r0); & \text{ME4} \\ & \Updownarrow(w1, r1)\} & \text{ME5} \end{array}$$

In March SME, $(op)^i$ represents the number of times (i) that an initialization operation (op) is performed. The test has a time complexity of $7n$+$3n^i$. This test ensures that the required operations are performed on a given cell that would yield exactly the proper range of initializing voltage. It ensures the detection of single-cell static faults both in the presence and absence of parasitic memory effect in the following way.

1. *Detection in the absence of parasitic memory effect*

In march element ME0 the entire memory is initialized to 0. ME1 starts by sensitizing TF_1 during the first $w1$ operation, then WDF_1 during the second $w1$ operation. These two faults are detected during the first $r1$ of ME2, which also sensitizes and detects SF_1, RDF_1 and IRF_1, as well as $DRDF_1$ in the subsequent read. In ME3, TF_0 is sentistized during the first $w0$ operation, and WDF_0 during the second $w0$. These two faults are detected during the first $r0$ of ME4, which also sensitizes and detects SF_0, RDF_0 and IRF_0, as well as $DRDF_0$ in the second $r0$.

2. *Detection in the presence of parasitic memory effect*

March element ME0 initializes the entire memory to 0, while a subsequent $(r0)^i$ initializes the required low V_n value for the fault TF_1 associated with the defect R2c shown in Table 6.10. ME1 starts by sensitizing TF_1 during the first $w1$ operation, which is detected during the first $r1$ of ME2. In addition, $(r1)^i$ of ME2 also initializes the required high V_n and ensures the sensitization and detection of IRF_1 associated with R3c. Subsequently in ME3, $(w0)^i$ ensures the initialization of the required V_n value and sensitization of WDF_0 associated with R6c, which is then detected by the first $r0$ in ME4. RDF_0 associated with R6c is also initialized in ME3, then sensitized and detected in ME4. Finally, ME3 also initializes the required V_n for sensitizing RDF_1 associated with R6, which is sensitized and detected in ME5.

Chapter 7

Conclusions and recommendations

In this thesis, the analysis of parasitic fails, fault modeling, memory test development and optimization for static faults in SRAMs have been presented. The first part of the analysis of parasitic fails investigated the impact of parasitic bit line coupling effect on the faulty behavior of SRAM memory cells. The defects analyzed include resistive opens, bridges and short defects. The analysis modeled the failure mechanisms and investigated the impact on the faulty behavior on single-cell and two-cell static faults, and determined the detection conditions of these static faults in the presence of parasitic bit line coupling. The second part of the analysis of parasitic fails investigated the impact of parasitic memory effect on the faulty behavior of memory cells in SRAMs. The modeling of the parasitic node component was presented, and the impact of the parasitic memory effect on the detection of static faults, and the necessary and sufficient detection conditions of these faults in the presence of parasitic memory effect were addressed. The results have been verified using circuit simulations. Thereafter, new systematic approaches and memory test generation techniques have been developed for generating new tests that detects static faults in the presence of the evaluated parasitic fails.

This chapter summarizes the overall investigation and results presented in this thesis. In Section 7.1, the key issues discussed in each chapter are summarized. Section 7.2 presents specific contributions to the frontiers of knowledge by this research, and thereafter Section 7.3 discusses recommendations for future research in this field.

7.1 Summary of thesis chapters

Chapter 1, introduction, presented an introduction to semiconductor memories. The chapter described semiconductor memory technology and discussed the two broad classes of memory, which are the random access memory (RAM) and the read only memory (ROM). For RAM, it discussed both the dynamic random access mem-

ory (DRAM) and the static access memory (SRAM), which is the focus of this thesis. The chapter also discussed the motivation for memory testing, challenges in testing memories and provided insight into future emerging memory technologies. Finally, it listed the main contributions of the research work, and gave a general outline of the rest of the discussions in the thesis.

Chapter 2, modeling memory devices, described semiconductor memory models and their usefulness towards detection and localization of faults. The models discussed were geometrical models, which provide layout implementation of the device; logical models that aim at localizing faults in the logic gates; electrical models, which give information on the electrical components and internal structures of a given device; functional models, which are derived from the functional specifications and partly the internal structures of the device; and behavioral models that represent a device based on its specification, with little or no information on the internal structures. The chapter also presented discussions on DRAM and SRAM models and their characteristics.

Chapter 3, functional fault modeling approaches, discussed the functional fault models and their corresponding fault primitives. It introduced the classification of fault primitives based on the number of cells involved or the number of operations performed. Furthermore, the chapter presented static fault models and explains single-cell, two-cell and multiple cell static faults.

Chapter 4, parasitic BL coupling effect, introduced parasitic fails focusing on parasitic BL coupling effect in SRAMs; something that can make detectable faults become undetectable. This chapter presented a modeling of the failure mechanism and established the impact parasitic BL coupling effect on the SRAM faulty behavior. The chapter reported the evaluation of the impact of this effect for resistive opens, bridges and shorts using detailed SPICE circuit simulations. The results showed that different data combinations referred to as coupling backgrounds (CBs), i.e., CB 00, CB 01, CB 10 and CB 11 when written in the two adjacent neighbors of a victim, can stress the SRAM faulty behavior in different ways. From the evaluation, the worst-case coupling backgrounds (WCBs) were derived, which stressed the faulty behavior. Thereafter, from these WCBs the necessary and sufficient detection conditions for detecting faults in the presence of parasitic BL coupling were derived.

Chapter 5, parasitic memory effect, introduced, analyzed and evaluated the presence of parasitic memory effect in SRAMs. Analysis of the scenario where a defective node in the memory cell array is characterized by both a resistive component and a parasitic capacitive component was evaluated. The modeling of the failure mechanism and the impact of the parasitic node capacitance were also analyzed and

described. It provided analysis of the faulty behavior of the SRAM in the presence of this effect, and explained why certain faults are undetectable (or detectable) in the presence of parasitic memory effect. Furthermore, the analysis in this chapter included the impact of the parasitic memory effect on the detection of static faults. Using detailed circuit simulations, the chapter lists the static faults detected (undetected) in the presence of parasitic memory effect. Results from this analysis underscores the importance of accounting for the impact of floating node voltages on static faults. Thereafter, the chapter presents the detection conditions required to properly detect faults in the presence of parasitic memory effect; something that must be considered while generating efficient memory tests.

Chapter 6, memory testing for parasitic fails, established the fact that new systematic techniques and testing approaches are required in order to properly detect faults in the presence of parasitic fails in SRAMs. In this chapter, the limitations of existing memory tests in detecting faults in the presence of parasitic fails were shown. For detecting single-cell static faults, the chapter presented March SSS, an optimal memory test that detects all single-cell static faults in the absence of BL coupling, and demonstrated how this test fails to detect the same faults in the presence of BL coupling. Therefore, March SSSc was introduced, an efficient test that accounted for the required detection conditions, and detects all single-cell static faults in the presence of BL coupling. Likewise, for two-cell static faults, March m-SSS was presented. This test is a modification of March MSS, a well-known industrial test for detecting all single-cell and two-cell faults, but which fails to detect all such faults in the presence of BL coupling. Because March m-MSS required the use of all possible CBs in order to ensure proper fault detection, the test time complexity was high, which invariably affect test cost. Therefore, a novel systematic approach for optimizing memory tests was developed. Thus, the chapter presented March BLC, an optimized memory test that detects all single-cell and two-cell static faults in the presence of BL coupling, with over 50% reduction in test time complexity with respect to March m-MSS.

Finally, for detecting static faults in the presence of parasitic memory effect, the chapter presented March SME, a test that takes variations in parasitic node voltages into consideration, while detecting all single-cell static faults in the presence of parasitic memory effect.

7.2 Specific thesis contributions

The thesis presented the analysis, modeling, simulation, evaluation and validation of the impact of parasitic component on the faulty behavior of memory devices. It has also presented new test methodologies and developed techniques for the generation of appropriate memory tests to detect memory fails in the presence of parasitic com-

ponents. The specific contributions of this thesis to scientific knowledge can then be summarized as listed below:

1. A theoretical framework that modeled BL parasitic capacitance, and the effect of capacitive BL coupling on the faulty behavior of the memory cell array. This behavior have been validated theoretically and through electrical SPICE simulations.
2. This thesis analyzed, simulated and evaluated the impact of parasitic BL coupling and neighborhood coupling data backgrounds on the faulty behavior of SRAMs. It validated this analysis through defect injection and circuit simulation of all possible spot defects in the memory cell array.
3. The thesis investigated and established the worst case coupling backgrounds required to induce worst case coupling effects in deep sub-micron devices. It presented the conditions necessary to ensure proper detection of memory faults, while taking BL coupling into consideration.
4. This thesis has clearly demonstrated the inadequacies and limitations of several well-known industrial tests in detecting memory faults in the presence of BL coupling effect.
5. The thesis has presented March SSS, March SSSc and March m-MSS, which are tests that target and detect all single and two-cell static faults in the presence and absence of BL coupling for any possible spot defect.
6. This thesis introduced a systematic approach for developing optimized tests for memory faults in the presence of BL coupling. It showed how to identify the required coupling backgrounds for all static memory faults, and presented March BLC, an optimized test that detects all static memory faults in the presence of BL coupling.
7. The thesis investigated and analyzed the impact of parasitic node capacitance on defective resistive nodes, referred to as parasitic memory effect. It demonstrated that the faulty behavior in the memory is exacerbated in the presence of parasitic node capacitance; something that reduces the fault coverage of current memory tests, and increases the DPM rate.
8. This thesis analyzed, evaluated and characterized parasitic memory effect, and the variation of the floating node voltages on the faulty behavior of the memory. It demonstrated that the detection of memory faults is not determined by the value of the defect resistance alone, but is significantly influenced by the parasitic components of the defective node; something that is often not accounted for during memory testing. It presented the detection conditions for faults in the presence of parasitic memory effect, and finally presented March SME, which detects all targeted faults in the presence of parasitic memory effect.

7.3 Recommendations for future research

This thesis has analyzed and evaluated parasitic fails caused by spot defects in deep sub-micron memory devices. It has presented new systematic approaches and test optimization techniques required to develop efficient memory tests that can detect such fails. New memory tests have also been proposed, which can detect static faults in the presence of parasitic fails. In view of the research work presented in this thesis, recommended areas for further investigation include:

- Dynamic faults: Based on the number of successive operations performed on a cell to sensitize a fault, memory faults can be divided into static and dynamic faults. Whereas for static faults at most one operation is required for fault sensitization, for dynamic faults multiple operations are required. The analysis for BL coupling effect focused so far on static faults. In addition to the work already completed for static faults, the analysis and test development could be extended to evaluate dynamic faults, determine the necessary and sufficient conditions for the detection of dynamic faults, and further develop efficient memory tests to detect these faults.
- Evaluation of the impact of BL coupling effect on faulty behavior using write operations: In the analysis presented in this thesis for BL coupling effect in SRAMs, read operations ($r0$ and $r1$) were considered. The reason is that the read circuitry is more complex than the write, and thus presents more vulnerability to fails. However, for reasons of completeness, we need to apply the analysis and evaluation using write operations.
- Impact of parasitic memory effect on two-cell static faults: The analysis in this thesis on the parasitic memory effect in SRAMs had focused on evaluating the impact of this effect on single-cell static faults. Detection conditions have been determined for the single-cell fault space and a memory test has been generated to detect such faults in the presence of parasitic memory effect. However, this analysis could be extended to evaluate the impact on two-cell faults, deriving the essential detection conditions, and thus, generating efficient memory tests that can detect two-cell static faults in the presence of parasitic memory effect.

Bibliography

[1] M.S. Abadir and J.K. Reghbati. Functional Testing of Semi-conductor Random Access Memories. *ACM Computing Surveys*, 15(3):175–198, 1983.

[2] A. Abramovici, M. Breuer, and A. Friedman. Digital System Testing and Testable Design. *IEEE press, New York*, 1990.

[3] R.D. Adams and E.S. Cooley. Analysis of a Deceptive Destructive Read Memory Fault Model and Recommended Testing. *In Proceedings of IEEE North Atlantic Test Workshop (NATW)*, 1996.

[4] Z. Al-Ars. *DRAM Fault Analysis and Test Generation*. A Doctoral Dissertation., 2005.

[5] Z. Al-Ars and S. Hamdioui. Automatic Analysis of Memory Faulty Behavior in Defective Memories. *In Proceedings of IEEE International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS)*, pages 41–46, 2007.

[6] Z. Al-Ars and S. Hamdioui. Evaluation of SRAM Faulty Behavior Under Bit Line Coupling. *In Proceedings of IEEE International Design and Test Workshop (IDT)*, 2008.

[7] Z. Al-Ars, S. Hamdioui, G. Gaydadjiev, and S. Vassiliadis. Test Set Development for Cache Memory in Modern Microprocessors. *In Proceedings of IEEE Transaction on VLSI Systems*, 16(6):725–732, 2008.

[8] Z. Al-Ars, S. Hamdioui, G.N. Gaydadjiev, and S. Vassiliadis. Test Development for Cache Memory in Modern microprocessor. *IEEE Transaction on Very Large Scale Integration Systems (TVLSI)*, 16(6):725–732, 2008.

[9] Z. Al-Ars, S. Hamdioui, G.N. Gaydadjiev, and S. Vassiliadis. Test Set development for Cache Memory in Modern Microprocessors. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 16(6):725–732, 2008.

[10] Z. Al-Ars, S. Hamdioui, and A.J. van de Goor. Effects of Bit Line Coupling on

the Faulty Behavior of DRAMs. *In Proceedings of IEEE VLSI Test Symposium (VTS)*, pages 117–122, 2004.

[11] Z. Al-Ars, S. Hamdioui, A.J. van de Goor, and S. Al-Harbi. Influence of Bit Line Coupling and Twisting on the Faulty Behavior of DRAMs. *IEEE Transactions on Computer Aided Design*, pages 2989–2996, 2006.

[12] Z. Al-Ars and A.J. van de Goor. Static and Dynamic Behavior of Memory Cell Array Opens and Shorts in Embedded DRAMs. *In Proceedings of the Design Automation and Test in Europe (DATE)*, pages 496–503, 2001.

[13] Z. Al-Ars and A.J. van de Goor. Static and Dynamic Behavior of Memory Cell Array Spot Defects in Embedded DRAMs. *IEEE Transactions on Computers*, pages 293–309, 2003.

[14] R. E. Aly, M. A. Elgamel, and M. A. Bayoumi. Dual Sense Ampified Bit Lines (DSABL) Architecture for Low-Power SRAM Design. *In proceedings of ISCAS*, pages 1650–1653, 2005.

[15] E. Barke. Line-to-Ground Capacitance Calculation for VLSI: A Comparison. *IEEE Transactions on Computer-Aided Design*, 7(2):295–298, 1988.

[16] 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. *In Proceedings of International Conference on Dependable Systems and Networks Workshops*, pages 95–100, 2010.

[17] A. Benso, A. Bosio, S. Di Carlo, G. Di Natale, and P. Prinetto. A Unique March Test Algorithm for the Wide Spread of Realistic Memory Faults in SRAMs. *IEEE Design and Diagnostics of Electronic Circuits and systems (DDECS)*, pages 155–156, 2006.

[18] A. Benso, A. Bosio, S. Di Carlo, G. Di Natale, and P. Prinetto. March Test Generation Revealed. *IEEE Transactions on Computers*, 57(12):1704–1713, 2008.

[19] A. Benso and P. Prinetto. Fault Injection Techniques and Tools for Embedded Systems Reliability Evaluation. *Kluwer Academic Publishers*, 2003.

[20] A. Bhavnagarwala, S. Borkar, T. Sakurai, and S. Nerendra. The Semiconductor Industry in 2025. *In Proceedings of the IEEE International Solid-State Circuits Conference Digest of Technical papers (ISSCC)*, pages 534–535, 2010.

[21] S. Borkar. Design and Test Challenges for 32 nm and Beyond. *keynote speech at IEEE International Test Conference (ITC)*, page 13, 2009.

[22] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. Parameter Variations and Impact on Circuits and Microarchitectures. *In Proceedings of IEEE Design Automation Conference (DAC)*, pages 338–342, 2003.

[23] S. Borri, M. Hage-Hassan, P. Girard, S. Pravossoudovitch, and A. Virazel. Defect Oriented Dynamic Faults Models for Embedded SRAMs. *In Proceedings of the 8th IEEE European Test Workshop*, 2003.

[24] M.A. Breuer and A.D. Friedman. *Diagnosis and Reliable Design of Digital Systems*. Computer Science Press., 1976.

[25] S. Di Carlo, A. Savino, A. Scionti, and P. Prinetto. Influence of Parasitic Capacitance Variations on 65nm and 32nm Predictive Technology Model SRAM Core-Cells. *In Proceedings of 17th Asian Test Symposium (ATS)*, pages 411–416, 2008.

[26] J.H. Chern, J. Huang, L. Arledge, P.C. Li, and P. Yang. Multilevel Metal Capacitance Models for CAD Design Synthesis Systems. *IEEE Electron Letters*, 13(1):32–34, 1992.

[27] J. A. Clark and D. K. Pradhan. Fault Injection: A Method for Validating Computer-System Dependability. *IEEE Computer*, 28(6):47–56, 1995.

[28] Z. Conroy, G. Richmond, G. Xinli, and B. Eklow. A Practical Perspective on Reducing ASIC NTFs. *In Proceedings of the IEEE International Test Conference (ITC)*, page 349, 2005.

[29] R. Dekker, F. Beenker, and L. Thijssen. A Realistic Fault Model and Test Algorithms for Static Random Access Memories. *IEEE Transaction on Computer Aided Design (CAD)*, 9(6):567–572, 1990.

[30] A. Deutsch, Coteus, P.W. Kopcsay, G.V. Smith, H.H. Surovic, C.W. Krauter, B.L. Edelstein, D.C., and P.L Restle. On-Chip Wiring Design Challenges for Gigahertz Operation. *In Proceedings of the IEEE*, 89(4):529–555, 2001.

[31] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, and M. Bastian. Analysis and Test of Resistive-Open Defects in SRAM Pre-charge Circuits. *Journal of Electronic Testing: Theory and Applications*, 23(5):435–444, 2007.

[32] L. Ding and P. Mazumder. The Impact of Bit-line Coupling and Ground Bounce on CMOS SRAM Performance. *In Proceedings of International VLSI Design*, pages 234–234, 2003.

[33] S.W. Director, W. Maly, and A.J. Storjwas. VLSI Design for Manufacturing Yield Enhancement. *Kluwer Academic Publishers*, 1990.

[34] T.D. Drysdale, A.R. Brown, G. Roy, S. Roy, and A. Asenov. Capacitance Variability of Short Range Interconnects. *Journal of Computational Electronics*, 7(3):124–127, 2007.

[35] P. Engelke, I. Polian, M. Renovell, and B. Becker. Simulating Resistive Bridging and Stuck-At Faults. *IEEE Transaction on CAD of Integrated Circuits and Systems*, 25(10):2181–2192, 2006.

[36] M. Favalli and M. Dalpasso. Bridging Fault Modeling and Simulation for Deep Submicron CMOS ICs. *IEEE Transactions on CAD*, 21(8):941–953, 2002.

[37] M. Favalli, M. Dalpasso, P. Olivo, and B. Ricco. Analysis of Dynamic Effects of Resistive Bridging Faults in CMOS and BiCMOS Digital ICs. *In Proceedings of IEEE International Test Conference (ITC)*, pages 865–873, 1993.

[38] H. Goto and K. Iwasaki. Experimental Fault Analysis of 2Mb SRAM Chips. *In Proceedings of the IEEE Design and Test Europe Conference (DATE)*, pages 623–630, 1997.

[39] H. Goto, S. Nakamura, and K. Iwasaki. Experimental Fault Analysis of 1Mb SRAM Chips. *In Proceedings of the IEEE VLSI Test Symposium (VTS)*, pages 31–36, 1997.

[40] J. Griffin, B. Matas, and C. de Suberbasaux. Memory 1997: Complete Coverage of DRAM, SRAM, EPROM and Flash Memory ICs. *Integrated Circuit Engineering Corporation, USA*, 1997.

[41] B. Grundmann, R.Galivanche, and S. Kundu. Circuit and Design Platform Challenges in Technologies Beyond 90nm. *In proceedings of the Design, Automation and Test in Europe (DATE)*, pages 44–47, 2003.

[42] S. Hamdioui. *Testing Multi-port Memories: Theory and Practice*. Doctoral Thesis, 2001.

[43] S. Hamdioui. *Testing Static Random Access Memories: Defects, Fault Models and Test Patterns*. Kluwer Academic Publishers, 2004.

[44] S. Hamdioui, Z. Al-Ars, G.N. Gaydadjiev, and A.J. van de Goor. An Investigation on Capacitive Coupling in RAM Address Decoders. *In proceedings of IEEE International Workshop on Memory Technology, Design, and Testing (MTDT)*, pages 9–14, 2007.

[45] S. Hamdioui, Z. Al-Ars, J. Jimenez, and J. Calero. PPM Reduction on Embedded Memories in System on Chip. *In Proceedings of IEEE European Test Symposium (ETS)*, pages 85–90, 2007.

[46] S. Hamdioui, Z. Al-Ars, L. Mhamdi, G. N. Gaydadjiev, and S. Vassiliadis. Trends in Tests and Failure Mechanisms in Deep Sub-micron Technologies. *In Proceesings of IEEE International Conference on Design and Test of Integrated Systems in Nanoscale Technology*, pages 216–221, 2006.

[47] S. Hamdioui, Z. Al-Ars, and A.J. van de Goor. Opens and Delay Faults in CMOS RAM Address Decoders. *IEEE Transaction on Computers*, 55(12):1630–1639, 2006.

[48] 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*, 19(2):195–205, 2003.

[49] 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 Transactions on CAD Integrated Circuits Systems*, 23(5):737–757, 2004.

[50] S. Hamdioui and A.J. van de Goor. Port Interference Faults in Two-Port Memories. *In Proceedings of the IEEE International Test Conference (ITC)*, pages 1001–1010, 1999.

[51] S. Hamdioui and A.J. van de Goor. An Experimental Analysis of Spot Defects in SRAMs: Realistic Fault Models and Tests. *Proceedings of the ninth Asian test symposium. (ATS)*, pages 131–138, 2000.

[52] S. Hamdioui, A.J. van de Goor, J.D. Reyes, and M. Rodgers. Memory Test Experiment: Industrial Results and Data. *IEEE Computers and Digital Techniques*, 153(1):1–8, 2006.

[53] H. Hao and E.J. McCluskey. Resistive Shorts within CMOS Gates. *In Proceedings of IEEE International Test Conference (ITC)*, pages 292–301, 1991.

[54] G. Harutunyan, V.A. Vardanian, and Y. Zorian. Minimal March Tests for Unlinked Static Faults in Random Access Memories. *In Proceedings of the VLSI Test Symposium (VTS)*, pages 53–59, 2005.

[55] Electronics Engineering Herald. SRAM Memory Interface to Microcontroller in Embedded Systems: Module 13. *EE Herald is an Electronics Design Magazine, Banglore*, 2006.

[56] R. F. Huang, Y. F. Chou, and C. W. Wu. Defect Oriented Fault Analysis for SRAM. *In Proceedings of 12th Asian Test Symposium (ATS)*, pages 256–262, 2003.

[57] I.S. Irobi, Z. Al-Ars, and S. Hamdioui. Bit Line Coupling Memory Tests for Single Cell Fails in SRAMs. *In proceedings of the VLSI Test Symposium (VTS)*, 2010.

[58] I.S. Irobi, Z. Al-Ars, and S. Hamdioui. Detecting Memory Faults in the Presence of Bit Line Coulping in SRAM Devices. *In Proceedings of IEEE International Test Conference (ITC)*, pages 1–10, 2010.

[59] I.S. Irobi, Z. Al-Ars, and M. Renovell. Parasitic Memory Effect in CMOS SRAMs. *In Proceedings of the IEEE International Design and Test Workshop (IDT)*, 2010.

[60] R.D. Isaac. The Future of CMOS Technology. *International Business Machine Corporation (IBM) Journal*, 44(3):369–378, 2000.

[61] G. Jervan, P. Eles, Z. Peng, R. Ubar, and M. Jenihhin. Test Time Minimization for Hybrid BIST of Core-Based Systems. *In Proceedings of IEEE Asian Test Symposium (ATS)*, pages 318–323, 2003.

[62] G. Jervan, Z. Peng, and R. Ubar. Test Cost Minimization for Hybrid BIST. *In Proceedings of IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT)*, pages 283–291, 2000.

[63] K. Kinoshita and K.K. Saluja. Built-In Testing using an On-Chip Compact Testing Scheme. *IEEE Transactions on Computers C-35(10)*, pages 862–870, 1986.

[64] H. Koike, T. Takeshima, and M. Takada. A BIST Scheme using Microprogram ROM for Large Capacity Memories. *In Proceeding of IEEE Custom Integrated Circuits Conference*, pages 815–822, 1990.

[65] E. Larsson, J. Pouget, and Z. Peng. Defect-Aware SOC Test Scheduling. *In proceedings of IEEE VLSI Test Symposium (VTS)*, pages 359–364, 2004.

[66] R. Madge, B.R. Benware, and W.R. Daasch. Obtaining High Defect Coverage for Frequency Dependent Defects in Complex ASICs. *IEEE Design and Test of Computers*, pages 46–52, 2003.

[67] P. Mazumder and J.K. Patel. An Efficient Design of Embedded Memories and their Testability Analysis using Markov Chains. *IEEE Transactions on Computers C-38(3)*, pages 394–407, 1989.

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

[69] G.E Moore. Cramming More Components onto Integrated Circuits. *Electronics magazine*, 38(8), 1965.

[70] 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*, 29:441–453, 2010.

[71] H. Nambu, K. Kanetani, Y. Idei, N. Homma, K. Yamaguchi, T. Hiramoto, N. Tamba, M. Odaka, K. Watanabe, T. Ikeda, K. Ohhata, and Y. Sakurai. High-Speed Sensing Techniques for Ultrahigh-Speed SRAMS. *IEEE Journal of Solid-State Circuits*, 27:632–640, 1992.

[72] H. Nambu, K. Kanetani, Y. Idei, T. Masuda, K. Higeta, M. Ohayashi, M. Usami, K. Yamaguchi, T. Kikuchi, T. Ikeda, K. Ohhata, T. Kusunoki, and N. Homma. A 0.65-ns, 72-kb ECL-CMOS RAM Macro for a 1-Mb SRAM. *IEEE Journal of Solid-State Circuits*, 30(4):491–499, 1995.

[73] K. Noda, K. Takeda, K. Matsui, S. Masuoka, H. Kawamoto, N. Ikezawa, Y. Aimoto, N. Nakamura, T. Iwasaki, H. Toyoshima, and T. Horiuchi. An Ultrahigh-Density High-Speed Loadless Four-Transistor SRAM Macro with Twisted Bitline Architecture and Triple-Well Shield. *IEEE Journal of Solid-State Circuits*, 36(3):510–515, 2001.

[74] K. Ohhata, Y. Sakurai, H. Nambu, K. Kanetani, Y. Idei, T. Hiramoto, N. Tamba, K. Yamaguchi, M. Odaka, K. Watanabe, T. Ikeda, and N. Homma. Noise Reduction Techniques for an ECL-CMOS RAM with a 2 ns Write Cycle Time. *In Proceedings of Bipolar/BiCMOS Circuits and Technology Meeting*, 1992.

[75] C.A. Papachristou and N.B. Saghal. An Improved Method for Detecting Functional Faults in Random Access Memories. *IEEE Transaction on Computers*, 34(3):110–116, 1985.

[76] S. Pasricha and N. Dutt. *On-Chip Communication Architectures - System On Chip Interconnect*. Morgan Kaufmann Publishers, USA, 2008.

[77] T. Powell, A. Kumar, J. Rayhawk, and N. Mukherjee. Chasing Subtle Embedded RAM Defects for Nanometer Technologies. *In Proceedings of the IEEE International Test Conference (ITC)*, pages 841–850, 2005.

[78] B. Prince. *Semiconductor Memories, A Handbook of Design and Manufacturing and Application, Second Edition*. John Wiley and Sons, West Sussex, 1991.

[79] B. Prince. *High Performance Memories. New Architecture DRAMs and SRAMs - Evolution and Function*. John Wiley and Sons, Chichester, 1996.

[80] I.M. Ratiu and H.B. Bakoglu. Pseudorandom Built-In-Self-Test Methodology and Implementation for the IBM RISC Systems/6000 Processors. *IBM Journal of Research and Development*, 34(1):78–84, 1990.

[81] S. Reddy, I. Pomeranz, T. Huaxing, S. Kajihara, and S. Kinoshita. On Testing of Interconnect Open Defects in Combinational Logic Circuits with Stems of Large Fanout. *In Proceedings of IEEE International Test Conference (ITC)*, pages 83–89, 2002.

[82] M. Redeker, B.F. Cockburn, and D.G. Elliott. An Investigation into Crosstalk Noise in DRAM Structures. *In Proceedings of IEEE International Workshop on Memory Technology, Design and Testing (MTDT)*, pages 123–129, 2002.

[83] M. Renovell, M. Comte, I. Polian, P. Engelke, and B. Becker. Analyzing the Memory Effect of Resistive Open in CMOS Random Logic. *In Proceedings of the IEEE Design and Test of Integrated Systems in Nanoscale Technology*, pages 251–256, 2006.

[84] M. Renovell, M. Comte, I. Polian, P. Engelke, and B. Becker. A Specific ATPG Technique for Resistive Open with Sequence Recursive Dependency. *In Proceedings of the IEEE Asian Test Symposium (ATS)*, pages 273–278, 2006.

[85] M. Renovell, P. Huc, and Y. Bertrand. The Concept of Resistance Interval: A New Parametric Model for Resistive Bridging Fault. *In Proceedings of IEEE VLSI Test Symposium (VTS)*, pages 184–189, 1995.

[86] R. Rodriguez-Montanes, P. Volf, and J.P. de Gyvez. Resistance Characterization for Weak Open Defects. *IEEE Design and Test of Computers*, 19(5):18–26, 2002.

[87] A. Scionti S. Di Carlo, P. Prinetto and Z. Al-Ars. Automating Defects Simulation and Fault Modeling for SRAMs. *In Proceedings of IEEE International High Level Design Validation and Test Workshop (HLDVT)*, pages 169–176, 2008.

[88] K.K. Saluja, S.H. Sng, and K. Kinoshita. Built-In-Self-Test RAM: A Practical Alternative. *IEEE Design and Test 4(1)*, pages 3.4.1–4, 1991.

[89] I. Schanstra and A.J. van de Goor. Industrial Evaluation of Stress Combinations for March Tests Applied to SRAMs. *In Proceedings of the IEEE International Test Conference (ITC)*, pages 983–992, 1999.

[90] I. Schanstra and A.J. van de Goor. Consequences of RAM Bitline Twisting for Test Coverage. *In Proceedings of Design Automation and Test in Europe (DATE)*, pages 1176–1177, 2003.

[91] A. Scionti, G. Politano, S. Di Carlo, P. Prinetto, and A. Savino. Genetic Defect Based March Test Generation for SRAM Computer-System Dependability. *In Proceedings of International Conference on Applications of Evolutionary Computation (EvoWorkshops)*, pages 141–150, 2011.

[92] J. Srinivasan, S.V. Adve, P. Bose, and J.A. Rivers. The Impact of Technology Scaling on Lifetime Reliability. *In Proceedings of International Conference on Dependable Systems and Networks*, pages 177–186, 2004.

[93] K. Takeda, Y. Aimoto, N. Nakamura, H. Toyoshima, T. Iwasaki, K. Noda, K. Matsui, S. Itoh, S. Masuoka, T. Horiuchi, A. Nakagawa, K. Shimogawa, and H. Takahashi. A 16-Mb 400-MHZ Loadless CMOS Four Transistor SRAM Macro. *IEEE Journal of Solid-State Circuits*, 35(11):1631–1640, 2000.

[94] A.J. van de Goor. *Testing Semiconductor Memories. Theory and Practice*. ComTex Publishing, The Netherlands., 1998.

[95] A.J. van de Goor and Z. Al-Ars. Functional Memory Faults: A Formal Notation and a Taxonomy. *In Proceedings of VLSI Test Symposium (VTS)*, pages 281–289, 2000.

[96] A.J. van de Goor and J. de Neef. Industrial Evaluation of DRAM Tests. *In Proceedings of the IEEE Design Automation and Test in Europe (DATE)*, pages 623–630, 2006.

[97] A.J. van de Goor, S. Hamdioui, and R. Wadsworth. Detecting Faults in the Peripheral Circuits and an Evaluation of SRAM Tests. *In Proceedings of IEEE International Test Conference (ITC)*, pages 114–123, 2004.

[98] H. Vierhaus, W. Meyer, and U. Glaser. CMOS Bridges and Resistive Transis-

tor Faults: IDDQ versus Delay Effects. *In Proceedings of IEEE International Test Conference (ITC)*, pages 83–91, 1993.

[99] S.C. Wong, G.Y. Lee, and D.J. Ma. Modeling of Interconnect Capacitance, Delay, and Crosstalk in VLSI. *IEEE Transactions Semiconductor Manufacturing*, 13(1):108–111, 2000.

[100] J.C. Yeh, C.H. Chen, C.W. Wu, and S.F. Kuo. a Systematic Approach to Memory Test Time reduction. *IEEE Design and Test of Computers*, 25(6):560–570, 2008.

[101] W. Zhao and Y. Cao. New Generation of Predictive Technology Model for Sub-45nm Early Design Exploration. *IEEE Transaction on Electron Devices*, 53(11):2816–2823, 2006.

[102] J. Zhou and H.J. Wunderlich. Software-based Self Test of Processors under Power Constraints. *In Proceedings of the IEEE Conference on Design Automation and Test in Europe (DATE)*, pages 1–6, 2006.

List of Publications

Journal papers

- **I.S Irobi**, Zaid Al-Ars "Parasitic Bit-Line Coupling in Deep sub-micron Devices: Analysis and Tests", *Submitted to* The IEEE Transactions on Computers.

International conferences

- **I.S Irobi**, Z. Al-Ars, S. Hamdioui. Bit Line Coupling Memory Tests for Single-Cell Fails in SRAMs. In Proceedings of 28th IEEE VLSI Test Symposium (VTS), Santa Cruz, California, USA. April 2010.

- **I.S. Irobi**, Z. Al-Ars, S. Hamdioui. Detecting Memory Faults in the Presence of Bit Line Coupling in SRAM Devices. In Proceedings of IEEE International Test Conference (ITC), Austin, Texas, USA. November 2010.

- **I.S Irobi**, Z. Al-Ars, M Renovell. Parasitic Memory Effect in CMOS SRAMs. In Proceedings of International Design and Test Workshop (IDT), Abu Dhabi, United Arab Emirates. December 2010.

- **I.S Irobi**, Z. Al-Ars, S. Hamdioui, M Renovell. Influence of Parasitic Memory Effect on Single-Cell Faults in SRAMs. In Proceedings of IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS), Cottbus, Germany. April 2011.

- **I.S Irobi**, Z. Al-Ars, S. Hamdioui. Memory Test Optimization for Parasitic Bit Line Coupling in SRAMs. In Proceedings of European Test Symposium (ETS), Trondheim, Norway. May 2011.

- **I.S Irobi**, Z. Al-Ars, S. Hamdioui, Claude Thibeault. Test for Parasitic Memory Effect in SRAMs. In Proceedings of Asian Test Symposium (ATS), New Delhi, India. November 2011.

- S. Hamdioui, Venkataraman, **I.S Irobi**, Z. Al-Ars. A new Approach on

Reducing No Trouble Found for Semiconductor Memories. In Proceedings of Asian Test Symposium (ATS), New Delhi, India. November 2011.

- Jude Angelo Ambrose, Ben Juurlink, **Sandra Irobi**. Scratchpad Memory Size Optimization for Real-Time Multiprocess Embedded Applications. In Proceedings of World Congress on Computer Science and Information Engineering (CSIE), 2011.

National conferences

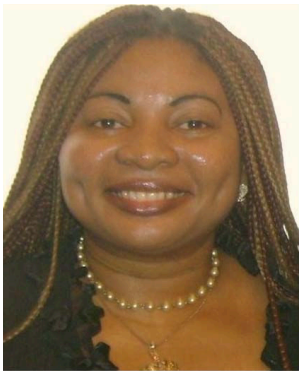
- **I.S Irobi**, Z. Al-Ars. Worst-Case Bit Line Coupling Backgrounds for Open Defects in SRAM Cells. In proceedings of Annual Workshop on Circuits, Systems and Signal Processing (ProRisc), Veldhoven, The Netherlands, November 2009.

- **I.S Irobi**, B.H.H Juurlink. On-chip Scratchpad Memory Size Prediction and Allocation for Multiprocess Embedded Applications. In Proceedings of the 17th Annual Workshop on Circuits, Systems and Signal Processing (ProRisc), pp. 270-276, Veldhoven, Netherlands. November 2006.

Other publications

- **Ijeoma Sandra Irobi**, Anders Wall. Algorithms for Determining the Adequacy Constraints for the Validation of Models of complex Real-Time Systems. In proceedings of International conference on Models, Simulation and Visualization (MSV), Las Vegas, Nevada, USA. June 2005.
- **Ijeoma Sandra Irobi**, Johan Andersson, Anders Wall. Correctness Criteria for Models' Validation - A Philosophical Perspective. In proceedings of International Conference on Models, Simulation and Visualization (MSV), Las Vegas, Nevada, USA. June 2004.
- Dodig-Crnkovic Gordana, **Ijeoma Sandra Irobi**. Model Validation, Evolutionary Systems and Semantics of Information. In proceedings of International Conference on Model Based Reasoning in Science and Engineering (MBR), Pavia, Italy. December 2004.

Curriculum vitae



Ijeoma Sandra Irobi comes from Imo state, Nigeria. In 1996 she obtained a Bachelor of Engineering degree in Electrical/Electronic engineering from Enugu State University of Science and Technology, Nigeria. Sandra started her work in the industry from Nigerian Telecommunications Limited (Long distance communications) in 1997, and later at Federal Housing Authority Homes, Abuja from 2001 to 2003, where she became Head, Information and Communications Technology (ICT) section. In 2004, Sandra obtained her first MSc degree in Computer Engineering from Malardalen University Sweden, majoring in Real-Time systems; and in 2005 obtained her second MSc degree in Computer Science from Uppsala University, Sweden, where she was the first graduated international student in the program.

In 2006, Sandra was awarded the competitive NFP grant for her PhD research at the Computer Engineering laboratory, department of Microelectronics and Computer engineering of Delft University of Technology (TU DELFT) in The Netherlands. She worked with the Memory and Logic Tesing (MaLT) Group on the ANTPAR project. Towards the end of her PhD, Sandra spent some months on internship at LACIME Laboratory, École de Technologie Supérieure, Université du Québec, Montreal, Canada.

Sandra initiated the mutual collaboration and capacity development program between University of Nigeria (UNN) and TU Delft (UNN-TU DELFT linkage), which has facilitated short and long-term training for UNN academic staff for advanced degrees at TU Delft since 2006. The second phase of this collaborative serves to establish a Microelectronics Research Center of Excellence in West Africa. Her research interests include design and testing of embedded memory, design for testability (DFT)

techniques, diagnosis and IC reliability, Built-In Self-Test (BIST), fault simulation, Inductive fault analysis (IFA) and yield improvement of defective ICs, Automatic test pattern generation, fault analysis and tests, real-time systems and requirement engineering.

Sandra's other interests include development issues, and techno-preneurship that is, turning scientific knowledge and technology into development-oriented innovative and profitable business enterprises. This is typified by her involvement in Young Entrepreneurs for Africa (YEFA). In 2010, Sandra was awarded the Nigeria Excellence Award by the Ambassador of Nigeria to The Netherlands for visible contributions to development.