A comprehensive comparison between design for testability techniques for total dose testing of flash-based FPGAs

Mohamed Ayman Ahmed Ibrahim

The American University in Cairo

School of Sciences and Engineering

A COMPREHENSIVE COMPARISON BETWEEN DESIGN FOR TESTABILITY TECHNIQUES FOR TOTAL DOSE TESTING OF FLASH-BASED FPGAS

A Thesis Submitted to

Electronics and Communications Engineering Department

in partial fulfillment of the requirements for
the degree of Master of Science

by Mohamed Ayman Ahmed Ibrahim

under the supervision of Prof. Ahmed Abou-Auf
August/2019

To my wonderful family

Father, Mother, Brother & Sister

ACKNOWLEDGMENTS

Firstly, I would like to express my sincere gratitude to my advisor Prof. Ahmed Abou-Auf for the continuous support of my M.Sc. study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my M.Sc. study.

Besides my advisor, I would like to thank the rest of my thesis defense committee: Prof. Yehea Ismail and Prof. Amr Wassal, for their insightful comments and encouragement.

I would also like to thank Eng. Mostafa Mahmoud, Eng. Mohamed Sami, Eng. Nancy Sayed and Eng. Assem Mohamed for their assistance throughout my research.

Last but not the least, I would like to thank my family: my parents, my brother and my sister for supporting me spiritually throughout my M.Sc. study and my life in general.

ABSTRACT

The American University in Cairo, Egypt

TID testing

Name: Mohamed Ayman Ahmed Ibrahim

Supervisor: Prof. Ahmed Abou-Auf

Radiation sources exist in different kinds of environments where electronic devices often operate. Correct device operation is usually affected negatively by radiation. The radiation resultant effect manifests in several forms depending on the operating environment of the device like total ionizing dose effect (TID), or single event effects (SEEs) such as single event upset (SEU), single event gate rupture (SEGR), and single event latch up (SEL).

CMOS circuits and Floating gate MOS circuits suffer from an increase in the delay and the leakage current due to TID effect. This may damage the proper operation of the integrated circuit. Exhaustive testing is needed for devices operating in harsh conditions like space and military applications to ensure correct operations in the worst circumstances. The use of worst case test vectors (WCTVs) for testing is strongly recommended by MIL-STD-883, method 1019, which is the standard describing the procedure for testing electronic devices under radiation. However, the difficulty of generating these test vectors hinders their use in radiation testing.

Testing digital circuits in the industry is usually done nowadays using design for testability (DFT) techniques as they are very mature and can be relied on. DFT techniques include, but not limited to, ad-hoc technique, built-in self test (BIST), muxed D scan, clocked scan and enhanced scan. DFT is usually used with automatic test patterns generation (ATPG) software to generate test vectors to test application specific integrated circuits (ASICs), especially with sequential circuits, against faults like stuck at faults and path delay faults. Despite all these recommendations for DFT, radiation testing has not benefited from this reliable technology yet. Also, with the big variation in the DFT techniques, choosing the right technique is the bottleneck to achieve the best results for TID testing.

In this thesis, a comprehensive comparison between different DFT techniques for TID testing of flash-based FPGAs is made to help designers choose the best suitable DFT technique depending on their application. The comparison includes muxed D scan technique, clocked scan technique and enhanced scan technique. The comparison is done using ISCAS’89 benchmarks circuits. Points of comparisons include FPGA resources utilization, difficulty of designs bring-up, added delay by DFT logic and robust testable paths in each technique.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

ASIC	Application Specific Integrated Circuits
ATPG	Automatic Test Pattern Generation
BIST	Built-In Self-Test
CMOS	Complementary Metal Oxide Semiconductor
CUT	Circuit Under Test
DFT	Design For Testability
EDA	Electronic Design Automation
FG	Floating Gate
FPGA	Field Programmable Gate Arrays
IC	Integrated Circuit
LFSR	Linear Feedback Shift Register
MOS	Metal Oxide Semiconductor
MOSFET	Metal Oxide Semiconductor Filed Effect Transistor
ORA	Output Response Analyzer
RILC	Radiation Induced Leakage Current
RTL	Register Transfer Level
SEE	Single Event Effect
SEGR	Single Event Gate Rupture
SEL	Single Event Latch up
SEU	Single Event Upset
SSI	Small Scale Integration
STA	Static Timing Analysis
STI	Shallow Trench Isolation
TID	Total Ionizing Dose
TPG	Test Pattern Generator
VLSI	Very Large Scale Integration
WCTV	Worst-Case Test Vector

Chapter 1

Summary

Radiation sources exist in different kinds of environments where electronic devices often operate. The radiation sources can be found in terrestrial, extra-terrestrial and man-made radiation environments. Correct device operation is usually affected negatively by radiation. The radiation resultant effect manifests in several forms depending on the operating environment of the device like total ionizing dose effect (TID), or single event effects (SEEs) such as single event upset (SEU), single event gate rupture (SEGR), and single event latch up (SEL).

CMOS circuits and Floating gate MOS circuits suffer from an increase in the delay and the leakage current due to radiation effects. The interest here is focused on the TID effect. TID effect may damage the proper operation of the integrated circuit. Exhaustive testing is needed for devices operating in harsh conditions like space and military applications to ensure correct operations in the worst circumstances. The use of worst case test vectors (WCTVs) for testing is strongly recommended by MIL-STD-883, method 1019, which is the standard describing the procedure for testing electronic devices under radiation. However, the difficulty of generating these test vectors hinders their use in radiation testing.

Testing digital circuits in the industry is usually done nowadays using design for testability (DFT) techniques as they are very mature and can be relied on. DFT techniques include, but not limited to, ad-hoc technique, built-in self test (BIST), muxed D scan, clocked scan and enhanced scan. DFT is usually used with automatic test patterns generation (ATPG) software to generate test vectors to test application specific integrated circuits (ASICs), especially with sequential circuits, against faults like stuck at faults and path delay faults. Despite all these recommendations for DFT, radiation testing has not benefited from this reliable technology yet. Also, with the big variation in the DFT techniques, choosing the right technique is a bottleneck to achieve the best results from TID testing.

In this thesis, a comprehensive comparison between different DFT techniques for TID testing of flash-based FPGAs is made to help designers choose the best suitable DFT technique depending on their application. The comparison includes muxed D scan technique, clocked scan technique and enhanced scan technique. Methodologies for applying the DFT techniques are developed. Points of comparisons include FPGA resources utilization, difficulty of designs bring-up, added delay by DFT logic and robust testable paths in each technique.

The comparison is done using ISCAS'89 benchmarks circuits. They are described as being suitable for researchers working in the field of scan designs for sequential circuits. The comparison is done on designs of different sizes. The designs are implemented using Microsemi ProASIC3 flash-based FPGAs. For each design, the methodologies for applying the three DFT techniques under investigation are applied. Results show that some techniques are more superior to others depending on the point of comparison. Recommendations on when to use each technique are presented and verified by the results of the comparison.

The rest of the thesis is organized as follows. Chapter two includes a review on different radiation sources and effects. It also focuses on the TID effect on CMOS circuits and floating gate MOS transistor which are the switch elements in flash-based FPGA. Chapter three introduces a review on different methodologies of design for testability like Ad-hoc designs, BIST designs and scan designs with a special focus on different scan design techniques. Also, the path delay testing is discussed and how it is done using scan designs. Chapter four discusses the previously developed efforts to generate WCTVs of ASICs and FPGAs. Chapter five explains the methodology of using each DFT technique to generate the WCTVs. Then, the results of the comparison between the DFT techniques on the ISCAS'89 benchmark designs are presented. Finally, the thesis ends by chapter six which concludes the work done in this thesis and possible future work to build upon the results concluded here.

Chapter 2

Radiation effects on CMOS circuits

Radiation sources surround us everywhere. They are found in terrestrial environments, extra-terrestrial environments and in some cases are even artificially made by man. Electronic devices operate in all of these environments. Radiation sources surely affect them and can cause severe damages to their operation.

One of the important effects of radiation is the Total Ionizing Dose effect “TID”. It has a severe damaging effect on CMOS circuits’ operation. The reason behind this is the ionizing–radiation effect on the materials used in the fabrication of CMOS circuits. In this chapter, different radiation sources are illustrated and their resultant radiation effects. Then, the TID effect on CMOS circuits is discussed along with TID testing techniques.

2.1 Radiation sources

It is important to understand the characteristics of radiation sources so that electronic devices are designed to properly operate in different environments.

In terrestrial environments, radiation can come from neutrons which are found in the atmosphere, or alpha particles which are emitted from defects found inside chips materials. While in extra-terrestrial environments like space, radiation comes from trapped particles around the Earth’s atmosphere, particles which originated from the sun and high energy galactic cosmic rays’ particles. Artificial man-made radiation originates from nuclear reactors, biomedical devices and high energy particles physics experiments equipment.

2.1.1 Terrestrial environment

In the terrestrial environment, there are two sources of radiation that highly affect electronic circuits. First one is the neutrons found in Earth’s atmosphere. The other is the alpha particles coming from decay inside the integrated circuit (IC) material.

2.1.1.1 Atmospheric neutrons

Aviation equipment has lots of electronics inside it so it is important to know the type of radiation it may encounter when flying at an altitude of 15 Km. Atmospheric neutrons have the highest radiation effect at this altitude [8].

When galactic cosmic rays hit Earth’s atmosphere, atmospheric neutrons are generated. These neutrons trigger some nuclear reactions. Their interaction with oxygen and nitrogen generate protons, muons, neutrons and pions. These particles are the radiation affecting the electronics inside the aviation equipment and they reach their maximum level at approximately 15 Km.

2.1.1.2 IC radioactive decay

Some elements like Uranium, Thorium and Platinum are used in integrated circuits fabrication process. Sometimes they are also found in the chip material as unwanted defects. These elements face radioactive decay producing alpha particles. These alpha particles are the most responsible source for soft errors happening in electronic devices. This type of radiation is gaining more and more importance compared to atmospheric neutrons as technology advances and the feature size of IC decreases. In Complementary Metal Oxide Semiconductor (CMOS) circuits, some particles like muons are becoming more important due to the threats they can cause as the device dimensions decreases [9].

2.1.2 Space environment

All nuclei that can occur naturally are found in the space environment. That is why the space environment is very harsh to electronic devices. These nuclei particles can come from geomagnetically trapped electrons and protons, high-energy cosmic-ray particles, particles of solar origin, and possibly particles trapped in the magnetospheres of other planets. The near-Earth environment where satellites operate is the one that concerns electronic circuit designers. In this region, the electrically charged particles trapped in the Earth’s magnetosphere and, high-energy particles of cosmic-ray origin are the most dominating radiation sources [30] as depicted in Fig.1.

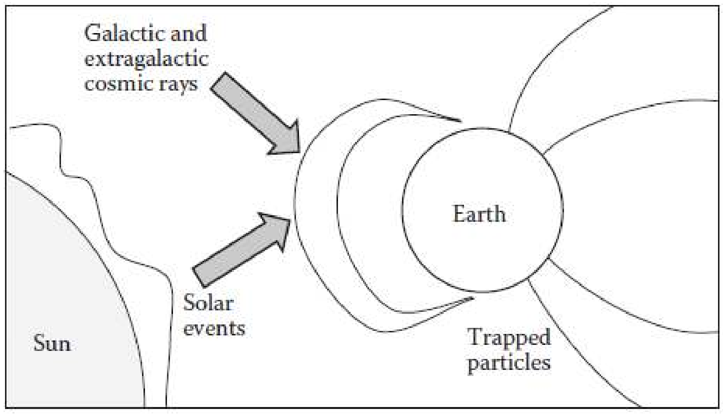


Figure 1: Schematic illustration of three main sources of radiation in space [8].

Due to the complexity of the space environment, it is very difficult to assess the amount of ionizing radiation hitting a system in space, and it is also dependent on the cycle of solar activity. Furthermore, the exact amount of radiation that affects a specific electronic device depends on its location inside the spacecraft or satellite that is operating due to the shielding effect of the materials used. In a context like space, one cannot just overdesign the electronic systems because the addition of this weight when applied to spacecraft or satellites has a high cost. Furthermore, these devices lack the power to support overdesigned electronic systems. This is why the complex simulation tools and models are used to predict the dose affecting the devices and design them within the appropriate margins [8].

2.1.3 Man-made radiation environment

Some radiation environments are artificially made by man. For example, the biomedical devices or high energy particles physics experiment equipment.

Large Hadron Collider (LHC) at CERN in Switzerland is one practical example. It will receive an upgrade, after which, doses may exceed 100 Mrad(Si). These are massive doses compared to most missions in space by the National Aeronautics and Space Administration (NASA). In these missions, expected doses are around 100 Krad(Si). For that, dedicated “rad-hard” libraries are used in LHC. These libraries have a layout that is made specially to avoid the problems of standard design and ultimately to withstand the high levels of radiation found in these environments.

Another practical example is the ITER neutral beam test facility. It contains a fusion reactor where electronic devices are expected to be hit by large fluxes of neutrons of energies up to 14 MeV, the doses expected in the ITER environment can reach a value of 50 rad(Si) in one operating hour [10].

2.2 Radiation effects

What is meant by radiation effects is the ability of radiation sources to cause damage that can either be reversible or not inside the electronic devices. This effect depends on the type of radiation source and the environment in which the device operates. Two of the most famous effects are Total Ionization Dose effects and Single Event effects.

2.2.1 Total Ionization Dose effects

Total Ionization Dose (TID) is the amount of energy resulting from ionization radiation at the target material. TID is measured either by rad or gray (Gy). Rad is the amount of energy equal to 100 ergs transferred to one gram of the target material. Since the amount of the energy transferred is material-dependent, the radiation dose is usually followed by the target material. Gray (Gy) is one-hundredth of rad.

One of the most susceptible materials to TID is the silicon oxide (SiO_2) which is one of the most essential parts in the Metal Oxide Semiconductor (MOS) structure. Two main effects result from TID in electronics devices [8]:

1. Positive charge (holes) trapping in the insulation layer
2. Generation of interface states in the insulation layer

The band diagram for an n-type MOS structure on a p-substrate and biased at positive voltage is illustrated in Fig.2.

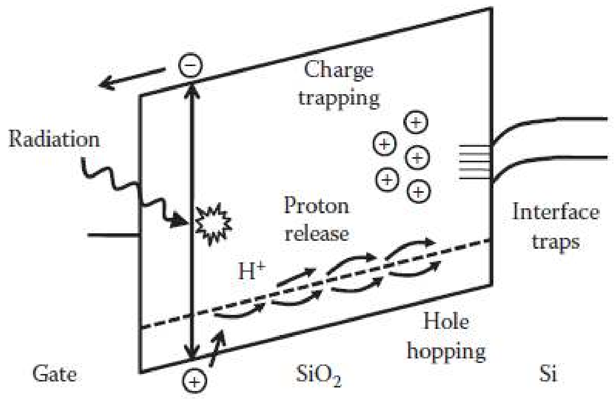


Figure 2: Band diagram of an n-type MOS biased at positive voltage [8]

When radiation hits a MOS structure, energy is transferred to the silicon oxide. This leads to the generation of electron-hole pairs inside SiO_2, as illustrated in Fig.3. After that, a process called initial recombination occurs, where some electron-hole pairs recombine. This depends on many factors: applied electric field, the transferred energy and the type of charged particle i.e. the source of radiation. Some pairs are not recombined in this process and they remain uncombined. They are called charge yield. After initial recombination process is finished, electrons and holes start to move where electrons are quickly attracted to the positively charged gate due to their high mobility, while, on the other hands, holes move slowly towards the silicon substrate due to their lower mobility. This difference in speed between electrons and holes cause temporary negative voltage shift in the characteristics of the MOS device, such as the threshold voltage [12].

After the holes reach the Silicon/Silicon Oxide (SiO_2) interface, some holes go to the Silicon substrate, and some others are trapped in defects sites. These defects sites have a higher density at the Silicon/ Silicon Oxide (SiO_2) interface. These trapped holes cause permanent negative voltage shift in the characteristics of the MOS structure [8][12]. This can affect the MOS in the following ways:

1. Can cause a voltage shift in the threshold voltage.
2. Can cause inversion of the channel leading to flow of leakage current in the OFF state.
3. Can cause an increase in the static power of the integrated circuits.

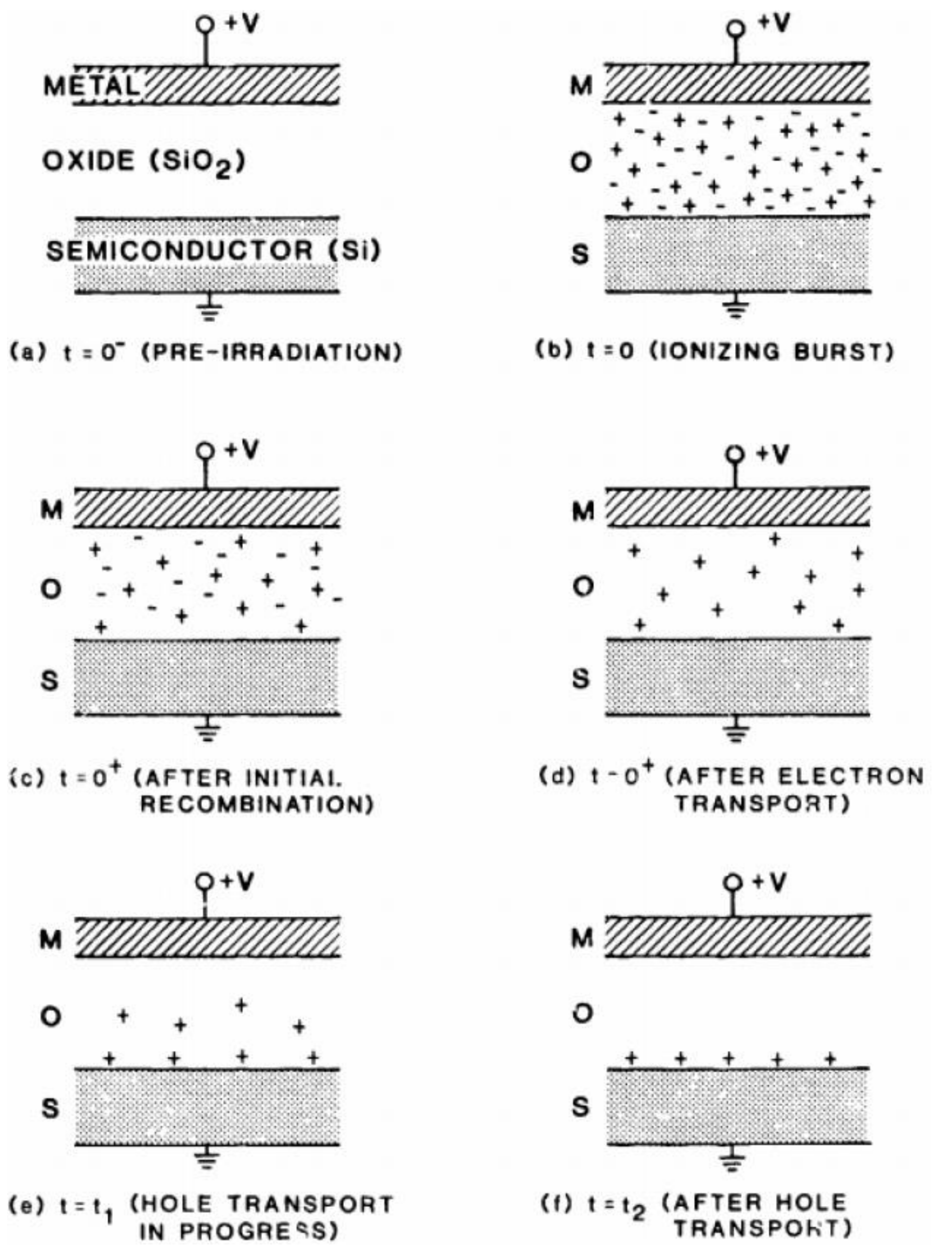

+V
METAL
OXIDE (SiO2)
SEMICONDUCTOR (Si)
(a) t = 0⁻ (PRE-IRRADIATION)
M
O
S
(b) t = 0 (IONIZING BURST)
(c) t = 0⁺ (AFTER INITIAL RECOMBINATION)
(d) t - 0⁺ (AFTER ELECTRON TRANSPORT)
(e) t = t1 (HOLE TRANSPORT IN PROGRESS)
(f) t = t2 (AFTER HOLE TRANSPORT)

Figure 3: Illustration of generation, recombination, transport, and trapping of charges in SiO_2 films [31].

In parallel to holes movement process to the Si/SiO_2 interface or when holes are trapped, some hydrogen ions (protons) can be released. Interface traps are created by the arrival of these hydrogen ions at the interface. These interface traps can exchange carriers with the channel, and their occupancy depends on the position of the Fermi level at the interface. This is a much slower process than the charge accumulation due to the trapped holes near the interface, but it also depends on the applied electric field [8][12]. These traps can affect the MOS in the following ways:

1. Can cause a positive shift in the threshold voltage of the NMOS transistor and a negative voltage shift in the threshold voltage of the PMOS transistor.
2. Can affect the mobility of the transistor.
3. Can decrease current capability in the transistor.
4. Can cause timing degradation of the integrated circuits.

2.2.2 Single-Event effects

Single-Event Effects (SEE) are caused by the passage of high-energy particles (heavy ion) through sensitive regions of an electronic device. They can be classified into three categories depending on their effects.

The first category is “Soft”, in which the damage is temporary e.g. soft errors in memory circuits. The second category is “Hard”, in which the damage is irreversible e.g. the rupture of the dielectric in the gate. The third category is the case between Soft and Hard, in which some other SEE may or may not cause damage depending on the operation of the electronic device. Example for the third category is the case of a single-event latch-up, in which the time to cut the power supply after the occurrence of the event decides if the damage happened to the device or not [8].

The most famous SEE are [8][13]:

1. Single-event upset (SEU):

   A soft SEE, in which the value of a single bit in memory is flipped because of a single ionizing particle. It is also known as a soft error. Rewriting the bit again restores the correct value of the memory bit i.e. the damage is just temporary.
2. Single-event gate rupture (SEGR):

   A hard SEE, in which the gate of a MOSFET is ruptured. This effect cannot be reversed i.e. the damage is permanent.

3. Single-event latch-up (SEL):

   A type of SEE where its damage depends on what the circuit does after being exposed to radiation. It is characterized by the activation of parasitic bipolar structures in CMOS circuits. This leads to a sudden increase in supply current. There may or may not be permanent damage to the device depending on the time taken to cut the power source of the circuit.

2.3 TID effect in MOS transistor

As explained in the previous sections, when ionization radiation hits MOS transistors, the high-energy particles interact with the atoms of the dielectric SiO_2 leading to the generation of electron-hole pairs. The density of the generated electron-hole pairs is determined by the amount of the energy transferred to the dielectric material along the track of the high-energy particles. Equation (1) describes the relation between the radiation energy that enters the plane of the target material (ΔE_E) and the radiation energy that leaves the material plane (ΔE_L). This is called Linear Energy Transfer (LET) or stopping power which expresses the loss of energy per unit length (dE/dx) [14].

$$\Delta E_L(\gamma) = \Delta E_E(\gamma)\, exp\left(-\frac{\mu_{en}}{\rho}\rho\Delta x\right), \tag{1}$$

where, $\mu_{en}/\,\rho$ is the mass attenuation coefficient of the target material, ρ is the target material density, and Δx is the thickness of the target material.

Figure 4 describes the LET for protons and electrons as a function of the particle energy. For protons, LET decreases monotonically, while in the case of electrons, the LET decreases as a function of particle energy for energies below 1 Mev, while it increases for higher energies [15].

The generation of the electron-hole pairs due to the ionization radiation causes what are called oxide traps and interface traps. Oxide traps result from the movement of the holes towards the oxide-Silicon (SiO_2/Si) interface. Some of the holes get trapped at the interface due to defects which results in the formation of positive oxide-trap charges [16]. Interface traps are also generated at the SiO_2/Si interface. They are the result of the interaction between the hydrogen ions that can drift to the interface. The interface traps generation process occurs in a long time compared to the oxide charges generation process. They act as energy levels within the silicon band-gap. Interface traps can be negative or positive, depending on the location of the Fermi level at the interface, if the Fermi level at the interface is below the trap energy, it acts as a donor and the interface trap is positively charged, and if the Fermi level at the interface is above the trap energy, it acts as an acceptor and the interface trap is negatively charged. Positive interface traps cause negative threshold voltage shifts for p-channel transistors, while negative interface traps cause positive threshold voltage shifts for n-channel transistors [12].

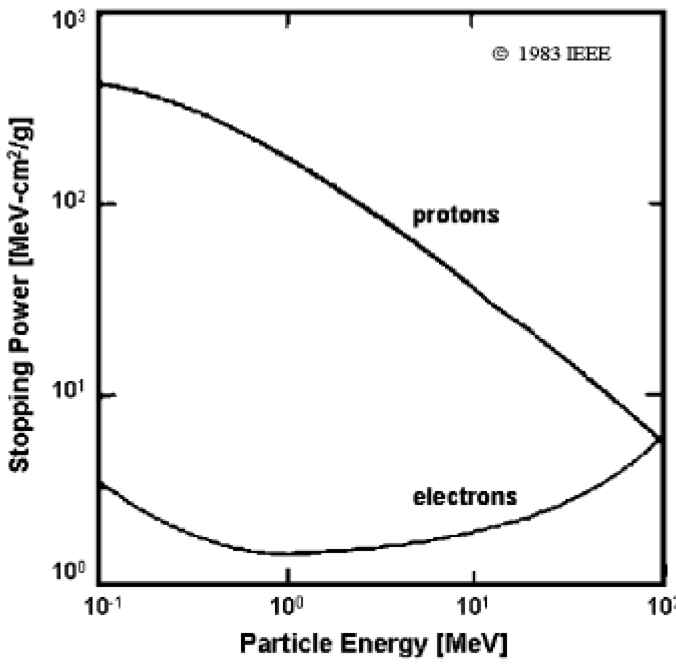


Figure 4: Linear energy transfer for protons and electrons as a function of particle energy [15]

With the mentioned oxide traps and interface traps, degradation of the MOS transistors characteristics occurs. These degradation can be seen in threshold voltage shifts, mobility degradation and induced leakage current.

2.3.1 Threshold voltage shifts

Threshold voltage shifts in MOS transistors can be caused by both oxide traps and interface traps. Equation (2) describes the total threshold voltage shifts.

$$\Delta V_{th} = \Delta V_{ot} + \Delta V_{it}, \tag{2}$$

where ΔV_{ot} is the threshold voltage shift due to oxide charge traps, and ΔV_{it} is the threshold voltage shift due to interface traps. Equation (3) describes how to determine them.

$$\Delta V_{ot,it} = \frac{-1}{C_{ox}\, t_{ox}} \int_0^{t_{ox}} \rho_{ot,it}(x) x dx, \tag{3}$$

where $\rho_{ot,it}(x)$ is the charge distribution of radiation-induced oxide charge traps or interface traps. Threshold voltage shift is negative for positive charges, while for negative charges, the threshold voltage shift is positive [12].

Figure 5 illustrates the effect of oxide charge traps. Their effect is always in the negative direction for both n-channel MOSFET and p-channel MOSFET. For n-channel MOSFET, the oxide charge traps shift the V_{gs} bias point by a negative value. This leads to a reduction in the threshold voltage and an increase of the drive current and the off-state current. For p-channel MOSFET, the oxide charge traps also shift its bias point by a negative value. This leads to an increase in the threshold voltage and a reduction of the drive and off-state currents [15].

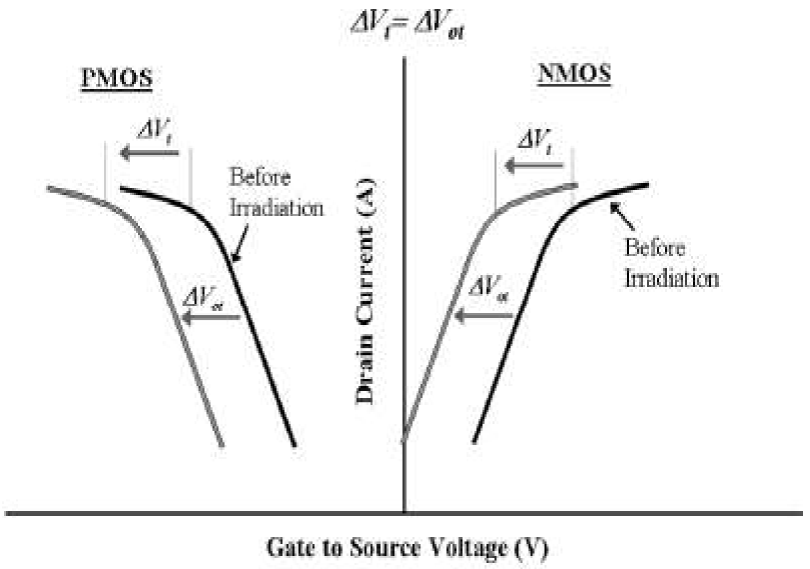
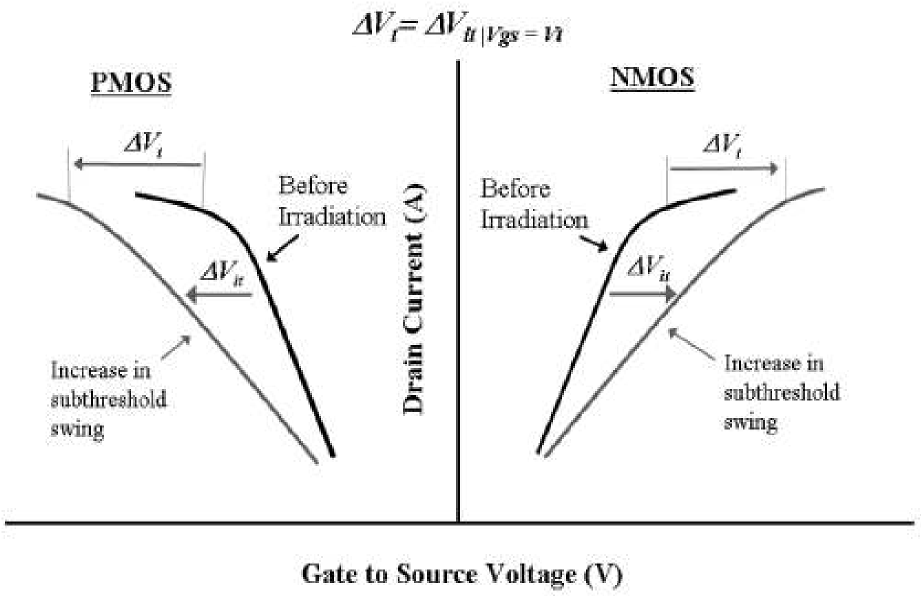


Figure 5: Illustration of the oxide charge traps on n and p channel MOSFETs [15].

Figure 6 illustrates the effect of interface traps. The threshold voltage is also impacted where an increase in the subthreshold swing of both n and p channel MOSFETs are caused by the interface traps. For n channel MOSFET, interface traps cause a shift in the threshold voltage by a positive value, while for a p channel MOSFET, interface traps cause a shift in the threshold voltage by a negative value [15].



$\Delta V_t = \Delta V_{it} |_{Vgs = Vt}$
PMOS
NMOS
ΔV_t
ΔV_t
Before Irradiation
Before Irradiation
ΔV_{it}
ΔV_{it}
Increase in subthreshold swing
Increase in subthreshold swing
Drain Current (A)
Gate to Source Voltage (V)

Figure 6: Illustration of the effect of the interface traps on n and p channel MOSFETs [15]

At high dose rate of radiation for a short time, Oxide charge traps can have a neutralization effect on the threshold voltage shift, while the effect of interface traps won't have enough time to build up. So, the effect of oxide charge traps will dominate leading to a negative and large shift in threshold voltage for both n and p channel MOSFETs. This will affect the n-channel MOSFET badly as it will lead to an increase in the leakage static power of the integrated circuit, which can lead eventually to a failure in the integrated circuit [12].

At moderate dose rates, both oxide charge traps and interface traps can happen to lead to large threshold voltage shifts. For n-channel MOSFET, oxide charge traps lead to a negative threshold voltage shift, while interface traps lead to positive threshold voltage shift, so, they compensate each other, and the failure level of the integrated circuit due to radiation can be relatively high. For p-channel MOSFET, both oxide charge traps and interface traps have a negative threshold voltage shift, so, they add to each other unlike the n-channel MOSFET case [12].

2.3.1.1 Oxide thickness

Oxide thickness affects the radiation induced charge buildup. As oxide thickness decreases, the radiation induced charge buildup decreases rapidly [15]. So, the threshold voltage shift is directly proportional to the oxide thickness by equation (4).

$$\Delta V_{ot} \propto t_{ox}^2 \tag{4}$$

As CMOS technology advances, the oxide thickness decreases so the threshold voltage shift is reduced, as shown in Fig.7. In shallow trench isolation (STI) dielectrics, the thickness is much larger which make STI dielectrics more dangerous in modern CMOS technologies. As a result, interface traps and oxide charge traps in the thin gate oxides are not a concern, and the total dose effects are dominated by oxide charge traps in the field oxides even at a low dose of radiation [12][15].

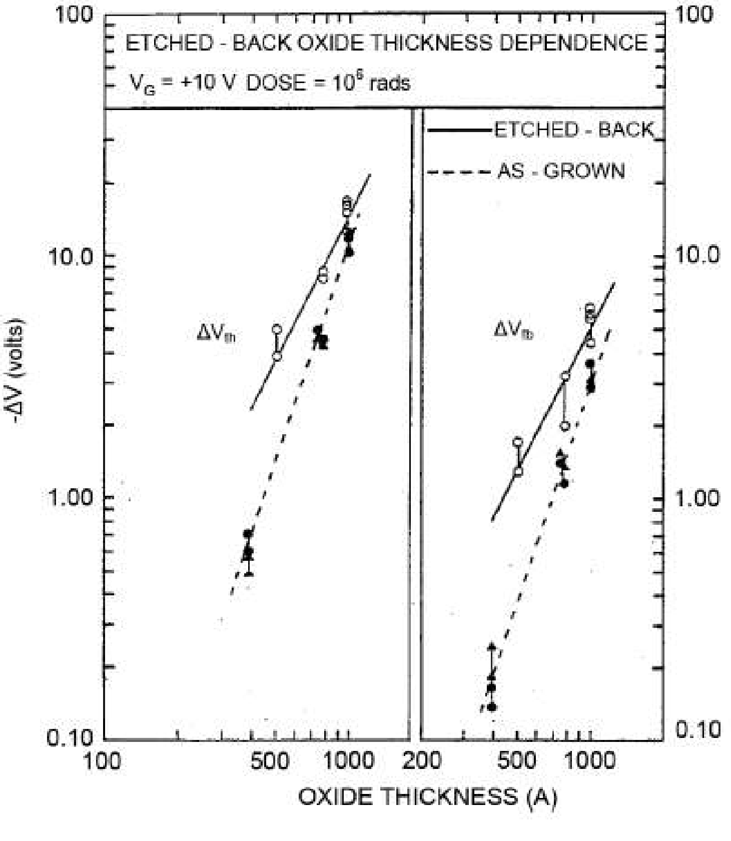


Figure 7: The effect of scaling down of the gate oxide on the threshold voltage shift [17]

2.3.1.2 Transistor dimensions

Transistor dimensions are another factor that can affect the value of the threshold voltage shift. For the transistor width (W), the threshold voltage shift increases with the decrease of the width. This is valid in both n-type and p-type transistors. When the width (W) value is greater than 1 µm, the threshold voltage shift should not be critical. The same is said for the transistor length. The shorter the length (L) of the transistor, the larger is the magnitude of the threshold voltage shift. Also as the length (L) increases, the threshold voltage shift becomes very small. So, as the device size increases, there is no important dependency on the transistor size [18].

2.3.2 Carriers mobility degradation

In addition to threshold voltage shift, Interface traps and oxide charge traps have other effects that can affect MOS transistors. Carriers mobility is one of these effects. Degradation of carrier mobility can cause degradation in the drive capabilities of the MOS transistor. This leads to degrading the timing parameters of an integrated circuit, which may cause timing failure in the electronic circuit operation.

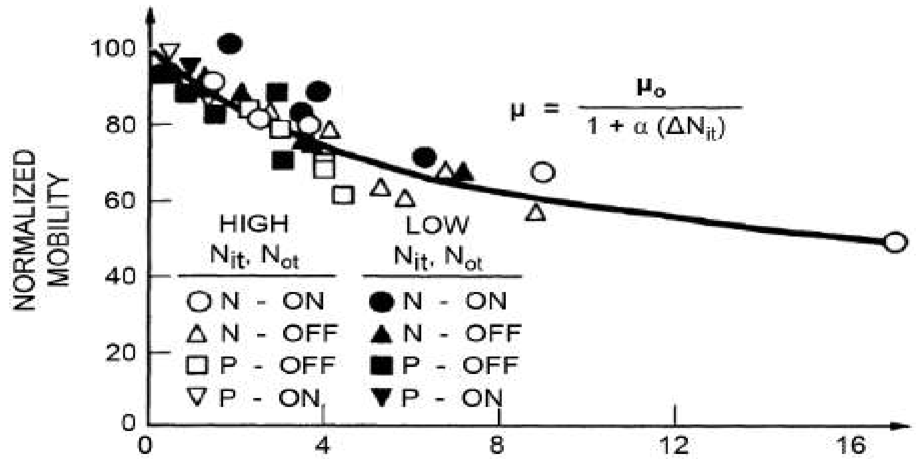


Figure 8: Mobility of carriers normalized to the pre-irradiation values as a function of interface traps density for devices with high and low interface traps and oxide charge traps densities [17]

Although both oxide charge traps and interface affects the mobility of carriers in a MOS transistor, interface traps have a first order effect on the effective mobility of carriers of a MOS transistor as shown in Fig. 8, while oxide charge traps have a much weaker effect. Interface traps are much more efficient when it comes to the scattering of carriers causing mobility degradation of carriers in a MOS transistor. This is because oxide charge traps are more distant from the oxide silicon interface [17].

2.3.3 Induced leakage current

Radiation Induced Leakage Current (RILC) is another form of damage to the CMOS devices resulting from TID. RILC is caused by two mechanisms. The first mechanism is the threshold voltage due to the silicon dioxide charge traps, and the second mechanism is the built-up charges in the field oxide p-substrate. Figure 9 shows the leakage gate current against the gate voltage before radiation and after being radiated to 5.3 Mrad(Si) with Co-60 gamma rays at a gate bias of 0.3 V[12].

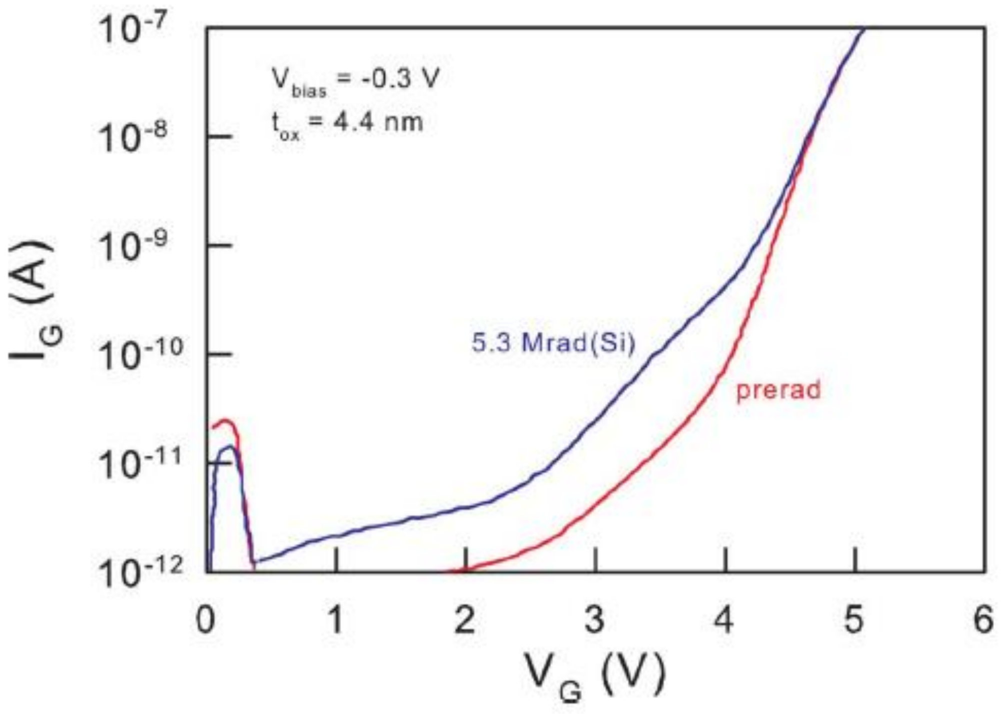


Figure 9: Gate oxide leakage current versus gate voltage for a non-irradiated capacitor and an irradiated capacitor to 5.3 Mrad(Si) [12]

As mentioned in the previous sections, as technology advances, gate oxide becomes thinner so it is less susceptible to radiation damage. However, the field oxides of advanced commercial technologies are still much thicker than the gate oxides. According to equation (4), radiation response increases proportionally with the oxide thickness. Therefore, the effect of field oxides is the most dominant effect of TID and its effect is the main radiation problem of these modern technologies [12].

Induced leakage current in shallow-trench isolation (STI) oxide can cause an increase in the standby current in modern CMOS integrated circuits. Figure 10 shows the increase of the standby current of CMOS shift registers using the technology of commercial 130 nm process [15].

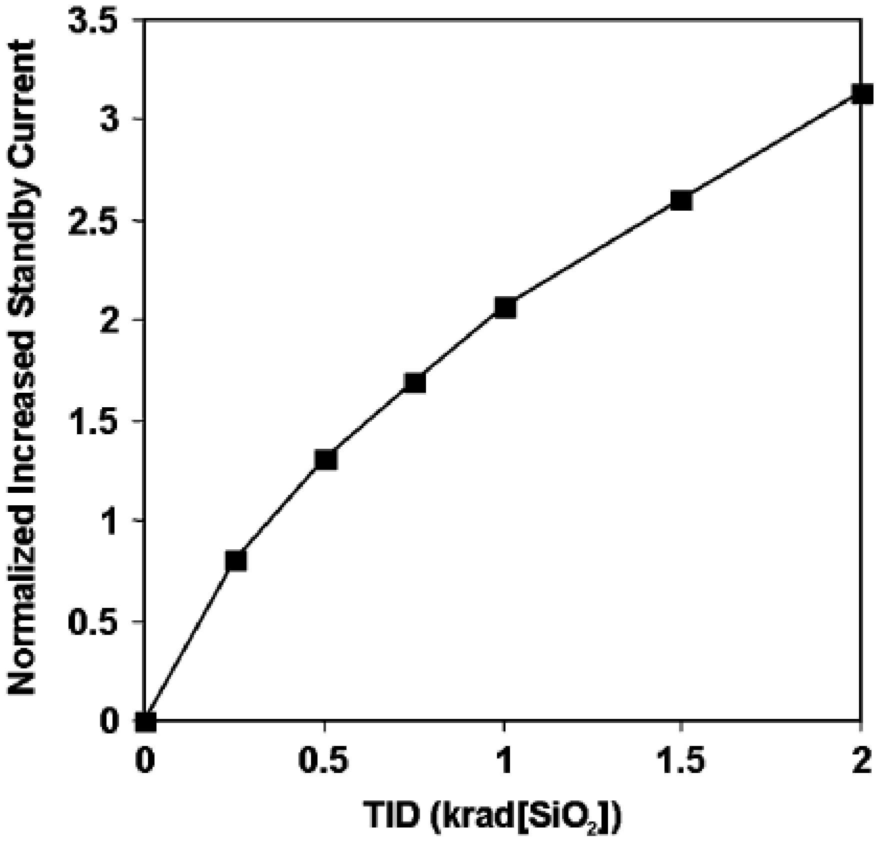


Figure 10: Normalized increased standby current in CMOS shift registers manufactured in 130 nm process as a function of the TID dose in krad [15]

In STI oxide, the built-up charges by the TID effect create leakage paths. This leads to an increase in the standby current in integrated circuits. The most known leakage paths are:

1. In a single n-channel MOSFET, drain to source leakage path.
2. Drain to source leakage between two n-channel transistors
3. Source to a well leakage between different devices.

In all cases, the mechanism is the same. Positively charged oxide traps invert an adjacent p-type silicon layer. This enables the flow of current from one isolation region to another [15].

2.3.3.1 Drain to source leakage

Drain to source leakage happens due to built-up charges in the isolation dielectric at the interface along the sidewalls of the STI oxide. This leakage is the main source for standby current in n-channel MOSFET. Figure 11 shows an illustration of the leakage path.

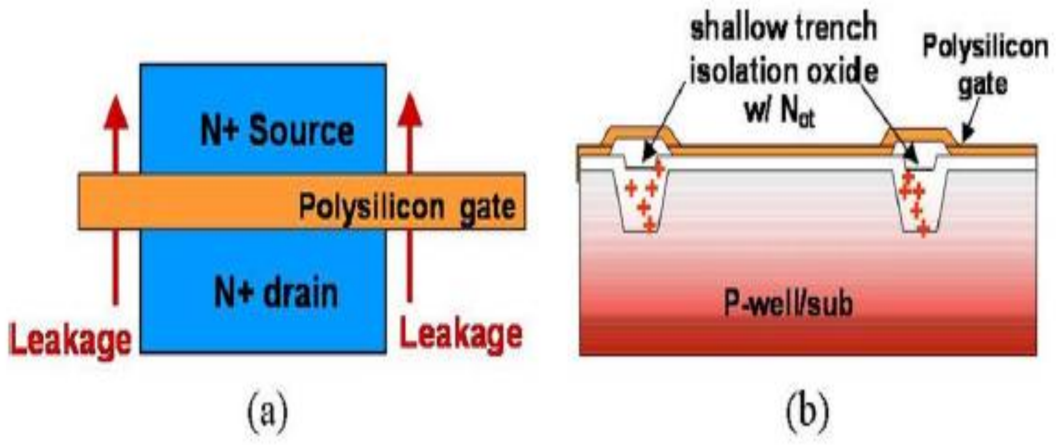


Figure 11: a) Illustration of drain source leakage path in a n-channel MOSFET and b) its cause; built-up charges in the isolation oxide [15]

Figure 12 shows the current-voltage characteristics of an n-channel MOSFET manufactured by a 180 nm process by the Taiwan Semiconductor Manufacturing Company (TSMC). Different radiation doses are shown in the graph. It is noted that above 200 Krad, the drain current increases a lot at the cut off region. At 500 Krad, the drain current becomes 0.1 μA [15]. This is a significant amount of current in the cutoff region.

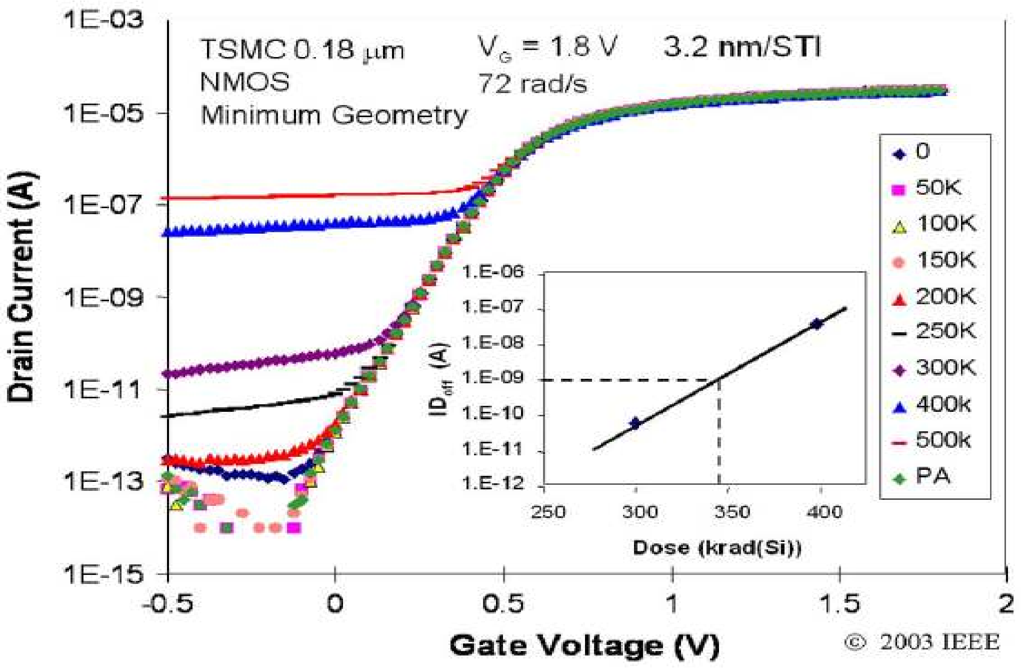


Figure 12: Impact of STI radiation damage on the current-voltage characteristics of n-channel MOSFET fabricated in TSMC 180 nm CMOS process [15]

2.3.3.2 Inter device drain to source leakage

The leakage paths in the case of inter device drain to source leakage are paths between two adjacent n-channel transistors unlike the case of the drain to source leakage in the same device[15]. Figure 13 describes the path of the leaked current.

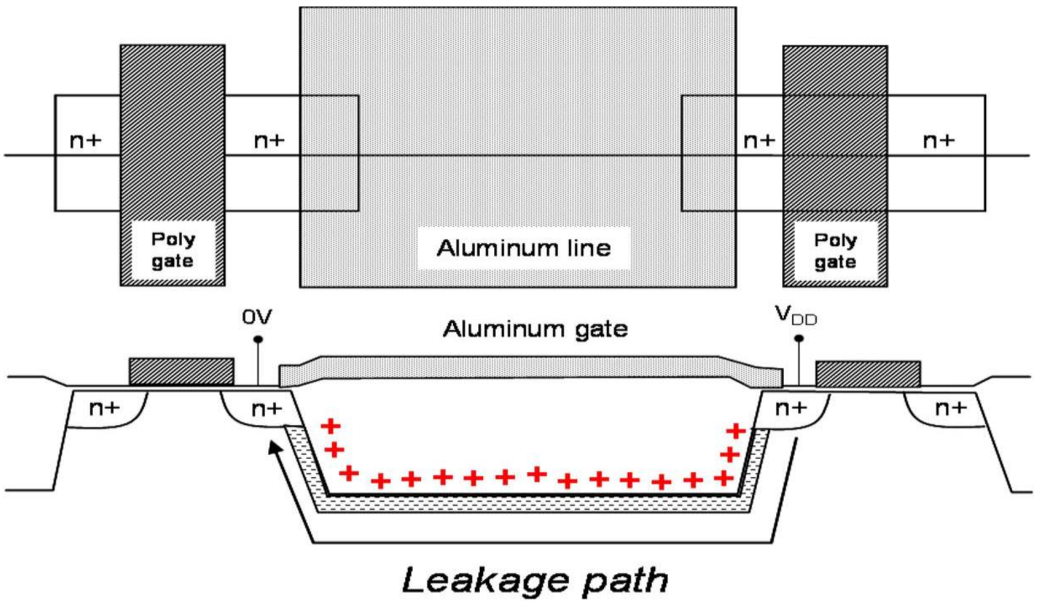


Figure 13: Leakage path between two adjacent n-channel transistors [15]

2.3.3.3 Inter device source to well leakage

Same as inter device drain to source leakage, the leakage paths in the case of inter device source to a well leakage are paths between two different devices. However, the devices here are an n-channel device and a p-channel device. The path is between the n+ drain/source of one n-channel transistor and the n-well of an adjacent p-channel transistor[15]. Figure 14 describes the path of the leaked current.

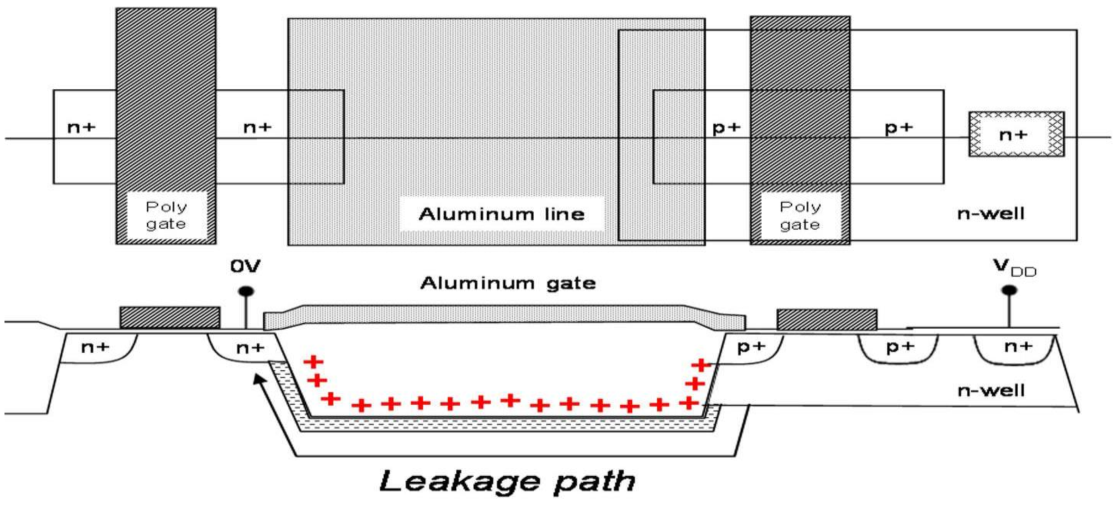


Figure 14: Leakage path between the source of n-channel MOSFET and the n-well of p-channel MOSFET [15]

2.4 TID effect in floating gate MOS

Field programmable gate arrays, usually known as FPGAs, are one of the most important devices in the electronics industry. They are characterized by:

1- high logic density
2- Fast deployment
3- Reprogram ability

If the project is limited in budget and resources or if it is on a tight schedule, then FPGA is the best choice. The switches in an FPGA consists of floating gate MOS. It happens to be that radiation effect on flash-based FPGAs is dominated by floating gate MOS. That is why it is important to study the effects of TID on floating gate MOS [19].

2.4.1 Flash-based FPGA switch

Floating gate MOS transistors are the building elements of flash-based FPGA switch. As shown in Fig.15, the floating gate MOS transistor has two poly-silicon gates. First, there is a tunnel oxide composed of SiO_2, then comes the floating gate, then another layer of inter-poly oxide-nitride-oxide (ONO) composite dielectric and at last the control gate [19].

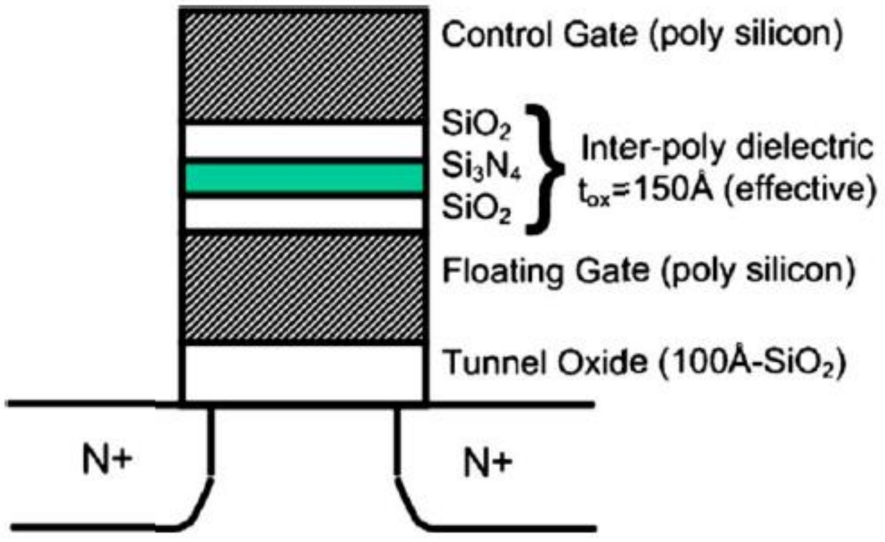

Control Gate (poly silicon)
SiO_2
Si_3N_4
SiO_2
Inter-poly dielectric
t_{ox}=150Å (effective)
Floating Gate (poly silicon)
Tunnel Oxide (100Å-SiO_2)
N+
N+

Figure 15: Schematic of the floating gate transistor in flash-based FPGA [20]

Figure 16 shows that the flash-based FPGA switch consists of two floating gate NMOS transistors: the switch transistor and the program/sense transistor. Both transistors have the same control gate and the same floating gate. The program/sense transistor is responsible for programming the floating gate voltage and sensing the current during threshold voltage measurement, while the switch transistor is responsible for turning on or off the data path.

For setting the threshold voltage, a mechanism called Fowler-Nordheim tunneling is used where charge tunneling occurs through the thin gate oxide leading to storage of charge in the floating gate. This tunneling process is used during programming and erasing of the FPGA. Depending on the threshold voltage, the switch transistor can be turned on or off. To turn the switch transistor on, a low threshold voltage must be programmed in the floating gate, and to turn the switch transistor off, a high threshold voltage must be "erased" in the floating gate [19].

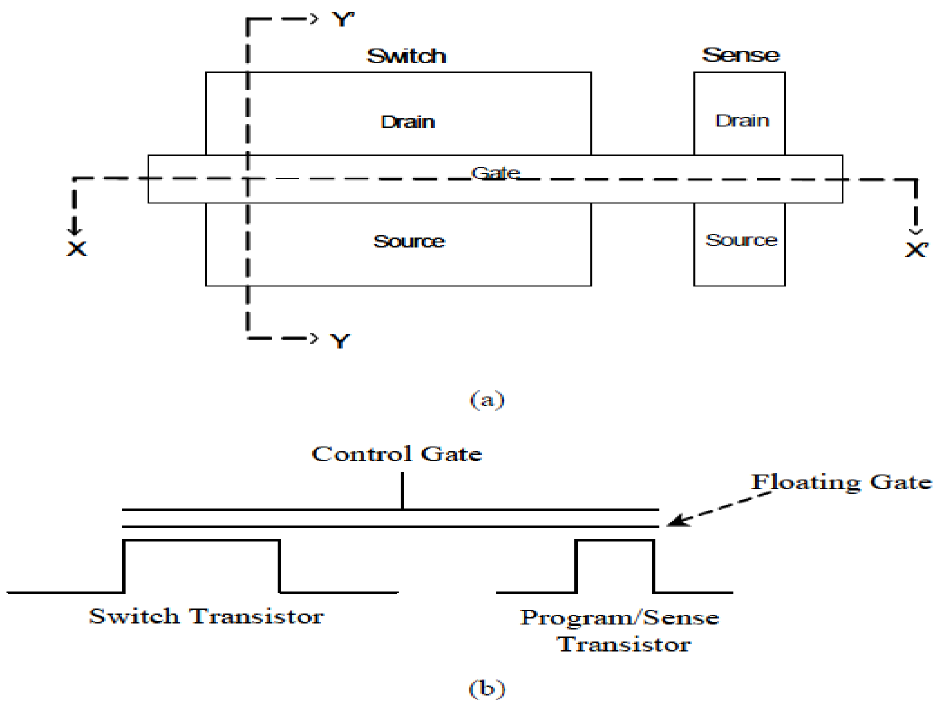


Figure 16: a) Layout of the switch element for the flash-based FPGA. b) Schematic showing the cross section X-X' [20]

The relation between the threshold voltage of the floating gate MOS transistor and the electronic charge stored on the floating gate can be determined as follows:

$$V_{th} = V_{si} + \frac{Q_{fg} d_{ono}}{\epsilon_{ox}}, \tag{5}$$

where V_{si} is the threshold voltage determined by processing and it is a function of many variables including the dielectric thickness; Q_{fg} is the net electronic charge per unit area stored on the floating gate; d_{ono} is the effective oxide-nitride-oxide (ONO) thickness; ϵ_{ox} is the oxide permittivity [19].

2.4.2 Threshold voltage shift

Radiation induces three mechanisms that affect the threshold voltage shift in the floating gate MOS transistor. These mechanisms are:

1- Injection of holes into the floating gate

2- Trapped holes into the oxide

3- Electrons emission over the poly-silicon/oxide barriers.

The first two mechanisms are caused by the generated electron-hole pairs generated from the radiation. The third mechanism, the electron emission, occurs when radiation energy is greater than the potential barrier. Threshold voltage shift is affected by these mechanisms. Equation (6) shows the dependency of the threshold voltage in the floating gate MOS transistor on the radiation induced mechanisms.

$$V_{th}(\gamma) = C_0 + BV_{bias} + [C_1 - BV_{bias}]\exp(-A\gamma) \tag{6}$$

where γ is the total ionizing dose, V_{bias} is the control gate bias, and C_0, C_1, B and A are physical constants [20].

Figure 17 illustrates the I_d versus V_g curves for the high threshold or "erased" state flash cell before and after irradiation. Figure 18 shows the same curves but for the "programmed" state or the low threshold voltage state.

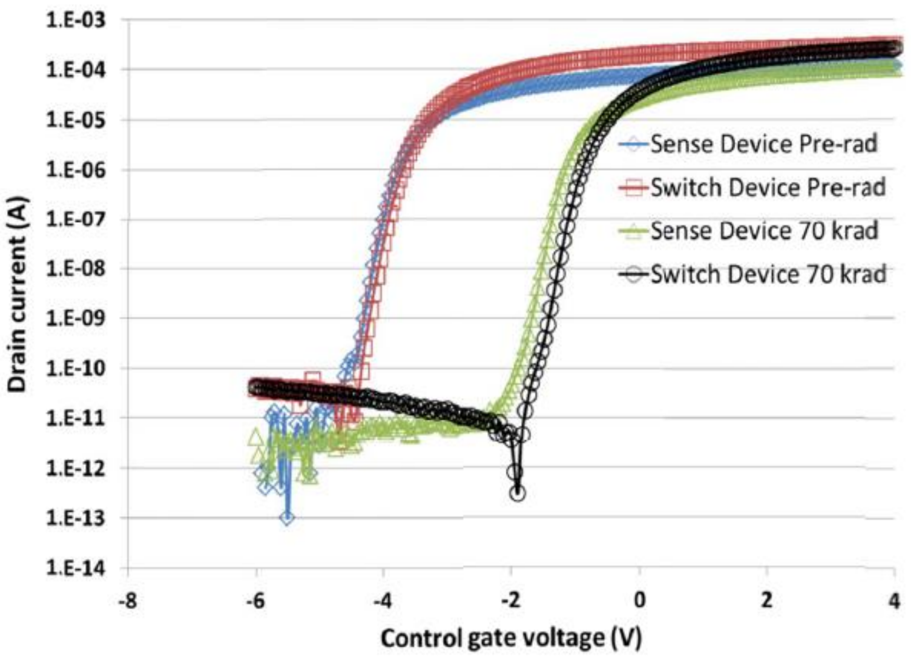


Figure 17: TID effect on the high threshold voltage flash cell [20]

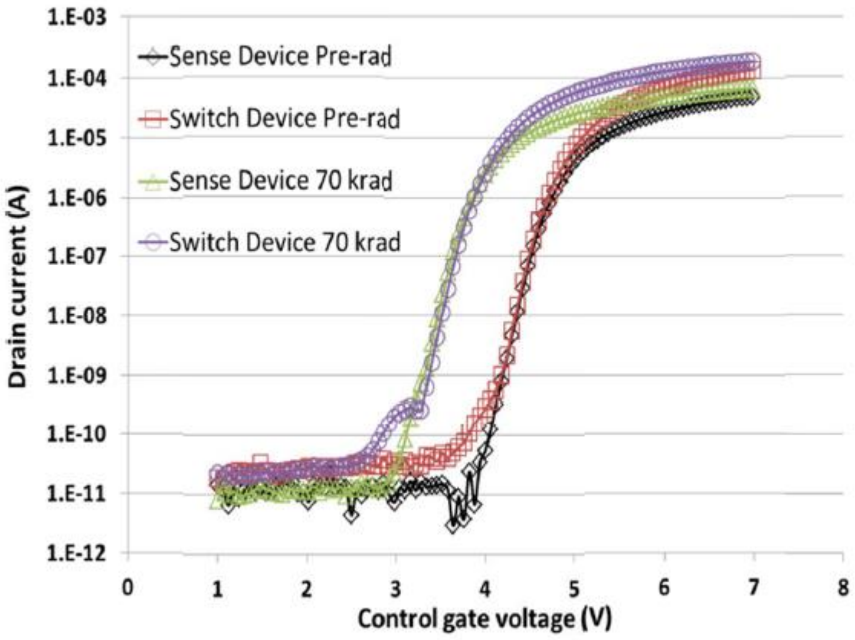


Figure 18: TID effect on the low threshold voltage flash cell [20]

Figure 19 shows the experimental data for threshold voltage with different radiation doses of both "erased" and "programmed" state flash cell compared to data from the model determined by equation (6) [20].

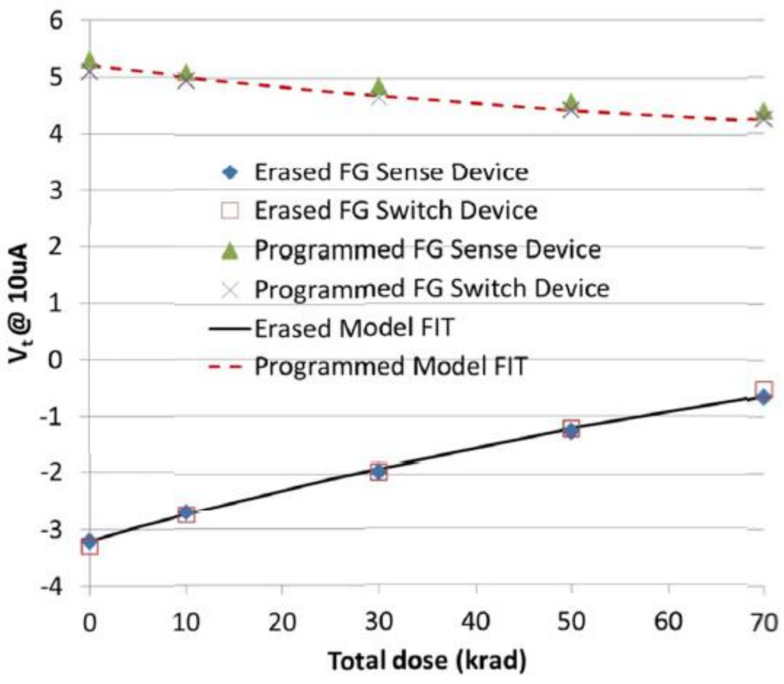


Figure 19: Experiment threshold voltage versus total dose for both low threshold and high threshold voltage flash cells and model prediction (dashed line) [20]

2.4.3 Propagation delay degradation

The degradation in the propagation delay is one of the most important side effects of the threshold voltage shift in the floating gate MOS transistor of flash-based FPGA. An experiment for the TID effect on propagation delay was conducted on a 1000-stage inverter string on a second-generation flash-based FPGA, named ProASIC^{PLUS} APA family. Figure 20 shows the experimental data for the propagation delay for both biased and unbiased cases. The total dose threshold for the unbiased case is higher than that of the unbiased case [19].

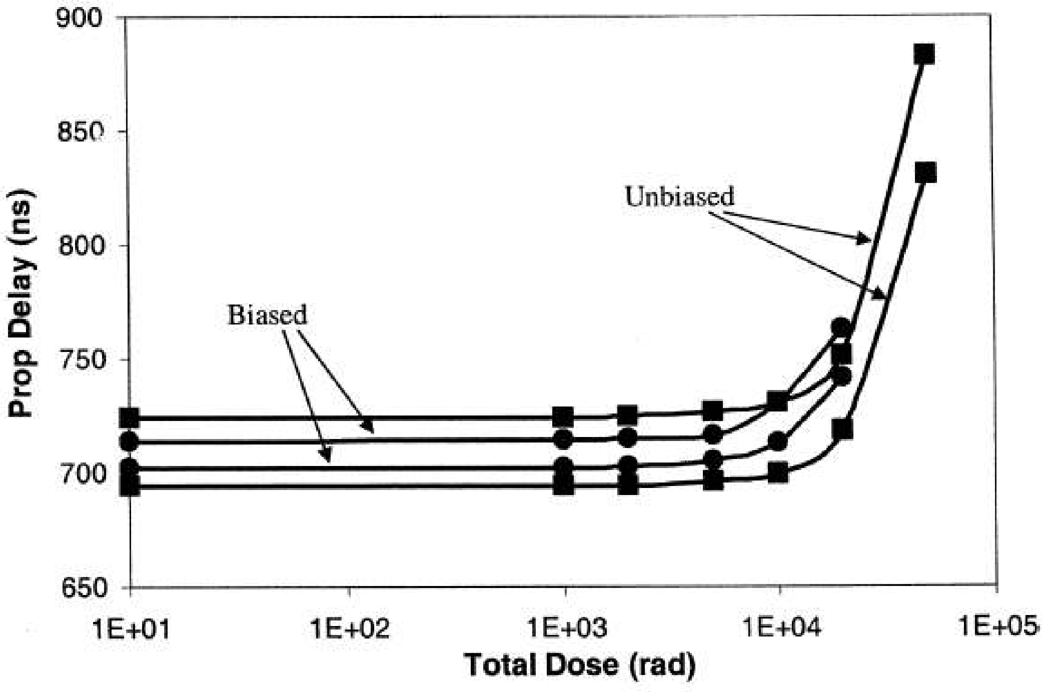


Figure 20: Experimental propagation delay versus total dose for 1000 inverter string [19]

Figure 21 shows the comparison between the spice simulation and the experimental data of the propagation delay for the unbiased case, while Fig.22 shows a comparison between the spice simulation and the experimental data of the propagation delay for the biased case. In the unbiased case, the simulation data is not in a perfect fit with the experimental data unlike the biased case [19].

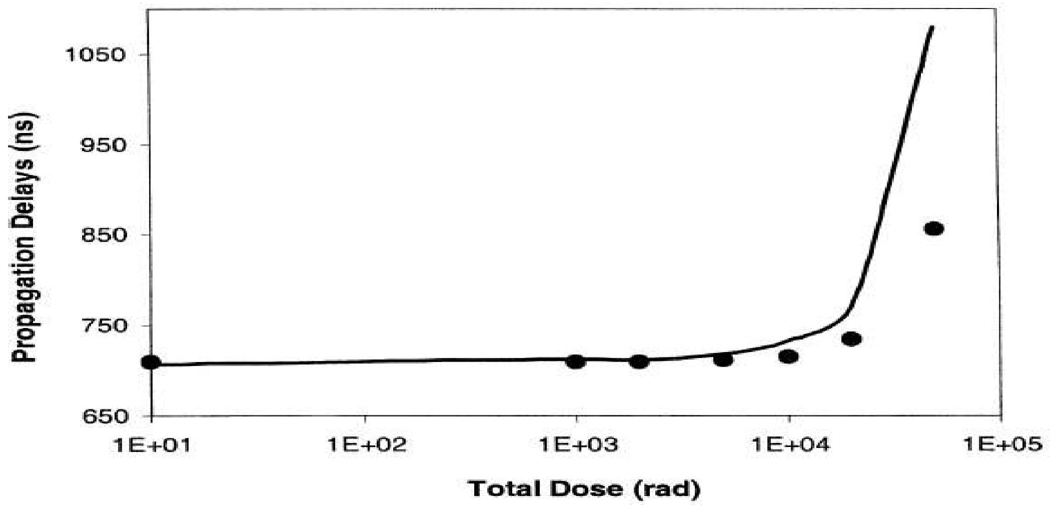


Figure 21: Propagation delay experimental data compared to SPICE simulation predictions for the unbiased case [19]

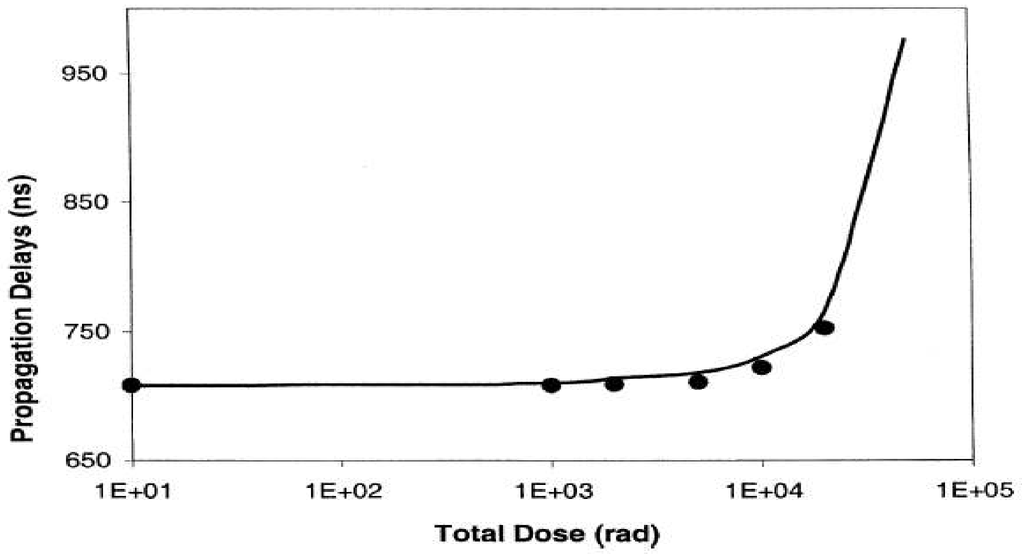


Figure 22: Propagation delay experimental data compared to SPICE simulation predictions for the biased case [19]

The placement and routing of the VersaTiles in the FPGA fabric also affects the degradation in the propagation delay. The study in [21] shows that for TID over 20 krad(Si), the placement and routing of VersaTiles in the critical path plays a significant role in determining the percentage of the propagation delay degradation.

2.5 TID testing

Understanding the nature of testing equipment and the environment is crucial to successfully simulate the real radiation effect. That is why the determination of the nature and specifications of radiation sources is an important issue. Also, the testing techniques should be standardized by a testing procedure to ensure the reliability of the applied test. For total ionizing dose (TID) effect testing, the MIL-STD-883, method 1019 is the standard test that should be followed in TID testing, also Gamma sources (especially cobalt-60 Co^{60}) are commonly used in TID testing.

The MIL-STD-883, method 1019, used in TID testing, emphasizes to use test vectors which should cause the worst radiation effect in the tested devices (worst case test vectors “WCTV”). However, it is very difficult to generate these worst-case test vectors due to the complexity of the designs. Actually, most TID testing don’t use WCTVs due to the difficulty of generating these vectors [7].

Chapter 3

Design for testability

Design for testability is a design methodology where the design phase and the testing phase of a digital circuit are both taken into account during design implementation. This is done to make the testing easier and more meaningful by increasing the observability and the controllability of the circuit. In this chapter, Different methodologies of design for testability are discussed like Ad-hoc designs, BIST designs and scan designs with a special focus on different scan design techniques. Also Path delay testing is discussed and how it is done using scan designs.

3.1 History of design for testability

Although design for testability is a known methodology in design nowadays, however, this was not the case in the early stages in the history of IC implementation. Usually, there were two groups of engineers, one for design and the other for testing. They worked independently, where the design group focused only on getting the design implemented and working correctly from the functionality point of view without any consideration of how it will be tested. The testing group had the job of finding effective ways to test the circuit within a reasonable time.

This approach was sufficient in the era of small scale integrated circuits (SSI). In this time, the circuits were simple combinational circuits or simple finite state machines. As the era of very large integrated circuits (VLSI) approached, the circuits became much more complicated and larger in size with lots of internal states. The previous testing approach was not suitable anymore.

In the early 1980s, a new approach was introduced. Fault coverage of the supplied functional patterns was measured through fault simulation. Designs has long sequential depth so functional patterns were provided to stimulate the internal states. This approach could not reach fault coverage higher than 80%. This means that testing was limited and some issues were not able to be detected. Although designs were good to go from a functionality point of view but performance-wise there was no way to be

100% sure due to the low quality of testing. At this point, Design For Testability (DFT) was introduced [24].

DFT is based on increasing the controllability and the observability of the circuits. Controllability is defined as the difficulty in setting a certain signal to a certain value. Observability is defined as the difficulty in observing the state of a certain signal. So, the goal of DFT is to be able to stimulate and access the internal states of a design in a simple way. The first trials to do this was through ad hoc testability enhancement. However, this method was not effective for large designs. Fault coverage could not exceed 90%. This is because generating test patterns in sequential circuits was much harder than the combinational circuits even after using ad hoc testing enhancements. A large number of internal states made the mission difficult. The solution was to get the sequential circuits to be as close as possible to combinational circuits during the testing phase. This led to the adoption of structured DFT techniques.

The concept behind structured DFT techniques is to be able to control and observe all internal states by providing direct external access to memory elements. These reconfigured memory elements are called “scan cells”. Since every memory element could be accessed and altered, the testing problem for sequential circuits became a simple combinational circuit problem. Many innovative algorithms were already developed to address this problem [24]. Scan designs became the most popular structured DFT technique. To from a scan cell, additional input is added to each memory element called “scan input” and an additional output called “scan output”. By connecting the scan output of a memory element to the scan input of the next memory element, a scan chain is formed. Many architectures for scan designs were introduced based on this idea.

Although scan designs improved quality, diagnostics ability, and testability of designs, the increased cost of traditional testing through automatic test pattern generation (ATPG) makes it inefficient to test deep submicron on nanometer VLSI designs. To solve this, a logic built-in self-test (BIST) is used alongside scan designs. This makes the circuits that generate the test patterns embedded in the chip with the circuit under test. Logic BIST is crucial for safety critical and mission critical applications which can be found in defense/aerospace, automotive, and banking industries [22][24].

3.2 Ad-hoc DFT technique

Ad-hoc DFT technique is not a systematic technique. A lot of ad hoc techniques were used to enhance the testability. Each design has its own local adjustments when using this technique. This means that ad hoc cannot be generalized. Ad hoc increases the testability of the designs but they need to be repeated for each design in a different way[24]. In general, ad hoc technique follows some guidelines that were learned by experience [23]. Some of these guidelines are:

1- Avoid combinational cycles:

   Asynchronous logic feedback causes combinational cycles which leads to having race conditions in the circuits. The automatic test pattern generation programs do not work in the presence of the combinational cycles. This is because test generation algorithms are only known to work with acyclic combinational circuits.

2- Avoid the large number of fan-in:

   Observability and controllability decrease significantly as the number of inputs of the gates increase. It becomes harder to control the output of the gates.

3- Initialization of flip flops:

   To be able to control the values of the internal states in the design, it is very useful to add set and reset input pins to flip flops to be able to initialize their values.

Test point insertion (TPI) is one of the most famous ad hoc DFT techniques. Testability analysis determines the points in the design that needs to be observed and controlled. TPI logic is inserted at these points. Example of how observation point insertion is done is shown in Fig.23. In this example, the design has three low observability points that need TPI. The signal from each of these points is connected to a multiplexer input 0 and the multiplexer output is connected to a register as shown at point OP_2. The registers of all the observation points are connected serially through the other multiplexer input 1 to for a shift register. The multiplexer selection signal is called scan enable (SE). The observation operation is done through two steps. At the first clock cycle, the SE signal is 0 to select the value of the observation point and load it into the register. In the second clock cycle, the SE signal is 1 to select the input from the previous register and the registers together operate as a shift register where the output of the last register is connected to an output port from the design (OP_output) [24].

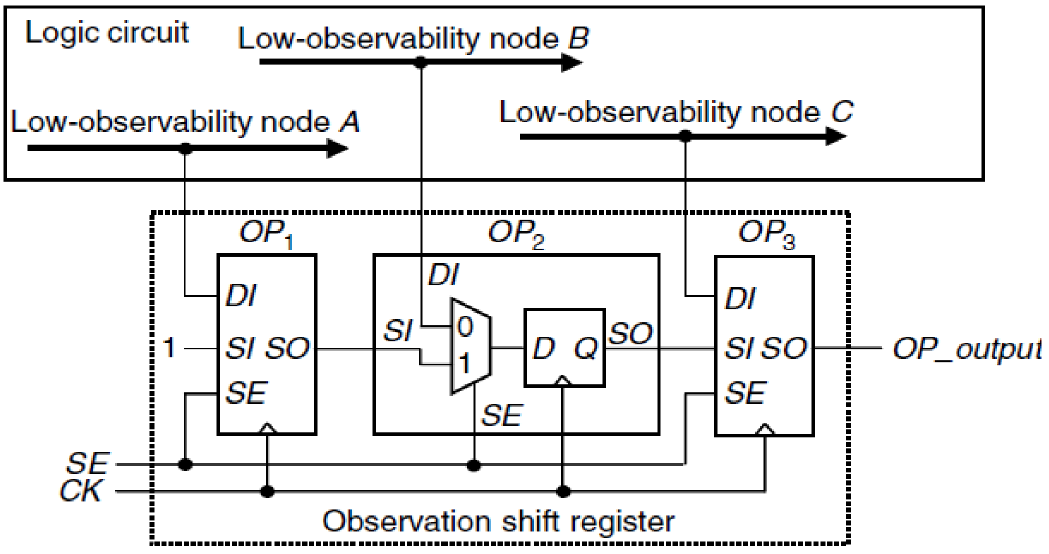


Figure 23: Example of an observation point insertion [24]

Example of how to control point insertion is done is shown in Fig.24. In this example, the design has three low controllability points that need TPI. The nodes that need to be controlled are cut and reconnected through a multiplexer on input 0. The other input of the multiplexer 1 is connected to a register as shown at point CP_2. The registers of all the control points are connected to each other to form a shift register. The multiplexer selection signal is called test mode (TM). The control operation is done through two steps. At the first, the test mode signal is selected to be 1 to connect the registers as shift registers. The inputs are loaded to the shift registers accordingly where there is a primary input signal connected to the first register called CP_input. The second step is to set the test mode signal to 0. In this case, the control nodes take their inputs from the registers and not from the original circuit. The drawback of this architecture is the increase in the delay of the circuit. That is why care must be taken in choosing the placement of controlling points, especially in the critical path. Also, it is recommended to insert a "scan point" which is a combination of control and observation point to be able to observe the source end as well [24].

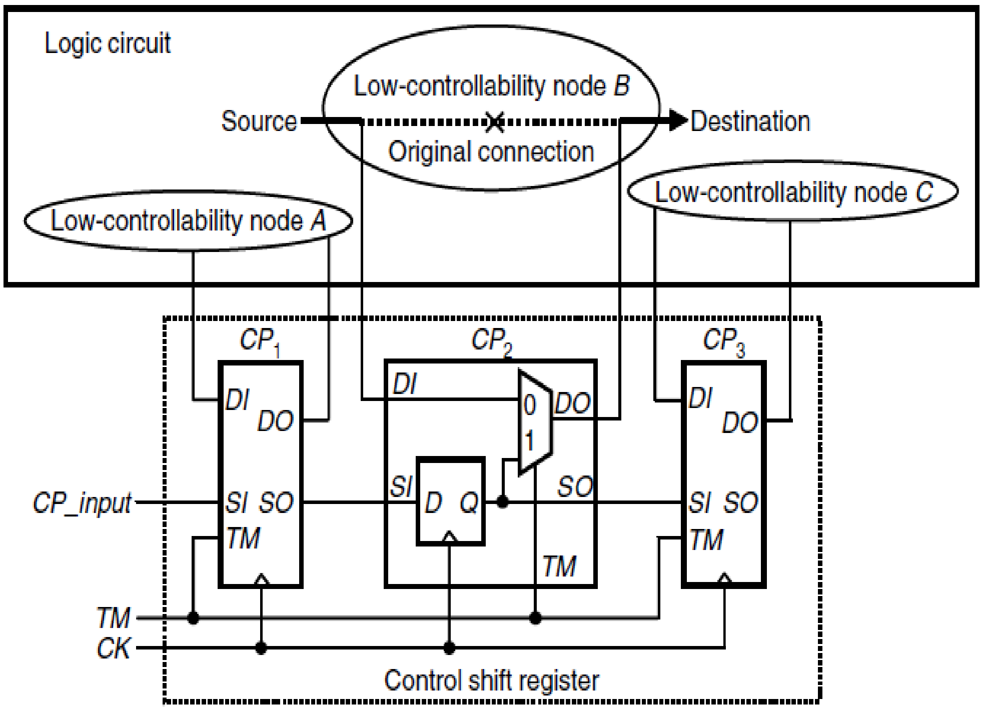


Figure 24: Example of a control point insertion [24]

3.3 Scan design

Ad-hoc DFT techniques have the problem of being local to each design and non-systematic. That is why structured DFT techniques are needed. Structured DFT techniques follow a systematic and methodical process. This allows for the testing process to be easily incorporated and budgeted for as part of the design flow, yielding the desired results. Another important aspect is the automation. Structured DFT techniques can be automated. Electronic Design Automation (EDA) companies provide tools to automate DFT tasks. One of the most popular and effective structured DFT technique is the scan design, which can achieve the targeted high fault coverage [24].

The most known and widely used structured DFT technique is the scan design. In scan design, all the registers are replaced with what is known as scan cells. These scan cells are just memory elements with additional input called "scan input" and additional output called “scan output”. When the scan output is of one cell is connected to the scan input of another cell, a scan chain is formed. In scan designs, there can be multiple scan chains. During operation, scan designs have three modes: normal mode, shift mode, and capture mode. Normal mode is the mode in which the circuit behaves normally without considering any testing logic. In shift mode and capture mode, all testing-related logic is turned on [22]. Many scan architectures have been proposed, some of which are described in the following subsections.

3.3.1 Muxed D scan design

Muxed D scan design architecture is the most famous scan design architecture. In this architecture, registers in the design are replaced with muxed D scan cells. Figure 25 shows the implementation of the muxed D scan cell. It consists of a multiplexer and a D flip flop. The multiplexer inputs are data input (DI), and scan input (SI). The scan enable (SE) is the control signal for the multiplexer [22].

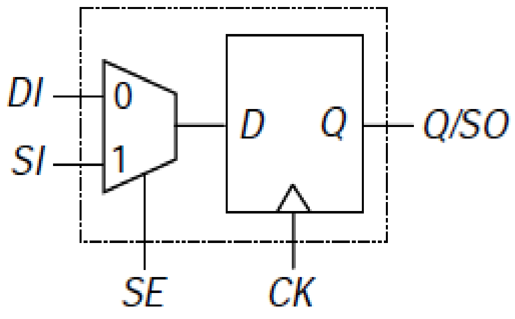


Figure 25: Illustration of a muxed D scan cell [22]

An example on how muxed D scan is implemented is shown in Fig.26 and Fig.27. In Figure 26, the original circuit has three registers FF_1, FF_2, and FF_3. Their inputs and outputs are connected to the combinational logic. In Figure 27, these registers are replaced with the scan cells SFF_1, SFF_2, and SFF_3 respectively. The DI ports of the scan cells and the output ports are connected to the combinational logic like the original circuit. The SI port of each scan cell is connected to the output of the previous scan cell. For the first scan cell in the scan chain, input SI is connected to a port. Also for the last scan cell in the scan chain, the output is connected to scan out port SO.

For normal operation mode, scan enable signal (SE) is 0 to select the data input to the mux. For testing, signal SE is connected to 1 so that the scan chain is connected as a shift register. This is called a shift mode. Inputs are fed through the scan input port and shifted till the scan chain is fully loaded. After that, the scan enable signal is set to 0 and capture mode is on. With the next clock cycle, the scan registers content is loaded into the design and the test response in captured from the combinational logic [22].

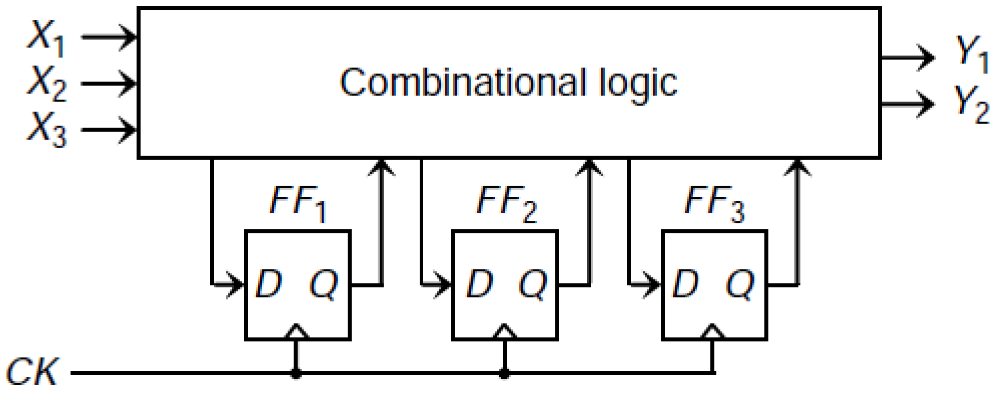


Figure 26: An example of a sequential circuit [22]

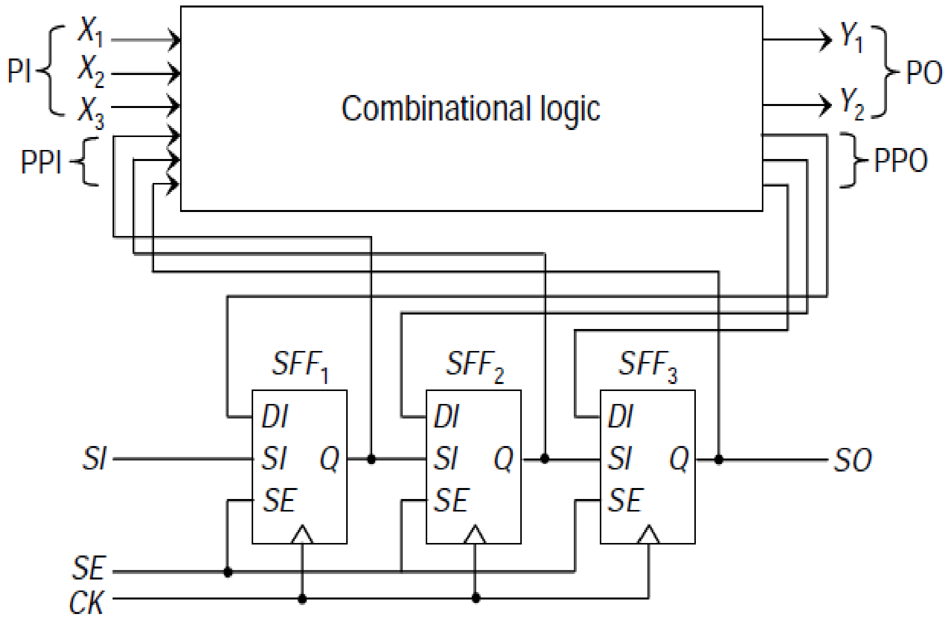


Figure 27: Corresponding muxed D full scan circuit of the sequential circuit in Fig.26 [22]

As shown in Fig.27, the inputs of the registers from the combinational logic are called pseudo primary output (PPO) while the output from the registers into the combinational logic is called pseudo primary input (PPI) [22].

The main issue with the muxed D scan design architecture is the addition of delay in the path of the normal operation of the circuit due to the added multiplexers in the scan cells.

3.3.2 Clocked scan design

Another scan design architecture is the clocked scan design. In this architecture, the registers in the design are replaced by clocked scan cells. Figure 28 shows the block diagram of the clocked scan cell. It is like a normal D flip flop but with two inputs: data input DI and scan input SI. Also, instead of a single clock, two clock sources are needed. The purpose of using two clocks is to select between the two inputs where data clock (DCK) is used to select the data input and the scan clock (SCK) is used to select the scan input [22].

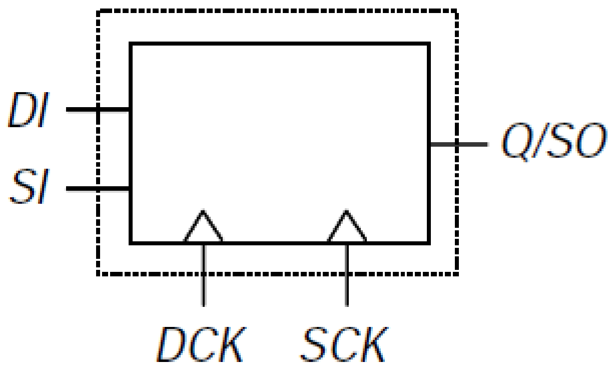


Figure 28: Illustration of clocked scan cell [22]

Figure 29 shows an example of the waveforms of the clocked scan cell. At the positive edge of the data clock DCK, the data input is transferred to the output pin. This is how normal mode works. In the case of testing, shift mode is applied by selecting the scan input at the positive edge of the scan clock SCK. Afterward, the capture mode is applied by capturing the combinational circuit outputs at the positive edge of the data clock DCK [24].

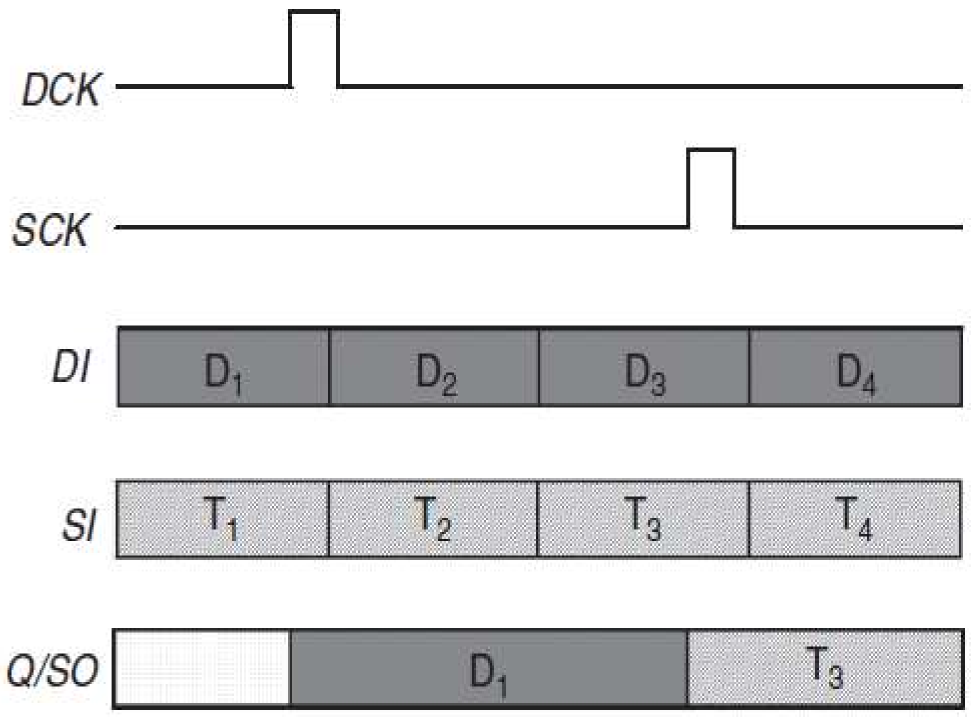


Figure 29: Example of the waveform of the operation of the clocked scan cell [24]

The same circuit example in Fig.26 is used to show how clocked scan design technique is implemented in Fig.30. The three registers in Fig,26 are replaced by the clocked scan cells in Fig.30. Figure 30 shows the corresponding clocked scan design of the sequential circuit in Fig.26. The difference between the two architecture is that in muxed D scan design, a scan enable (SE) signal is used to distinguish between the test and normal operations, while in the clocked scan design, two independent clock sources, data clock (DCK) and shift clock (SCK), are used to distinguish between these two operations [22].

The advantage of the clocked scan design technique over the muxed D scan design technique is that no added delay is needed in the path of the data in the normal operation due to eliminating the usage of the multiplexer. However, clocked scan cell requires additional clock routing for the shift clock source [24].

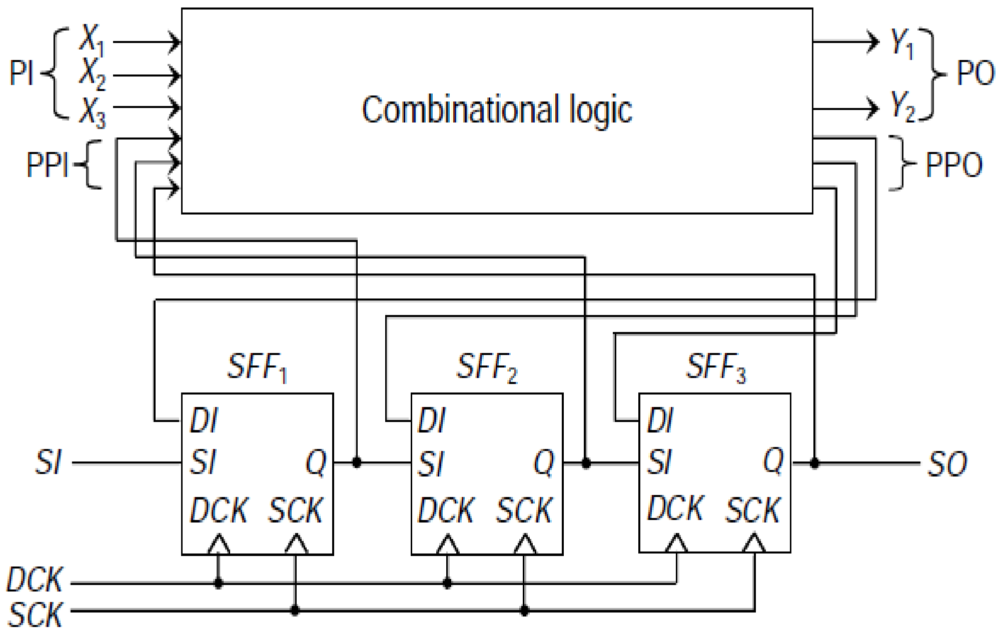


Figure 30: Clocked scan full design of the sequential circuit in Fig.26 [22]

3.3.3 Enhanced scan design

This scan design technique is built on either muxed D scan design technique or clocked scan design technique. The simple idea is to add another level of latches after the normal scan cell which leads to increase the capacity of a normal scan cell. The goal behind this is to be able to store two data vectors and apply them to the circuit in an at-speed manner [22].

The enhanced scan design technique is very useful when testing the path delay fault. It is possible to apply two completely independent and arbitrary vectors to the circuit under test. This will increase the detection capability of the delay fault. Other scan design techniques do not have this flexibility. They can only use two functionally dependent vectors, which are generated from the combinational logic in testing path delay fault.

The same circuit example in Fig.26 is used to show how the enhanced design technique provides an additional degree of freedom, as shown in Fig.31. Figure 31 shows the normal muxed D scan cells followed by another level of latches. The control signal for all the latches is the same and is called UPDATE. To apply a pair of test vectors <V_1, V_2>, the first vector V_1 is shifted in the muxed D scan cells of the design by setting SE signal to 1. Then it is stored in the D type latches, by making the update signal set to a value of 1. After that, the update signal is set to a value of 0 to keep the values of the first vectors in the latches. The second vector is then shifted into the scan cells of the design the same as the first vector. Then, the update signal is set again to a value of 1 to change the stored value in the D type latches from V_1 to V_2. This is done while applying the clock source after exactly one clock cycle to capture the output response of the test vector in an at speed manner [22].

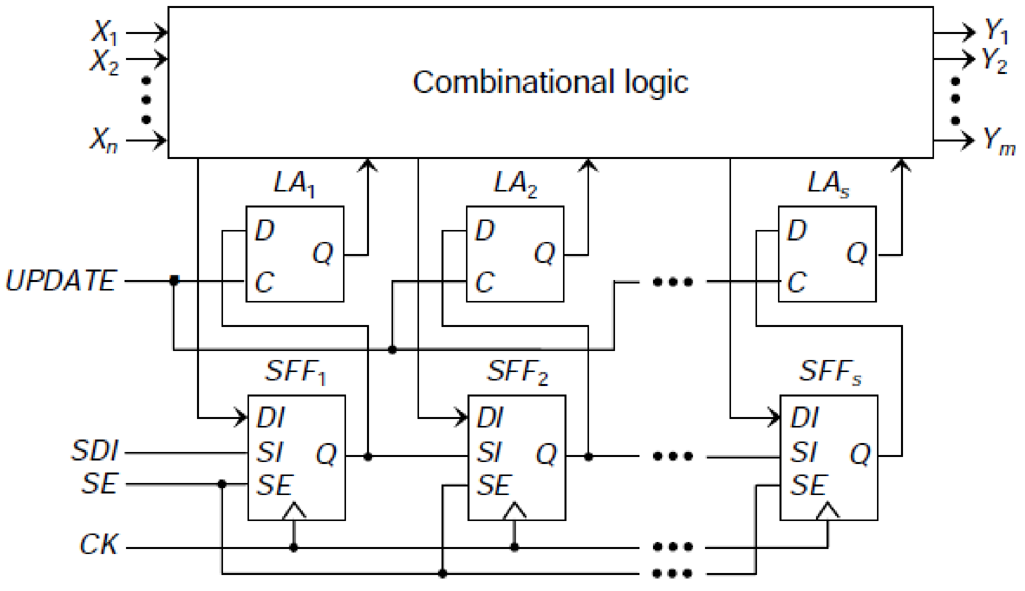


Figure 31: An example of an enahnced scan design [22]

Increase of path delay fault coverage is the main advantage of enhanced scan design. This is done by adding the degree of freedom of applying two independent pair of vectors to test delay fault possible. However, the drawbacks for this technique are not to be neglected. The addition of a D Type latch means more delay in the path of data in normal operation. Also, it may be difficult to maintain the timing between the update signal and the clock source in the testing operation. Another issue is the problem of over-testing that may result from activation of many false paths during the test operation, instead of functional data paths. To better handle these issues, there are lots of other delay fault techniques that avoid using additional latches like launch-on-shift (also called skewed load) and launch-on-capture (also called broadside) [22]. It is worth mentioning that Intel Pentium 4 processor used the enhanced scan design technique in its testing.

3.4 Logic Built-in self-test (BIST)

The idea behind logic built-in self-test (BIST) is to have all the needed circuits for testing and verifying the circuit under test on the same chip or at least on the same board. Figure 32 illustrates the system level design of a logic BIST system.

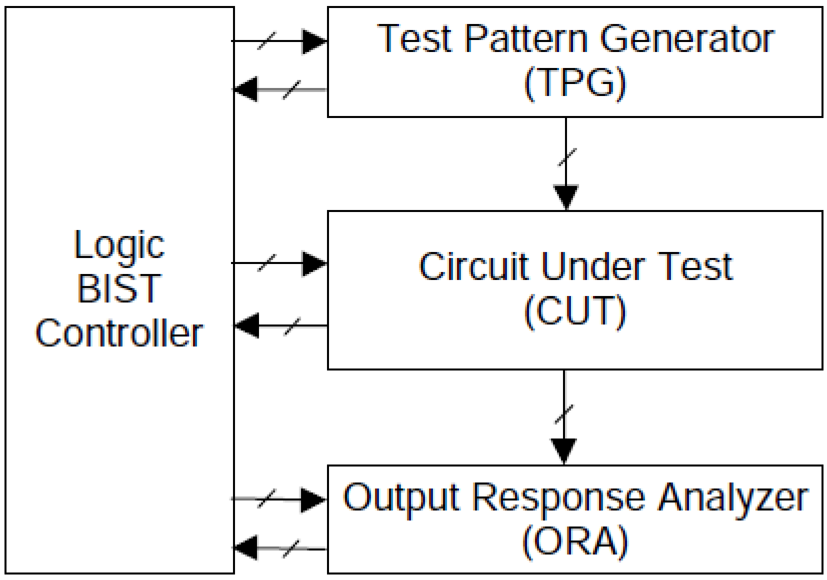


Figure 32: Common logic BIST system [22].

The system consists of four main modules. The first module is the Test Pattern Generator (TPG) which is responsible for automatic generation of test patterns to the circuit. The second module is the Circuit Under Test (CUT). The third module is the Output Response Analyzer (ORA). This module is responsible for putting the output responses of CUT in a compact form. The fourth module is Logic BIST Controller which is responsible for generating BIST timing control signals like scan enable signals and clocks to coordinate the operation of the other units TPG, CUT, and ORA [22].

Automatic test pattern generators usually use Linear Feedback Shift Registers (LFSRs) for this mission, whatever the type of testing is. It can be used for pseudo random testing, pseudo exhaustive testing, and exhaustive testing. Exhaustive testing is the type of testing where all possible combinations should be examined. In the case of a CUT having n inputs, it is required to apply all 2^n test patterns possible to do exhaustive testing. This guarantees single stuck fault coverage of 100%. The drawback is the large testing time required if a circuit has a large number of inputs. That is why pseudo random testing can be used instead. In this type of testing, a subset of random 2n test patterns is generated. The fault coverage in this type is not 100% and should be measured for each design to know the exact fault coverage. A solution in the middle is to test using pseudo exhaustive testing. In this type, the TPG generates 2^w test patterns, where $w< n$ and each output of the CUT depends at most on w inputs. By this way, it is possible to maintain 100% fault coverage [22].

Compaction of outputs from the CUT in the ORA is usually done by Multiple Input Signature Registers (MISRs). It is similar to LFSRs but with their inputs are connected to an XOR gate. This is done to reduce the hardware overhead in the ORA [22].

3.5 Path delay testing

As technology advances, the device dimensions scale down. This leads to defects related to timing. This type of defects is newly introduced and poses new challenges for test engineers. In the earlier days, the stuck-at fault model was enough for testing. This model checked if the designated signal has been set to the value of 1 or 0 and cannot be changed or takes a very long time to change its value from 0 to 1 or vice versa. However, this check is not enough in new technologies. In new designs, very small delay defects can happen due to process variation. Therefore, there is a need for a new fault model that should address these small defects. It should be characterized by the ability to be applied in an at-speed manner to ensure proper operation of the device under the test.

Path delay fault model is the new model that is introduced to detect accumulated delay defects in critical paths of the design. The scaling down of dimensions causes more delay variations so normal static timing analysis (STA) tools cannot fully address the effect of the defects. So the need for the path delay fault model is inevitable to ensure the proper operation of a circuit within the range of the operating frequency. This will result in the end in an increase in the quality of the manufactured devices [23][25].

3.5.1 Path delay classification

Unlike stuck-at faults, the detection of path delay faults is dependent on the test vector applied. This means that not all test vectors can detect the path delay fault. Some test vectors are independent and can guarantee the fault detection without being affected by the surrounding conditions. Other test vectors can detect the faults depending on the conditions of the surroundings. Path delay faults are classified to four main categories according to the sensitization criteria [26]:

1- Robust
2- Non-robust
3- Validatable non-robust
4- Functional sensitizable.

Before diving into the details of each category of the path delay faults, some important terminologies should be defined. “Controlling value” of the gate is the gate input value that determines the output value of the gate regardless of the values of other gate inputs. “Non-controlling value” is the complement of the “controlling value” for a designated gate. By taking the AND gate as an example, the controlling value for the gate is “0” because the value of the other input will not matter as in all cases the output value is “0”. The non-controlling value is “1”, the complement for the controlling value. Another example is the OR gate. It has a controlling value of “1” and the non-controlling value is “0”. Another terminology is “on-input” signal, it is a signal affecting path P that exists on the path P, while an “off-input” signal for a path P is a signal which resides in an input to a gate in path P but not an “on-input” signal.

3.5.1.1 Robust path delay faults

Robust paths are the paths on which you can detect the fault independently of the delays of “off-input” signals. Some conditions must be met to ensure robust sensitization of path delay fault. Consider a sequence of test vectors V_1 and V_2 applied to the circuit. If the on-inputs to gates on the path are transitioning from the controlling value to the non-controlling value, then the off-inputs can have either controlling or non-controlling values while applying the first vector V_1 but must have non-controlling values while applying vector V_2. If the on-inputs to gates on the path are transitioning from non-controlling values to controlling values, then the off-inputs must have non-controlling values during applying both vectors V_1 and V_2.

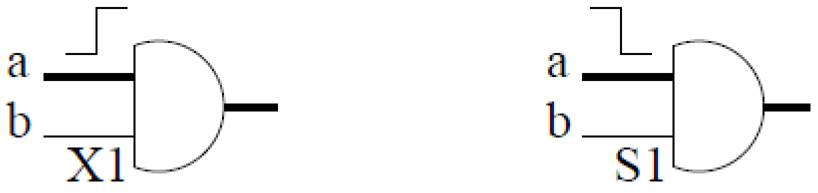


Figure 33: Robust sensitization criterion for an AND gate [26]

By taking the AND gate shown in Fig.33 as an example, input “a” is the on-input while input “b” is the off-input. In the first case, the signal on input “a” is transitioning from “0” to “1”, in other words, from a controlling value to a non-controlling value. In this case, off-input “b” can have either “0” or “1” before the transition but must have a value of “1” as the non-controlling value during the transition. That is why “X1” is the value of the “b” input. “X” indicates either “0” or “1” before the transition during first test vector V_1 and “1” is the must-have value during the transition when applying vector V_2. In the second case, the on-input “a” is transitioning from a non-controlling value to a controlling value, so input “b” must be a stable non-controlling value which is “1” in this case. “S1” on the “b” input indicates a stable value of “1” before and during the transition while applying both vectors V_1 and V_2.

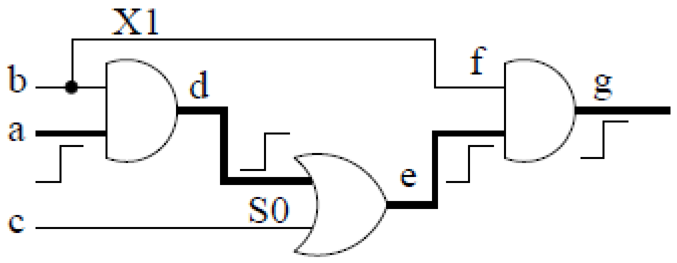


Figure 34: An example of a robust testable path delay fault [26]

Robust testable path delay faults are characterized by the existence of a sequence of test vectors that activates the required transition on the target path to be tested, also satisfying the early mentioned conditions for the off-inputs for every gate in the targeted path. Figure 34 shows an example of a robust testable path delay fault. Path (a, d, e, g) is the one under test. There are 3 off-inputs in the circuit “b”, “f” and “c”. For “b” and “f”, they are both inputs to AND gates, where the on-inputs in both cases are transitioning from a controlling value to non-controlling value. Accordingly, the off-inputs “b” and “f” can have any value during applying the first test vector while they should have a non-controlling value during applying the second test vector. For off-input “c”, it is an input to an OR gate where the on-input is transitioning from non-controlling value to controlling value. That is why the value for “c” should be a stable “0” during applying first and second test vectors. [26].

3.5.1.2 Non-robust path delay faults

Non-robust paths are the paths on which you can detect the path delay fault depending on the delays of the off-input signals. The conditions for non-robust sensitization are less strict than robust sensitization.

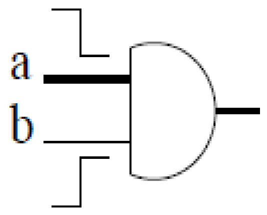


Figure 35: An example of non-robust sensitization of an AND gate [26]

Figure 35 shows an example of non-robust sensitization of an AND gate. “a” is the on-input, while “b” is the off-input. The transition on “a” is from non-controlling value to the controlling value. However, the value of the off-input “b” is not a steady non-controlling value like the case of robust sensitization. It is a transition from controlling value to non-controlling value. This means that the fault is observable depending on when exactly the transition of the off-input occurs. If the transition of the input “b” arrives before the transition of input “a”, then the off-input will mask the on-input transition to the output and the fault will go undetected. On the other hand, if the off-input transition arrives after the transition of the on-input, the fault will be detected at the output [26].

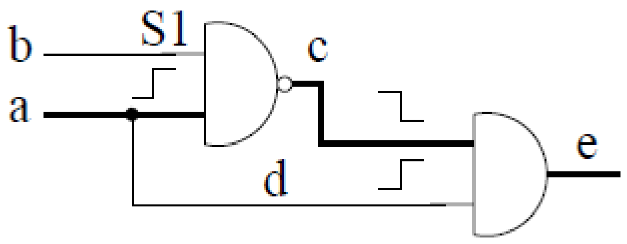


Figure 36: An example of a non-robust testable path delay fault [26]

Non-robust testable path delay faults are characterized by the existence of both a sequence of test vectors that activates the required transition on the target path to be tested and at least one off-input signal satisfies the early mentioned condition in the targeted path. Figure 36 shows an example of a non-robust testable path delay fault. The path is (a, c, e) and the off-input “d” shows the kind of the transition causing the path to be non-robust. If the transition at “d” arrives before the transition at “c”, then the transition at “c” will be masked at the output “e”. If the transition at “d” arrives after the transition at “c”, then the fault will be observable at output “e” [26].

3.5.1.3 Validatable non-robust path delay faults

This type of path delay faults is the same as non-robust path delay fault except for a small difference. To explain this type, consider the example at Fig.36, transition at signal “d” is the one causing the path to be non-robust as it can mask the fault propagation of path (a, c, e). If the path of signal “d”, which is a robust path, is tested and found to be faulty, then the circuit is faulty and no need for the testing path (a, c, e). However, if the path of signal “d” is not faulty, then the transition at “d” arrives before the transition at “c”. This makes the non-robust path validatable as the fault is observable at “e” [26].

3.5.1.4 Functional sensitizable path delay faults

Functional sensitizable path delay fault is similar to non-robust path delay fault in the fact that it depends on the delays of the off-input signals of the path to be tested. However, in this type, there must be at least one gate along the path where both the on-inputs and the off-inputs are transitioning from non-controlling value to controlling value.

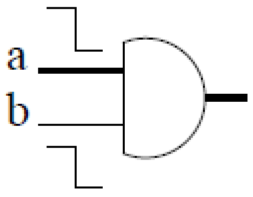


Figure 37: An example of functional sensitization of an AND gate [26].

Figure 37 shows an example of functional sensitization of an AND gate. “a” is the on-input while “b” is the off-input. Both “a” and “b” have transitions from non-controlling value to controlling value. This results in an observable fault at the output of the gate depending on the transition arrival times of both the on-input and the off-input [26].

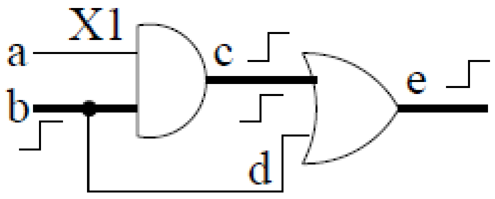


Figure 38: An example of a functional sensitizable path delay fault [26]

Functional sensitizable path delay faults are characterized by the existence of a sequence of test vectors that activates the required transition on the target path to be tested, and at least one gate in the target path has a transition from non-controlling value to controlling value in both its on-input and off-input. Figure 38 shows an example of a functional sensitizable path delay fault. The path is (b, c, e) where a rising transition is tested at the output. Any sequence of test vectors will lead to this path being functional sensitizable as off-input “d” will always have a rising transition which is a transition from non-controlling value of the OR gate the controlling value same as signal “c”. The observation of the fault propagation will depend on the arrival times of transitions of both “c” and “d” [26].

3.5.2 Path delay test methodologies

In path delay fault testing, the applied test is decided by the type of the circuit under test and the DFT. In this section, several path delay test methodologies are introduced.

3.5.2.1 Slow-clock combinational test

This methodology can be used for combinational circuits. Also, it can be used for sequential circuits, only if, the flip-flops are present at the primary inputs and primary outputs.

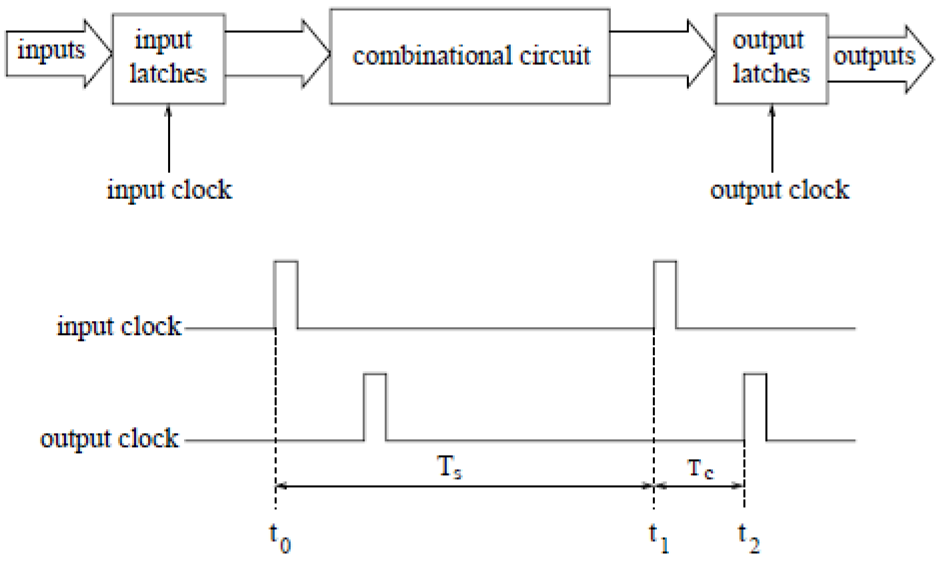


Figure 39: Slow-clock combinational test methodology [26].

Figure 39 illustrates how this methodology is applied. At first, latches are added at the inputs and outputs of the circuit. In normal circuit operation, only one clock source which is the system clock is used to control both latches. In the testing mode, two clock sources are needed. One source for the input latches and another for the output latches. There must be a phase difference between them or a skew equal to the period of the system clock T_c. After that, two test vectors are applied. The first test vector is applied at time t_0 at the clock edge of the input latches. After that, a time span T_s is given that should be more than the system clock period to ensure stabilization of the circuit. Then the positive second test vector is applied and the output latch will capture the output after exactly one system clock cycle T_c. That is why the phase difference between the two source clocks is equal to exactly T_c. If the captured output at the latch is the expected output of the second test vector then the circuit passes the test successfully. If the captured output is the expected one for the first vector then this is a faulty path as the output of the second test vector exceeded the delay of the circuit [23][26].

3.5.2.2 Normal-scan sequential test

This methodology only applies to the sequential circuits. As mentioned before, path delay testing required the application of a sequence of two vectors. In scan circuits, the first vector controls both the primary input and the internal states of the design while the second input determines the primary inputs only and the value of the internal states depend on the previous values from the first vector. This means that the second vector should be a function of the first vector. This is done by one of two methods:

1- Launch-on-shift method
2- Launch-on-capture method.

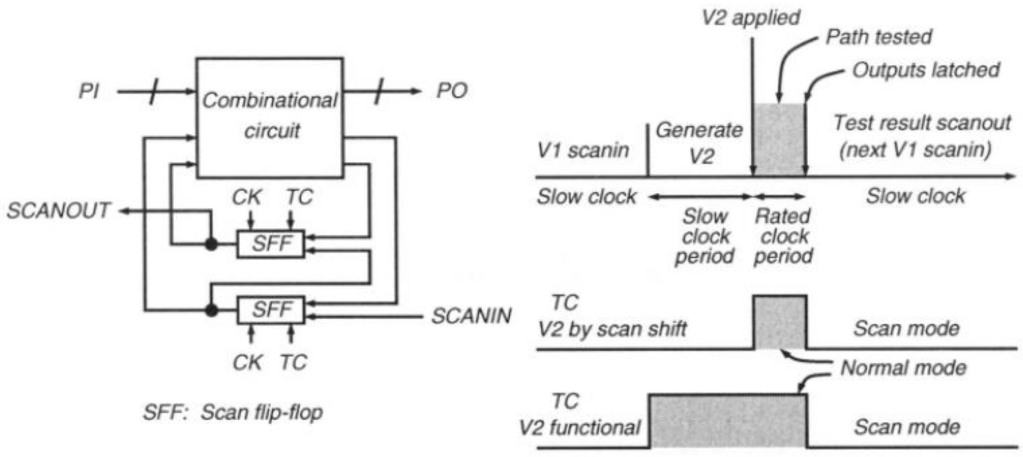


Figure 40: Normal-scan sequential test methodology [23]

Figure 40 shows the two methods. In the launch-on-shift method, the second vector is generated by applying a one-bit shift to the scan register, so the internal states of the second states will be one bit shifted than the internal states of the first vector. In the launch-on-capture test, the second vector is generated by propagating the combinational output due to the first vector into the scan registers [22][23][26].

In the launch-on-shift test, the first vector V_1 is scanned into the scan registers using a slow clock source. After that, one more period of this slow clock is applied to shift the bits in the scan register, and the internal states of V_2 is applied to the scan register. As soon as the primary inputs of the second vector are applied to the circuit under test, the normal operation is on for exactly one rated clock period, and the outputs are latched. Primary outputs are observed and the internal bits of the scan register are scanned out to compare them with the expected outputs to check whether a fault occurred or not [23].

In the launch-on-capture test, the first vector V_1 is also scanned into the scan registers using a slow clock source, and then the normal mode is on by making the signal test control TC equal to 1. This operation should be also controlled by the slow clock source. As a result, the combinational output due to the first vector is latched inside the scan register which are the internal states of the second vector V_2. While the normal operation is on, one rated clock period should be applied to have the transition of $V_1 \rightarrow V_2$. At the end of this rated clock period, the outputs are latched, and the scan registers can be scanned out and compared with the expected outputs to check whether there is a fault or not [23].

3.5.2.3 Enhanced-scan test

The enhanced-scan methodology is also applied to sequential circuits only. It is the same as the normal-scan methodology but it differs in that any arbitrary two vectors can be chosen to test a certain path. Unlike the normal-scan methodology, the second vector is not dependent on the first vector. This is achieved by the additional latches added to the normal-scan design, and the hold signal, which allows the storing of the internal states of the first vector until the application of the second vector [23].

Figure 41 shows how this methodology operates. The first vector's internal states are first scanned into the scan registers via the SCANIN input while setting the test control TC signal to 0. This operation is usually done with a slow clock source to decrease power dissipation and to ensure that no delay fault in the scan paths interferes with the test process. Then, the hold signal is set to 1 to transfer these internal states to the hold latches. The internal states of the second vector are then scanned into the scan registers while the signals due to the first vector stabilize. After that, the primary inputs of the second vector are applied while setting the hold signal to 1 and the test control to 1 to have a normal operation for exactly one rated clock period. This will result in the required transition $V_1 \rightarrow V_2$ in the hold latches and the inputs of the combinational circuit. At the end of this rated clock period, the outputs are latched, and the internal states are scanned out to be compared with the expected outputs to check if there is a fault or not [23].

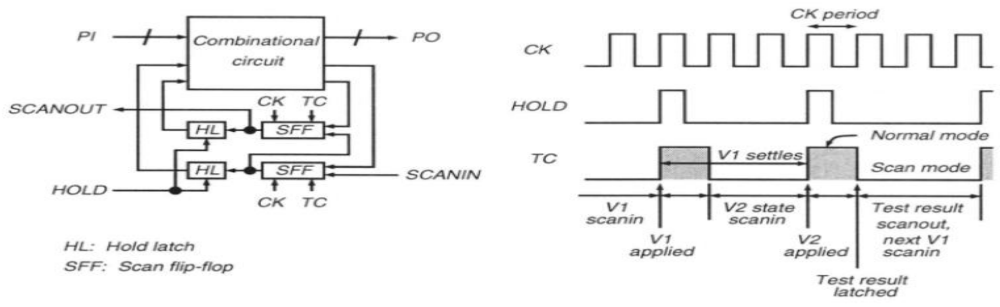


Figure 41: Enhanced-scan test methodology [23]

Chapter 4

WCTV generation

Worst case test vectors (WCTV) are preferred to be used to test the electronic circuits. This is recommended by MIL-STD-883, method 1019, which is the standard that should be followed to test electronic circuit under TID effect. However, due to the difficulty of the generation of the worst case test vectors for electronic circuits, they are not usually used in testing especially in FPGAs. For application specific integrated circuits (ASICs) exposed to TID effect, lots of methodologies were proposed but for FPGAs, a very limited effort was done. In this chapter, an overview of WCTV generation for ASIC designs is presented followed by an overview of the effort done for WCTV generation so far in FPGA-based designs.

4.1 WCTV for ASIC designs

Since most of the WCTV generation methodologies available in the literature is based on ASIC designs, it is important to highlight the effort done here as it represents the basis for WCTV generation for FPGA designs as well. The major difference, however, is that these methodologies depended on the transistor level circuit information, which can be easily extracted in case of a design using standard cell based ASIC. In FPGAs, this information is proprietary and cannot be shared with normal users.

In all methodologies for WCTV generation in ASIC designs, Verilog/VHDL functions implementing the fault models for each cell is developed at first. This step is mandatory and cannot be replaced. The steps for developing these functions are as follows. First, the failure in each cell in the library used is analyzed, and a fault model for every cell is developed. Then, a SPICE simulation is used to validate these fault models using the target process transistor parameters and parametric degradation from total dose experiments. At the end, a package of VHDL/Verilog functions is developed implementing the fault models mentioned before. Methodologies for WCTV generation differ in the type of the detected faults, type of logic for which they are used (combinational or sequential) and the algorithms used to find these vectors.

4.1.1 WCTV for leakage current faults

The methodology in [1] generates worst case test vectors for leakage current failure induced by TID in ASIC cells for both combinational and sequential circuits. Figure 42 shows the test bench setup where two identical instances of netlist of the circuit under test, generated from normal synthesis tools, are added to the testbench, allowing simultaneous simulation of the circuit under test under both irradiation input vectors (I), and post irradiation input vectors (P). By using normal simulation tools, all possible combinations of irradiation input vectors and post irradiation input vectors are applied to the circuit under test, to find the vectors that will make the circuit exhibits maximum leakage current (max_IL).

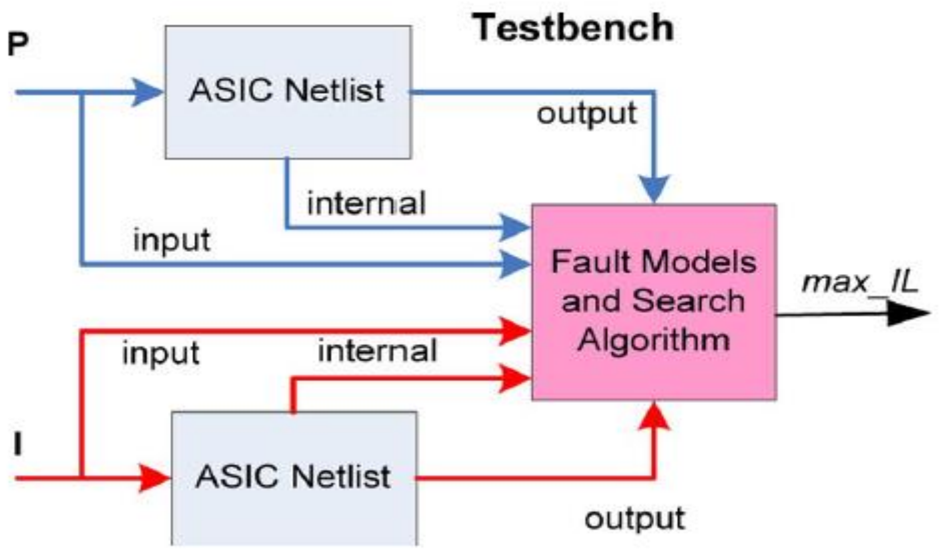


Figure 42: Testbench setup to identify worst case test vectors for leakage current failure in ASICs [1].

A new methodology is proposed in [3] that makes use of the genetic algorithm in the identification of WCTV for leakage current failure in ASICs exposed to TID effect. A smart search algorithm based on genetic algorithm principles is then used instead of the exhaustive search method which can take a very long time in designs with a large number of inputs for the identification of the WCTV. This search algorithm is written in System Verilog simulating the design under test using normal simulation tools. The authors compare the exhaustive search algorithm and the search algorithm based on genetic algorithms for an example design of an 8x8 multiplier. The search algorithm based on genetic algorithm significantly reduced the time for the identification of WCTV to an order of a few seconds compared to a whole two days that the exhaustive search requires to generate WCTV for a simple design like an 8x8 multiplier.

4.1.2 WCTV for logic faults

A methodology proposed in [2] generates worst cast test vectors for logic failure in combinational circuits of ASIC exposed to TID effect. Depending on the type of the cell, the developed fault model can be stuck at 0 or stuck at 1. A ranking system is developed to order the cells per their sensitivity to TID effect induced by radiation. The methodology then targets the cells with the highest sensitivity, and automatic test pattern generation (ATPG) tools like Mentor Graphics FastScan is used to insert a stuck at 0 or stuck at 1, depending on the targeted cell and its fault model, at the output of the cell. The ATPG tool will generate input test patterns, which will be the WCTV of the design under test, and the fault will manifest at the primary output of the circuit.

The advantage of this methodology is that it decreases significantly the time needed if an exhaustive search is used for the identification of WCTVs, especially large design with high number of inputs and a large number of transistor count. However, the disadvantage of this methodology is that it only applies to combinational circuits of ASIC, but most ASIC designs are composed of sequential circuits.

In [4] another methodology is proposed for the identification of WCTV for logic faults in ASIC exposed to TID effect. This methodology is identical to the one in [2] as it also applies to the combinational logic only. However, the difference is in the consideration of field oxide (FOX) edge leakage during the development of fault models by using process technologies that exhibit this type of leakage. Including the FOX leakage in the fault model of the cells will increase the accuracy of the failure analysis of cells, because as the technology advances, the FOX leakage becomes the dominant contributor to the device failure.

A new methodology designed especially for ASIC composed of sequential circuits is proposed in [5]. The difference between combinational circuits and sequential circuits is that the later are characterized by the presence of memory elements. So, in order to test sequential circuits for a single fault, a sequence of test vectors is required to be applied to the circuit to initialize the memory elements to a known state, as opposed to the case of combinational circuits where a single test vector can be applied to the circuit to test it for a single fault. In this methodology, an equivalent combinational circuit is constructed from the original sequential circuit by removing its memory elements. Then, an ATPG tool is used to generate input test vectors to exhibit stuck at 0 or stuck at 1 fault at the outputs of the most sensitive cells in the design depending on the developed fault models. These input test vectors are the WCTV for the designated failure of the original sequential circuit. These vectors must be applied for a number of clock cycles equal to the number of the flip-flops in the design under test to fill the memory elements with the required states.

The disadvantage of this methodology is that it cannot be applied to large designs with large number of memory elements, because it will be harder and even impossible to generate input vectors sequence that will initialize memory elements to known states.

4.1.3 WCTV for delay faults

Another effort is presented in [6] to generate WCTV for delay failure in ASIC exposed to TID effect. The methodology starts first by introducing a novel fault model for delay failure in sequential circuits. The methodology explains that the maximum frequency that a sequential circuit can operate depends on the maximum delay in the combinational logic between any two flip-flops or what is called a critical path. In order to have a delay failure induced by TID effect, the delay in the combinational logic in a chosen critical path must exceed the slack made by the operating frequency. The methodology generates WCTV using two steps. The first step is to identify the number of critical paths that can be candidates for the testing, these paths should have the longest delay between primary inputs and primary outputs, and by using the directed graph and the developed fault models, the candidates for testing can be identified. The second step is to identify a set of irradiation and post irradiation input vectors that will exhibit a maximum delay in the paths generated from the first step. For this step, an algorithm based on a genetic algorithm is developed to identify the test vectors maximizing the delay of the candidates' critical paths generated from the first step using the developed fault model.

The advantage of this methodology is that it didn’t rely on an exhaustive search for the generation of WCTV, which in large design with high number of inputs and sequential circuits with large number of states can take a very long time to implement or it can be even impossible to simulate such huge number of possibilities. Instead, the methodology used genetic algorithm basics to develop a search algorithm to generate WCTV, which in this case will take much less time to complete compared to the exhaustive search method. Giving this, the methodology can be applied to large designs which are characterized by large number of inputs and internal states.

4.2 WCTV for FPGA

All the mentioned methodologies for the WCTV generation in ASIC designs depend on the transistor level circuit information, which can be easily extracted in case of a design using standard cell based ASIC, however, this information is proprietary and cannot be shared with normal users in case of FPGAs. That is why, it has been hard to identify WCTV for FPGA exposed to TID effect. Only two methodologies are available for WCTV generation for FPGA exposed to TID effect. The first methodology in [7] is applicable only for combinational logic circuits while the second methodology in [32] is applicable for sequential logic circuits.

4.2.1 WCTV for combinational logic

The methodology in [7] proposes a way to identify WCTV for delay failure in flash-based FPGA. The authors built their methodology on the fact that the floating gate transistor, which is the switch element in flash-based FPGA, is the dominant factor in the degradation of flash-based FPGA induced by TID effect. Since the transistor level circuit of each cell in the FPGA cannot be shared with normal user, the state of each floating gate transistor (whether it is used or not) in each cell cannot be known, that is why the methodology depends on probability analysis to estimate the number of floating gate transistors used in every cell in the FPGA. The authors then explain that in order for a delay failure to manifest in the operation of the FPGA, the delay of the combinational logic between two flip-flops must exceed the slack value for the operating frequency of the FPGA. The identification of WCTV is done through three steps. First, by using the static timing analysis (STA) tool of the FPGA vendor, some candidate critical paths with the highest delay are identified. Second, from these critical paths, the path with the highest estimated number of floating gate transistors and the highest estimated probability to occur is chosen. Third, ATPG tool like Mentor Graphics FastScan is used to identify input test vectors that will allow a signal to toggle its value from 0 to 1 without being masked by other signals in the chosen path. The toggling must manifest its value at the primary output of the circuit under test to be able to observe the fault. These input test vectors are the WCTV for the design implemented in the FPGA.

Although this methodology was one of its kind, because it was the first effort to identify WCTV for FPGA design, and it only depends on the information that the FPGA vendor give to normal users, however, this methodology only applies to combinational circuit of sequential circuits characterized by a flip-flop at the input and a flip-flop at the output of the circuit, and that is very rare to find in today's design, as most of the designs consist of complex sequential circuits with many flip-flops and internal states.

4.2.2 WCTV for sequential logic

The methodology in [32] is based on the delay failure analysis and fault model developed in [7]. This means that this methodology is also built on the fact that the floating gate transistor, which is the switch element in flash-based FPGA, is the dominant factor in the degradation of flash-based FPGA induced by TID effect. However, the methodology in [7] is only limited to combinational circuits, unlike this one.

Generating test patterns for sequential circuits is harder than generating test patterns for combinational circuits. This is because in sequential circuits, in order to test a certain fault e.g. path delay fault, not only a pair of vectors must be applied to the primary inputs of the circuit under test, but also the internal states of registers in the sequential circuit must be controlled and initialized to a determinant value. Controlling these internal registers is difficult and maybe impossible for complex designs with high number of registers [24].

To overcome this issue, the DFT technique is used in this methodology to increase the controllability and observability of these internal registers. The increase in the controllability and observability is done by replacing normal registers with scan cells. DFT techniques were usually used in testing ASIC devices, however, DFT is never meant for FPGA designs. That is why scan cells are not included in the FPGA macro libraries.

Muxed D scan architecture is the DFT architecture used in this methodology. Every register in the design is replaced with a scan cell having extra input called "scan input" and a control signal "scan enable" to choose between the two inputs. Figure 43 shows how scan cells replace registers and how the scan chains are formed.

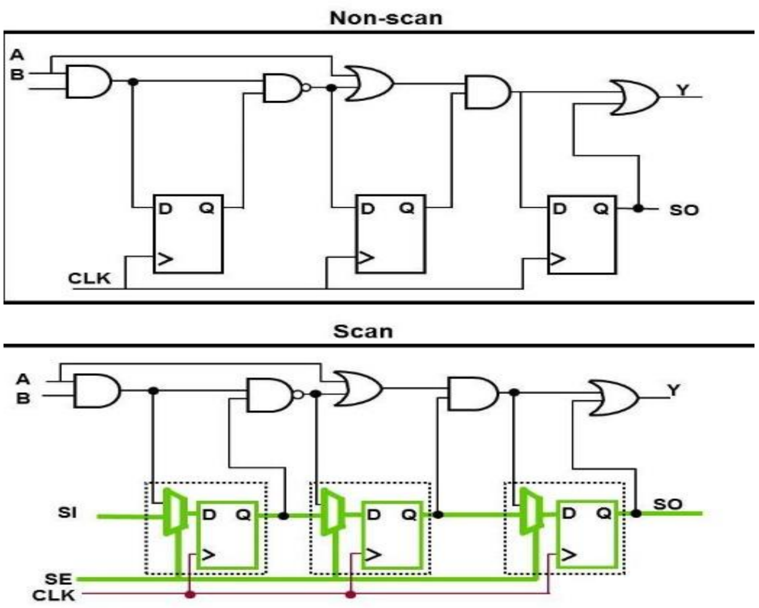


Figure 43: An example of a sequential circuit and its corresponding muxed D scan design [32]

The new netlist generated from the DFT tool containing scan cells is delivered to the ATPG tool to generate test patterns for path delay fault of the target path. In this methodology, a number of critical paths of the design under test are chosen based on the maximum estimated number of FG switches along the path. ATPG tools such as Mentor Graphics FastScan are used to generate the test vectors [32].

Chapter 5

Comparison of DFT techniques

The success of using muxed-D scan DFT technique in the methodology mentioned in [32] for the generation of worst case test vectors in sequential circuits targeting flash-based FPGAs paves the road to trying other DFT techniques. Although muxed-D scan technique is doing what is required to generate the WCTVs, it is still interesting to try other DFT techniques and compare them together to find out which suits the design under test in the best way.

Some of the interesting points of comparison to be considered for each technique are FPGA resources utilization, the added delay in data path due to the added DFT logic, the difficulty of bring-up of the testing environment and defining the robust testable paths in the design. In this work, the comparison is done between the three most famous techniques, muxed-D scan, clocked scan and enhanced scan. The comparison is done on designs from ISCAS'89 benchmarks. Also, the comparison is done under the assumption that all DFT techniques will yield the same amount of TID for the circuits to fail functionally.

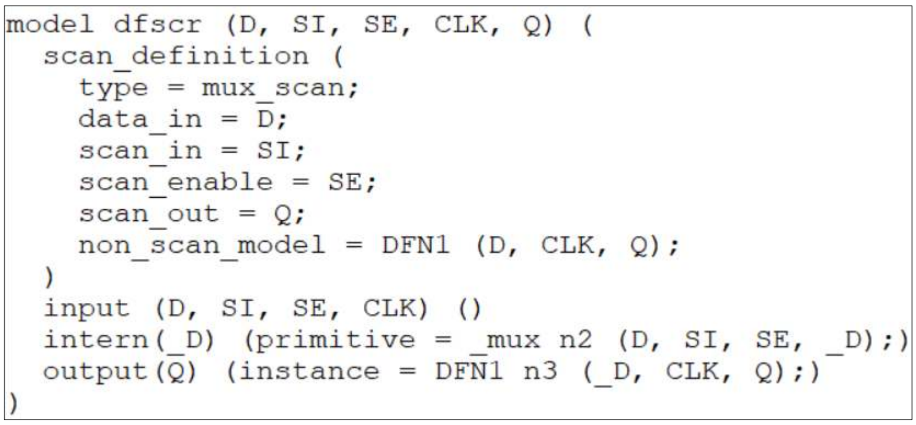
In this chapter, the methodology of using each DFT technique to generate the WCTVs is explained. Then, the results of comparison between the DFT techniques on the benchmarks designs are presented.

5.1 Muxed-D Scan

Muxed-D scan DFT technique is the one used in [32]. This technique represents the most basic idea for DFT designs where the D flip-flops in the design are replaced with the scan cells having added input ports which are a scan input instead of only a data input and a scan enable input to select between the scan input and the data input. Section 3.3.1 talks in details about this technique.

This section is concerned about the methodology of using muxed-D scan DFT technique to generate a testing environment of a design from its original RTL. At first, the original RTL is synthesized using the Synplify synthesis tool embedded within Microsemi Libero. The output netlist from the Synplify is exported. This netlist consists of Microsemi primitives used in ProASIC3 flash-based FPGAs. Some of these primitives are the memory elements representing the flip-flops in the design. These memory elements should be replaced by the corresponding scan cells to form the scan chains.

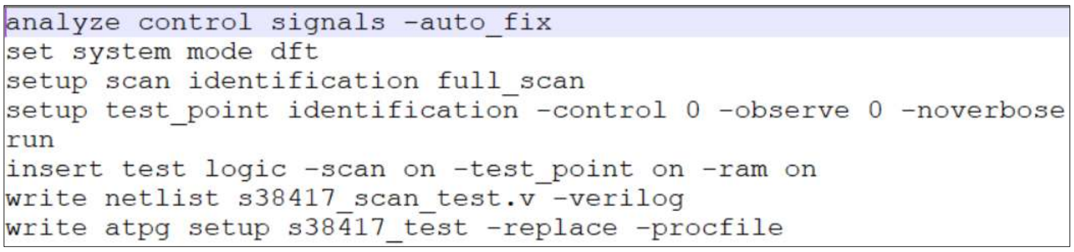
In the next step, the exported synthesized netlist from the Synplify is used as an input to the DFT tool called "DFTAdvisor". This is a Mentor Graphics tool used to add the DFT logic to design netlist to form scan chains to increase the controllability and observability of all internal states of the design under test. To be able to replace the memory elements in the netlist with the scan cells, an "adk" library is provided describing the primitives used in the input netlist and the model of the scan cells used to replace them. Figure 44 shows the scan model of one of the D-flip flops primitives "DFN1" as described in the "adk" library. The scan model is called "dfscr". This will replace all DFN1 primitive cells in the design. It is worth mentioning that adk libraries do not contain the scan models for FPGAs primitives as they are usually used for ASIC designs. Therefore, these models had to be developed to be able to use DFTAdvisor with FPGA netlists.



```
model dfscr (D, SI, SE, CLK, Q) (
  scan_definition (
    type = mux_scan;
    data_in = D;
    scan_in = SI;
    scan_enable = SE;
    scan_out = Q;
    non_scan_model = DFN1 (D, CLK, Q);
  )
  input (D, SI, SE, CLK) ()
  intern(_D) (primitive = _mux n2 (D, SI, SE, _D);)
  output(Q) (instance = DFN1 n3 (_D, CLK, Q);)
)
```

Figure 44: scan model of DFN1 primitive as described in the adk library

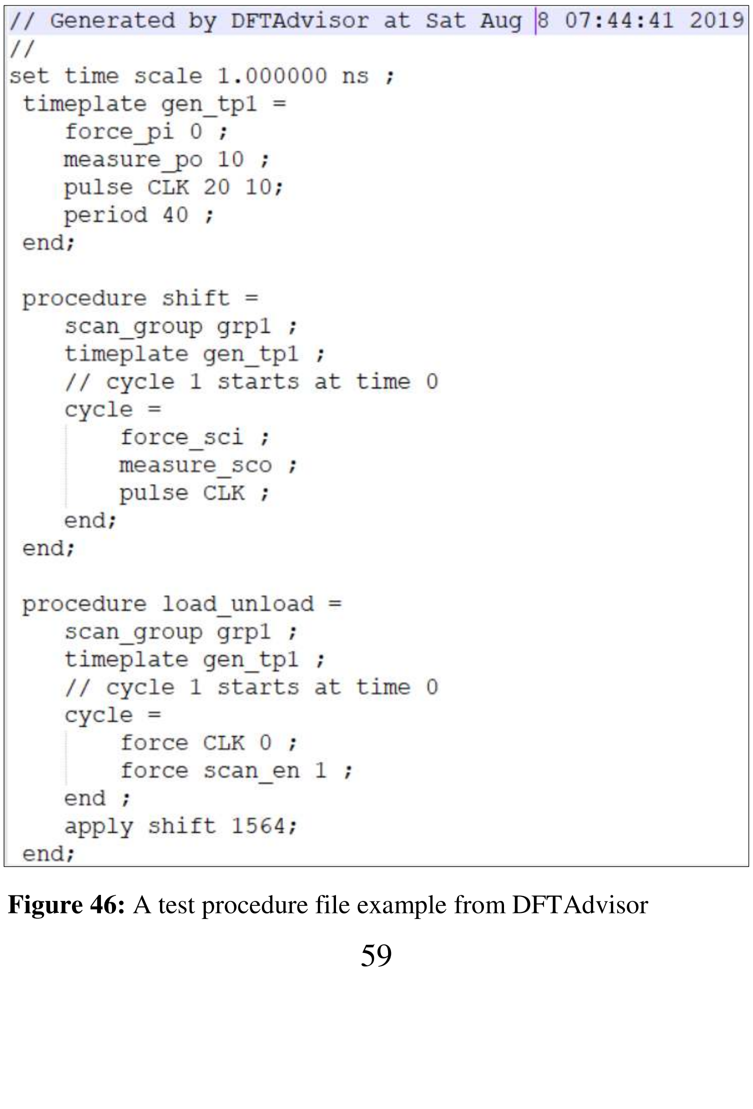
After preparing the input netlist and the scan models in the "adk" library, a do file script is used to run DFTAdvisor tool to produce the netlist after scan chain insertion. The goal behind using scripts to run different tools along the flow is to automate the flow for faster usage, ensure consistency between different runs for different designs and to avoid human errors as much as possible. Figure 45 shows an example of the do-file for DFTAdvisor.



```
analyze control signals -auto_fix
set system mode dft
setup scan identification full_scan
setup test_point identification -control 0 -observe 0 -noverbose
run
insert test logic -scan on -test_point on -ram on
write netlist s38417_scan_test.v -verilog
write atpg setup s38417_test -replace -procfile
```

Figure 45: A do-file example representing the automation of the flow in DFTAdvisor

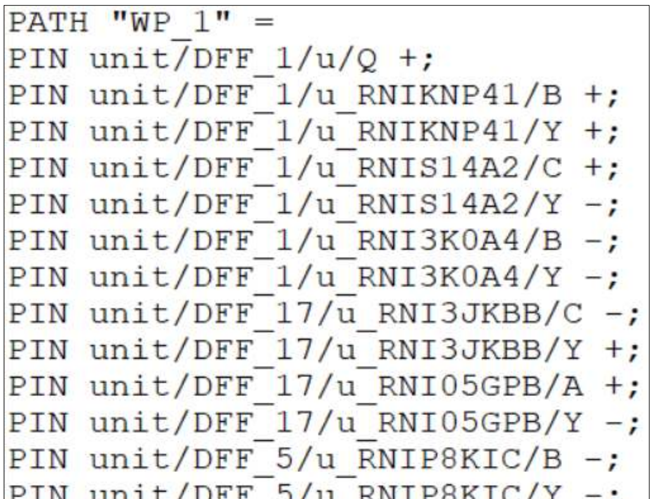
Other than the output netlist provided by DFTAdvisor, it also produces a procedure file for automatic test pattern generation (ATPG) tools. This file describes how to deal with the scan chains inserted by DFTAdvisor. Figure 46 shows an example of how shifting, load and unload operations of a scan chain occur. Shifting operation is done by pulsing the clock while the scan enable signal is high.



```
// Generated by DFTAdvisor at Sat Aug 8 07:44:41 2019
//
set time scale 1.000000 ns ;
 timeplate gen_tp1 =
    force_pi 0 ;
    measure_po 10 ;
    pulse CLK 20 10;
    period 40 ;
 end;

 procedure shift =
    scan_group grp1 ;
    timeplate gen_tp1 ;
    // cycle 1 starts at time 0
    cycle =
        force_sci ;
        measure_sco ;
        pulse CLK ;
    end;
 end;

 procedure load_unload =
    scan_group grp1 ;
    timeplate gen_tp1 ;
    // cycle 1 starts at time 0
    cycle =
        force CLK 0 ;
        force scan_en 1 ;
    end ;
    apply shift 1564;
 end;
```

Figure 46: A test procedure file example from DFTAdvisor

After completing the DFTAdvisor step, it comes the step of automatic test pattern generation. A tool called “FastScan”, which is a Mentor Graphics tool, is used for this purpose. The inputs to this tool to be able to provide the test patterns are the design netlist after inserting the scan chains, the test procedure file explaining how to deal with the scan chains in the design, the adk library and files in asci format describing the data paths to be tested. The first three inputs are already provided from DFTAdvisor. The fourth input, which is the file describing the data paths to be tested, should be written in what is called asci format. This file can describe the path in details by showing how each pin in a cell is connected to the next. The path should only consist of combinational cells (i.e. it should start from the output of a memory element till the input of another memory element). Figure 47 shows an example of a data path written in asci format.



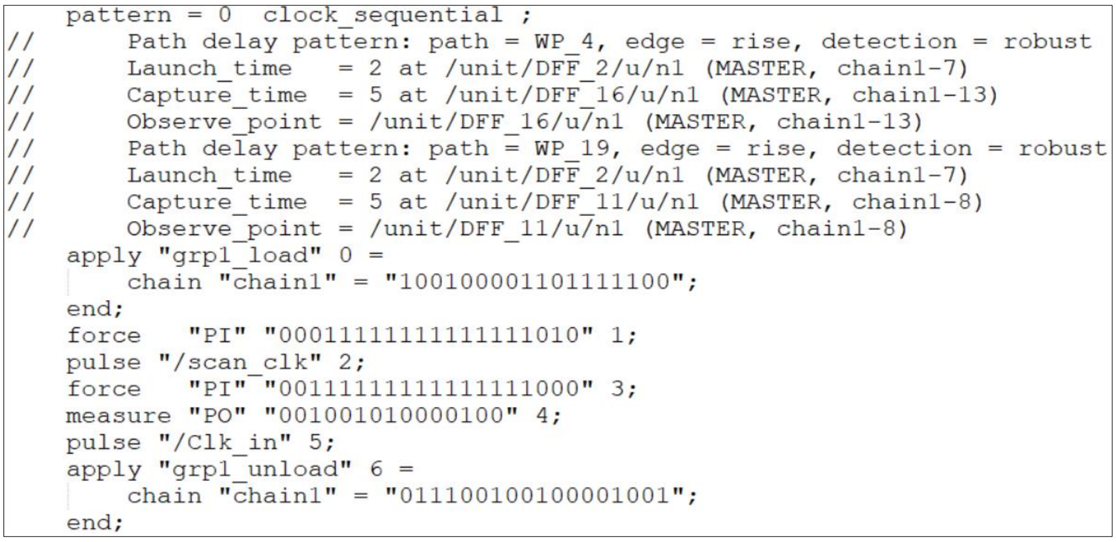
```
PATH "WP_1" =
PIN unit/DFF_1/u/Q +;
PIN unit/DFF_1/u_RNIKNP41/B +;
PIN unit/DFF_1/u_RNIKNP41/Y +;
PIN unit/DFF_1/u_RNIS14A2/C +;
PIN unit/DFF_1/u_RNIS14A2/Y -;
PIN unit/DFF_1/u_RNI3K0A4/B -;
PIN unit/DFF_1/u_RNI3K0A4/Y -;
PIN unit/DFF_17/u_RNI3JKBB/C -;
PIN unit/DFF_17/u_RNI3JKBB/Y +;
PIN unit/DFF_17/u_RNI05GPB/A +;
PIN unit/DFF_17/u_RNI05GPB/Y -;
PIN unit/DFF_5/u_RNIP8KIC/B -;
PIN unit/DFF_5/u_RNIP8KIC/Y -;
PIN unit/DFF_12/u_RNO/B -;
PIN unit/DFF_12/u_RNO/Y -;
PIN unit/DFF_12/u/D -;
END ;
```

Figure 47: An example of a data path written in asci format

To get the Worst Case Test Vectors (WCTVs) for path delay testing, we should run FastScan on the paths with the lowest time slack. The paths of the lowest time slack can be extracted from the Timing Analyzer tool in Libero. To do this, the placement phase in Libero is done at first before running Timing Analyzer to know the detailed path of the lowest time slack. However, two usability problems are faced in this phase.

First, writing the asci file manually is a lot of effort and can lead to many man-errors and worst slack paths are usually long so writing the asci file is highly error-prone. Second problem is that the chosen path may not be a robust testable path and the process of writing the asci file for another path will be repeated for several times. In some large designs, the first robust testable path can be close to the hundredth path. This is a lot of manual effort and countless number of trials. In this work, to avoid the previously mentioned problems, an automation code is written (available at Appendix A) that uses the timing report generated from Libero to write files for all paths in the design in asci format and also produces a one file containing all the paths in the asci format. In this way, from a single run on the FastScan, a detailed report is generated for all the robust paths in the design and their test vectors. By choosing the test vectors of the first robust path with minimum slack, we ensure that the chosen test vectors are the WCTVs for the design that can be applied for path delay testing.

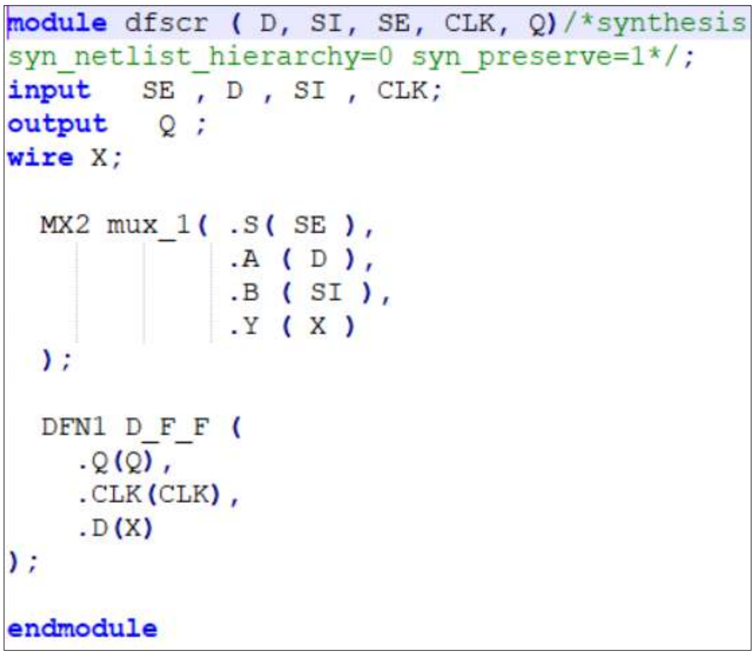
Figure 48 shows an example of the FastScan report. Path delay testing can be applied for two paths "WP_4" and "WP_16" using the same test vectors. The testing procedure is described where first the scan chain "chain1" should be loaded with the provided vector then two primary inputs "PI" should be applied. A gold primary output "PO" is provided for the expected primary output at this stage. Then, by unloading the scan chain, a gold chain output is provided as the expected scan chain output at that stage.



```
    pattern = 0  clock_sequential ;
//      Path delay pattern: path = WP_4, edge = rise, detection = robust
//      Launch_time   = 2 at /unit/DFF_2/u/n1 (MASTER, chain1-7)
//      Capture_time  = 5 at /unit/DFF_16/u/n1 (MASTER, chain1-13)
//      Observe_point = /unit/DFF_16/u/n1 (MASTER, chain1-13)
//      Path delay pattern: path = WP_19, edge = rise, detection = robust
//      Launch_time   = 2 at /unit/DFF_2/u/n1 (MASTER, chain1-7)
//      Capture_time  = 5 at /unit/DFF_11/u/n1 (MASTER, chain1-8)
//      Observe_point = /unit/DFF_11/u/n1 (MASTER, chain1-8)
    apply "grp1_load" 0 =
        chain "chain1" = "10010000110111100";
    end;
    force   "PI" "00011111111111111010" 1;
    pulse "/scan_clk" 2;
    force   "PI" "00111111111111111000" 3;
    measure "PO" "001001010000100" 4;
    pulse "/Clk_in" 5;
    apply "grp1_unload" 6 =
        chain "chain1" = "01110010010000l001";
    end;
```

Figure 48: An example of FastScan report for muxed-D scan design

After getting the WCTVs for path delay testing and the procedure to apply the test from FastScan, it comes the step of preparing the test bench. The test bench is just a mapping of the test procedure produced by FastScan to Verilog code. It is made sure that the test bench is synthesizable so that the whole testing environment is available on the FPGA. The test bench instantiates the netlist that DFTAdvisor produced, where DFT logic is added and the scan chain is inserted. In this netlist, the D-flip flop primitive “DFN1” is replaced by the scan model cell defined in the adk library called “dfscr”. The scan cell is not defined in Libero so an implementation of it is developed as shown in Fig.49. It is just a D-flip flop preceded by a multiplexer to choose between the data input and the scan input.



```
module dfscr ( D, SI, SE, CLK, Q)/*synthesis
syn_netlist_hierarchy=0 syn_preserve=1*/;
input   SE , D , SI , CLK;
output   Q ;
wire X;

  MX2 mux_1( .S( SE ),
             .A ( D ),
             .B ( SI ),
             .Y ( X )
  );

  DFN1 D_F_F (
    .Q(Q),
    .CLK(CLK),
    .D(X)
);

endmodule
```

Figure 49: muxed D scan cell model implementation

After the test bench is ready, the whole libero flow is run starting from synthesis ending to the placement and routing on the FPGA. The simulation is done on various steps in the flow to make sure of correct functionality. Simulation is done on original RTL, post-synthesis and post layout.

5.2 Clocked Scan

Clocked scan DFT technique is the second technique used in this comparison. The difference between this technique and the Muxed-D scan technique is that instead of a scan enable signal to choose between the scan input and data input, another clock called scan clock is used to enable the scan input. The main advantage of clocked scan technique is to avoid the additional multiplexer in the data path of the original design and avoid its additional delay. Section 3.3.2 talks in details about this technique.

This section is concerned about the methodology of using clocked scan DFT technique to generate a testing environment of a design from its original RTL. The steps are similar to the muxed-D scan technique. At first, the synthesized netlist is exported from Synplify synthesis tool embedded in Microsemi Libero. The memory elements in the netlist will be replaced by the corresponding clocked scan cells in DFTAdvisor to form the scan chains. In DFTAdvisor step, the adk library did not have the clocked scan model to replace the memory element "DFN1" in the synthesized netlist. However, DFTAdvisor supports the clocked scan technique so the scan model had to only be defined manually. Figure 50 shows the developed clocked scan model.

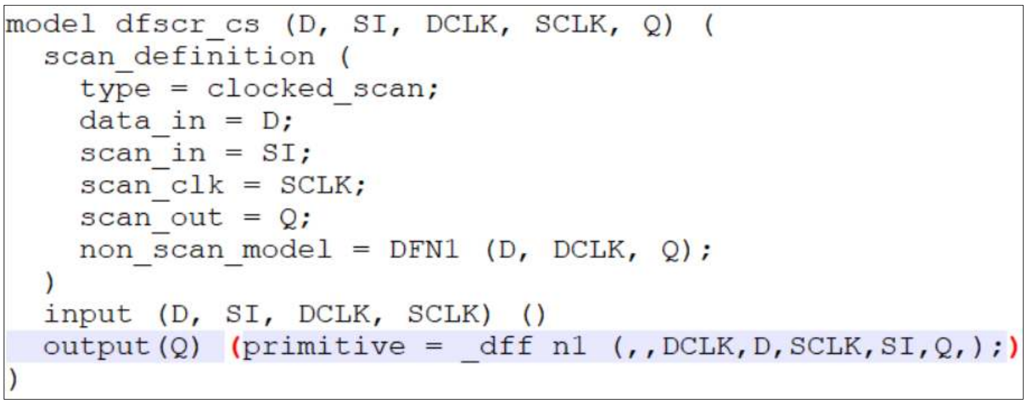
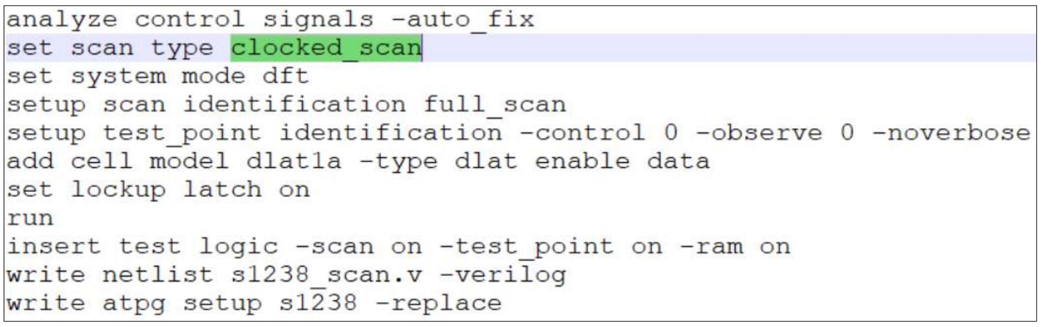

```
model dfscr_cs (D, SI, DCLK, SCLK, Q) (
  scan_definition (
    type = clocked_scan;
    data_in = D;
    scan_in = SI;
    scan_clk = SCLK;
    scan_out = Q;
    non_scan_model = DFN1 (D, DCLK, Q);
  )
  input (D, SI, DCLK, SCLK) ()
  output(Q) (primitive = _dff n1 (,,DCLK,D,SCLK,SI,Q,);)
)
```

Figure 50: clocked scan model of DFN1 primitive as described in the adk library

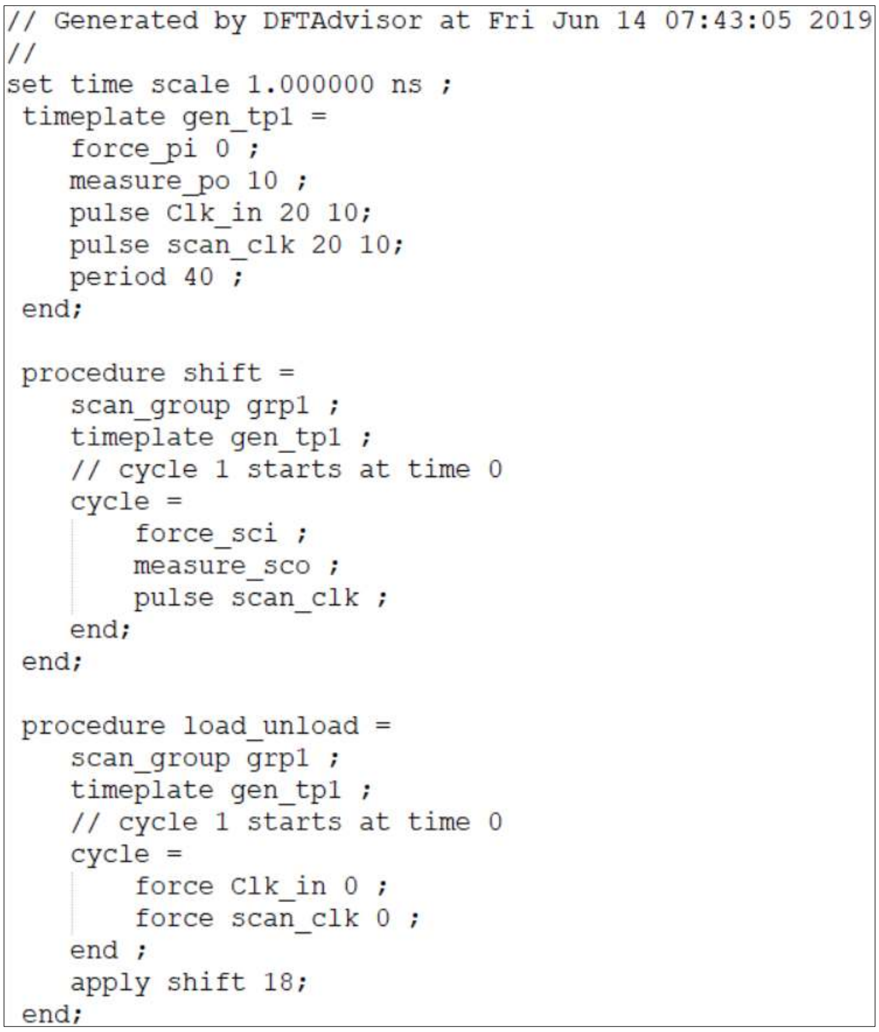
The scan model is called "dfscr_cs". It has two clock inputs, two data inputs, and one output. After preparing the input netlist and the clocked scan model. A do file is prepared to run DFTAdvisor automatically. This is similar to the one used in muxed-D scan technique except that it determines the DFT type to be clocked scan instead of the default muxed D. Figure 51 shows the do file where the scan type is set to clocked scan.



```
analyze control signals -auto_fix
set scan type clocked_scan
set system mode dft
setup scan identification full_scan
setup test_point identification -control 0 -observe 0 -noverbose
add cell model dlat1a -type dlat enable data
set lockup latch on
run
insert test logic -scan on -test_point on -ram on
write netlist s1238_scan.v -verilog
write atpg setup s1238 -replace
```

Figure 51: A do file example representing the automation of the flow in DFTAdvisor for clocked scan

Similar to muxed-D scan, DFTAdvisor produces the output netlist and a procedure file for automatic test pattern generation (ATPG) tools. Of course the shifting, load and unload operations are different in this technique. Figure 52 shows an example of that where shifting is done by pulsing “scan_clk” while normal design clock is forced to zero value.

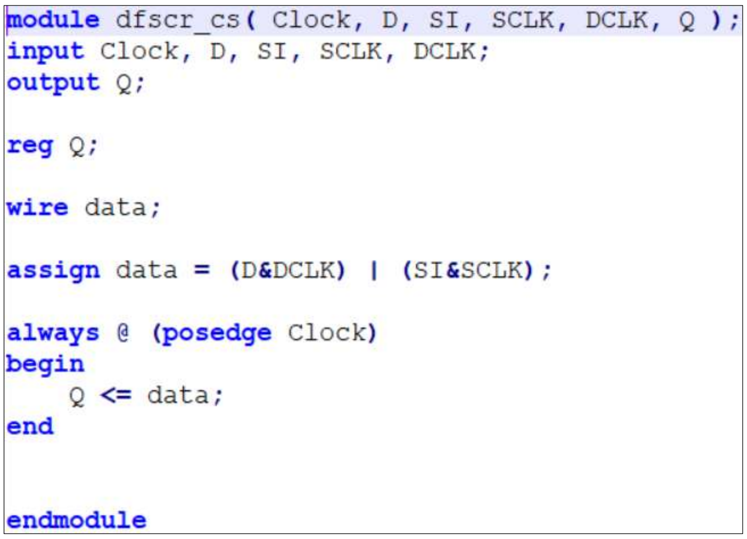


```
// Generated by DFTAdvisor at Fri Jun 14 07:43:05 2019
//
set time scale 1.000000 ns ;
 timeplate gen_tp1 =
    force_pi 0 ;
    measure_po 10 ;
    pulse Clk_in 20 10;
    pulse scan_clk 20 10;
    period 40 ;
 end;

 procedure shift =
    scan_group grp1 ;
    timeplate gen_tp1 ;
    // cycle 1 starts at time 0
    cycle =
        force_sci ;
        measure_sco ;
        pulse scan_clk ;
    end;
 end;

 procedure load_unload =
    scan_group grp1 ;
    timeplate gen_tp1 ;
    // cycle 1 starts at time 0
    cycle =
        force Clk_in 0 ;
        force scan_clk 0 ;
    end ;
    apply shift 18;
 end;
```

Figure 52: A test procedure file example from DFTAdvisor for clocked scan

Next is the step of automatic test pattern generation (ATPG) using “FastScan”. First, the asci files for the paths in the design are generated from using the automation script mentioned in “Appendix A” on Timing Analyzer report, the same as muxed-D scan technique. In this way, from a single run on FastScan, a detailed report is generated for all the robust paths in the design and their test vectors. By choosing the test vectors of the first robust path with minimum slack, we ensure that the chosen test vectors are the WCTVs for the design that can be applied for path delay testing.

After getting the WCTVs for path delay testing and the procedure to apply the test from FastScan, it comes the step of preparing the test bench. Similar to Muxed-D scan, the test bench is just a mapping of the test procedure produced by FastScan to Verilog code. The test bench instantiates the netlist that DFTAdvisor produced, where DFT logic is added and the scan chain is inserted. The scan cell is not defined in Libero so an implementation of it is developed. However, the implementation here is not as simple as in muxed-D scan. The microsemi ProAsic library does not contain a flip flop with two data inputs and two clock inputs. Figure 53 shows the proposed implementation for the clocked scan model “dfscr_cs”.



```
module dfscr_cs( Clock, D, SI, SCLK, DCLK, Q );
input Clock, D, SI, SCLK, DCLK;
output Q;

reg Q;

wire data;

assign data = (D&DCLK) | (SI&SCLK);

always @ (posedge Clock)
begin
    Q <= data;
end


endmodule
```

Figure 53: clocked scan cell model implementation

The cell has an additional input clock port that is not there in the scan model. This input clock is the result of an OR operation between the data clock and the scan clock. The input data is also an OR operation between the data input and the scan input when their corresponding clocks are enabled. Unfortunately, this means that there is an added logic in the data path of the original design. Clocked scan technique goal was to avoid any added logic in this path but due to a limitation in the hardware, this cannot be avoided. The added port in the “dfscr_cs” cell led to changes in the netlist provided by DFTAdvisor to add this port to all instantiations of this cell.

Similar to muxed-D scan, simulation is done on various steps in the flow to make sure of correct functionality. The simulation is done on original RTL, post-synthesis and post layout.

5.3 Enhanced Scan

Enhanced scan DFT technique is built on either muxed D scan design technique or clocked scan design technique. In this work, it is built on the muxed D scan. The simple idea is to add another level of latches after the muxed D scan cell which leads to increasing the capacity of a normal muxed D scan cell. The goal behind this is to be able to store two data vectors and apply them to the circuit in an at-speed manner. The true advantage behind this technique is to be able to apply two completely independent and arbitrary vectors to the circuit under test. This will increase the detection capability of the delay fault. Section 3.3.3 talks in details about this technique.

This section is concerned about the methodology of using enhanced scan DFT technique to generate a testing environment of a design from its original RTL. The difficulties in applying this methodology are due to the fact that this scan technique is not supported by the used DFT tool “DFTAdvisor” so the scan logic cannot be inserted automatically. Also, the ATPG tool “FastScan” does not support this scan technique so the test vectors are not automatically generated. This means that the whole methodology should be done manually. This is a big drawback from other scan techniques from the point of view of the ease of bring-up of the test environment.

The approach followed in this work to apply the enhanced scan based on muxed-D scan is as follows. At first, the complete methodology of muxed D scan needs to be applied as the outputs will be used to help apply the enhanced scan technique. This means that DFTAdvisor and FastScan are run targeting the muxed D scan technique. The output netlist from DFTAdvisor is then altered such that an additional latch is added after each scan register. Figure 54 shows the added muxed D scan cells in the green box and the latches that need to be added after each register in the red box to have the enhanced technique DFT logic ready. This process cannot be done manually for designs having hundreds or thousands of registers. To ease up this process, automation scripts (mentioned in Appendix B) are developed to be able to insert the latches whatever the size of the design is without exposing the process to human errors.

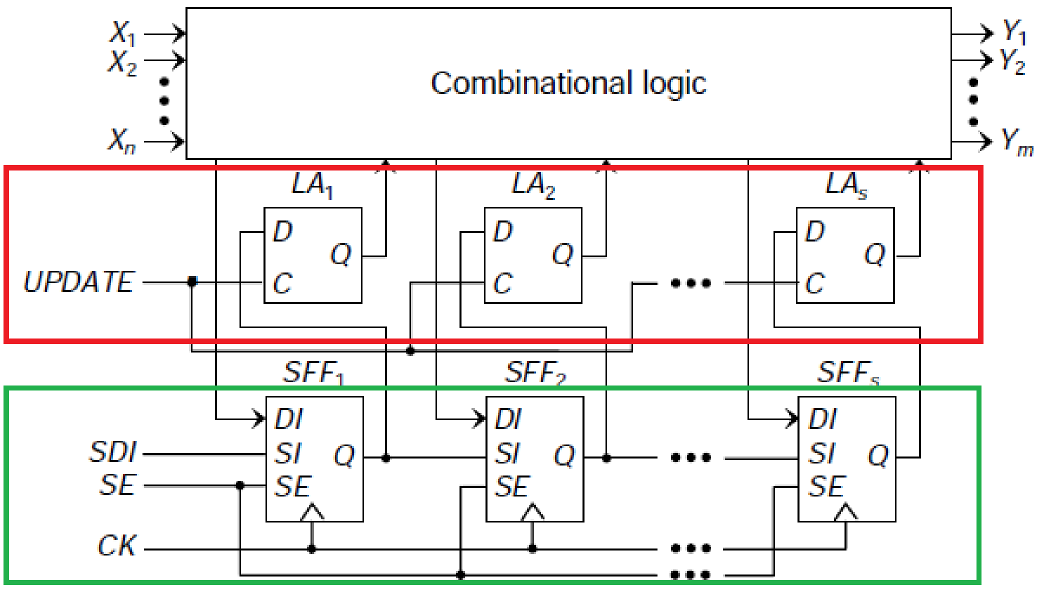
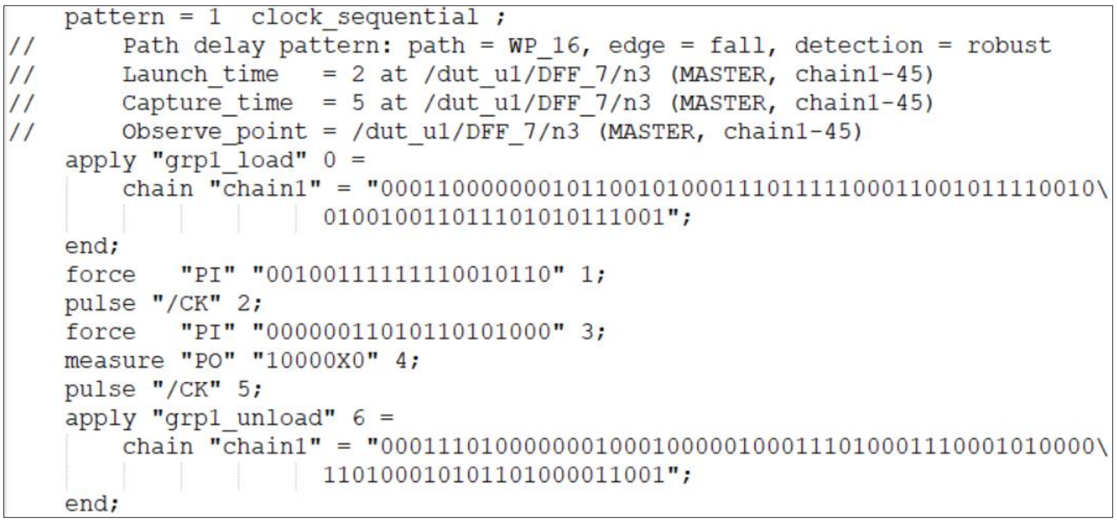


Figure 54: Enhanced scan DFT logic insertion

After the enhanced scan netlist is ready, it comes the step of the process of generation of the test vectors for the enhanced scan technique from the test vectors generated by FastScan for the muxed D scan technique. Figure 55 shows an example of test vectors generated by FastScan for the muxed-D scan technique. Since the idea in enhanced scan technique is to apply two test vectors to the scan chain in an at-speed manner, so the goal is to extract the two scan chain test vectors to apply from the test vectors generated for muxed-D scan. The first scan chain test vector is already provided by default in Figure 55. The second scan chain test vector is the value of the scan chain after applying the first primary input “PI”. Therefore, to get this vector, muxed-D scan test bench is prepared using the DFTAdvisor netlist (before modification) and simulation is run in the same way as done for standalone muxed-D scan technique. From the simulation, the value of the second scan chain test vector is extracted.



```
    pattern = 1  clock_sequential ;
//      Path delay pattern: path = WP_16, edge = fall, detection = robust
//      Launch_time   = 2 at /dut_u1/DFF_7/n3 (MASTER, chain1-45)
//      Capture_time  = 5 at /dut_u1/DFF_7/n3 (MASTER, chain1-45)
//      Observe_point = /dut_u1/DFF_7/n3 (MASTER, chain1-45)
    apply "grp1_load" 0 =
        chain "chain1" = "000110000000101100101000111011111000110010111100010\
                          01001001101110101011100 1";
    end;
    force   "PI" "00100111111110010110" 1;
    pulse "/CK" 2;
    force   "PI" "00000011010110101000" 3;
    measure "PO" "10000X0" 4;
    pulse "/CK" 5;
    apply "grp1_unload" 6 =
        chain "chain1" = "000111010000000100010000010001110100011100010100000\
                          110100010101101000011001";
    end;
```

Figure 55: an example of test vectors generated by FastScan for the muxed-D scan technique

The procedure for applying the test vectors for the enhanced scan technique is to first fill the scan chain by the first scan chain test vector while the enable signal of the latches is low. After the scan chain is fully populated, the enable signal for the latches is high for one clock cycle to move the first vector into the latches. Afterward, the second scan chain test vector is applied to fill the scan chain. After the scan chain is fully populated, the second primary input from the FastScan report is applied in the same clock cycle as the enable signal for latches becomes high to enable the second scan chain vector to be applied at an at-speed manner. The scan chain in the next clock cycle is shifted to be read from the scan output port. The output is compared to the gold scan chain provided by the FastScan report.

The above procedure is written in a synthesizable Verilog code as the testbench of the design. After the test bench is ready, the whole libero flow is run starting from synthesis ending to the placement and routing on the FPGA. Simulation is done on various steps in the flow to make sure of correct functionality. The simulation is done on original RTL, post-synthesis and post layout.

5.4 Benchmark designs

The benchmark designs used in the comparison between different DFT techniques are all from ISCAS’89 benchmarks [29]. They are described as being suitable for researchers working in the field of scan designs for sequential circuits. The three chosen benchmark designs are different in size where “S1238” is the smallest, “S5378” is middle sized and “S38417” is the largest. This is to make sure that the results of the comparison are consistent and not design-dependent. For each design, the methodologies for the three DFT techniques described earlier are applied.

5.4.1 S1238

S1238 is the smallest benchmark used in this comparison with the lowest number of flip flops among other benchmarks used in this comparison. It is described in [29] as a combinational circuit with randomly inserted flip flops. Table I shows the design specifications.

Table I: Design specifications of S1238 benchmark

Number of inputs	14
Number of outputs	14
Number of flip flops	18
Number of gates	508

The used FPGA for experimentation with this design is Microsemi ProASIC A3P125-PQ208 flash-based FPGA. It is the same used for all DFT techniques for a fair comparison. Also, the path for producing WCTV shown in Figure 56 is the same used for all the techniques.

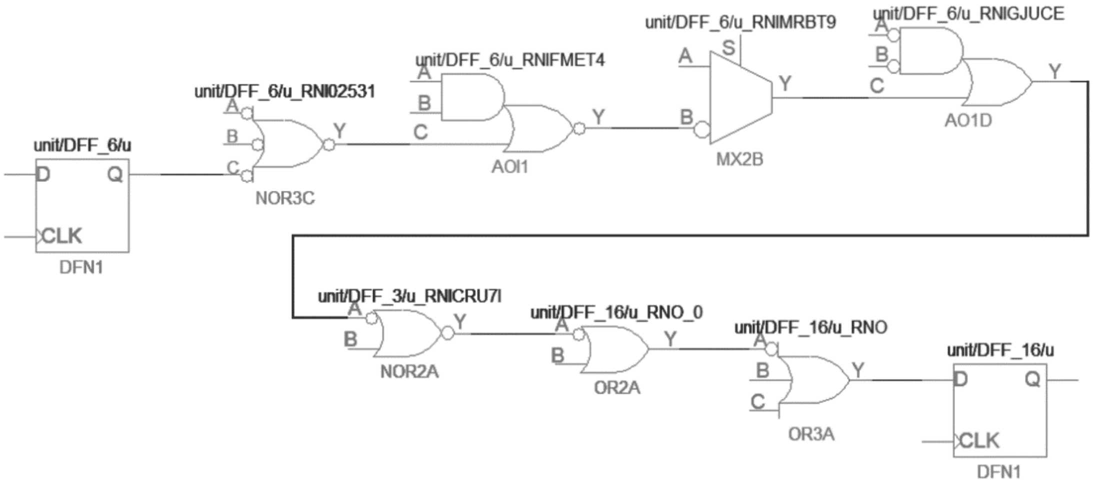


Figure 56: Worst-case path for S1238 benchmark design

The WCTVs reported by FastScan for the three techniques targeting the path mentioned in Figure 56 is summarized in Table II.

Table II: Summary of the test vectors of S1423 circuit with different DFT techniques

DFT Technique	Test Vectors

Muxed D scan	Scan chain = 1001011100011110000
	PI1 = 00110010011100
	PI2 = 10110011010100
	Gold scan chain = 0010101011000100001

Clocked scan	Scan chain = 1000010100000100011
	PI1 = 01110010011100
	PI2 = 01110010010100
	Gold scan chain = 0010101111001111001

Enhanced scan	Scan chain1 = 1001011100011110000 Scan chain2 = 0110101111110000001 PI = 10110011010100 Gold scan chain = 0010101011000100001

For logic utilization in each case, Table III shows a summary of the versatile/core utilization in the three scan techniques used. Clearly, Clocked scan technique yields the least utilization among the three techniques. It is worth mentioning that a unit core can be configured as a three-input logic function or a D-flip-flop or latch (with or without enable) by programming the appropriate Flash switch interconnections.

Table III: Summary of the core utilization for S1423 circuit with different DFT techniques

DFT Technique	Core utilization
Muxed D scan	Used: 264 , Total: 3072 (8.59%)
Clocked scan	Used: 199 , Total: 3072 (6.48%)
Enhanced scan	Used: 238 , Total: 3072 (7.75%)

Another point of comparison is the maximum frequency at which each technique can be functionally correct. The considered paths are the register to register paths. This gives an indication on which DFT technique has the worse effect on the design frequency. Timing Analyzer tool embedded in Libero is the one used for this purpose. Table IV shows a summary of the minimum clock period for each technique. Clearly, enhanced scan technique has the upper hand when it comes to the maximum frequency.

Table IV: Summary of the minimum clock period for S1423 circuit with different DFT techniques

DFT Technique	Minimum clock period
Muxed D scan	6.981 ns
Clocked scan	7.292 ns
Enhanced scan	4.269 ns

Another aspect to consider in the comparison is the number of robust paths for path delay testing that the automatic test pattern generation (ATPG) tools can detect while using different DFT techniques. In this comparison, FastScan is the used ATPG tool. Table V shows a summary of the number of robust paths in each technique. In clocked scan technique, the number of robust paths is greater than that of muxed D scan. Since the enhanced scan technique is built upon muxed D scan technique, the number of robust paths is the same as muxed D scan.

Table V: Number of robust paths for S1423 circuit with different DFT techniques

DFT Technique	Number of robust paths
Muxed D scan	26
Clocked scan	32
Enhanced scan	26

The simulation was done for each DFT technique on original RTL, post-synthesis and post layout. Figure 57 shows an example of the waveform from the simulation of muxed D scan technique. Signal “out_reg” contains the output from unloading the scan chain while signal “gold_out_reg” is the expected value from FastScan after unloading the scan chain. Figure 57 shows that they are equal and another signal called “op_check” goes high if the test was successful.

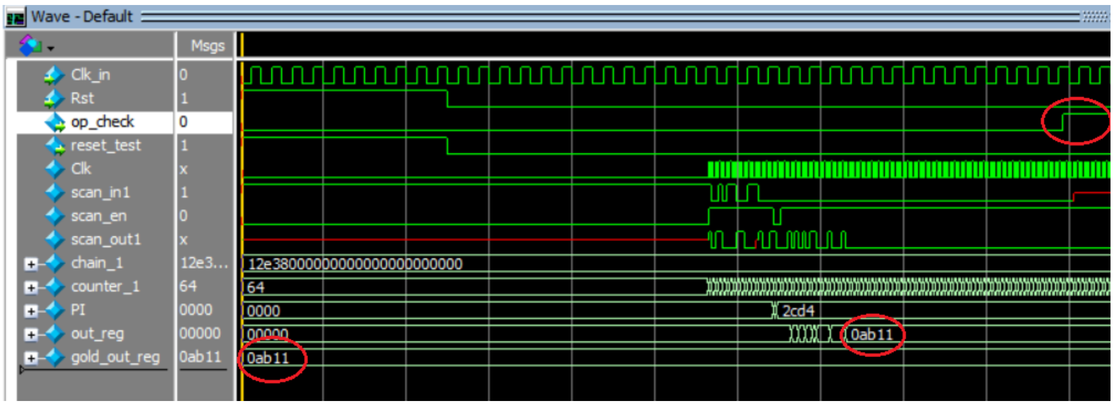


Figure 57: an example of the waveform from simulation of muxed D scan technique

5.4.2 S5378

S5378 is a middle-sized benchmark used in this comparison. Table VI shows the design specifications.

Table VI: Design specifications of S5378 benchmark

Number of inputs	35
Number of outputs	49
Number of flip flops	179
Number of gates	2779

The used FPGA for experimentation with this design is Microsemi ProASIC A3P125-PQ208 flash-based FPGA. It is the same used for all DFT techniques for a fair comparison. Also, the path for producing WCTV shown in Fig.58 is the same used for all the techniques.

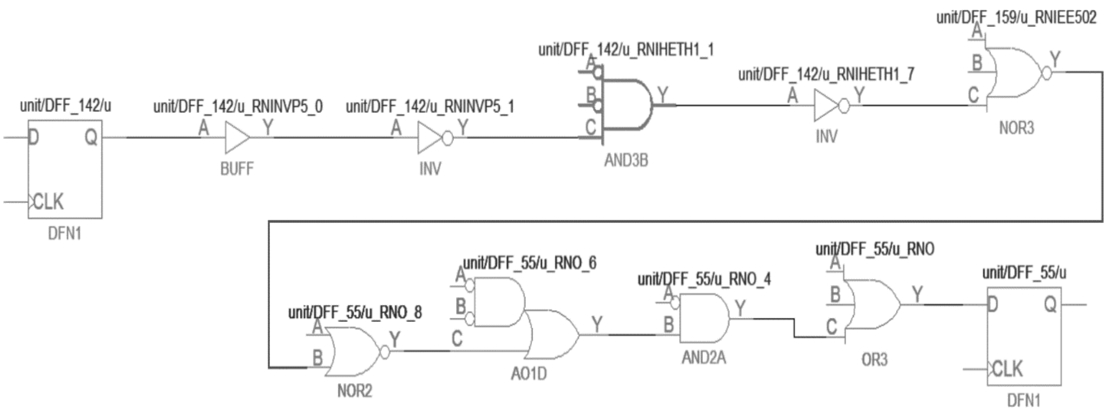


Figure 58: Worst-case path for S5378 benchmark design

The WCTVs reported by FastScan for the three techniques targeting the path mentioned in Fig.58 is summarized in Table VII.

Table VII: Summary of the test vectors of S5378 circuit with different DFT techniques

DFT Technique	Test Vectors
Muxed D scan	Scan chain = 1F01C73C70A7D081DCCCD810B607F8022950BDBDC79CF PI1 = 001110111001101000000100010000010101 PI2 = 111000101110010010000111000000010101 Gold scan chain = 19244204E18C10A03114091A1E452786ED7AEAE346D46
Clocked scan	Scan chain = 5E5BECC4D85389FB9943C593AA5272F7D07755D3C03E6 PI1 = 001010111111001111001011110010101 01 PI2 = 101001101110010110010000100000011001 Gold scan chain = 5ADCD0940C8093D08776A8482A48742B08D8A4155802D
Enhanced scan	Scan chain1 = 1F01C73C70A7D081DCCCD810A0619E644F36110EF4FA9 Scan chain2 = 10B7D54A4819BA40F3F10130BB6CD72730D3DDC2940ED PI = 111000101110010010000111000000010101 Gold scan chain = 19244204E18C10A03114091A1E452786ED7AEAE346D46

For logic utilization in each case, Table VIII shows a summary of the versatile/core utilization in the three scan techniques used. Clearly, Clocked scan technique yields the least utilization among the three techniques which is the same result as benchmark design S1238.

Table VIII: Summary of the core utilization for S5378 circuit with different DFT techniques

DFT Technique	Core utilization
Muxed D scan	Used: 2193 , Total: 3072 (71.39%)
Clocked scan	Used: 2130 , Total: 3072 (69.34%)
Enhanced scan	Used: 2256 , Total: 3072 (73.44%)

Another point of comparison is the maximum frequency at which each technique can be functionally correct. The considered paths are the register to register paths. This gives an indication on which DFT technique has the worse effect on the design frequency. Table IX shows a summary of the minimum clock period for each technique. Enhanced scan technique has a slightly larger maximum frequency.

Table IX: Summary of the minimum clock period for S5378 circuit with different DFT techniques

DFT Technique	Minimum clock period
Muxed D scan	21.094 ns
Clocked scan	21.056 ns
Enhanced scan	20.655 ns

Another aspect to consider in the comparison is the number of robust paths for path delay testing that the automatic test pattern generation (ATPG) tools can detect while using different DFT techniques. Table X shows a summary of the number of robust paths in each technique. They are nearly equal in all techniques.

Table X: Number of robust paths for S5378 circuit with different DFT techniques

DFT Technique	Number of robust paths
Muxed D scan	184
Clocked scan	185
Enhanced scan	184

Similar to S1238 benchmark, the simulation was done for each DFT technique on original RTL, post-synthesis and post layout to ensure correct functionality in all cases.

5.4.3 S38417

S38417 is one of the largest benchmark designs in ISCAS'89 and is used in this comparison. It is described in [29] as a real chip-based design that relies in its operation on the partial scan. Table XI shows the design specifications.

Table XI: Design specifications of S38417 benchmark

Number of inputs	28
Number of outputs	106
Number of flip flops	1636
Number of gates	22179

The used FPGA for experimentation with this design is Microsemi ProASIC A3P1000-PQ208 flash-based FPGA which is much larger than the one used for the other two benchmark designs. It is the same used for all DFT techniques for a fair comparison. Also, the path for producing WCTV shown in Fig.59 is the same used for all the techniques.

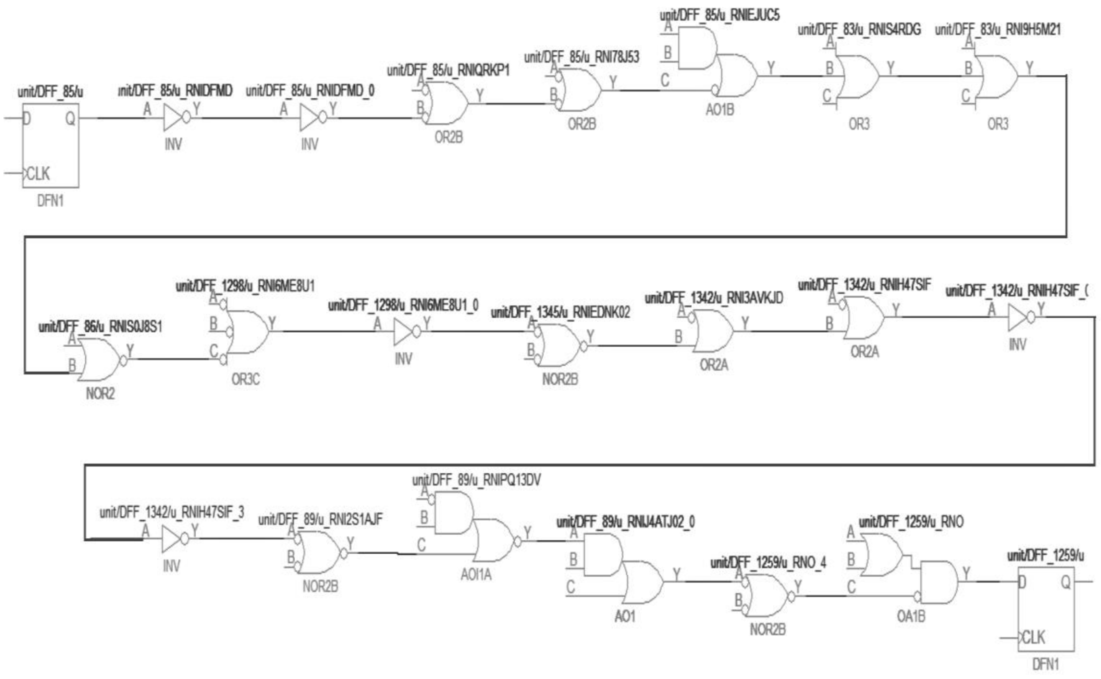


Figure 59: Worst-case path for S38417 benchmark design

The WCTVs reported by FastScan for the three techniques targeting the path mentioned in Fig.59 is summarized in Table XII.

Table XII: Summary of the test vectors of S38417 circuit with different DFT techniques

DFT Technique	Test Vectors
Muxed D scan	Scan chain = 99305DBB35D0429B82F5F1DE4EF9C33B3A39285188E7CF8 A25E43D6B9B75E4C98EC174ED759CB3684315D84BC469B 0E8218C3E0905D46D4AF5E6C61FAA0E9C1FE207EEBC994 6B2B492E6208A6AACCDC3FD78FE63D464E93165B936334 5EC745562E8561E8E920CDD8467F72085F47530F857C82865 E7B66921B8C6B5AD9D2CB5CB31FE9DD5E7AC416033BB7 A9C87D5A9EB87AA7D75ABDE228EA03693FC26E10A71EE 5C22188BCF90295FFD380F03BA828E3426A35FBBA8579EF 71E22E6926557C1AA86C PI1 = 10000110000100110010000001101 PI2 = 11101011000011100011110011111 Gold scan chain = D132DCBB05E4C2B3B6F6F11E4EE4487B323DECD90A474F EA64643D75AB6DE8C79E1006AE279CBF7843B7C88EE469 82F8318C9679D4512D4AE3E6E79FBB0DDA3E16AEAFBD9 866FEBC14E620880AE938C3DD6ABCB79616EA596538725 B046E344963E8465A0F02443D96C7E2078276753CD1C76A7 83CEFB2EE72BAC69D2C952CB5CE41FE3B87FB80000043C FE232474C83FE9F89F147C3FBE78F00FA07FC8BE421E39FF 00F403F1288E75BE8F0BF0F720E08F48E8F17EFB81FEAEE3 E2D9842F75F3B8E970
Clocked scan	Scan chain = B2D8475BC1007D6F9AFC45D75042BC5B9D20902E8B567E 5BF9D7B280A57F349D4739DB9A3365DC23AD38DD005A4 C86DD0190E6565748DB5491C750902A881807DEE7C6F63F DF8303303ECAA80C480CF8EDA935EB24D055DEC86E8DF 0B5B9CC4EC1ADEBEAD5C081FE1B3BF9713BC75DCC0E8 FB809FE1DD8514BBD599B3DA69B30F8D92A4436956BCD

	2637F661EB6B871A79288FA7A05A0DDAE83E8479A27FFE 73B45ACD6880D647AED1C3863984E5403DAC07E260595F4 AF461007B13EA66C010F3BF29D PI1 = 1111011101011010110111100110 PI2 = 1000001111101001000110010100 Gold scan chain = 83D9575DCB82DEEF8DFE249569463C5B0F8CAD0EF18637 4B3E1DB396A75E393953AC162AB141F1B77DC9ED0D30E9 EEC42129AEC8951E93449E8944853A434F85F6A28BFAFF5 E43255416CBA80CE00DD063A733FBA5F85D53B865EAF8F 3B1021CB5ACEBBC4DE199FC085BE6503E6D61C90E06BD 856845609F48B41D9B2CA2A490D0684546A79553EC5435D E686B02F5CE7A998FBF88081C8882241C2181FFDE23C6531 42C30F6361ED13BB29D85E45039AA53BF4EC8936E40B149 E7F3ED049000D336790
Enhanced scan	Scan chain1 = 99305DBB35D0429B82F5F1DE4EF9C33B3A39285188E7CF8 A25E43D6B9B75E4C98EC174ED759CB3684315D84BC469B 0E8218C3E0905D46D4AF5E6C61FAA0E9C1FE207EEBC994 6B2B492E6208A6AACCDC3FD78FE63D464E93165B936334 5EC745562E8561E8E920CDD8467F72085F47530F857C82865 E7B66921B8C6B5AD9D2CB5CB31E69DD5E7AA415FFC448 57D04E5A86BB7AA828A5427DD715F096C03E91EC40E11B BC5877431EFDAA03207F0FC457DB10BD99AA04857A8669 7DE222A926957C1AA9EA Scan chain2 = 81325CB335FEC08B36F5F1DE56F1423B321DEED88A774F4 A34640D738B75E0C5DC4116AA379CB7384337C80FC469B2 E8318C9619C5C12D6AE166C61DAA49DE3E0EBFAEFD9A6 6BAB494E6208A43A92D43F57ADEF3BE16EA1B253866734 1E8345D6AEB551F8F920CD996C7F78600F67520C147C8382

	4E7328921B8C6B52D1F6CB5CA61E739E3EFCE827FAC190 AD88492527F3FC4075C8C449E703F42D8038A1DDE9C311F E2BEE0651F80403F4FA0F80EF83347B07E6084A770459EDE 170021E7AF97C751E PI = 1110101100001110001111001111 Gold scan chain = D132DCBB05E4C2B3B6F6F11E4EE4487B323DECD90A474F EA64643D75AB6DE8C79E1006AE279CBF7843B7C88EE469 82F8318C9679D4512D4AE3E6E79FBB0DDA3E16AEAFBD9 866FEBC14E620880AE938C3DD6ABCB79616EA596538725 B046E344963E8465A0F02443D96C7E2078276753CD1C76A7 83CEFB2EE72BAC69D2C952CB5CE41FE3B87FB80000043C FE232474C83FE9F89F147C3FBE78F00FA07FC8BE421E39FF 00F403F1288E75BE8F0BF0F720E08F48E8F17EFB81FEAEE3 E2D9842F75F3B8E970

For logic utilization in each case, Table XIII shows a summary of the versatile/core utilization in the three scan techniques used. Clearly, Clocked scan technique yields the least utilization among the three techniques which is the same result as benchmark designs S1238 and S5378.

Table XIII: Summary of the core utilization for S38417 circuit with different DFT techniques

DFT Technique	Core utilization
Muxed D scan	Used: 18362 , Total: 24576 (74.72%)
Clocked scan	Used: 15429 , Total: 24576 (62.78%)
Enhanced scan	Used: 19611 , Total: 24576 (79.80%)

Another point of comparison is the maximum frequency at which each technique can be functionally correct. The considered paths are the register to register paths. This gives an indication on which DFT technique has the worse effect on the design frequency. Table XIV shows a summary of the minimum clock period for each technique. Unlike other benchmark designs, enhanced scan technique has the lowest maximum frequency.

Table XIV: Summary of the minimum clock period for S38417 circuit with different DFT techniques

DFT Technique	Minimum clock period
Muxed D scan	33.975 ns
Clocked scan	33.518 ns
Enhanced scan	39.430 ns

Another aspect to consider in the comparison is the number of robust paths for path delay testing that the automatic test pattern generation (ATPG) tools can detect while using different DFT techniques. Table XV shows a summary of the number of robust paths in each technique. They are nearly equal in all techniques.

Table XV: Number of robust paths for S38417 circuit with different DFT techniques

DFT Technique	Number of robust paths
Muxed D scan	13
Clocked scan	38
Enhanced scan	13

Similar to S1238 and S5378 benchmark designs, the simulation was done for each DFT technique on original RTL, post-synthesis and post layout to ensure correct functionality in all cases.

5.4.4 Results summary

The first point of comparison is the difficulty of the bring-up of the testing environment for the design. Muxed-D scan DFT technique is the simplest and easiest in this point. It is fully supported by DFT tools and ATPG tools. Also, its scan cell model is easy to implement. Clocked scan DFT technique is also supported by DFT tools and ATPG tools. However, its scan cell model is difficult to implement as it requires alterations in the design DFT exported netlist. Enhanced scan DFT technique is the hardest in bring-up. Although it is built on muxed-D scan technique, it is not supported by DFT tools or ATPG tools. It requires severe alterations in the design DFT exported netlist to add its logic correctly.

The second point of comparison is the FPGA resources utilization. It is clear from the three benchmark designs that clocked scan DFT technique consumes fewer resources compared to Muxed-D scan technique and enhanced scan technique that are close to each other in this point.

The third point of comparison is the maximum frequency at which each technique can be functionally correct. This is an important indication of the amount of added delay by each technique. The results were not very conclusive at this point. Enhanced scan DFT technique was far better in the small benchmark design s1238, but in the middle-sized design s5378, it was nearly the same as the two other techniques, while it was much worse in the largest benchmark design s38417. This means that most probably this point will be design dependent.

The fourth and last point of comparison is the number of robust testable paths as detected by ATPG tools. Results recorded in the benchmark designs show that clocked scan DFT technique is friendlier to ATPG tools and yields larger number of robust testable paths.

Table XVI shows the summary of the comparison between the DFT techniques. The recommendation for fast design bring-up is to use muxed-D scan technique. If the FPGA resources is an issue or if the ATPG tools are struggling to find robust paths with muxed-D scan technique then clocked scan technique will be more suitable. It was expected from enhanced scan technique to shine when it comes to testing with higher frequencies but the last experiment with the large benchmark design showed that it may not always be better to use the enhanced scan technique for this purpose.

Table XVI: summary of the comparison between the DFT techniques

	Muxed-D scan	Clocked scan	Enhanced scan
Easiest Bring-up	1st	2nd	3rd
Resources utilization	2nd	1st	3rd
Maximum frequency	Design-dependent		
Number of robust paths	2nd	1st	2nd

Chapter 6

Conclusion and future work

6.1 Conclusion

In this thesis, a comprehensive comparison between different DFT techniques for TID testing of flash-based FPGAs was made to help designers choose the best suitable DFT technique depending on their application. The comparison included the most famous and used DFT techniques, muxed D scan technique, clocked scan technique and enhanced scan technique. Methodologies for applying the DFT techniques were developed. Points of comparisons included FPGA resources utilization, difficulty of designs bring-up, added delay by DFT logic and robust testable paths in each technique.

The comparison was done using ISCAS'89 benchmarks circuits. Designs of different sized were chosen for the comparison. The designs were implemented using Microsemi ProASIC3 flash-based FPGAs. For each design, the methodologies for applying the three DFT techniques under investigation were applied. Results showed that some techniques were more superior to others depending on the point of comparison.

Recommendations that were reached from the results of the comparison were as follow. For fast design bring-up, it is recommended to use muxed-D scan technique. If the FPGA resources is an issue or if the ATPG tools are struggling to find robust paths with muxed-D scan technique then clocked scan technique will be more suitable. Enhanced scan technique may not always be the best choice when testing with higher frequencies.

6.2 Future work

Although the comparison in this work assumes that all of the DFT techniques will yield the same results under TID effect, it is still important to perform the total ionization dose experiments on the designs and add their results as another field in the comparison. Also, a similar comparison should be made between DFT techniques to generate worst-case test vectors for other types of faults other than delay faults like leakage current faults.

References

[1] A. A. Abou-Auf, “Total-Dose Worst-Case Test Vectors for Leakage Current Failure Induced in Sequential Circuits of Cell-Based ASICs,” *IEEE Trans. Nucl. Sci.*, vol. 56, no. 4, pp. 2189–2197, 2009.

[2] A. A. Abou-Auf, H. A. Abdel-Aziz, and M. M. Abdel-Aziz, “Fault Modeling and Worst-Case Test Vectors for Logic Failure Induced by Total-Dose in Combinational Circuits of Cell-Based ASICs,” *IEEE Trans. Nucl. Sci.*, vol. 57, no. 4, pp. 1978–1985, Aug. 2010.

[3] A. A. Abou-Auf, H. A. Abdel-Aziz, M. M. Abdel-Aziz, and A. G. Wassal, “Fault Modeling and Worst-Case Test Vectors for Leakage Current Failures Induced by Total Dose in ASICs,” *IEEE Trans. Nucl. Sci.*, vol. 57, no. 6, pp. 3438–3442, 2010.

[4] A. A. Abou-Auf, H. A. Abdel-Aziz, and A. G. Wassal, “Worst-Case Test Vectors for Logic Faults Induced by Total Dose in ASICs Using CMOS Processes Exhibiting Field-Oxide Leakage,” *IEEE Trans. Nucl. Sci.*, vol. 58, no. 3, pp. 1047–1052, 2011.

[5] A. A. Abou-Auf, M. M. Abdel-Aziz, H. A. Abdel-Aziz, and A. G. Wassal, “Fault Modeling and Worst-Case Test Vectors of Sequential ASICs Exposed to Total Dose,” *IEEE Trans. Nucl. Sci.*, vol. 59, no. 4, pp. 829–837, Aug. 2012.

[6] A. A. Abou-Auf, M. M. Abdel-Aziz, H. A. Abdel-Aziz, A. G. Wassal, and I. E. Talkhan, “Fault Modeling and Worst-Case Test Vectors for Delay Failures Induced by Total Dose in ASICs,” *IEEE Trans. Nucl. Sci.*, vol. 59, no. 6, pp. 2930–2935, Dec. 2012.

[7] A. A. Abou-Auf, M. M. Abdel-Aziz, M. A. Abdel-Aziz, and A. A. Ammar, “Fault Modeling and Worst Case Test Vector Generation for Flash-Based FPGAs Exposed to Total Dose,” *IEEE Trans. Nucl. Sci.*, vol. 64, no. 8, pp. 2250–2258, Aug. 2017.

[8] M. Bagatin and S. Gerardin, *Ionizing radiation effects in electronics: from memories to imagers*, vol. 50. CRC Press, 2015.

[9] B. D. Sierawski, B. Bhuva, R. Reed, and K. Ishida, “Bias dependence of muon-induced single event upsets in 28 nm static random access memories,” in *2014 IEEE International Reliability Physics Symposium*, 2014, p. 2B.2.1-2B.2.5.

[10] M. Bagatin, A. Coniglio, M. D’Arienzo, and A. De Lorenzi, “Radiation Environment in the ITER Neutral Beam Injector Prototype,” *IEEE Trans. Nucl. Sci.*, vol. 59, no. 4, pp. 1099–1104, 2012.

[11] E. H. Ibe, *Terrestrial radiation effects in ULSI devices and electronic systems*. IEEE, 2015.

[12] J. R. Schwank, M. R. Shaneyfelt, D. M. Fleetwood, and J. A. Felix, “Radiation Effects in MOS Oxides,” *IEEE Trans. Nucl. Sci.*, vol. 55, no. 4, pp. 1833–1853, 2008.

[13] D. Munteanu and J.-L. Autran, “Modeling and Simulation of Single-Event Effects in Digital Devices and ICs,” *IEEE Trans. Nucl. Sci.*, vol. 55, no. 4, pp. 1854–1878, 2008.

[14] N. A. Estep, J. C. Petrosky, J. W. McClory, and Y. Kim, “Electromagnetic Interference and Ionizing Radiation Effects on CMOS Devices,” *IEEE Trans. Plasma Sci.*, vol. 40, no. 6, pp. 1495–1501, 2012.

[15] H. J. Barnaby, “Total-Ionizing-Dose Effects in Modern CMOS Technologies,” *IEEE Trans. Nucl. Sci.*, vol. 53, no. 6, pp. 3103–3121, 2006.

[16] R. D. Schrimpf and D. M. Fleetwood, *Radiation effects and soft errors in integrated circuits and electronic devices*, vol. 34. World Scientific Pub, 2004.

[17] D. M. Fleetwood, “Total Ionizing Dose Effects in MOS and Low-Dose-Rate-Sensitive Linear-Bipolar Devices,” *IEEE Trans. Nucl. Sci.*, vol. 60, no. 3, pp. 1706–1730, 2013.

[18] D. M. Colombo, A. Rosseto, G. I. Wirth, and S. Bampi, “Total Dose Effects on Voltage References in 130-nm CMOS Technology,” *IEEE Trans. Device Mater. Reliab.*, vol. 18, no. 1, pp. 27–36, 2018.

[19] J. J. Wang, S. Samiee, H.-S. Chen, and C.-K. Huang, “Total ionizing dose effects on flash-based field programmable gate array,” *IEEE Trans. Nucl. Sci.*, vol. 51, no. 6, pp. 3759–3766, 2004.

[20] N. Rezzak, J. J. Wang, C. K. Huang, V. Nguyen, and G. Bakker, “Total Ionizing Dose Characterization of 65 nm Flash-Based FPGA,” in *2014 IEEE Radiation Effects Data Workshop (REDW)*, 2014, pp. 1–5.

[21] F. L. Kastensmidt, E. C. P. Fonseca, R. G. Vaz, and O. L. Goncalez, “TID in Flash-Based FPGA: Power Supply-Current Rise and Logic Function Mapping Effects in Propagation-Delay Degradation,” *IEEE Trans. Nucl. Sci.*, vol. 58, no. 4, pp. 1927–1934, 2011.

[22] L.-T. Wang, C. E. Stroud, and N. A. Touba, *System-on-Chip Test Architectures: Nanometer Design for Testability*. San Francisco, UNITED STATES: Elsevier Science & Technology, 2007.

[23] M. L. Bushnell and V. D. Agrawal, *Essentials of electronic testing for digital, memory, and mixed-signal VLSI circuits*. Boston, MA: Kluwer Academic, 2000.

[24] L.-T. Wang *et al.*, *VLSI Test Principles and Architectures: Design for Testability*. San Francisco, UNITED STATES: Elsevier Science & Technology, 2006.

[25] M. Tehranipoor, K. Peng, and K. Chakrabarty, *Test and Diagnosis for Small-Delay Defects*. Springer Verlag, 2011.

[26] A. Krstic, “Delay fault testing for VLSI circuits,” ProQuest Dissertations Publishing, 1998.

[27] *Microsemi ProASIC3 Flash Family FPGAs with Optional Soft ARM Support*, Advanced Datasheet, Actel, Aliso Viejo, CA, USA, March 2016.

[28] I. G. Harris, P. R. Menon, and R. Tessier, “BIST-based delay path testing in FPGA architectures,” in *Proceedings International Test Conference 2001 (Cat. No.01CH37260)*, 2001, pp. 932–938.

[29] F. Brglez, D. Bryan, and K. Kozminski, “Combinational profiles of sequential benchmark circuits,” in *IEEE International Symposium on Circuits and Systems*, 1989, pp. 1929–1934 vol.3.

[30] T.P. Ma, Paul V. Dressendorfer “Ionizing Radiation Effects in MOS Devices and Circuits” Wiley, 1989.

[31] Timothy R Oldham “Ionizing Radiation Effects in MOS Oxides” *International Series on Advances in Solid State Electronics and Technology*, vol. 3, World Scientific, 1999.

[32] M. S. Abdelwahab, M. M. Abdel-Aziz, M. M. Abdelgawad, A. A. Abou-Auf and M. A. Ibrahim, "Worst Case Test Vectors for Sequential Circuits in Flash-Based FPGAs Exposed to Total Dose," *in IEEE Transactions on Nuclear Science*, vol. 66, no. 7, pp. 1642-1650, July 2019.

Appendix A

Automatic generation of FastScan asci files

Python script:

```
import os
import re

if os.path.exists("register_to_register_timing.txt"):
  os.remove("register_to_register_timing.txt")

f1 = open("register_to_register_timing.txt","w+")

input_file = open("Timing.rpt", "r")
for line in input_file:
  endline = re.match( r'END SET Register to Register', line, re.M)
  f1.write(line)
  if endline:
    break

input_file.close()
f1.close()

f2 = open("register_to_register_timing.txt", "r")

flag_main = 1
flag1 = 0
count1 = 0
flag2 = 0
count2 = 0
no_of_paths = 0

for line in f2:
    match = re.match( r'^Expanded Path (\d*)', line, re.M)
    if(match):
        if os.path.exists("WP_%s.asci" %match.group(1)):
            os.remove("WP_%s.asci" %match.group(1))
        path_name = "WP_%s" %match.group(1)
        file_handler = "WP_%s" %match.group(1)
        file_handler = open("WP_%s.asci" %match.group(1), "w+")
        flag_main = 1
        flag1 = 1
        count1 = 0
        no_of_paths = no_of_paths + 1
    elif(flag_main):
        if(flag1):
            count1 = count1 + 1
        if(count1 == 14):
            flag1 = 0
            count1 = 0
            match2 = re.match( r'^\s*\d*.\d*\s*(\S*) \((\w)\)', line,
re.M)
            first_edge = match2.group(2)
            file_handler.write("PATH \"%s\" =\n" %path_name)
            file_handler.write("PIN %s +;\n" %match2.group(1))
            flag2 = 1
            count2 = -1
        if(flag2 and count2 != 2):
            count2 = count2 + 1
            if(re.match( r'^\s*$', line, re.M)):
                flag_main = 0
                flag2 = 0
                count2 = 0
                file_handler.write("END ;\n")
                file_handler.close()
        if(count2 == 2):
            count2 = 0
            match3 = re.match( r'^\s*\d*.\d*\s*(\S*) \((\w)\)', line,
re.M)
            if(match3.group(2) == first_edge):
                edge = "+"
            else:
                edge = "-"
            file_handler.write("PIN             %s              %s;\n"
%(match3.group(1),edge))
```

```
f2.close()

if os.path.exists("worst_path.asci"):
  os.remove("worst_path.asci")
f3 = open("worst_path.asci", "w+")
for path_number in range(1, no_of_paths+1):
  f4 = open("WP_%s.asci" %path_number, "r")
  count = 0
  for line in f4:
    if(count == 0 or count == 1):
      f3.write(line)
    elif(re.match( r'^END ;$', line, re.M)):
      f3.write(previous_line)
      f3.write(line)
      f4.close()
      break
    else:
      previous_line = line
    count = count + 1
f3.close()
```

Appendix B
Automatic DFT netlist alteration for Enhanced scan technique

The below codes are run in the same order as written for correct generation of the netlist after adding the latches. These are perl codes run using a unix environment.

Code 1:

```
#!/usr/bin/perl
if ($ARGV[0]) {
`grep "scan_cell =" $ARGV[0] | sed 's/^.*"\\(\\/.*u\\)".*\$/\\1/' >
dff_cells`;
} else {
`grep     "scan_cell     ="     patterns_detailed.txt     |     sed
's/^.*"\\(\\/.*u\\)".*\$/\\1/' > dff_cells`;
}
`sed -i 's/\\//\\\\/' dff_cells`;
my $fh;
open($fh, "<./dff_cells") or die "Couldn't open file dff_cells, $!";
my $assign = "assign scan_reg = {";
while ( my $line = <$fh> )  {
    chomp $line;
    $assign = $assign."unit.$line .Q, ";
}
close $fh;
$assign = substr $assign, 0, -2;
$assign = $assign."};";
print $assign."\n";
open($fh, ">./assign.txt");
print $fh $assign;
close $fh;
```

Code 2:

```
#!/usr/bin/perl

my $fh1;
my $fh2;
open($fh1, "<./s38417.v") or die "Couldn't open file s1238.v, $!";
open($fh2, ">./s38417_modified.v");
my $ins_name;
my $flag1 = 0;
my $flag2 = 0;
my $flag3 = 0;
my $flag4 = 0;
my $flag5 = 0;
my $flag6 = 0;
my $add_line;
my $input_line;
my $scan_in;
while ( my $line = <$fh1> ) {
    if ($flag1 eq 1) {
      $line =~ m/.Q\s*\(\s*(\S*)\s*\)/;
      print $1."\n";
      if ($1) {
         $out = $1;
         $add_line = "  DLN1 ${ins_name}_latch ( .D( ${out}_latch ) ,
.G( update ) , .Q( $out ) );\n  MX2 ${ins_name}_mux ( .S( scan_en ) ,
.A( $out ) , .B( ${out}_latch ) , .Y( ${out}_mux ) );\n";
         $flag3 = 1;
         $out = substr $out, 1;
         $line =~ s/$out/${out}_latch/;
      } else {
         $flag2 = 1;
      }
      if ($flag6 eq 1) {
        $line =~ m/\s*\(\s*(\S*)\s*\)/;
        $scan_in = $1;
        unless ($scan_in eq "scan_in1") {
          $scan_in = substr $scan_in, 1;
          $line =~ s/$scan_in/${scan_in}_latch/;
        }
        $flag6 = 0;
        print "SI issue fixed in $line \n";
      }
    } elsif ($flag2 eq 1) {
      $line =~ m/\s*\(\s*(\S*)\s*\)/;
      print $1."\n";
      $out = $1;
      $add_line = "  DLN1 ${ins_name}_latch ( .D( ${out}_latch ) , .G(
update ) , .Q( $out ) );\n  MX2 ${ins_name}_mux ( .S( scan_en ) , .A(
$out ) , .B( ${out}_latch ) , .Y( ${out}_mux ) );\n";
      $flag3 = 1;
      $out = substr $out, 1;
      $line =~ s/$out/${out}_latch/;
      $flag2 = 0;
    }
    if ($line =~ m/dfscr\s*(\S*)\s*/) {
      print $line."\n";
      print $1."\n";
      $ins_name = $1;
      $flag1 = 1;
      if ($line =~ m/.SI\s*\(\s*(\S*)\s*\)/) {
        $scan_in = $1;
        unless ($scan_in eq "scan_in1") {
          $scan_in = substr $scan_in, 1;
          $line =~ s/$scan_in/${scan_in}_latch/;
        }
      } else {
        $flag6 = 1;
        print "Couldn't find SI in $line \n";
      }
    } else {
      $flag1 = 0;
    }
    if (($line =~ m/module\s*\S*\s*\(/) and ($flag5 eq 0)) {
      $line =~ s/\(/\( update, /;
      $flag5 = 1;
    } elsif (($line =~ m/input /) and ($flag4 eq 0)) {
      $input_line = "input   update;\n";
      print $fh2 $input_line;
      $flag4 = 1;
    }
    print $fh2 $line;
    if ($flag3 eq 1) {
       print $fh2 $add_line;
       $flag3 = 0;
    }
}
close $fh1;
close $fh2;
```

Code 3:

```
#!/usr/bin/perl
my $fh1;
my $fh2;
open($fh1, "<./s38417_modified.v") or die "Couldn't open file s1238.v,
$!";
open($fh2, ">./s38417_modified2.v");
my $ins_name;
my $flag1 = 0;
my $flag2 = 0;
my $flag3 = 0;
my $flag4 = 0;
my $flag5 = 0;
my $flag6 = 0;
my $add_line;
my $input_line;
my $scan_in;
while ( my $line = <$fh1> ) {
    if ($flag2 eq 1) {
      $line =~ m/\s*\(\s*(\S*)\s*\)/;
      print $1."\n";
      $out = $1;
      $add_line = "  DLN1 ${ins_name}_latch ( .D( ${out}_latch ) , .G(
update ) , .Q( $out ) );\n  MX2 ${ins_name}_mux ( .S( scan_en ) , .A(
$out ) , .B( ${out}_latch ) , .Y( ${out}_mux ) );\n";
      $flag3 = 1;
      $out = substr $out, 1;
      $line =~ s/$out/${out}_latch/;
      $flag2 = 0;
    }
    if ($line =~ m/DFN1\s*(\S*)\s*/) {
      print $line."\n";
      print $1."\n";
      $ins_name = $1;
      if ($line =~ m/.Q\s*\(\s*(\S*)\s*\)/) {
        print $1."##########################\n";
           $out = $1;
           $add_line = "  DLN1 ${ins_name}_latch ( .D( ${out}_latch )
, .G( update ) , .Q( $out ) );\n  MX2 ${ins_name}_mux ( .S( scan_en )
, .A( $out ) , .B( ${out}_latch ) , .Y( ${out}_mux ) );\n";
           $flag3 = 1;
           $out = substr $out, 1;
           $line =~ s/$out/${out}_latch/;
      } else {
         $flag2 = 1;
      }
    }
    print $fh2 $line;
    if ($flag3 eq 1) {
       print $fh2 $add_line;
       $flag3 = 0;
    }
}
close $fh1;
close $fh2;
```

Code 4:

```
#!/usr/bin/perl
my $fh1;
my $fh2;
open($fh1,  "<./s38417_modified2.v")  or  die  "Couldn't  open  file
s1238.v, $!";
`cp s38417_modified2.v s38417_modified2_tmp.v`;
my $ins_name;
my $flag1 = 0;
my $flag2 = 0;
my $flag3 = 0;
my $flag4 = 0;
my $flag5 = 0;
my $flag6 = 0;
my $add_line;
my $input_line;
my $scan_in;
my $mux_out;
while ( my $line = <$fh1> ) {
    if ($line =~ m/MX2\s*\S*\s*\( .S\( scan_en \).*.Y\( (\S*)_mux/) {
      print $line."\n";
      print $1."\n";
      $mux_out = $1;
      $mux_out = substr $mux_out, 1;
      open($fh2, "<./s38417_modified2_tmp.v");
      open($fh3, ">./s38417_modified3.v");
        while ( my $line2 = <$fh2> ) {
          if ($line2 =~ m/\(\s*\\*$mux_out\s*\)/) {
            print "##### match $mux_out\n";
            unless(($line2 =~ m/\s*DLN1/) or ($line2 =~ m/\s*MX2/)) {
              $line2 =~ s/$mux_out/${mux_out}_mux/;
            }
          }
          print $fh3 $line2;
        }
      close $fh3;
      close $fh2;
      `mv s38417_modified3.v s38417_modified2_tmp.v`;
    }
}
close $fh1;
`mv s38417_modified2_tmp.v s38417_modified3.v`;
```

Code 5:

```
#!/usr/bin/perl
my $fh1;
my $fh2;
```

```
open($fh1,  "<./s38417_modified3.v")  or  die  "Couldn't  open  file
s1238.v, $!";
`cp s38417_modified3.v s38417_modified3_tmp.v`;
my $ins_name;
my $flag1 = 0;
my $flag2 = 0;
my $flag3 = 0;
my $flag4 = 0;
my $flag5 = 0;
my $flag6 = 0;
my $add_line;
my $input_line;
my $scan_in;
my $ins_name;
my $DFN1_in;
my $mux_in1;
open($fh3, ">./s38417_modified4.v");
while ( my $line = <$fh1> ) {
    if ($line =~ m/DFN1\s*(\S*)\s*\(.D\s*\(\s*(\S*)\s*\)/) {
      print $line."\n";
      print $1."\n".$2."\n";
      $ins_name = $1;
      $DFN1_in = $2;
      open($fh4, "<assign.txt");
      while ( my $line3 = <$fh4> ) {
        if ($line3 =~ m/unit.\\$ins_name .Q, unit.(\S*) .Q/) {
          $ins_prev = $1;
          print $1."######################\n";
        }
      }
      close $fh4;
      open($fh2, "<./s38417_modified3_tmp.v");
        while ( my $line2 = <$fh2> ) {
          if                        ($line2                          =~
m/DLN1\s*\\${ins_prev}_latch\s*\(\s*.D\(\s*(\S*)\s*\)/) {
            $mux_in1 = $1;
            print $1."#\n";
          }
        }
      close $fh2;
      $second_mux = "  MX2 ${ins_name}_secondmux ( .S( load ) , .A(
$DFN1_in ) , .B( $mux_in1 ) , .Y( ${DFN1_in}_secondmux ) );\n";
      print $fh3 $second_mux;
      $DFN1_in = substr $DFN1_in, 1;
      $line =~ s/$DFN1_in/${DFN1_in}_secondmux/;
    }
    if (($line =~ m/module\s*\S*\s*\(/) and ($flag5 eq 0)) {
      $line =~ s/\(/\( load, /;
      $flag5 = 1;
    } elsif (($line =~ m/input /) and ($flag4 eq 0)) {
      $input_line = "input   load;\n";
      print $fh3 $input_line;
      $flag4 = 1;
    }
  print $fh3 $line;
}
close $fh3;
close $fh1;
`rm s38417_modified3_tmp.v`;
```