

Doctoral Dissertation
Doctoral Program in Computer and Control Engineering (33^{rd} cycle)

New Reliable Operation Infrastructure for Dynamic, High-dependability Applications

Thomas Lange

* * * * *

Supervisors
Prof. Luca Sterpone, Supervisor
Dr. Dan Alexandrescu, Supervisor

Doctoral Examination Committee:
Prof. Fernanda Lima Kastensmidt, Federal University of Rio Grande do Sul
Prof. Luis Entrena, University Carlos III de Madrid
Prof. Milos Krstic, University of Potsdam
Prof. Diana Göhringer, Dresden University of Technology

Politecnico di Torino
April 2021

I hereby declare that, the contents and organisation of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

. . . . . .

Thomas Lange
Turin, April 2021

Summary

Due to the technology scaling, lower supply voltages and higher operating frequencies, modern electronic devices become more and more vulnerable to transient faults. At the same time the number of transistors in a single chip and the complexity of modern systems is increasing. This creates a challenge for performing a reliability assessment on today's circuits and requires many resources in terms of human efforts, processing power and cost. The thesis addresses some of these issues by advancing current analysis techniques on higher abstraction levels, especially focusing on a functional analysis.

In the first part, two new machine learning based approaches are presented to assist the functional failure analysis of complex circuits. The methodologies aim to reduce the efforts needed to determine the functional failure rate of the circuit's sequential logic. The machine learning models use a feature set which was developed to characterizes each sequential element in the circuit. The features combine attributes from static elements and dynamic elements.

The objective of the first approach is to accelerate a fine grained functional failure analysis. It reduces computational cost to determine the Functional De-Rating (FDR) factors of the circuit's sequential logic. The aim is to predict factors per individual instances, which is particularly difficult to obtain using classical approaches such as clustering, selective fault simulation or fault universe compaction techniques. The methodology was applied in a practical example where several machine learning models were evaluated, predicting different failure classes. It was shown that the cost of a fault injection campaign can be reduced by a factor of 2 up to 5 in comparison to a classical statistical fault injection campaign. Further, the ability of the method to be used in early design phases has been assessed. Therefore, a feature subset was identified which can be extracted from an elaborated Register-Transfer Level (RTL) description of the design. The results have shown that the impact on the prediction performance was marginal.

The second machine learning based methodology uses clustering techniques to group flip-flops together which are expected to have a similar contribution to the overall functional failure rate. In this way the fault space is reduced which allows a more efficient fault injection strategy. The advantage in comparison to already other already existing clustering approaches is that this approach is more flexible and no assumption of the circuit or its representation is made. The effectiveness of the grouping by different

machine learning clustering algorithms was evaluated on a practical example and compared to a random and ideal solution. With the approach it was possible to reduce fault injection efforts by a factor of 5× to 20×.

Typical design flows are hierarchical and rely on assembling many individual technology elements from standard cells to complete boards. Providers use compact models to provide simplified views of their products to their users. Designers group simpler elements in more complex structures and have to manage the corresponding propagation of reliability and functional safety information through the hierarchy of the system, accompanied by the obvious problems of IP confidentiality, the possibility of reverse engineering, etc. Therefore, in the second part of the thesis, a methodology is proposed which aims to help experts to deal with the complexity of hierarchical modelling of reliability and functional safety metrics. The presented approach allows the use, elaboration and distribution of compact machine learning models in a uniform and systematic manner, minimizing both human and CPU efforts while maintaining high accuracy and fidelity. The trained machine learning models will be able to quickly evaluate a large variety of effects, encapsulating very useful reliability and functional safety data in a compact and efficient solution that can be used and reused further down the design and manufacturing flow.

In the third part of the thesis, the focus is shifted to Single-Event Transients (SETs) in Clock Distribution Networks (CDNs). A methodology is proposed to analyse how SETs in the clock distribution network are impacting the functional behaviour of a circuit. A methodology and a fault model are presented which implement the main radiation-induced effects in clock networks. The method enables the computation of the functional failure rate in a logic-level simulation based on the RTL description of the design. Thus, a faster evaluation can be performed than by simulating on the electrical level or gate-level. The analysis is extended by introducing a Temporal Masking effect for SETs in clock distribution networks. The Temporal Masking is based on the shortest input path delay of the flip-flops and the shortest output path delay, which are defining an SET Timing Window within the clock cycle. SETs occurring outside of this window are masked. The fault model was extended considering this Temporal Masking effect which allows to compute the functional failure rate weighted with Temporal De-Rating (TDR) of a circuit. The approach was applied in a practical example where SET were injected into the clock network of the circuit under test in a fault injection campaign. It was shown that the proposed Temporal Masking implementation is able to compute a pessimistic worst case.

Acknowledgements

Firstly, I wish to show my gratitude to my thesis supervisor Luca Sterpone, Vice Head of the Department of Control and Computer Engineering of Politecnico di Torino and Associate Professor in the CAD group, for supporting my research, for his guidance, criticism and encouragement.

To my thesis co-supervisor Dan Alexandrescu, CEO of iRoC Technologies, I wish to express my sincere appreciation. I am grateful for his support and encouragement, as well as for granting me time to write my thesis.

My sincere thanks also go to the entire staff at iRoC Technologies for providing a professional environment which allowed me to substantially learn about the field of Soft Errors. Especially, I am grateful to Dr. Maximilien Glorieux, with whom I collaborated on several topics.

Finally, I thank Maksim Jenihhin, Professor in the Department of Computer Systems of Tallinn University of Technology, for originating and managing the RESCUE project, which funded my research position. The project gave me the opportunity to meet other Early Stage Researchers and have enriching discussions at the numerous workshops.

Contents

List of Tables

List of Figures

Glossary

ASIC Application Specific Integrated Circuit. 53, 55, 60, 62

CDN Clock Distribution Network. iv, 2, 3, 86

CMOS Complementary Metal-Oxide-Semiconductor. 8

CPU Central Processing Unit. 15, 24, 31, 55

CRC Cyclical Redundancy Check. 23

CUT Circuit Under Test. 14

DRAM Dynamic Random-Access Memory. 7

ECC Error Correcting Code. 55

EDC Error Detection Correction. 55

EDR Electrical De-Rating. 8, 68

FDR Functional De-Rating. iii, 13, 27, 31, 60, 68, 76, 78, 85

FET Field Effect Transistor. 7

FIFO First-In First Out. 23, 74

FIT Failure in Time. 1, 62, 81

FPGA Field Programmable Gate Array. 7

FuSa Functional Safety. 53, 55, 57, 64

GRF Geomagnetic Rigidity Factor. 59

IC Integrated Circuit. 1, 2

IP Intellectual Property. 55

LDR Logical De-Rating. 12, 60, 68

MBU Multiple Bit Upset. 54

MCU Multiple Cell Upset. 54, 55

ML Machine Learning. 57

MOS Metal-Oxide-Semiconductor. 7

NF Neutron Flux. 59

PDK Process Design Kit. 53, 54

PIPB Propagation Induced Pulse Broadening. 12

PLL Phase Locked Loop. 54, 68

PVT Process, Voltage, Temperature. 66

PW Pulse Width. 10, 11, 58, 62

RTL Register-Transfer Level. iii, iv, 69, 71, 74, 86

SBU Single Bit Upset. 54, 55

SDF Standard Delay Format. 76

SEB Single Event Burnout. 7

SEE Single-Event Effect. 5, 8, 16, 54, 55, 59

SEFI Single-Event Functional Interrupt. 7

SEGR Single-Event Gate Rapture. 7

SEL Single-Event Latch-up. 7

SEMT Single-Event Multiple Transient. 6

SEMU Single-Event Multiple Upset. 6

SER Soft Error Rate. 2, 16, 58–60, 62

SET Single-Event Transient. iv, 2, 3, 6, 8, 54, 58, 82, 83, 86, 87

SEU Single-Event Upset. 6, 8, 54, 55

SRAM Static Random-Access Memory. 1

STA Static Timing Analysis. 74

TDR Temporal De-Rating. iv, 9, 60, 68, 87

TID Total Ionizing Dose. 66

TMR Triple Modular Redundancy. 45

Chapter 1

Introduction

The advancement in technology enabled the development of complex electronic systems and made it possible to embed more and more functionality into single devices and manufacture Integrated Circuits (ICs) with tens of millions of transistors. This trend led to a higher integration of electronics in our day-to-day lives, including critical applications where human lives is at stake. Especially, for these critical applications a reliable operation is extremely important.

A significant threat to the reliable operation of a system are transient faults induced by natural radiation. Without any mitigation the fault can propagate and manifest as an error in the system. This generated error can then lead to misbehaviour of the functionality and thus, creating a failure [57].

In the past radiation induced faults were mainly a problem for applications which demand a very high reliability, such as aero-space, nuclear facilities, and medical devices. However, due to the technology advancements the size of transistors is scaling down, devices operate with lower supply voltages and at higher operating frequencies which makes them more vulnerable to faults induced by radiation [8, 9]. Thus, radiation induced faults become also a greater hazard for systems working in normal environments and it is no longer possible to ignore them.

Today's reliability standards and customers' expectations set tough targets for the quality of electronic devices and systems. Among other reliability threats, the transient faults are known to contribute significantly to the overall failure rate of the system and possibly exceeding the set reliability targets. As an example, standard flip-flops and Static Random-Access Memory (SRAM) memories, manufactured in relatively recent technologies exhibit error rates of hundreds of events (Failure in Times (FITs); failures that can be expected in one billion working hours of operation) [9, 67]. Complex circuits using such cells can easily overshoot 10 FIT target mandated by the ISO 26262 standard [38] for an automotive ASIL D application.

A vast amount of research is devoted to analyse and mitigate these effects, with the focus on achieving accurate results when analysing relatively small circuits. Current

trends show, however, that the total number of ICs is growing and a large part of performance increase nowadays comes from parallelism, which translates in an increase of the total number of transistors per IC (Moore's Law [52]). This results in larger and more complex systems and the required effort, in terms of human resources, computation power and cost, is vastly increasing.

1.1 Structure of the Thesis

The focus of the thesis is on the advancement of the soft error analysis on higher abstraction levels. The main contributions consist of improvement in fault-injection simulations, a technique to reduce the fault space and a technique to model Single-Event Transients (SETs) in Clock Distribution Networks (CDNs) on the functional level.

The thesis will be organized as follows:

- Chapter 2: This chapter provides a summary of single-event effects, their causes and the basic masking mechanism. An overview of techniques to analyse fault propagation and determine the de-rating factors are described. Finally, the common methodology to calculate the overall Soft Error Rate (SER) is presented.

- Chapter 3: This chapter investigates the usage of machine learning techniques for the fault analysis on the functional level. Two new machine learning based methodologies are presented, which are using a feature set developed to characterise each sequential element in the circuit.

  The first approach aims to accelerate fine grained functional fault analysis on sequential cells. It uses regression models in a supervised approach to train machine learning models on the functional failure rates of the sequential cells in the circuit. The objective of the second approach is to reduce the fault space of the functional fault analysis. The technique is based on machine learning clustering techniques and groups flip-flops together which are expected to have a similar sensitivity to faults.

- Chapter 4: This chapter proposes a machine learning based methodology to tackle the complexity of hierarchical modelling of reliability and functional safety metrics. The presented approach allows the use, elaboration and distribution of compact machine learning models in a uniform and systematic manner, minimizing both human and CPU efforts while maintaining high accuracy and fidelity.

- Chapter 5: This chapter describes a methodology to simulate Single-Event Transients in the Clock Distribution Network on a higher abstraction level. A fault model was developed which implements the main radiation induced faults in clock networks by performing logic-level simulation. Additionally, the Temporal Masking effect for this type of faults is explored and the fault model is extended accordingly.

- Chapter 6: This is the last chapter and concludes the thesis and discusses plans for future work.

Overall, the thesis seeks to advance the fault analysis on the functional level. Therefore, chapter 2 describe the state-of-the-art methodologies and Chapter 3 applies Machine Learning techniques and thus, present novel methods to increase the efficiency of the fault analysis. Chapter 5 shifts the focus to the analysis of SETs in CDNs. Together, these techniques facilitate the analysis of radiation induced faults in complex integrated circuits on a higher abstraction level.

Chapter 2

Single-Event Effects

The amount of information processed of modern system is increasing which requires a higher performance and results in a higher complexity of the implemented circuits. This trend is also requested in markets which demand a high reliability and are usually running safety-critical applications, such as aerospace and the automotive market. These new requirements lead to the usage of smaller transistors technology, lower voltages and higher operating frequencies which makes them more prone to radiation induced transient faults. In order to cope with the additional complexity and new technologies of modern systems the research aims at the development of new assessment and mitigation techniques for transient faults and new test methodologies for complex electronic systems.

Due to technology scaling, lower supply voltages and higher operating frequencies, modern circuits become more and more vulnerable to reliability threats. Additionally, today's reliability standards and customers' expectations set tough targets for the quality of electronic devices and systems. Especially, transient faults, such as Single-Event Upsets and Single-Event Transients in the individual sequential and combinational cells, have been identified as the leading contributor to the overall failure rate for many applications [8, 9, 66].

2.1 Mechanism and Classification

Single-Event Effects (SEEs), are caused by a single, energetic particle striking through a silicon device and depositing electrical charge, as shown in figure 2.1. On ground, the main sources of these energetic particles are alpha particles, high-energy neutrons and thermal neutrons [9]. The radiation environment in space is much harsher. Primarily, there are cosmic rays which are consisting of protons, alpha particles and heavy nuclei [35]. The earth's atmosphere and magnetic field blocks most of these cosmic particles. However, these particles can strike atoms in the atmosphere and create chains of secondary and tertiary particles [82].

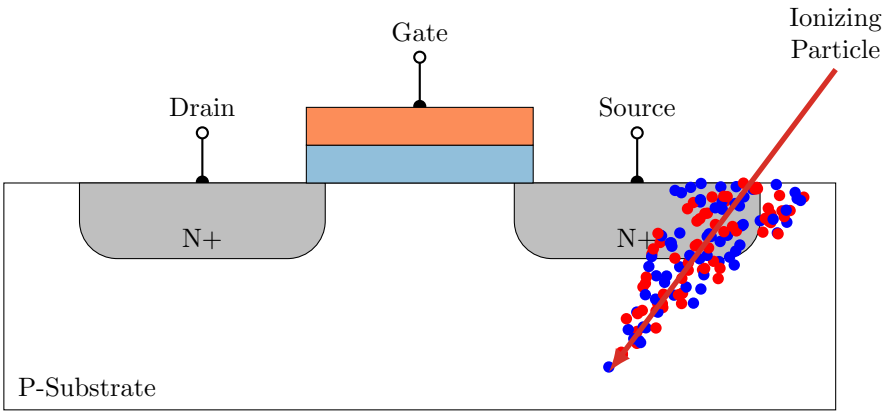


Figure 2.1: Energetic particle striking through a transistor.

The occurring subsequent effects in an electronic circuit, due to radiation effects, depend on the type of circuit, the energy of the particles and the nature of the strike. The effects can be classified in hard errors, soft errors and cumulative effects. While it may be possible to recover from soft errors by a reset of the application, a power cycle of the device, or by overwriting the corrupted information, hard errors are non-recoverable. In the following a summary of the different effects is provided.

2.1.1 Soft Errors

Single-Event Transient

This effect occurs when the energetic particle is depositing the charge near the combinational logic cells of the circuit. The charge is then collected at one of the transistor nodes of the logic cells and thus, creating a transient current pulse, also called SET. SETs usually appear in the combinational cells of the circuit. They can also appear in the sequential element (such as a flip-flop or a latch) in the case the Single-Event only affects the output stage but not the internal feedback loop of the cell [3]. When a single particle affects more than one combinational gate, the effect is called Single-Event Multiple Transient (SEMT) [34]. This fault becomes an error when it propagates through the combinational logic and eventually is latched by at least one of the sequential elements during the sampling window [29].

Single-Event Upset

A Single-Event Upset (SEU) occurs when the induced charge of the energetic particle is causing the logic state of a discrete sequential element, such as a latch, a flip-flop or a memory cell, to change. In the case where a single particle affects more than one sequential cell, the effect is called Single-Event Multiple Upset (SEMU) [21].

Single-Event Functional Interrupt

The Single-Event Functional Interrupt (SEFI) refers to a loss of functionality in a more complex device, e.g. Dynamic Random-Access Memory (DRAM), Field Programmable Gate Array (FPGA), Microprocessor, etc. This can be the case, for example, when an SEU occurs in the control logic of the circuit. An SEFI does not lead to a permanent damage and can be recovered by resetting the device or performing a power cycle.

Single-Event Latch-up

A Single-Event Latch-up (SEL) is a latch-up caused by a radiation induced event. The deposited charge by the energetic particle triggers a parasitic structure in a transistor and thus, creating a short circuit from power to ground [68]. This disrupts the function of the transistor and due to the increased current consumption, can lead to a destruction of the part and thus, to a hard error. A power cycle is required in order to restore to normal operation.

2.1.2 Hard Errors

Single-Event Burnout

When the energetic particle causes a power Field Effect Transistor (FET) to enter a second breakdown, the event is called Single Event Burnout (SEB). This results in a very high current and causes the device metal traces to vaporize, bond wires to fuse open and silicon regions to melt due to the thermal increase [68].

Single-Event Gate Rapture

The Single-Event Gate Rapture (SEGR) is initiated when the energetic particle strikes through a Metal-Oxide-Semiconductor (MOS) power transistor. The additional induced carriers are accelerated by the electric field, which in turn increases the field and eventually may cause the dielectric to breakdown [68]. This permanently damages the transistor.

2.1.3 Cumulative Effects

Cumulative effects describe the long-term effects on the electronic devices, and the resulting degradation, due to exposure to radiation.

Total Ionizing Dose

The deposited charge by the radiation can be trapped in the oxide layer of the transistor. This can cause a shift in the switching characteristics, increased device leakage and

power consumption, and timing changes. This leads to a decreased functionality of the transistor and in the worst case can cause a circuit failure.

Displacement Effects

When highly energized particles striking the device they may displace the atoms in the silicon lattice. Although Complementary Metal-Oxide-Semiconductor (CMOS) devices are known not to be sensitive to displacement damage, bipolar and optical devices are very sensitive to this effect.

2.1.4 Summary of Radiation Effects

In order to analyse the overall SEE sensitivity of a system, all the described effects must be considered. However, in large digital circuits, the majority of failures are usually caused by SEU and SET. Therefore, the focus of this thesis is on analysing these effects.

2.2 Masking Mechanisms

When a radiation induced fault propagates in the system it can lead to observable effects up to the system level. Fortunately, not all faults necessarily manifest themselves as errors or failures in the system. There are several effects which can mask the event and de-rate the raw rate of faults. These effects are quantified by the de-rating factors which then can be used to determine the effective error rate.

The de-rating factor in this work is defined in such a way that, a value of 1.0 indicates that the faults are not masked and all of them propagate, leading to an error. A value of 0.0 on the other hand indicates that all faults are masked and none of them propagates.

In [55, 2] four main masking mechanisms are defined with their corresponding de-rating factors which can reduce the effect of SEEs on the actual error rate significantly.

2.2.1 Electrical Masking

Digital logic operates between logical zero and one values. To discriminate between these values the signals have to cross a switching threshold. The current pulse created from an SET may not be strong enough to change the voltage above the switching threshold on the effected node. Further, due to the capacitance and limited slew rates of the digital logic gates, a transient current pulse is usually narrowed and its rise and fall time is increased during the propagation. By the time it reaches the end of the path, either it has been completely filtered or the voltage transition is below the switching threshold! [32, 75]. The reduction of the fault rate due to these effects can be described as Electrical De-Rating (EDR).

2.2.2 Temporal Masking

Almost all digital circuits are synchronous. This means for a fault to propagate, it must be sampled by the sequential elements. A fault must occur within an opportunity window (OW) to be latched by a sequential element on the downstream path. If the fault misses this opportunity window it does not affect the behaviour of the circuit and is masked [65, 56, 30, 13]. The Temporal De-Rating (TDR) describes the reduction of the fault rate due to these effects and can be calculated depending on the type of fault.

Temporal De-Rating for SEUs

An SEU in a sequential logic must occur sufficiently early in the clock cycle in order to meet the setup time of the flip-flops on the downstream path. Figure 2.2 shows the masked and unmasked case of an SEU in a flip-flop which is propagating to the flip-flops connected on the downstream path. Unless the SEU is masked otherwise, the fault will always propagate through the combinational network and reach the next sequential logic stage. The faulty flip-flop value remains on the output until the new value is latched in the next clock cycle.

The occurrence of an SEU can be considered uniform within the clock cycle. The opportunity window depends on the slack of the considered path, as well as the setup and hold time of the flip-flop on the downstream path [57]. The Temporal De-Rating for SEUs $\mathrm{TDR}_{\mathrm{SEU}}$ is then defined as the ratio between the opportunity window and the clock period T_{cycle}

$$\mathrm{TDR}_{\mathrm{SEU}} = \frac{t_{\mathrm{slack}} + \frac{t_{\mathrm{setup}}}{2} - \frac{t_{\mathrm{hold}}}{2}}{T_{\mathrm{cycle}}} \tag{2.1}$$

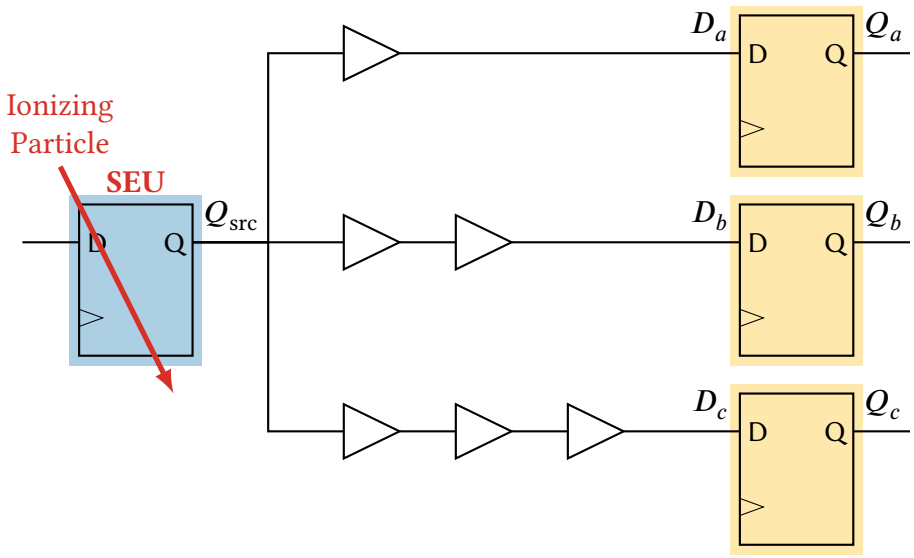
According to the equation it can be derived that the opportunity window for an SEU is increasing with the slack of the downstream path and thus, the probability of an SEU being captured. In the case the SEU caused a setup or hold validation it is uncertain whether the fault is captured or not. The Temporal De-Rating can also be approximated with the ratio of the path slack to the clock period

$$\mathrm{TDR}_{\mathrm{SEU}} \approx \frac{t_{\mathrm{slack}}}{T_{\mathrm{cycle}}} \tag{2.2}$$

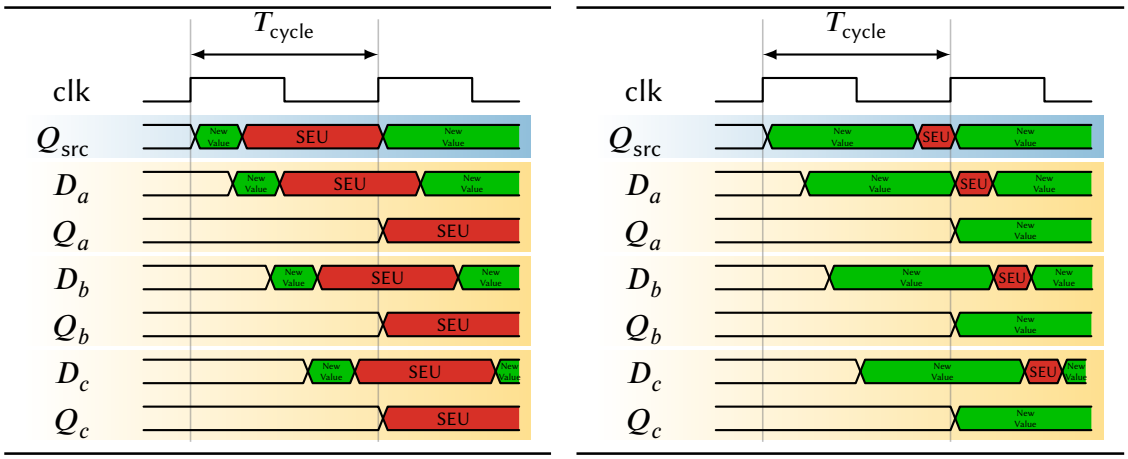
For the critical path in the circuit the slack is very low. For these paths the opportunity window is quite short which reflects in a low probability for the fault to propagate. Therefore, it can be reasoned that the Temporal Masking of SEUs is increasing with higher clock frequencies and vice versa, is increasing with lower clock frequencies.

Temporal Masking for SETs

Similar to SEUs, SETs can also be subject to temporal masking. Single-Event Transients cause short pulses on the output of the affected cell. This transient pulse has to occur



(a) SEU in a Flip-Flop



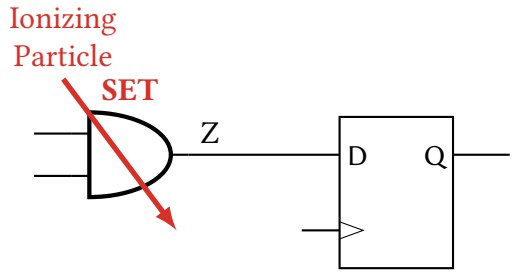
(b) SEU sampled (not masked)

(c) SEU temporally masked

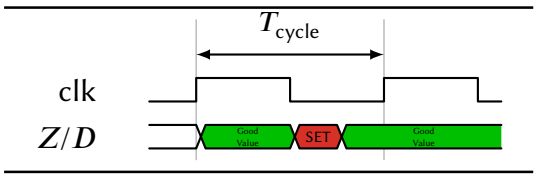
Figure 2.2: Temporal Masking of SEUs.

and propagate to the input of the sequential logic during its latching window. The masked and unmasked cases of an SET are shown in figure 2.3.

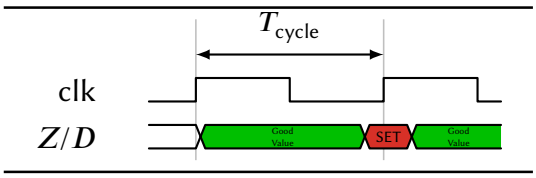
Unless otherwise masked, the SET is propagating thorough the combinational network and eventually, will reach a sequential element. Since the occurrence of the SETs can be considered as uniform within the clock cycle, a first approximation of the Temporal De-Rating TDR_{SET} can be calculated as the ratio of the transient's Pulse Width



(a) SET in a combinational cell



(b) SET masked



(c) SET sampled (not masked)

Figure 2.3: Temporal Masking of SETs.

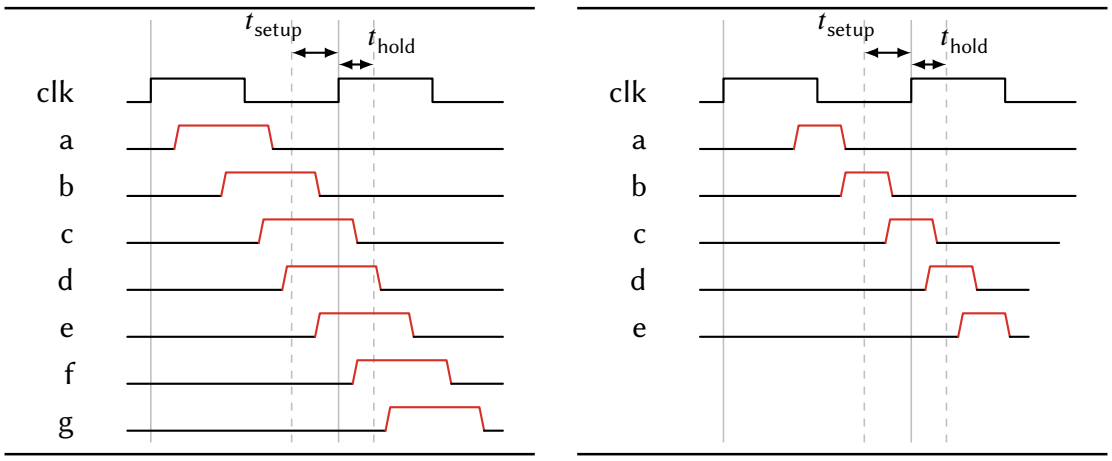
(PW) to the clock period T_{cycle}

$$\mathrm{TDR}_{\mathrm{SET}} = \frac{\int_{w=\min(\mathrm{PW})}^{w=\max(\mathrm{PW})} w \, \mathrm{d}w}{T_{\mathrm{cycle}}} \tag{2.3}$$

Many of today's circuit still run with clock periods in the nanosecond range. For recent technologies, studies show that the pulse duration of the combinational cells is in the range of picoseconds [19]. With the equation (2.3) it can be concluded that the Temporal Masking for SETs is quite high. However, the trend of modern circuits is going to higher operating frequencies which decreases the Temporal Masking and as further studies suggest, the effect of SETs become severe [56, 31, 47].

SET Pulse Alignment Cases The exact probability of the transient pulse to be latched depends on when it arrives at the input of the sequential element and how it is aligned to the sampling clock edge. Figure 2.4 shows all of these possible alignment cases. Figure 2.4a shows the case when the PW of the transient is longer than the setup-hold window of the sequential element and figure 2.4a shows the case when the pulse width is shorter. An Overlapping Width is defined as the part of the pulse that falls within the setup-hold window. In the case the pulse width is longer than the setup-hold window ($\mathrm{PW} > t_{\mathrm{setup}} + t_{\mathrm{hold}}$), the probability the fault is captured is considered to be proportional to the ratio of Overlapping Width (OW) to the setup-hold window $\frac{\mathrm{OW}}{t_{\mathrm{setup}}+t_{\mathrm{hold}}}$. In the case the pulse width is shorter than the setup-hold window ($\mathrm{PW} < t_{\mathrm{setup}} + t_{\mathrm{hold}}$),

the caused violations by the fault are treated together. The error latching probability is assumed to be linear with the Overlapping Width ratio [19].



(a) PW > $t_{setup} + t_{hold}$ (b) PW < $t_{setup} + t_{hold}$

Figure 2.4: SET pulse alignment cases.

For each case the corresponding latching probability is shown in table 2.1. Since an SET can occur uniformly within the clock cycle, the latching probability can be averaged over the full clock period. The overall latching probability for both cases is then $\frac{\mathrm{PW}}{T_{\mathrm{cycle}}}$.

For this analysis, the discussed pulse width in figure 2.4 and table 2.1 is the one seen at the input of the sequential element. As the transient pulse propagates through the combinational network, it might be distorted. This effect is called Propagation Induced Pulse Broadening (PIPB) [26, 73, 27].

2.2.3 Logical Masking

If the propagation of a fault is prevented due to the state of the combinational network the fault is considered as logically masked. This can be due to a controlling input of a gate such as a zero value on a 2-input AND gate, as shown in figure 2.5. An SEU can either be masked in the same cycle it occurs (figure 2.5a), or it may be masked several cycles later (figure 2.5b). An SET can masked in the same way as SEUs as shown in figure 2.5c [57, 78].

The reduced fault rate due to these effects are quantified by the Logical De-Rating (LDR). For the evaluation of the LDR only a representation of the circuit without any knowledge of the function is required. Usually, the propagation of the fault starting from the output of the affected cell to the inputs of a sequential element is considered.

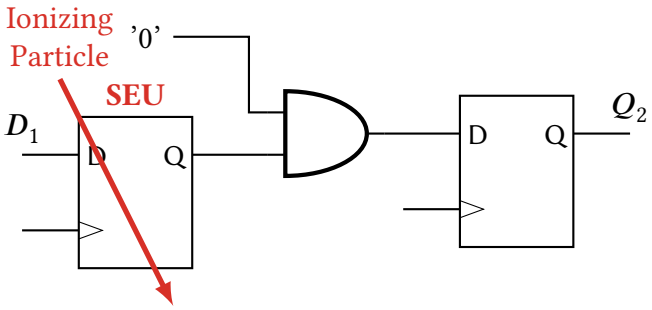
Table 2.1: SET Pulse Alignment Cases and Corresponding Capture Probabilities

Pulse Width	Case	Error Probability	Comments
$PW > t_{setup} + t_{hold}$	a	0	correct value latched
	b	$\frac{1}{T_{clk}} \cdot \int_0^{t_{setup}} \frac{OW}{t_{setup}+t_{hold}} dOW$	setup time violation
	c	$\frac{1}{T_{clk}} \cdot \int_{t_{setup}}^{t_{setup}+t_{hold}} \frac{OW}{t_{setup}+t_{hold}} dOW$	hold time violation
	d	$\frac{PW-t_{setup}-t_{hold}}{T_{clk}}$	wrong value latched
	e	$\frac{1}{T_{clk}} \cdot \int_{t_{hold}}^{t_{setup}+t_{hold}} \frac{OW}{t_{setup}+t_{hold}} dOW$	setup time violation
	f	$\frac{1}{T_{clk}} \cdot \int_0^{t_{hold}} \frac{OW}{t_{setup}+t_{hold}} dOW$	hold time violation
	g	0	correct value latched
$PW < t_{setup} + t_{hold}$	a	0	correct value latched
	b	$\frac{1}{T_{clk}} \cdot \int_0^{t_{setup}} \frac{OW}{t_{setup}+t_{hold}} dOW$	setup time violation
	c	$\frac{PW \cdot (t_{setup}+t_{hold}-PW)}{T_{clk} \cdot (t_{setup}+t_{hold})}$	metastability
	d	$\frac{1}{T_{clk}} \cdot \int_0^{t_{hold}} \frac{OW}{t_{setup}+t_{hold}} dOW$	hold time violation
	e	0	correct value latched
Overall	-	$\frac{PW}{T_{clk}}$	-

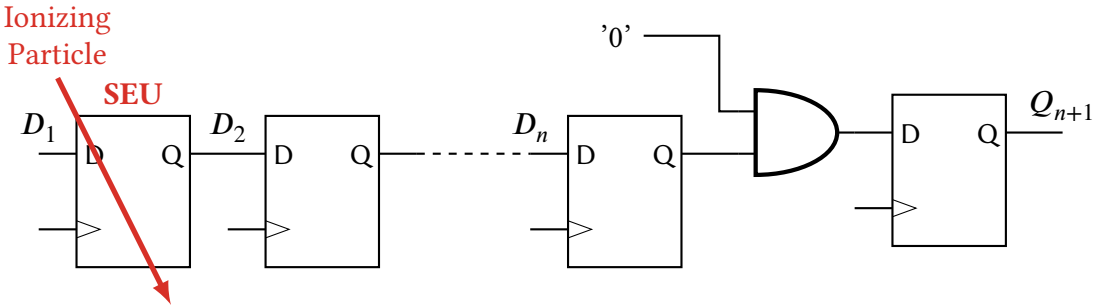
2.2.4 Functional Masking

For the functional masking the fault is considered at an applicative level and takes the actual usage of the circuit and the function of the system into account. This means even when an SEU or SET is able to propagate (is not masked by any other masking effect), the impact at the function of the circuit can vary. The SEU or SET might significantly change the state of the circuit, however in many cases is benign. The fault could, for example, lead to a faulty pixel in a video stream or cause a delay of a data packet in a networking application. The effect on the functional behaviour in these cases is probably very minor.

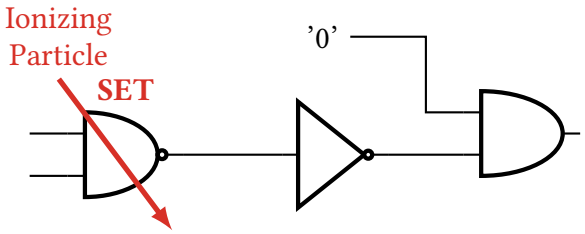
The de-rating due to the functional behaviour of the circuit is referred to as Functional De-Rating (FDR) and can be quite significant [71, 57]. In comparison to the other masking effects the de-rating highly depends on the criteria defining the acceptable behaviour of the circuit during the execution of an application. This can include objective



(a) Logical Masking of an SEU



(b) Logical Masking of an SEU after n clock cycles



(c) Logical Masking of an SET

Figure 2.5: Logical Masking of SEUs and SETs.

aspects, as well as subjective aspects to define a fault-free or faulty behaviour of the system. An objective criterion would be for example, the behaviour of the primary outputs of the Circuit Under Test (CUT). Any difference to the fault-free reference could be classified as a failure. A subjective criterion, however, might take the actual criticality of the observed difference into account. The observed difference in the primary output might not be relevant for specific cases or applications and thus, can be ignored for the analysis. Usually, the effects of errors are divided into classes based on their system-level severity (correctable, uncorrectable, not detected by the hardware but detected by the software, if a retry is possible, if there is a time limit to receive the correct result, etc.).

This means, that there is subjectivity involved in the failure classification. The functional failure analysis is largely driven by the actual usage of the circuit and different

function modes or applications will lead to different failure rates. In the cases the circuit under test has a clearly defined function, the functional failure analysis has to take this specific function into account and the Functional De-Rating characterises faults masked due to this functional behaviour. The analysis becomes more complex for more general circuits, such as Central Processing Units (CPUs), since the actual application is unknown and the usage field is large.

The Functional Masking analysis takes the propagation of the fault over several clock cycles into account. Commonly it is evaluated if the fault has been silently masked and does not have any further effect on the functional behaviour of the circuit. Another usual criterion which is evaluated is, if the fault remains in the circuit in a latent sate which does not have any observable effect on the function. Additionally to these evaluations usually, several failure classes are defined for a given application which result in not just a single Functional De-Rating factor.

2.2.5 Summary of Masking Mechanisms

The described masking/de-rating mechanisms are used to evaluate the probability of the propagation of a fault and are usually determined by using probabilistic algorithms and simulation-based approaches. All the different evaluation steps can require significant investment in terms of human efforts, processing resources and licenses for different tools. Thereby, especially the simulation-based approaches to determine the Functional De-Rating are very computationally intensive. Techniques to determine the different de-rating factors and how to calculate an over error rate induced by soft errors are described in the next section.

2.3 Soft Error Analysis

While in the past Single-Event Effects were mostly considered in radiation harsh environments, such as space, recent technologies may suffer from Single-Event Effects even in terrestrial applications [8, 9]. For instance, Single-Event Effects may happen even at high altitude for avionic applications, close to nuclear reactor or just from the natural decay in the device material itself. Therefore, it is fundamental to investigate and characterize the susceptibility of a device to Single-Event Effects. Hence, in the last decades research put a lot of efforts in estimating the devices and applications dependability with respect to SEUs and SETs [45, 58, 10, 46, 20].

As discussed in the previous sections, the majority of faults does not propagate due to the various masking mechanism. It is necessary to determine the corresponding derating factors, in order to obtain a realistic failure rate of the system which is not fully overestimated. A high-level view of the Single-Event Effects analysis from the fault to system level failures is shown in figure 2.6. The faults which are induced by the sequential elements and the combinational cells are shown on the left. Due to the various

masking effects many of the faults are not propagating (Electrical Masking, Temporal Masking and Logical Masking). The lowering of the probability is quantified by the respective de-rating factors. After applying the de-rating factors, the effective rate of observable errors is calculated. Afterwards the effect of the errors on the functional behaviour of the system is analysed. This is done in the Functional Masking analysis and quantified by the Functional De-Rating. Obtained are the results which characterises the probability for the defined failures on the system level [23].

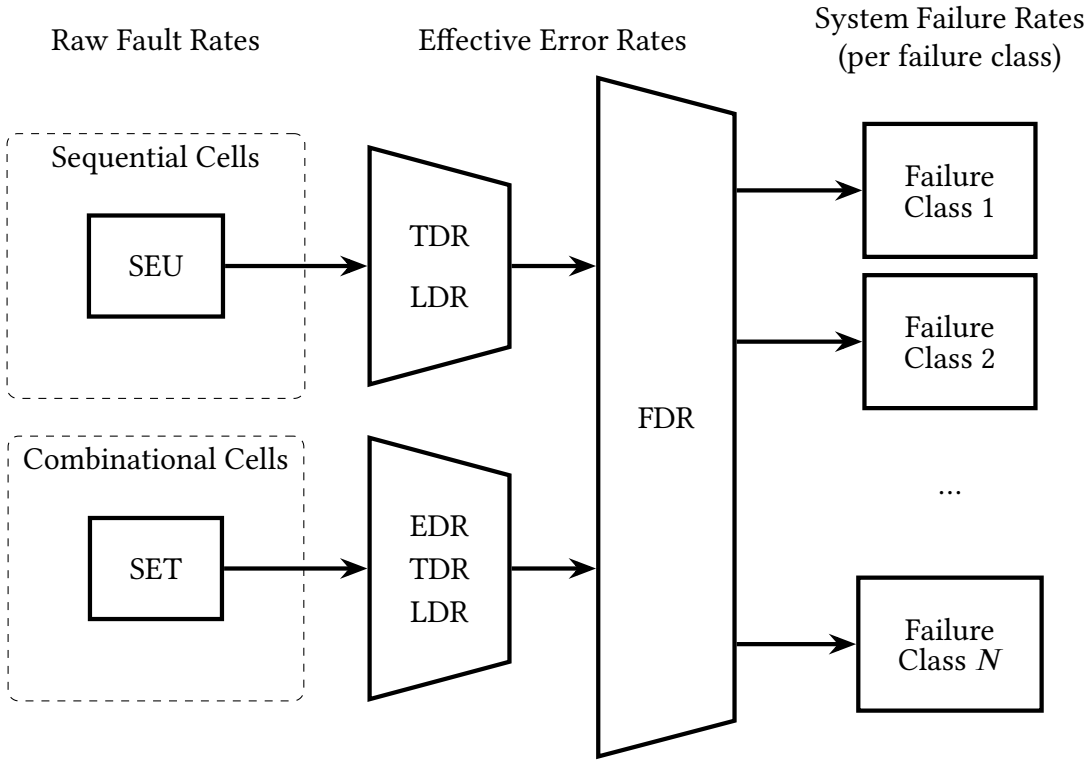


Figure 2.6: Single-Event Effects analysis.

The objective of the Soft Error analysis is to determine the probability of radiation induced faults resulting in a system-level error. The considered system-level failures must be defined prior to the analysis. With the quantify failure rates of the defined failure classes it can be ensured that the design meets the specified targets and complies with certain standards and regulations (e.g., ISO 26262 [38], DO-254 [25], IEC 61508 [37]]).

To summarize, the Soft Error analysis aims to compute the rate of system-level failures starting from the rate of technology level faults due to soft errors. In the following the three main steps are presented to perform a SER analysis. The technology SER characterization of standard cells and memory blocks, the various de-rating factors and finally the overall SER calculation. The SER methodology presented focuses on non-destructive SEEs: Bit Upsets in sequential cells, Transients in combinational cells.

2.3.1 Technology Soft Error Rate Characterization

The technology SER characterization is the first step of the SER analysis. Raw SER data should be provided in terms of raw (intrinsic) rate/probability of occurrence of logic SEU or SET for combinational, sequential and memory cells for a specific environment. The final operating environment should be also carefully analysed, of a particular interest to most commercial and aeronautical applications is the natural background, terrestrial environment characterized by a natural contribution of atmospheric neutrons and internal alpha particles from contaminants. The neutron SER is specific to the technology and the environment (altitude and localization). The alpha contribution depends strongly on the sensitivity of the cell to alpha particles and the alpha emissivity rate of the packaging materials.

2.3.2 De-Rating Characterization

The most common techniques to evaluate the propagation of a fault are based on fault injection or analytical approaches. In the fault injection analysis, many faults are injected into the circuit while it is measured if there is any observable effect. The fault injection can be performed on a model of the circuit. Depending on the used implementation of the design, such as the RT-level description or the gate-level netlist, the accuracy of the results might differ and which de-rating factors can be obtained.

An alternative to the fault injection techniques are the analytical approaches. They use a mathematical or abstracted model of the circuit to evaluate the effect of faults. Further techniques are based on a formal representation of the circuit as well as a definition of properties the circuit has to meet. Using tools for model checking or symbolic simulation, it can be verified whether the properties hold in the presence of faults. Another set of approaches is based on associating probability distributions to the nodes in a circuit and then using these to compute the probability that faults on a given node will propagate to an output.

2.3.3 Computing the Overall Soft Error Rate

To obtain the overall Soft-Error Rate (SER) for a full chip the SER of the used technology needs to be combined with the determined de-rating factors. For a given application and a specific failure class, the overall failure rate $\mathrm{SER}_{\mathrm{total}}$ can be calculated by

$$\mathrm{SER}_{\mathrm{total}} = \mathrm{SER}_{\mathrm{seq}} + \mathrm{SER}_{\mathrm{comb}} \tag{2.4}$$

where $\mathrm{SER}_{\mathrm{seq}}$ is the contribution of the sequential elements, such as flip-flops and latches, and $\mathrm{SER}_{\mathrm{comb}}$ the contribution of the combinational elements, such as gates.

The individual parts are then calculated by

$$\mathrm{SER}_{\mathrm{seq}} = \sum_{i \in \text{Flip-Flops}} \mathrm{FIT}_{\mathrm{SEU},i} \cdot \mathrm{TDR}_i \cdot \mathrm{LDR}_i \cdot \mathrm{FDR}_i \tag{2.5}$$

$$\mathrm{SER}_{\mathrm{comb}} = \sum_{i \in \text{Gates}} \int_{w_{\min}}^{w_{\max}} \mathrm{FIT}_{\mathrm{SET},i}(w) \cdot \mathrm{EDR}_i(w) \cdot \mathrm{TDR}_i(w) \cdot \mathrm{LDR}_i \cdot \mathrm{FDR}_i \, \mathrm{d}w \,. \tag{2.6}$$

where

- $\mathrm{FIT}_{\mathrm{SEU},i}$ represents the SEU rate for the sequential element i,
- $\mathrm{FIT}_{\mathrm{SET},i}(w)$ represents the SET rate of the combinational element i depending on the transient pulse width w and
- the corresponding de-rating factors EDR_i, TDR_i, LDR_i, FDR_i for element i

Chapter 3

Machine Learning Techniques for Functional Failure Rate Analysis

3.1 Introduction

In chapter 2 it was discussed that flip-flops are the major contributor to the overall Soft-Error Rate of integrated circuits and one of the major metrics used in today's functional safety analysis are de-rating or vulnerability factors. Using classical methods, such as fault simulations, to obtain accurate per-instance Functional De-Rating data for the full list of circuit instances is a complex and computationally intensive task. Therefore, the objective of the work presented in this chapter is to explore how machine learning techniques can be used to advance the functional failure analysis of complex circuits.

Previous works have shown that certain characteristics of the circuit, such as structural properties and signal probabilities, can be related to the masking effect and thus, used to estimate vulnerability factors [14, 79, 61, 63]. Since machine learning algorithms are very suitable to learn even complex relationships, it can be expected that these models are able to learn and predict the Functional De-Rating by using similar circuit features.

Two approaches are proposed in the following sections which accelerate and assists the functional failure analysis with the help of machine learning techniques. They aim in reducing fault injection efforts in terms of computing resources, human efforts and tool licenses. The first approach uses supervised machine learning algorithms to learn and estimate the Functional De-Rating factors for individual flip-flops in the circuit. The second approach uses machine learning clustering techniques to group flip-flops with a similar sensitivity to faults together and thus, reducing the fault space.

3.2 Machine Learning

Machine Learning is the concept of a machine learning from examples and making predictions based on its experience, without being explicitly programmed. Machine learning algorithms are usually built upon a mathematical model which uses sample data (also called training data) in order to make predictions. The machine learning process generally consists of two phases, namely the training or learning phase and the prediction phase. The learning phase can be further grouped in [5, 12]

- supervised learning,
- unsupervised learning and
- reinforcement learning

The supervised learning algorithms try to model the dependency between the inputs and the target output in such a way that the output values for new data points can be predicted based on the learned relationships. The input data is used for training and consists of a set of training samples. A training sample describes the input values and the corresponding and expected output value. For the mathematical model, the training samples are represented by a vector, also called feature vector. The full training data is then represented by a matrix. in an iterative optimization process the training data is used to adjust parameters of the underlying mathematical function of the model. The learned function can be used to predict an output associated with new data input samples, which were not a part of the training data [62, 51, 5].

The main tasks of supervised learning models are classification and regression. While classification algorithms are used when the outputs are restricted to a limited set of values, regression algorithms are used when the outputs may have any numerical value within a range [5].

In contrary to supervised learning, unsupervised learning models try to find structures in the data set without external labelling or classification. This means, that the training data samples contain only input values for the model and not the expected corresponding output values. Instead of optimizing a mathematical function, unsupervised learning algorithms search for similarities in the training data. The reaction of the model on new data inputs is based on the found similarities and depends on how close each new data point resembles to the already trained data [12].

The two main tasks in this type of unsupervised machine learning methods are clustering and dimensionality reduction. In the cluster analysis the set of training data is divided into subsets, called clusters. Data points within the same cluster are similar according to the predefined criteria, while data points from different clusters are dissimilar. The various clustering algorithms make different assumptions on the structure of the data and therefore, use different similarity metrics to evaluate the similarity of

the members of the same clusters or evaluate the difference between clusters. The dimensionality reduction is a process to reduce the dimension of the feature set, often by eliminating or extract features.

The first developed approach which is presented in section 3.5 is based on supervised learning models and predicting Functional De-Rating factors of individual flip-flops by performing a regression. The second approach presented in section 3.6 uses clustering analysis to group flip-flops together with a similar sensitivity to faults in order to decrease the faults space. Both approaches are based on a feature set which is presented in the next section.

3.3 Flip-Flop Feature Set

For the machine learning based approaches proposed in the following sections a feature set was developed, which characterises each flip-flop in the circuit. The extraction of the flip-flop feature set needs to be efficient in order to compete with the classical fault injection approach. Therefore, the developed feature set to characterise each flip-flop instance, contains only simple characteristics which are easy and fast to obtain. This feature set combines static elements, such as cell properties, circuit structure and synthesis attributes, as well as dynamic elements, such as the signal activity. The initial version of the feature set was published in [42] and was then extended in [44] and [43]. The features are described in detail in table 3.1.

The feature set can be divided into 3 parts. The structural related features which describe a flip-flop in relation with other flip-flops without taking the (technology dependent) combinational logic into account. The synthesis related features which describe a flip-flop with their assigned cell attributes and the combinational logic at the input and output. These synthesis related features can differ from one technology to another. To consider the workload of the circuit, a variety of features are evaluated to describe the dynamic behaviour of the flip-flops. As an initial approach it was considered that the information related to the signal activity (state distribution, transitions) could be an appropriate starting set of features.

The structural and synthesis related features can be extracted from the gate-level netlist of the circuit. The gate-level netlist is converted into a graph representation. Graph algorithms, such as Dijkstra's algorithm to find the shortest path, can be used to extract the features. The structural related features, can also be extracted just by using the elaborated RTL description of the design. The elaboration of the RTL description is faster than performing a full synthesis and does not necessarily need a full-fledged synthesis tool. This has the advantage that the analysis can already be performed in an early design stage. The signal activity related features are extracted by running a simulation using the corresponding testbench instantiating the RTL description or the gate-level netlist of the circuit and tracing the simulation signals.

Table 3.1: Feature Set to Characterise a Flip-Flop Instance FF_i

Feature Name	Type	Description
Structural Related Features		
# FF at Startpoint/Endpoint	Numerical	The number of flip-flops directly connected to/by FF_i of the next/previous flip-flop stage.
# Connections from/to FF	Numerical	The number of flip-flops connected to/by FF_i within the circuit (over several flip-flop stages).
# Connections from/to Primary Input/Output	Numerical	The number of primary inputs/outputs connected to/by FF_i.
# FF Stages to/from Primary Input/Output (max/avg/min)	Numerical	Number of flip-flop stages from/to the primary input/output of the circuit.
# Constant Drivers	Numerical	The number of constant drivers directly connected to FF_i.
Has Feedback	Categorical	Describes if FF_i has a feedback loop (directly or over several flip-flops stages).
Feedback Depth	Numerical	The depth of the shortest feedback loop in terms of flip-flops stages.
Is Part of Bus	Categorical	Describes if FF_i is part of a bus signal.
Bus Position	Numerical	Describes the position of FF_i within the bus.
Bus Length	Numerical	Describes the total length of the bus signal FF_i is part of.
Bus Label	Categorical	The number/label of the bus signal FF_i is part of.
Module Label	Categorical	The number/label of the hierarchical module FF_i is part of.
Signal Activity Related Features		
@0/@1	Numerical	The relative time FF_i output is at logical 0/1.
State Changes	Numerical	The number of state changes.
Synthesis Related Features		
Drive Strength	Numerical	The drive strength of the FF_i cell instance.
Depth combinational Path	Numerical	The depth of the combinational stage from FF_i output.
# Combinational Cells at/from Input/Output	Numerical	The total number of combinational cells in FF_i fan-in/fan-out cone.

3.3.1 Feature Preprocessing

In general, machine learning algorithms perform better if the data set is standardised. Ideally the data set is normally distributed with zero mean and unit variance. Further, for most machine learning algorithms categorical values have to be represented as numbers. They can be converted by using an ordinal encoder. However, such a number representation could be interpreted as if the categories are in a specific order. This is often not the case and therefore, a one-hot encoding should be used. This encoding transforms each categorical feature with n categories into n binary features.

The presented feature set consists of two different types of features, the numerical and categorical type. The numerical features are expressed as a number and the categorical features can be expressed by a number, label or Boolean value. The number representation for the numerical feature does have a meaning as a measurement. However, if a categorical feature is represented by a number, it does not have a mathematical meaning and the assignment is arbitrary.

For the presented approach in this chapter, the feature set can be extracted for the full circuit before the training of the models. This has the advantage that the distribution of the values is already known beforehand. Therefore, the standardisation on the numerical features and the one-hot encoding on the categorical features is applied on the full feature set before the training.

3.4 Circuit Under Test

The two proposed techniques which are presented in the following sections are applied on a practical example. The used circuit under test is the Ethernet 10GE MAC Core from OpenCores. This circuit implements the Media Access Control (MAC) functions as defined in the IEEE 802.3ae standard [36]. The 10GE MAC core has a 10 Gbps interface (XGMII TX/RX) to connect it to different types of Ethernet PHYs and one packet interface to transmit and receive packets to/from the user logic [6]. The circuit consists of control logic, state machines, First-In First Outs (FIFOs) and memory interfaces. It is implemented at the Register-Transfer Level (RTL) and is publicly available on OpenCores.

The corresponding testbench writes several packets to the 10GE MAC transmit packet interface. As packet frames become available in the transmit FIFO, the MAC calculates a Cyclical Redundancy Check (CRC) and sends them out to the XGMII transmitter. The XGMII TX interface is looped-back to the XGMII RX interface in the testbench. The frames are thus processed by the MAC receive engine and stored in the receive FIFO. Eventually, the testbench reads frames from the packet receive interface and prints out the results [6]. During the simulation all sent and received packets to and from the core are monitored and recorded. This record is used as the golden reference for the fault injection campaign.

The elaboration of the RTL implementation of the circuit identifies 1234 flip-flops.

The design was synthesised with Synopsys Design Compiler by using the NanGate FreePDK45 Open Cell Library [74]. Due to logic optimization by the synthesis tool the obtained gate-level netlist contains 1202 flip-flops, 32 flip-flops less than the elaborated RTL implementation.

3.4.1 Failure Classes and Fault Injection Campaign

In order to evaluate the performance of the developed machine learning approaches in the following sections, the sensitivity of the considered design was measured by performing an exhaustive flat statistical fault injection campaign. In this way the sensitivity of each flip-flop was determined and thus, the Functional De-Rating. For comparison, the fault injection campaign was performed on both, the RTL implementation and gate-level netlist of the circuit. The fault injection mechanism is implemented by inverting the value stored in a flip-flop using a simulator function.

For the analysis, three different failure classes are considered. In case the injected fault propagates to the primary outputs of the circuit and thus, the output values are altered in comparison to the golden reference, an *Output Failure* is counted. If the payload of the final received packets is corrupted or the circuit stopped sending or receiving any data, the simulation run was considered as an *Application Failure*. In networking applications, such as the considered design, important data is protected by checksums. This means that a minor payload corruption can be handled by the error correction algorithm. However, in case the fault causes the circuit to stop working and interrupting the flow of sending packages or data is continuously corrupted, then the effect can be considered as critical and a *Critical Failure* is counted.

In each flip-flop 200 faults were injected at a random time during the active phase of the test-case. The fault injection simulations were parallelised by using 7 independent processes. To perform the fault injection campaign on the RTL design in total 12 hours 14 minutes and 31 seconds were needed. Performing the campaign on the gate-level netlist required in total 19 hours 38 minutes and 3 seconds[1]. The failure rate of each flip-flop is calculated by dividing the number of simulation runs which lead to a failure with the number of total simulation runs. The failure classes Output Failure, Application Failure and Critical Failure are considered separately and the overall results of the flat statistical fault-injection campaign are presented in table 3.2.

Register-Transfer Level Design Versus Gate-Level Netlist

The elaborated RTL design is not optimized by a synthesis tool and thus, contains 32 more flip-flops than the final gate-level netlist. To perform a comparison, the flip-flops from the gate-level netlist have been matched with the corresponding RTL signal names.

[1]Computations were performed on a PC with an Intel Xeon E5-2687W CPU (8 cores/16 threads 3.10 GHz).

Table 3.2: SEU Fault Injection Campaign Results

	RT-Level		Gate-Level Netlist	
	Total	Per Injection	Total	Per Injection
Injection Targets (FFs)	1234	-	1202	-
Injected Faults (SEU)	246800	-	240400	-
Output Failure	90306	36.59 %	84159	35.01 %
Application Failure	55187	22.36 %	53756	21.78 %
Critical Failure	14012	5.68 %	12332	5.13 %

The fault injection simulation campaign on the RTL design was performed with the same parameters (number of injections and injection time) as for the gate-level netlist. The failure rates for almost all flip-flops were identical, except of the additional flip-flops and some flip-flops which are connected to/by these additional flip-flops. The difference between the failure rates were less than 2 %.

3.4.2 Feature and Target Pre-Analysis

With the considered circuit under test and the results from the performed fault injection campaigns, the relationship between the features itself and the features and the target output can be pre-analysed. The gate-level netlist and the testbench of the circuit under test are used to extract the respective features described in 3.3 for each flip-flop. The feature extraction is automated and overall, takes 186 seconds.

The Pearson Correlation is used to analyse the relationship between the features and the targets. The Pearson Correlation is a measure of linear correlation between two sets of data, by calculating the covariance of two variables and dividing it by the product of their standard deviations. It is essentially a normalised measurement of the covariance, which reflects a linear correlation between the variables. The results of the Pearson Correlation are shown in figure 3.1

The matrix in figure 3.1 shows that some features have light correlation between each other. This can indicate some redundancy in the feature set. Analysing the correlation between the features and the target output, the Functional De-Rating factors, some correlation can be observed between the failure rate and

- the total number of connections from the flip-flop FF_i to the primary output,
- the number of flip-flop stages to the primary output (minimum, average, maximum),
- the feedback loop and the depth of the feedback loop and
- the bus the flip-flop FF_i is part of.

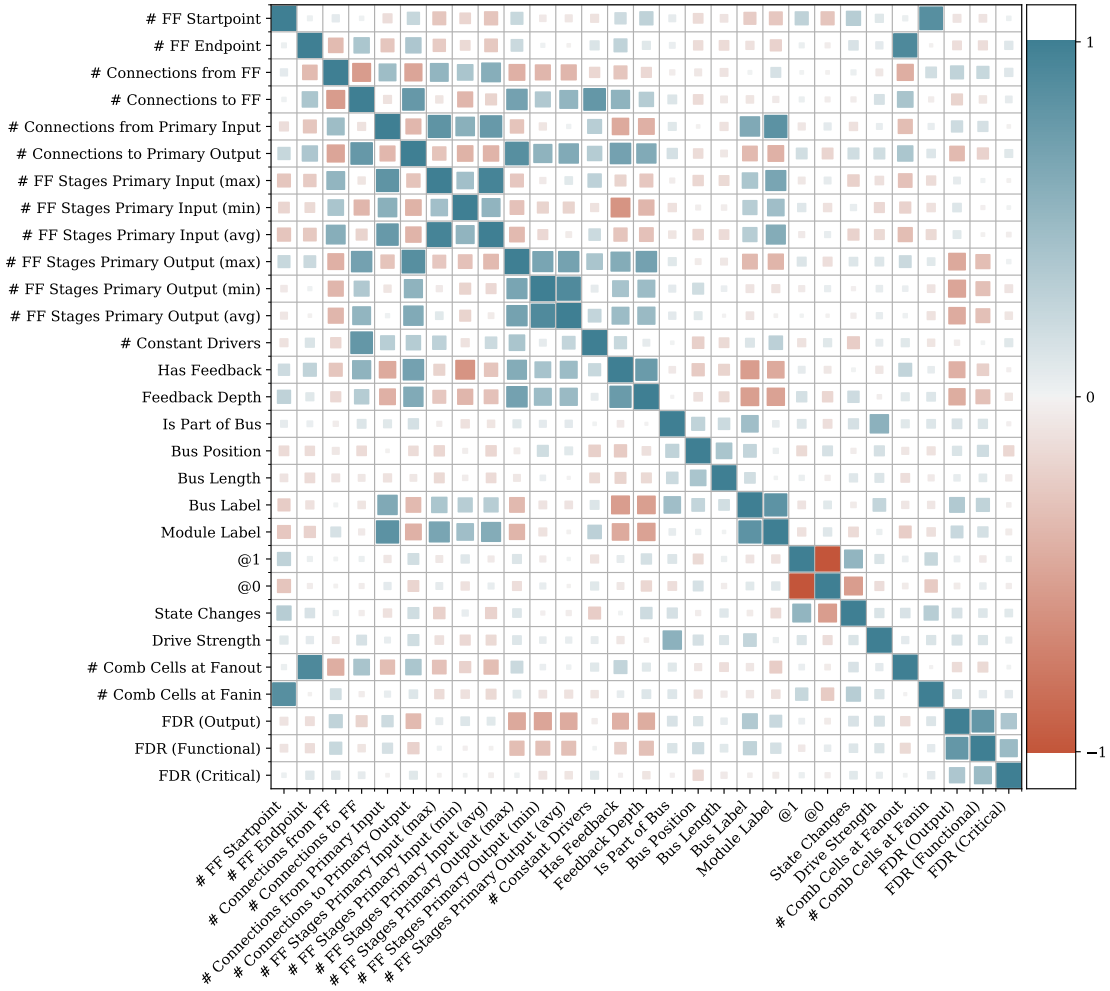


Figure 3.1: The Pearson Correlation of features and targets.

The correlation is decreasing for the different failure classes, starting highest from the Output Failure class, Functional Failure class to the Critical Failure class.

Some minor correlations can be observed for

- the total number of connections from and to the flip-flop FF_i,
- the total number of connections from the primary input to the flip-flop FF_i and
- the module the flip-flop FF_i belongs to.

Here, the correlation is also decreasing for the different failure classes.

The observed correlation is a first indication that machine learning models might be able to learn the relationship between the derived features and the failure rate. It should be noted however, that the Pearson Correlation only shows linear correlation.

Depending on the machine learning model, it might even be able to learn more complex relationships than linear dependencies.

3.5 Machine Learning Regression for Predicting Functional De-Rating Factors

Determine accurate de-rating factors of each instance in the circuit is an exhaustive procedure and especially on the functional level computationally intensive fault injection campaigns are necessary. In this section a machine learning based methodology is proposed which estimates the Functional De-Rating factors for individual flip-flops in the circuit with the help of supervised regression models. These models try to learn the dependency between a set of input features and the target output variable, usually based upon a mathematical model. The initial methodology was presented in [42], an improved approach with an extended analysis was published in [44]. In the next sections, the proposed methodology is presented in detail with an extended evaluation of the performance.

3.5.1 Methodology

The approach aims to predict the Functional De-Rating (FDR) factor of individual flip-flop instance. The target variable, the Functional De-Rating factor, is a continues variable with values in the range of [0.0; 1.0] (see section 2.2) which makes regression models suitable for the learning and prediction task.

The implemented methodology uses a supervised approach to train the regression models and is shown in figure 3.2. The RTL description or the gate-level netlist (GLN) of the circuit and a corresponding testbench are used to extract the features for each flip-flop in the circuit, as described in section 3.3. These extracted features function as the input of the model. The circuit description and the testbench are also used in a statistical fault injection simulation to determine the FDR factors for one part of the circuit. The determined FDR factors per flip-flop and the associated flip-flop features form the training data set, used to train the machine learning model. Eventually, the trained model can be used to estimate the FDR values of the remaining flip-flops which were not used for the training.

An important parameter of the procedure is the training size. The training size defines the size of the training data set. This is connected to the number of reference FDR factors needed for the training and thus, determines how many fault injection simulations need to be performed. This basically means, that the training size also controls the savings of fault injection efforts gained by the presented approach. A larger training size would probably result in more accurate prediction results, but more efforts for the fault injection simulation campaigns. In contrary, very low training size would require low efforts for the fault injection campaigns but might result in a poor prediction

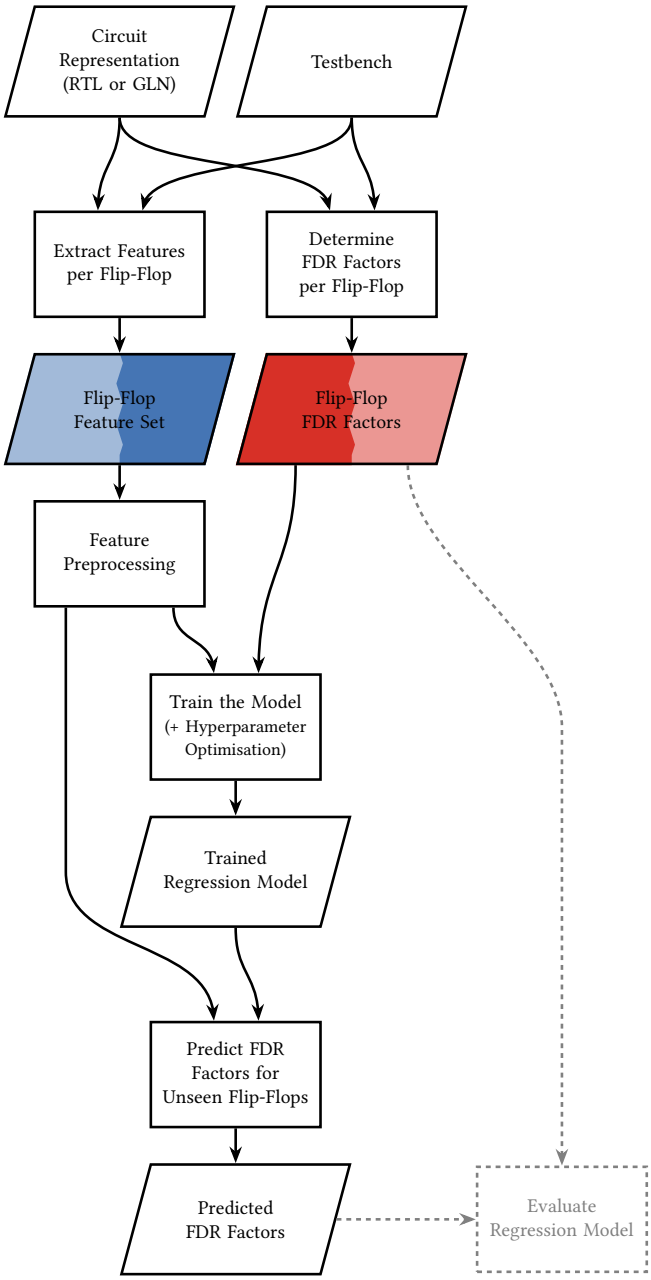


Figure 3.2: Functional De-Rating estimation and evaluation flow.

accuracy. It is important to find a good training size in order to save sufficient fault injection efforts, but maintain a good prediction accuracy. This will be analysed in detail in section 3.5.2.

Model Training and Hyperparameter Optimisation

Machine learning models are usually represented by internal parameters or an internal state. These parameters or the state are determined during the training process by the machine learning algorithm. Additionally, most of the machine learning algorithms can be controlled by hyperparameters. In contrast to the internal parameters or state, these hyperparameters are not derived by the training algorithm and need to be manually set before the training process.

The problem of finding the optimal set hyperparameters for the model is called hyperparameter optimisation. Therefore, several instances of the model need to be trained and evaluated for different tuples of hyperparameters. The tuple that minimises a predefined loss function or evaluation metrics yields an optimal model. A common approach to perform the optimisation is a random search method combined with a grid search method. There, the model is first evaluated for parameter values randomly generated in a given distribution. Afterwards a more detailed grid search is performed within the region of the values obtained by the random search [11]. This is the used approach in the proposed methodology and indicated in figure 3.2.

Model Performance Evaluation

The performance of the machine learning model is required to perform the hyperparameter optimisation. Further, in the following section the performance of a model is used to compare and evaluate various regression models with using different training data sizes.

In order to measure and evaluate the performance of a machine learning model different metrics are used to cover certain aspects of the prediction performance. In the following description of the considered metrics, $\hat{y}_i$ is the value of the i-th sample predicted by the model and y_i is the corresponding true/expected value.

Mean Absolute Error The mean absolute error (MAE) describes the average absolute difference of the expected values to the predicted values. It is calculated over n_{samples} by the following equation (values closer to zero are better)

$$\text{MAE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} |y_i - \hat{y}_i| \tag{3.1}$$

Maximum Absolute Error The maximum absolute error (MAX) describes the maximum difference of the expected values to the predicted values. The equation

$$\text{MAX}(y, \hat{y}) = \max_{i \in [1, n_{\text{samples}}]} |y_i - \hat{y}_i| \tag{3.2}$$

calculates the metrics (values closer to zero are better).

Root Mean Squared Error The root-mean-square error (RMSE) describes the square root of the quadratic error of the expected values. In comparison to the mean absolute error the root-mean-square error gives a higher weight to larger errors. It is calculated over n_{samples} by the following equation (values closer to zero are better)

$$\text{RMSE}(y, \hat{y}) = \sqrt{\frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} (y_i - \hat{y}_i)^2} \tag{3.3}$$

Explained Variance The Explained Variance (EV) measures the proportion to which a model accounts for the variation (dispersion) of a given dataset. If $\text{Var}(X)$ is the variance, the square of the standard deviation, of a random variable X then the explained variance is calculated by

$$\text{EV}(y, \hat{y}) = 1 - \frac{\text{Var}(y - \hat{y})}{\text{Var}(y)} \tag{3.4}$$

The best possible value is 1 and lower values are worse.

Coefficient of Determination The coefficient of determination (R^2) provides a measure of how well future samples are likely to be predicted by the model. If $\bar{y}$ is the mean of the expected values, the coefficient of determination can be calculated by

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n_{\text{samples}}} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n_{\text{samples}}} (y_i - \bar{y})^2} \tag{3.5}$$

and the best possible value is 1 (lower values are worse).

Cross-Validation A problem when evaluating the performance of a machine learning model can occur, when only one training and test data set is used to calculate the considered performance metrics. The used training and test data set can be particularly good or bad and bias the results in a certain direction. Cross-validation is used to ensure that the model is not only trained for one particular training and test data set. There, the model is trained and evaluated against multiple train and test splits of the data. Several subsets, or cross validation folds, of the data set are created and each fold is used to train and evaluate a separate model. Thus, instead of relying only on one single training and test data set, a more stable performance measure is obtained which indicates how the model is likely to perform on average [40]. For the detailed evaluation in the next section a cross-validation of 10 is used.

3.5.2 Evaluation of Machine Learning Regression Models for Predicting Functional De-Rating Factors

In this section the presented methodology is applied on a practical example and various machine learning models are evaluated and compared with using different training data

sizes. Therefore, the results of the full flat statistical fault injection campaign described in section 3.4.1 are used to get the Functional De-Rating factors for each flip-flop of the gate-level netlist. The netlist and the testbench of the described circuit are used to extract the respective features for each flip-flop. The full dataset provides an objective measure to evaluate the presented technique. Part of the dataset is used to train the machine learning models. The remaining part of the dataset which was not used for training, is used to evaluate the trained models.

The performance of the prediction is evaluated for different training sizes by using several metrics. Additionally, the fit and prediction time for different training sizes were measured and compared to each other, as well as the time to perform the hyperparameter optimisation, which is compared against the full fault injection campaign[2].

Evaluated Regression Models

Several machine learning models were used to predict the two failure classes, Output and Application Failure. Therefore, the data obtained from the fault injection campaign and the extracted flip-flop features form the training and test data set. All evaluated models are implemented using Python's scikit-learn Machine Learning framework [59].

Prior to the model training, the feature set is preprocessed, by removing the mean and scaling to unit variance on the numerical features and applying a one-hot encoding on the categorical features. Further, since the FDR factors are within the range of 0 to 1, the predicted values are clipped to the expected output range. For the hyperparameter search and evaluation a cross validation fold of 10 and a training size of 50 % are used. In order to keep the computational time low and get a substantial advantage over the fault injection simulation, the number of hyperparameter combinations was chosen, such that the total needed training time is about 30 minutes.

The investigated models are briefly described in the following (for a more detailed description see [53], [51] or [5]) and the prediction performances for the two failure classes are given in table 3.3. For a selection of models, the prediction of one test data fold is shown in figure 3.3 for the Output Failure and figure. 3.5 for the Application Failure. Further, figure 3.4 and figure 3.6 show the learning curves, which describes the performance of the model for different training sizes.

Linear Least Squares Regression The Linear Least Squares algorithm is based on a linear model. The target output variable is represented as a linear combination of the input feature variables. Thereby, the algorithm targets to minimise the sum of squared residuals, the squared sum of the difference between the true value in the training dataset and the predicted value by the linear approximation.

[2] All computations were performed on a PC with an Intel Xeon E5-2687W CPU (8 cores/16 threads 3.10 GHz).

Table 3.3: Performance Results for the Evaluated Regression Models (With Cross Validation = 10 and Training Size = 50 %).

(a) Output Failure

Regression Model	MAE	MAX	RMSE	EV	R^2	Hyperparameter Combinations	Training Time / s	Fit Time / s	Prediction Time / s
Linear Least Squares	0.048	0.991	0.141	0.795	0.794	-	-	0.005	0.001
k-Nearest Neighbors	0.028	0.782	0.096	0.904	0.903	5260	311	0.004	0.06
Decision Tree	0.035	0.755	0.105	0.887	0.887	175 000	1864	0.004	0.001
Ridge w/ Linear Kernel	0.042	0.768	0.104	0.889	0.889	70 000	1786	0.006	0.002
Ridge w/ Polynomial Kernel	0.041	0.718	0.102	0.894	0.893	20 000	1820	0.007	0.002
Ridge w/ RBF Kernel	0.037	0.757	0.102	0.894	0.894	50 000	1618	0.007	0.003
Ridge w/ Sigmoid Kernel	0.11	0.737	0.16	0.743	0.742	25 000	2083	0.033	0.002
SVR w/ Linear Kernel	0.037	0.816	0.109	0.88	0.879	75 000	1917	0.125	0.017
SVR w/ Polynomial Kernel	0.036	0.797	0.109	0.879	0.878	15 000	2003	0.116	0.014
SVR w/ RBF Kernel	0.03	0.792	0.095	0.907	0.906	50 000	1867	0.077	0.016
SVR w/ Sigmoid Kernel	0.23	0.789	0.284	0.189	0.183	75 000	1634	0.029	0.025
MLP Neural Network	0.047	0.806	0.102	0.897	0.892	300	2090	3.923	0.011

(b) Application Failure

Regression Model	MAE	MAX	RMSE	EV	R^2	Hyperparameter Combinations	Training Time / s	Fit Time / s	Prediction Time / s
Linear Least Squares	0.032	0.934	0.112	0.787	0.784	-	-	0.006	0.002
k-Nearest Neighbors	0.016	0.616	0.07	0.917	0.917	5260	311	0.003	0.013
Decision Tree	0.021	0.65	0.076	0.904	0.904	175 000	1783	0.004	0.001
Ridge w/ Linear Kernel	0.023	0.61	0.066	0.928	0.927	70 000	1818	0.006	0.002
Ridge w/ Polynomial Kernel	0.023	0.611	0.067	0.927	0.926	20 000	1880	0.007	0.002
Ridge w/ RBF Kernel	0.021	0.615	0.068	0.922	0.922	50 000	1571	0.007	0.003
Ridge w/ Sigmoid Kernel	0.103	0.698	0.145	0.661	0.658	25 000	2090	0.032	0.002
SVR w/ Linear Kernel	0.03	0.597	0.069	0.922	0.922	75 000	1793	0.08	0.008
SVR w/ Polynomial Kernel	0.03	0.631	0.074	0.909	0.909	15 000	1712	0.109	0.007
SVR w/ RBF Kernel	0.022	0.537	0.066	0.927	0.927	50 000	1900	0.059	0.015
SVR w/ Sigmoid Kernel	0.21	0.884	0.241	0.071	0.056	75 000	1555	0.025	0.021
MLP Neural Network	0.031	0.575	0.074	0.909	0.907	300	1921	1.102	0.005

k-Nearest Neighbors Regression The k-Nearest Neighbor (k-NN) algorithm exploits feature similarity to predict values of new data points. During the training phase, the training data set is only indexed and stored into a database. Then, the value of a new data point is predicted based on how closely it resembles to the points in the training set. A weighted average of the k-nearest neighbors is used to predict the value, where the weight is calculated by the inverse of the distances and the distance itself can be any metric measure, such as the Manhattan or Euclidean distance. Hence, the model hyperparameters are the number of nearest neighbors k taken into account and the used distance metrics.

During the hyperparameter optimisation it was found that the model performed the best with $k = 5$ and Manhattan distance as metric measure for both failure classes.

Decision Tree Regression Decision Trees in machine learning are models which recursively partitioning the input feature space by inferring simple decision rules from the training data. This is usually represented by a tree structure where the decision rules are defined in the branches of the tree and the leaves contain the trained value. In general, the deeper the tree, the more different decision rules it has which results in a more complex model. However, this can also lead to over-complex trees that do not generalise the training data, also called overfitting.

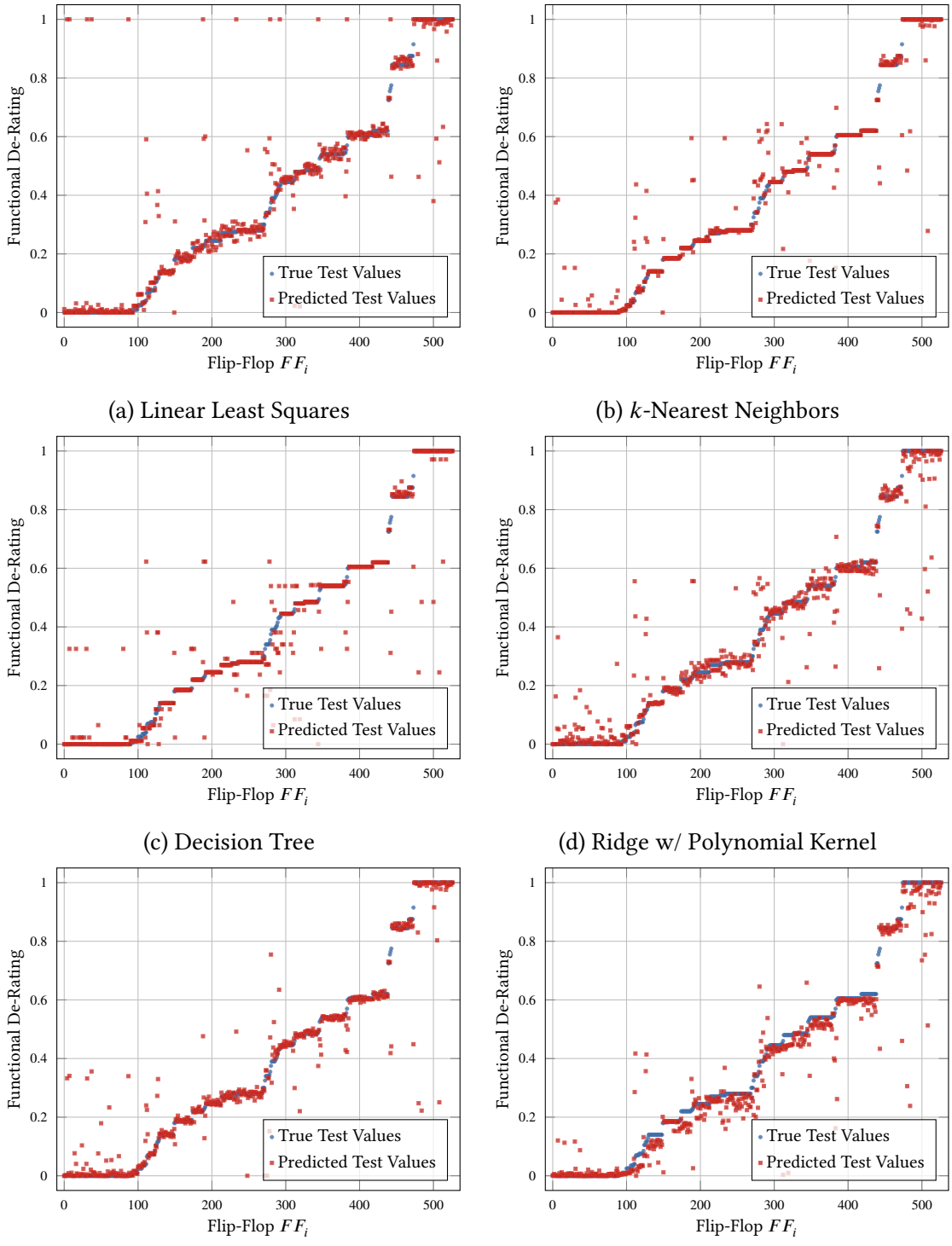
The considered hyperparameters for this model are controlling the structure of the tree, such as the maximum depth, the maximum number of leaf nodes and the balance of the tree. Further parameters defined by the framework are set to their default values.

The hyperparameter optimisation has shown that the model performs at best for both failure classes when the tree structure is not restricted (no maximum for the depth and number of leaf nodes is set and no restriction to balance the tree).

Kernel Ridge Regression The Ridge regression algorithm is modifying the ordinary least squares regression by imposing a penalty on the size of the coefficients (regularisation). Another advantage of the Ridge regression is, that it can be extended to use kernel functions. These functions perform a transformation of the input values and map them to a higher dimensional space. Thus, it is possible to learn a linear function in the space induced by the respective kernel. For non-linear kernels, this corresponds to a non-linear function in the original space. This is useful for regression problems which cannot adequately be described by linear models.

The hyperparameter of the model are the regularisation hyperparameter α, γ to control the kernel function, d to define the degree of the polynomial kernel and the independent term r in the polynomial and sigmoid kernel.

The best performance when using a Ridge regression with a polynomial kernel was found with degree of $d = 2$ for both failure classes.



(a) Linear Least Squares

(b) k-Nearest Neighbors

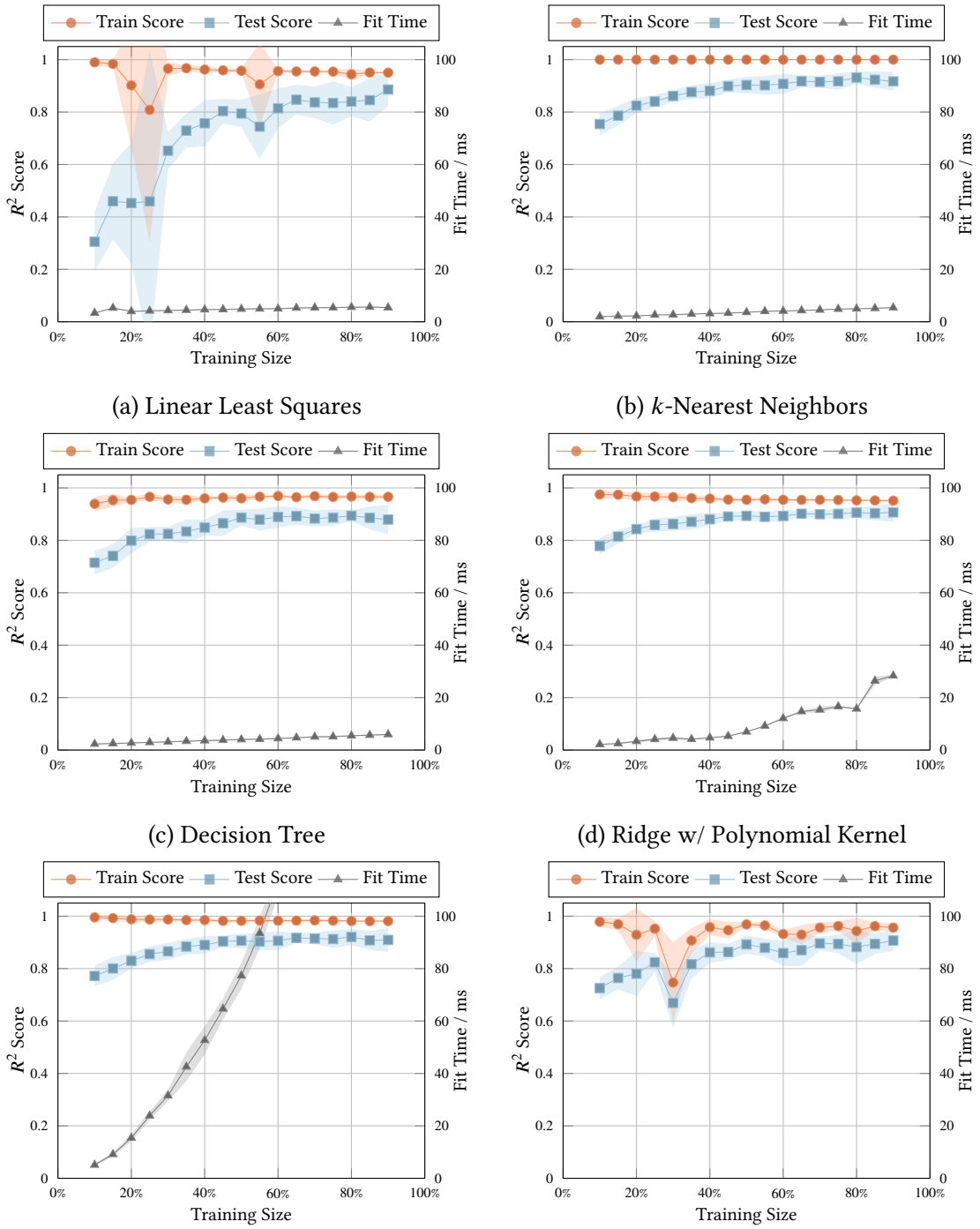
(c) Decision Tree

(d) Ridge w/ Polynomial Kernel

(e) SVR w/ RBF Kernel

(f) MLP Neural Network

Figure 3.3: Prediction of the Output Failure for one test data fold (training size = 50 %).



(a) Linear Least Squares

(b) k-Nearest Neighbors

(c) Decision Tree

(d) Ridge w/ Polynomial Kernel

(e) SVR w/ RBF Kernel

(f) MLP Neural Network

Figure 3.4: Learning curve and fit time predicting the Output Failure (cross validation = 10).

Support Vector Regression The Support Vector Regression (SVR) is similar to the Ridge Regression where the loss function is modified in such a way that predictions only depend on a subset of the training data, known as support vectors. The goal is to find a function where each training data points within an ε-tube are not penalised, and at the same time is as flat as possible. SVR can also operate with the kernel trick and the used ML framework provides the same kernels as for the Ridge Regression. The model defines several hyperparameters, such as the penalty factor C, the size of the ε-tube, γ to control the kernel function, d to define the degree of the polynomial kernel and the independent term r in the polynomial and sigmoid kernel.

The SVR with a polynomial kernel operated the best when a polynomial degree of $d = 2$ for both failure classes was chosen.

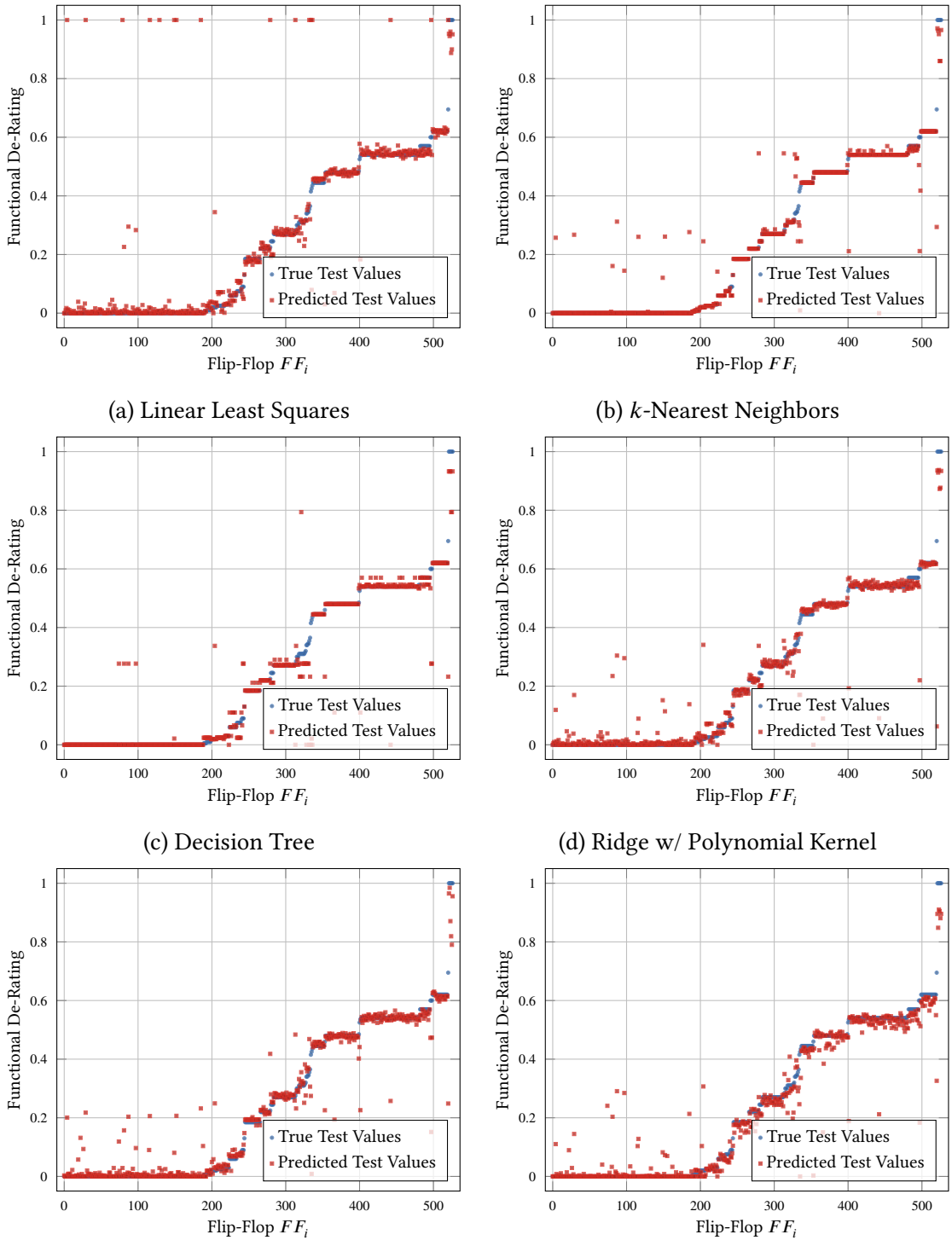
Multilayer Perceptron Neural Network A multilayer perceptron (MLP) belongs to the class of feedforward artificial neural networks (ANN). An MLP Neural Network consists of at least three layers of nodes. The first layer is the input layer, followed by one or more hidden layers and an output layer. Except for the input nodes, each node is a perceptron. A single perceptron has one or more inputs, a bias, an activation function, and one output. The received input is multiplied by a weight and passed to the activation function, which produces the output. MLP utilizes a supervised learning technique called backpropagation for training. Thereby, the predicted output of the network, is compared to the expected value and the internal weights are adjusted accordingly. This process is repeated until a maximum number of iterations or an acceptable error rate is reached.

The scikit-learn framework supports 4 different activation functions. The identity or linear function, the logistic or sigmoid function, the hyperbolic tan function and the rectified linear unit (RELU) function. This activation function together with the network topology define the hyperparameters of the model which will be optimised. Therefore, random networks are generated with a size for the hidden layer in the range of $[1; 25]$ and each layer can consist of $[1; 250]$ perceptrons.

RELU was the optimal activation function for both classes. The optimal network topology was achieved with a $(219, 85, 94)$ hidden network layer when predicting the Output Failure and a $(170, 87, 140, 244, 113)$ hidden network layer when predicting the Application Failure.

Comparison and Discussion

Prediction Performance Comparing the general performance of the different models shown in table 3.3, it can be seen that the Linear Least Squares regression and the regression models using sigmoid kernels are rated the worst. The Decision Tree regression, instance-based k-NN algorithm, as well as the kernel-based algorithms using regularisation on the parameters performing much better (except by using the Sigmoid



(a) Linear Least Squares

(b) k-Nearest Neighbors

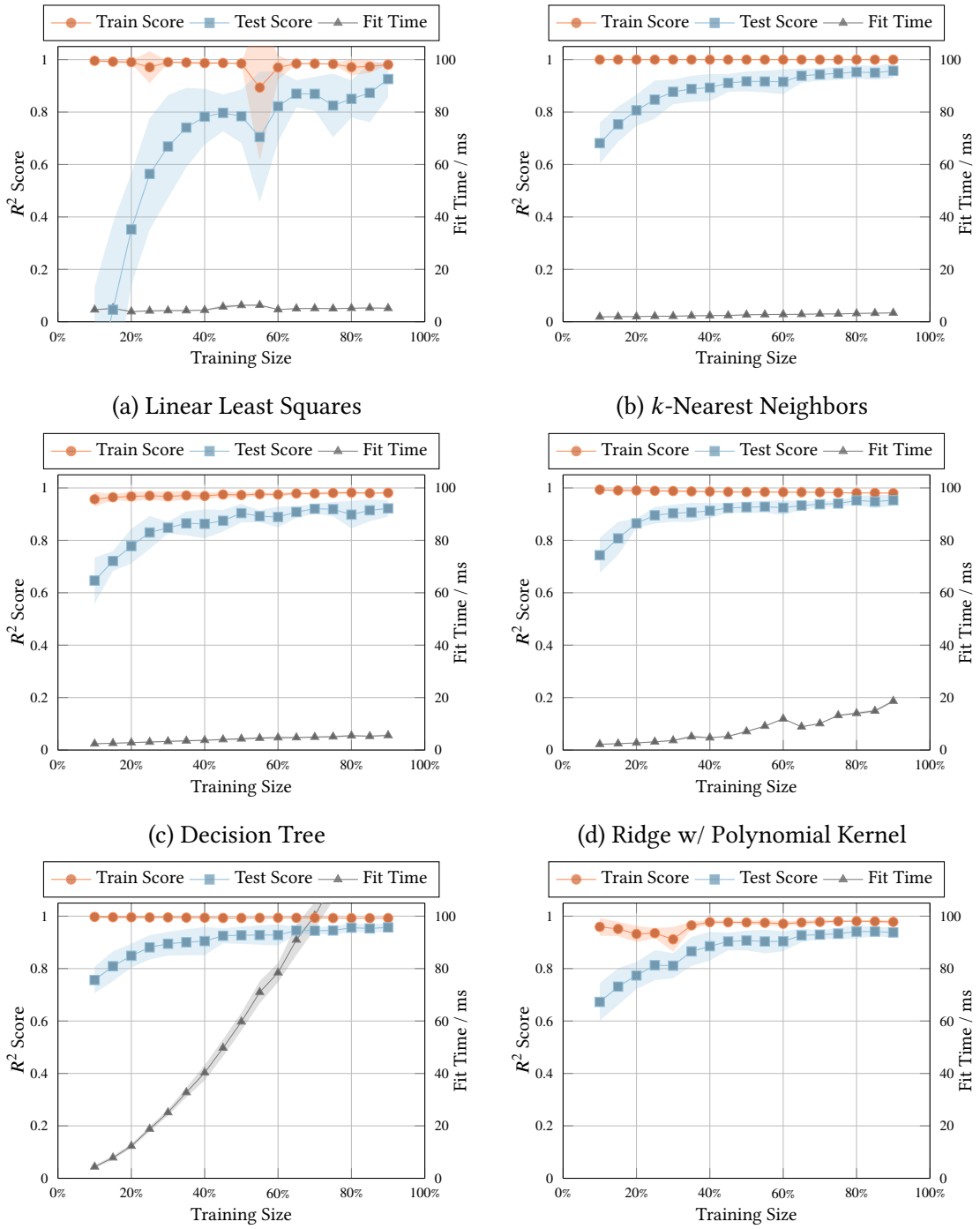
(c) Decision Tree

(d) Ridge w/ Polynomial Kernel

(e) SVR w/ RBF Kernel

(f) MLP Neural Network

Figure 3.5: Prediction of the Application Failure for one test data fold (training size = 50 %).



(a) Linear Least Squares

(b) k-Nearest Neighbors

(c) Decision Tree

(d) Ridge w/ Polynomial Kernel

(e) SVR w/ RBF Kernel

(f) MLP Neural Network

Figure 3.6: Learning curve and fit time predicting the Application Failure (cross validation = 10).

kernel). Especially, the k-NN and SVR with RBF kernel yield very good overall prediction results.

The prediction performance of the Output Failure compared to the Application Failure is quite similar in all cases. This demonstrates that the presented approach and the used feature set is suitable to learn both failure classes. It should be noted that for training sizes lower than 50 % the models predict Output Failure slightly better.

The learning curves in Fig. 3.4 and Fig. 3.6, show that the performance does not significantly improve with training sizes higher than 50 %. This means, by using the proposed method to assist a fault injection campaign, the time needed to obtain a detailed list of Functional De-Rating factors can be reduced by half. For most of the models, the cost can even be reduced further, in exchange of a slight reduction in accuracy of about $5 - 10$ %. Thus, a more aggressive optimisation, a cost reduction up-to 5×, can be achieved.

Training and Prediction Time The fit and prediction time of a model defines how many hyperparameter combinations can be tested in order to find the best performance in a reasonable amount of time. As mentioned before, the time for the hyperparameter optimisation was restricted to about 30 minutes. Thus, the total time to extract the features and train a model would be less than 2 % of the total time needed to perform the full fault injection campaign. In addition, it should be noted that the used machine learning framework does not require any licenses and offers several functions to parallelise the computation on clusters. Some machine learning frameworks even support to accelerate the computation by using GPUs. Hence, testing the hyperparameter combinations can be accelerated more easily than accelerating the fault injection campaign.

Looking at the fit time in relation to the training size, shown in Fig. 3.4 and Fig.3.6, it can be observed that with more training data the time needed to fit a model is increasing. Especially, the Ridge algorithm and SVR are known for a quadratic dependency of the training data. In contrary, the fit time of the k-NN and Decision Tree algorithm is increasing linearly with the training data [53]. This might make them more suitable to learn very large circuits with a high number of flip-flops.

3.5.3 Functional De-Rating Prediction on the Register-Transfer Level

As mentioned in section 3.3 some features can already be extracted from the RTL description of the circuit. The structural related features can be obtained after an elaboration of the RTL description. The elaboration of the design can be performed much faster than a full synthesis and additionally, the elaborated design is technology independent. Further, the signal activity related features can be obtained when simulating on RT level.

Using the approach on the RT level has the advantage, that the analysis can be performed in an early design stage and the target technology does not need to be known.

Critical circuit parts could be identified in an early design phase and mitigation techniques could be applied accordingly. Further, RT level simulations can usually be performed faster than on the gate-level netlist. Thus, a fault injection simulation can be performed quicker.

Nevertheless, by using only the RTL description of the design, it is not possible to extract the synthesis related features. These might be beneficial for the overall performance of the prediction. Therefore, in this section the approach will be applied on the RTL description of the circuit from section 3.4. The prediction performance will be compared to the performance when the full feature set, extracted from the gate-level netlist, is used.

The structural related and signal activity related features were extracted from the elaborated design. The actual values of the features extracted from the elaborated RTL were compared the actual values of the features extracted from the gate-level netlist. The only difference was observed in features which characterise the relation of the target flip-flop FF_i to the other flip-flops in the circuit and when the additional flip-flops were involved (such as # FF at Startpoint/Endpoint, # Connections from/to FF and Bus Length).

Evaluate RTL Based Functional Failure Rate Prediction

Table 3.4 compares the prediction performance (in terms of the R^2 score) when the full feature set (Structural, Signal Activity and Synthesis related features), extracted from the gate-level netlist, is used against a partial feature set, extracted from the RTL description (Structural and Signal Activity related features). Additionally, the prediction performance is shown when only the structural related features are used. Ridge and Support Vector Regression with a Sigmoid kernel are not considered in the table due to their low performance.

In the table it can be seen that the overall performance is getting lower when a reduced feature set (structural and signal activity related features) is used for most of the models. Interestingly, it can be also noted that when only the structural related features are used, the prediction performance is increasing. This can suggest two things: First, the here used signal activity related features are not linked with the Functional De-Rating. In this case new features need to be identified which better represent the workload of the circuit. Second, it is well known, that a high number of features can decrease the performance due to the curse of dimensionality [76]. By using a smaller feature subset this effect can increase the observed prediction performance.

Nevertheless, the differences in the prediction performance in all the cases is marginal. This means the presented approach is also applicable on the RT level with only a small impact, if any, on the overall prediction performance. It seems that the strongest correlation between the features and the Functional De-Rating lies in the structural related features. Results are technology independent and can be obtained faster without performing a full synthesis and thus, the approach is suitable in an early design phase.

Table 3.4: Performance Comparison by Using Different Feature Subsets (With Cross Validation = 10 and Training Size = 50 %)

(a) Output Failure

Regression Model	R^2 Score		
	Strucutral + Signal Activity + Syntehsis	Strucutral + Signal Activity	Strucutral
Linear Least Squares	0.794	0.794	0.797
k-Nearest Neighbors	0.903	0.902	0.9
Decision Tree	0.887	0.871	0.875
Ridge w/ Linear Kernel	0.889	0.889	0.89
Ridge w/ Polynomial Kernel	0.893	0.893	0.893
Ridge w/ RBF Kernel	0.894	0.891	0.892
SVR w/ linear kernel	0.879	0.88	0.88
SVR w/ polynomial kernel	0.878	0.879	0.879
SVR w/ RBF Kernel	0.906	0.904	0.904
MLP Neural Network	0.892	0.894	0.882

(b) Application Failure

Regression Model	R^2 Score		
	Strucutral + Signal Activity + Syntehsis	Strucutral + Signal Activity	Strucutral
Linear Least Squares	0.784	0.795	0.794
k-Nearest Neighbors	0.917	0.913	0.922
Decision Tree	0.904	0.871	0.865
Ridge w/ Linear Kernel	0.927	0.927	0.927
Ridge w/ Polynomial Kernel	0.926	0.926	0.926
Ridge w/ RBF Kernel	0.922	0.923	0.922
SVR w/ linear kernel	0.922	0.922	0.924
SVR w/ polynomial kernel	0.909	0.908	0.909
SVR w/ RBF Kernel	0.927	0.924	0.926
MLP Neural Network	0.907	0.906	0.911

3.6 Machine Learning Clustering for Selective Mitigation

Especially, the Soft Errors in flip-flops are a major concern and countermeasures have to be taken into consideration by using hardening techniques, such as Triple Modular Redundancy (TMR). However, a fully protected chip might not meet the system requirements in terms of area, power or target frequency. Since for many applications it is not necessary to decrease the vulnerability to a possible minimum, Selective Mitigation can

be used. Thereby, only the most critical elements of the circuit are protected against Soft Errors and thus, the failure rate of the system is decreased to meet all requirements [60, 50, 77].

In order to perform Selective Mitigation an exhaustive failure analysis is required to identify and rank the most vulnerable elements of the circuit. Especially, the failure analysis on a functional level grows with the design size, the number of workloads to analyse and the duration in cycles of each workload. A detailed functional failure analysis requires a significant investment in terms of human efforts, processing resources and tool licenses. Studies have shown that exhaustive fault simulation is not feasible for today's complex circuits [81].

Identifying and ranking the sequential elements which are most vulnerable to transient faults, usually requires computationally intensive fault-injection simulation campaigns. This procedure can be optimized by grouping flip-flops together which are expected to have a similar sensitivity to faults. Fault injection campaigns can then be performed on a per-group basis and thus, significantly reduce the time and cost of the evaluation [22]. However, this optimization heavily relies on the effectiveness of the grouping methodology. Therefore, the second machine learning based techniques proposed in this chapter, is an approach to effectively group flip-flops together which are expected to have a similar sensitivity to functional failures. The approach is based on machine learning clustering techniques by using the prior described set of features which characterises each flip-flop in the circuit (see section 3.3). Machine learning clustering algorithms are evaluated and compared to an ideal selective mitigation obtained by exhaustive fault-injection simulation.

3.6.1 Clustering Techniques for Selective Mitigation

The approach of protecting only the smallest set of elements in a circuit to meet a specified reliability target is called selective mitigation. Therefore, the individual circuit elements of a circuit need to be ranked from the most to the least sensitive. In the case of transient faults in the sequential logic which lead to a functional failure this usually requires exhaustive fault injection simulation, which might not be feasible for large and complex circuits.

In order to reduce the mentioned fault injection efforts, fault simulation campaigns can be performed on a group basis. Therefore, prior any simulations, flip-flops are grouped together and the statistical fault injection is performed on each of these groups. This can significantly reduce the time and cost of the evaluation. However, the accuracy of this coarse-grained fault injection solely relies on an effective approach to group flip-flops together which have highly similar sensitivity to faults.

Current clustering techniques are based on buses, design hierarchy, or a hybrid approach using buses, hierarchy and signal naming information [22]. These approaches have several drawbacks. The bus based and hierarchical based approach are only able to provide a fixed number of clusters. Thereby, the hierarchical based approach often

provides a low number of clusters which can be very heterogeneous. The bus based approach often results in a high number of clusters with small number of flip-flops per cluster and one larger cluster containing all the flip-flops which do not belong to any bus. Thus, the reduction of the number of fault injections is limited and the large cluster tends to be heterogeneous which negatively impacts the effectiveness of the clustering. The hybrid approach overcomes the problem of the fixed number of clusters by combining the bus and hierarchy based approaches and also taking the signal names into account. The approach assumes that flip-flops with a similar naming also have a similar function and thus, have a similar sensitivity to faults. However, this relies on a consistent naming convention and strong correlation between the naming and the function within the circuit.

Machine Learning Clustering Techniques

In order to tackle the drawbacks of the previous studied clustering approaches a machine learning clustering techniques was developed. Clustering techniques in the machine learning domain belong to the unsupervised learning category (see section 3.2). In general, these algorithms try to group similar objects together based on a given set of features. The feature set characterizes the objects and the clustering algorithms group objects together which have similar feature values, while objects from a different group should have highly dissimilar feature values [12]. The proposed Machine learning clustering approach uses the feature set described in section 3.3 to group flip-flops together. The first version of the approach was published in [43].

3.6.2 Methodology

The proposed methodology is based on clustering techniques for selective mitigation. In contrary to previous work the clustering approach uses machine learning clustering algorithms and a feature set to characterize each flip-flop in the circuit. In this way no assumptions are made about the design and a general methodology is provided.

The steps to perform a selective mitigation are illustrated in figure 3.7. First, the features for each flip-flop in the design are extracted by using the RTL description or gate-level netlist of the design and a corresponding testbench. Second, the machine learning clustering algorithm is applied to the obtained feature set and the flip-flop groups are obtained. The resulting number of groups N_c can be adjusted by the parameters of the machine learning clustering algorithm. The number of groups also dictates the effort needed for the next step: the statistical fault injection. The fault injection campaign is performed on the computed flip-flop cluster and thus, needs more efforts with a higher number of cluster and vice versa. Eventually, the sensitivity to faults for each cluster is obtained and they can be ranked from the most sensitive to the least sensitive. The selective mitigation will be applied starting from the most sensitive cluster until the reliability requirement is met.

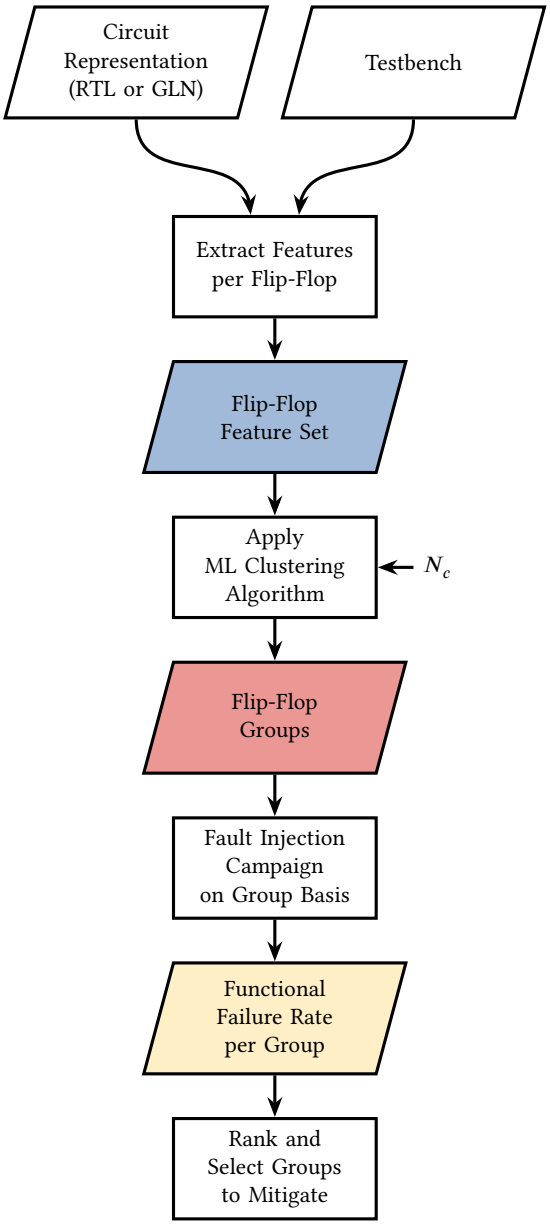


Figure 3.7: Selective mitigation by using machine learning clustering.

3.6.3 Evaluating Machine Learning Clustering for Selective Mitigation

In this section the machine learning clustering approach is evaluated on a practical example. Different clustering algorithms are used to group the flip-flops based on the feature set presented in section 3.3. For this analysis, only the structural related features are used, extracted from the RTL design of the circuit, and the sensitivity of the flip-flops

to critical failures is considered.

The effectiveness of the clustering algorithms is measured by evaluating the created flip-flop cluster. The goal is to create flip-flop groups which have a similar vulnerability to critical failures. Therefore, the exhaustive full flat statistical fault injection campaign is used, which provides the sensitivity to failures for each flip-flop as an independent measure. This data is used to evaluate the different clustering algorithms against an ideal and random approach.

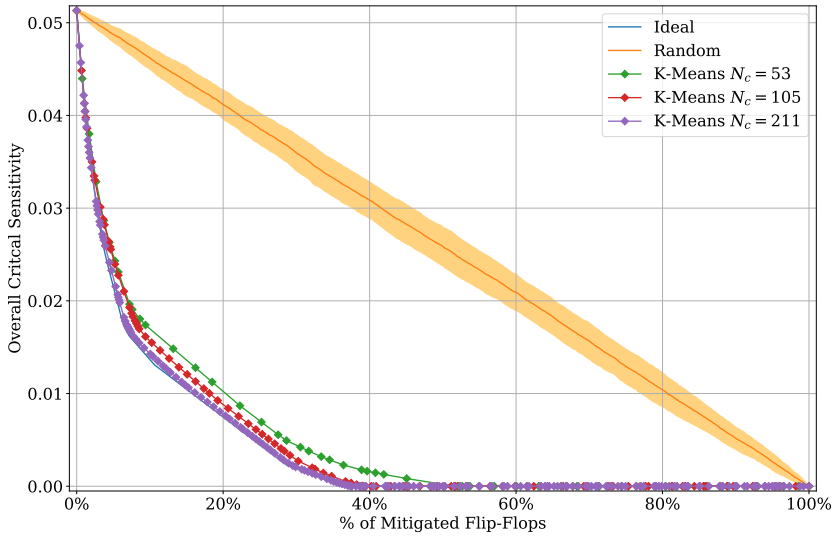
Evaluated Clustering Algorithms

The effectiveness of the clustering is evaluated considering they would be used to selectively mitigate against the critical failures. Therefore, the data obtained from the exhaustive statistical fault injection is used to compute the sensitivity to critical faults for each cluster. Then, the reduction of the overall sensitivity of the circuit was calculated by varying the number of groups being protected. For the protection it is assumed that the considered flip-flops within the group are substituted by hardened cells, Triple Modular Redundancy (TMR) or other approaches. It is assumed that after mitigation, the sensitivity to Single-Event Upsets is zero[3]. The flip-flops to protect were selected starting with the most sensitive clusters first. The results are compared against an ideal and a random approach. In the ideal approach the most sensitive flip-flops are selected based on the exhaustive fault injection campaign. The random approach selects flip-flops to protect randomly (averaged over 100 independent runs).

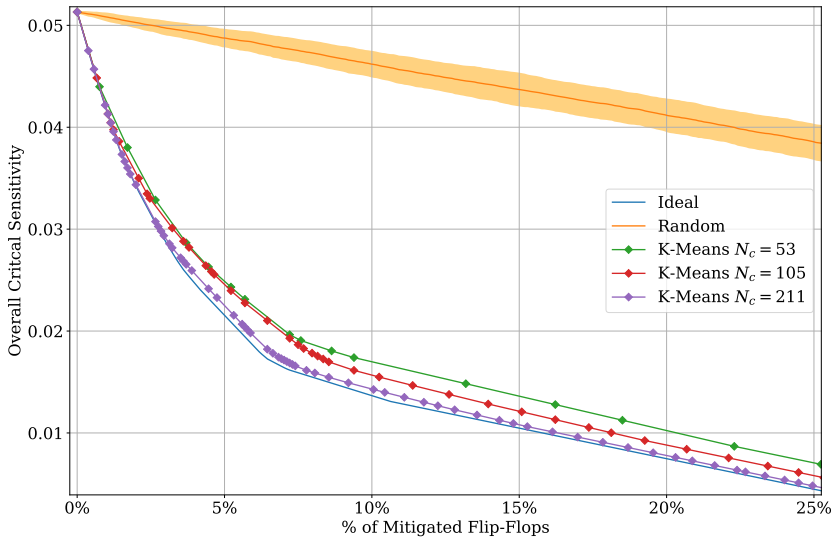
The clustering algorithms were implemented by using Python's scikit-learn Machine Learning framework [59] and applied to the extracted flip-flop feature set. This process took only several seconds and is negligible. Each considered clustering algorithm has different parameters which can be adjusted. They affect the performance of the clustering and also the resulting number of clusters. For some of the algorithms the number of clusters can be specified directly. Other algorithms try to find the optimal number clusters within the constrained parameters. In the following evaluation the parameters were chosen in a way that the number of resulting clusters are about 20 %, 10 % and 5 % of the number of flip-flops in the circuit. This would correspond to a reduction of the fault injection efforts by 5×, 10× and 20× respectively.

K-Means Clustering K-Means clustering aims to partition the given data points into k clusters. The algorithm computes the Euclidean distance for each data point in the

[3]This is a simplification and important aspects related to physical design are not considered. However, the here presented approach focuses on the evaluation on the functional level by using the RTL description of the circuit. To obtain a more complete analysis, the presented approach could be combined with e.g., the classical analysis for electrical and temporal masking by using post place and route gate-level netlist.



(a) From 0 % to 100 % of mitigated flip-flops

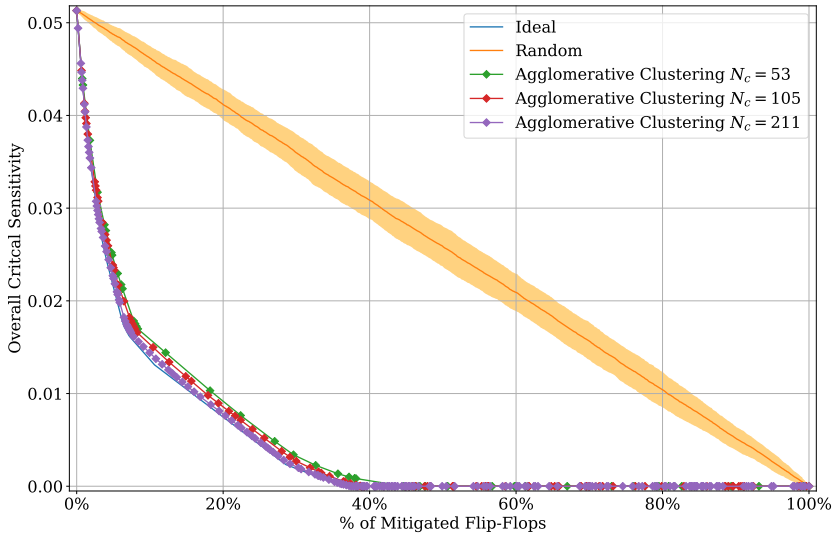


(b) Section from 0 % to 25 % of mitigated flip-flops

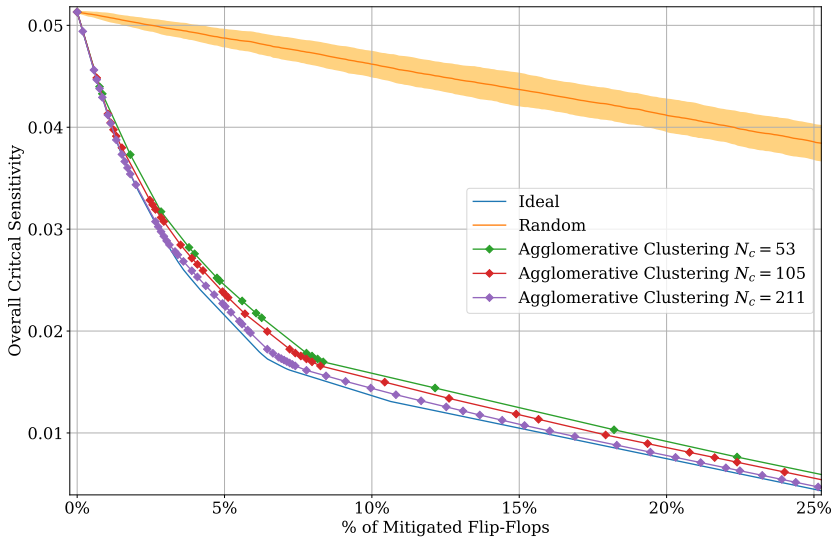
Figure 3.8: K-Means Clustering

feature space. The data samples are then separated in such a way the variance within each cluster is equal. The algorithm requires the number of clusters N_c to be specified.

Figure 3.8 shows the overall critical sensitivity when the grouping is performed by the K-Means clustering with different number of clusters N_c. The flip-flops to protect were selected starting with the most sensitive clusters first until all flip-flops are mitigated. It can be noted that the grouping performs better when the number of clusters is higher.



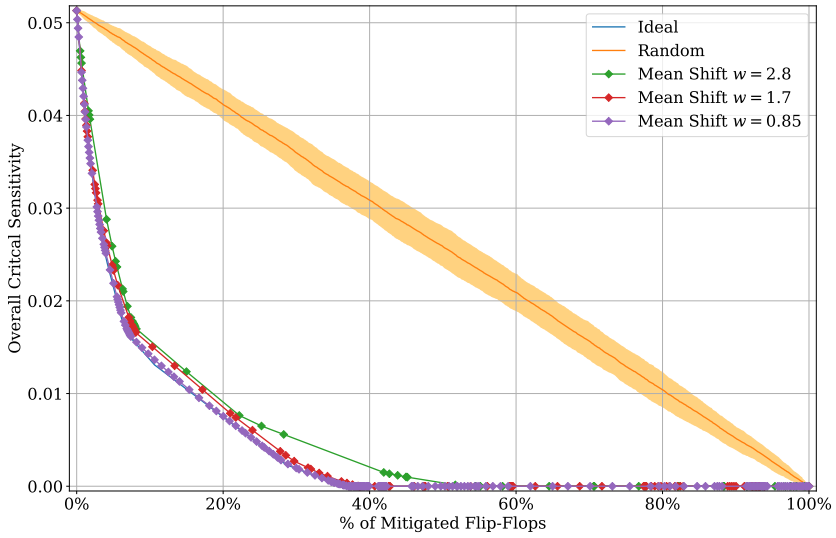
(a) From 0 % to 100 % of mitigated flip-flops



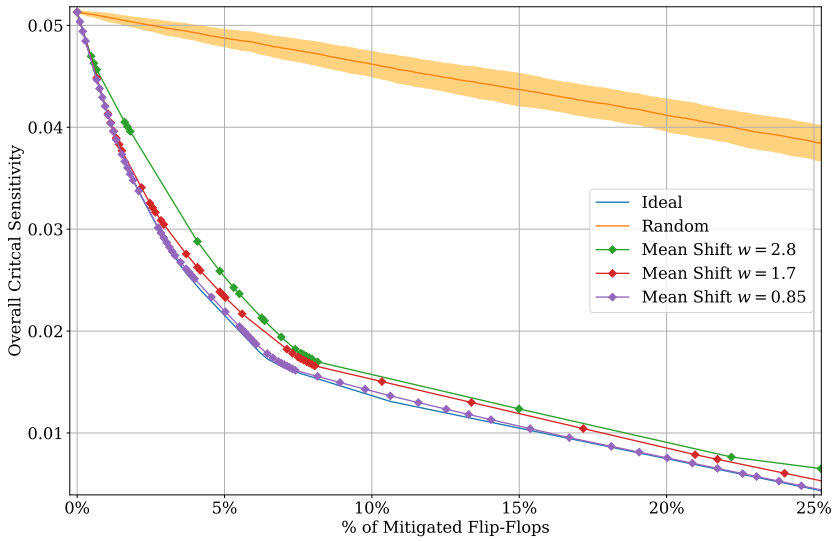
(b) Section from 0 % to 25 % of mitigated flip-flops

Figure 3.9: Agglomerative Clustering

Agglomerative Clustering Agglomerative Clustering performs a hierarchical clustering by creating nested clusters, which are represented as a tree. The advantage of hierarchical clustering is that any valid measure of distance can be used, in comparison to K-Means clustering which performs on a Euclidean distance metric. Agglomerative Clustering uses a bottom-up approach where each data point starts as its own cluster. Clusters are merged together by following the linkage criterion. This criterion defines the metric used for the merge strategy. It was noted that the best results were obtained by using the maximum or complete linkage, which minimizes the maximum distance



(a) From 0 % to 100 % of mitigated flip-flops



(b) Section from 0 % to 25 % of mitigated flip-flops

Figure 3.10: Mean Shift Clustering

between data points of pairs of clusters and a Manhattan distance metric (l1 norm).

In figure 3.9 the results of the selective mitigation are shown when Agglomerative Clustering is used for different number of clusters N_c. Similar to the K-Means clustering the effectiveness of the selective mitigation is increasing with a higher number of clusters. However, the improvements from 53 to 105 and 211 clusters are very minor. When comparing the Agglomerative Clustering to K-Means Clustering it can be noted that Agglomerative Clustering performs generally better.

Mean Shift Clustering Mean Shift Clustering aims to find dense areas of data points. The algorithm is based on a sliding-window which is shifted towards regions of higher density. The density of the sliding-window is proportional to the number of data points within the window it. The goal is to locate center points for each group in the dataset. For the previous described clustering algorithms, the number of clusters had to be specified manually. In Mean Shift clustering the number of clusters is determined by the algorithm. Further, K-Means clustering assumes spherical distribution shape of the clusters in the feature space. For the algorithm the window size w needs to be specified.

A general problem when using Mean Shift Clustering is to choose the correct windows size. In this analysis the window sizes were chosen in a way the resulting number of clusters are close to the number of clusters used for the previous algorithms. By using window sizes of $w = 2.8$, $w = 1.7$ and $w = 0.85$ the number of clusters were resulting in 52, 105 and 210, respectively. Figure 3.10 shows the effectiveness of the algorithm when using the obtained clusters for selective mitigation. As for the previous algorithms the effectiveness is increasing with a higher number of clusters (smaller window sizes). It can be seen, that the results with a window size of $w = 0.85$, which results in 210 groups, is almost as good as the ideal solution.

Comparison and Discussion

The results of the considered clustering algorithms with different resulting number of clusters are summarized in table 3.5. Since the number of resulting clusters was chosen to be about the same, the average cluster size is identical or very similar. The standard deviation of the cluster sizes however shows that Agglomerative and Mean Shift clustering tend to create cluster with more variety in the cluster size.

In order to quantify the effectiveness of the algorithm to create flip-flop groups with similar functional failure rate, two metrics were derived from the results: the average variance of the functional failure rate and the maximum difference of the functional failure rate of flip-flops within the same cluster. An additional metrics was created where the average variance of the functional failure rate within one cluster is weighted with the cluster size. In this way a large cluster, which has the same variance within the cluster as a small cluster, is penalized more.

The results verify what was observed from the figure 3.8, 3.9 and 3.10. In general, the algorithms perform better with a higher number of clusters. Agglomerative Clustering performs slightly better than k-Means and Mean Shift clustering performs worst with low number of clusters and close to ideal with the highest considered number of clusters.

Using an Independent Metric In order to evaluate the performance of the clustering algorithm without knowledge of the actual/reference values, several metrics exists which quantify certain characteristics of the created clusters (such as the separation of the data). These metrics can be used when the presented approach is applied to a new

Table 3.5: Comparison of the Different Clustering Algorithm

Clustering Algorithm	# Cluster	Mean Cluster Size	Standard Deviation Cluster Size	Mean Functional Failure Variance	Mean Weighted Functional Failure Variance	Max Functional Failure Difference	Davies-Bouldin Index
k-Means	53	23.28	13.01	0.009	0.095	0.92	0.73
	105	11.75	7.45	0.010	0.041	0.92	0.68
	211	5.85	3.80	0.003	0.008	0.64	0.51
Agglomerative Clustering	53	23.28	21.39	0.015	0.094	0.92	0.78
	105	11.75	11.76	0.011	0.040	0.92	0.57
	211	5.85	5.32	0.003	0.007	0.64	0.54
Mean Shift	52	23.73	28.71	0.017	0.167	1	0.60
	105	11.75	14.12	0.006	0.038	0.92	0.42
	210	5.88	6.64	0.002	0.005	0.47	0.39

unknown circuit and different algorithms are evaluated or the algorithm parameters are fine tuned. For this analysis the Davies-Bouldin index was used and results are shown in table 3.5. This index can be used to evaluate the separation between the clusters. It signifies the average similarity between clusters by comparing the distance between clusters with the size of the clusters themselves. An index of 0 is the lowest possible score and values closer to zero indicate a better partition.

The Davies-Bouldin index does not fully correlate with the functional failure variance or difference metrics. However, it follows the general direction and a lower index can be observed with higher number of clusters. Further, similar scores are obtained for the k-Means and Agglomerative Clustering as also seen when comparing the functional failure metrics. The best index, the closest to zero, is also obtained for the best result, Mean Shift clustering with 210 clusters.

3.7 Conclusion

Two machine learning techniques to assist the Functional Failure analysis of complex circuits were proposed in this chapter. The presented methodologies reduce the computational cost to determine the Functional De-Rating factors of the circuit's sequential logic.

For the presented approaches a feature set was developed. This feature set characterise each flip-flop in the circuit. The features combine attributes from static elements and dynamic elements.

The first approach aims to predict the De-Rating factors per individual instances, which is particularly difficult to obtain using classical approaches such as clustering, selective fault simulation or fault universe compaction techniques. The methodology was applied in a practical example where several machine learning models were evaluated, predicting two different failure classes, first, predicting the propagation of a fault to the primary output and second, predicting the rate of a fault leading to a functional failure. The performance comparison has shown the instance-based k-NN model, the Decision Tree regression or kernel-based algorithm with non-linear kernels and training sizes of 20% to 50% are most suitable. This means, the cost of a fault injection campaign can be reduced of a fault injection campaign can be reduced by a factor of 2 up to 5 times in comparison to a classical statistical fault injection campaign. Further, the comparison between the two failure classes has shown that the machine learning models are able to learn and predict the fault propagation to the output, as well as the functional failure rate.

Additionally, the ability of the method to be used in early design stages has been assessed. Therefore, a feature subset was identified which can be extracted from an elaborated RTL description of the design. It was shown that the impact on the prediction performance was marginal and, in some cases, even increased. It was noted that the performance increased, especially when the prediction was performed without the

signal activity related features. This suggests that new features should be identified to characterise the workload of a circuit more adequately.

The second methodology uses machine learning clustering techniques to group flip-flops together which are expected to have a similar contribution to the overall functional failure rate. The advantage in comparison to other existing approaches is that the approach is more flexible and no assumption of the circuit or its representation is made. Further, the number of clusters can be chosen by the user, which determines the needed efforts for the fault injection campaign.

The effectiveness of the grouping by different machine learning clustering algorithms were evaluated on a practical example and compared to an ideal solution. Good results were obtained by choosing the number of clusters with 5 % of the total number of flip-flops in the circuit and results close to the ideal solution were obtained with number of clusters corresponding to 10 % to 20 % of the number of flip-flops. This would mean that the fault injection efforts could be reduced by a factor of 5×, 10× or 20× respectively.

Chapter 4

Cross-Layer Reliability and Functional Safety Assessment Through Machine Learning

4.1 Introduction

High quality, reliable and safe electronics requires massive cooperation and exchanges across all the partners of the supply, design and manufacturing chain. A huge quantity of information, addressing functional and extra-functional qualities must be produced accurately, exchanged without loss of fidelity and applied as intended.

One of the most self-evident examples of such flow of information is the typical

$$\text{Foundry} \rightarrow \text{Designer} \rightarrow \text{Integrator}$$

process. In this process, the technology provider prepares a complex Process Design Kit (PDK) that includes technological data, simulation models, design rule information, primitives and possible standard cell libraries from the foundry vendor. Designers make use of this information during the preparation of cells, IP blocks and ultimately components integrating the technology elements. Components are used by system integrators on boards, sub-systems and systems. Some information (such as recommended supply voltage) will be valuable through the whole flow while other data can be consumed in one design stage to ensure the fulfilment of specific requirements.

Design paradigms, workflows, practices and expectations are progressing continuously. While in the past the most important parameters for a typical Application Specific Integrated Circuit (ASIC) design process were area, frequency and power, today's requirements are presented as a vast set of functional and extra-functional specifications. Functional Safety (FuSa) and Reliability requirements are increasingly present because of implicit or explicit customer expectations and formal standards such as IEC 61508 [37], ISO 26262 [38] and others. These standards demand the calculation

and presentation of functional safety and reliability metrics at system-level, in quantitative and qualitative terms. However, computing the failure rate of a system to make sure that it fulfils the < 10FIT requirement for an ASIL D product involves data that has been produced by at least three individual entities and transformed by tens of engineers (or even companies) during the design flow. There is a huge potential for loss of fidelity, data misuse, omissions and translation errors. In addition, much of this information is highly proprietary and the transmission of the information from one partner to another can be misused by restrictions and disclosure limitations. Some of the data can also facilitate possible reverse engineering or at least disclose critical design information that was intended to be kept secret.

In this chapter a uniform methodology is presented to evaluate reliability and functional safety metrics based on using compact models build with Machine Learning models. The compact models can be used at any design abstraction level. The compact models of a design element (standard cell, IP, block, component, sub-system or system) at a given hierarchical level can be combined through machine learning techniques in a single compact model that can then be used for the next design stage or by the next user. This approach ensures that relevant parameters are retained and impactful during the design flow and that the contribution of the various design elements is well represented at any calculation stage. Additionally, it may provide a way to obscure technology and design information, safeguarding critical IP, while still equipping the users at any design stage with the information they need for their work. The approach was initially published in [4].

4.2 Motivation

To motivate the approach a metric calculation flow and the associated risks and works, the evaluation of SEEs to the failure rate of a system is considered: Firstly, the technology provider (foundry), in a possible collaboration with a library vendor must provide technological (raw) event rate for the various technology elements (standard cells, memory blocks, analog IP such as Phase Locked Loops (PLLs), etc.) that is used by the customer. Sequential cells can be affected by SEUs with rates that depend on the cell state, voltage and temperature. SETs in combinational cells can depend on voltage and temperature and the cell fanout. Single Bit Upset (SBU), Multiple Bit Upset (MBU) or Multiple Cell Upset (MCU) can impact the data stored in memory blocks and their occurrence rate can depend on physical implementation (aspect, column muxing, scrambling) or design choices such as error management schemes. Ideally, all this data deserves to be captured in a detailed, high precision format that accurately reproduces the error rate of the researched event according to a reasonable set of parameters.

Today's PDK contain large databases, multi-index tables and process information in a variety of formats. The current State-Of-The-Art to deliver the technological information of SEEs consists in PDF test reports or a spreadsheet with limited summary

information. The component or IP designer must use this information as an input of its own event rate calculation methodology. The unit SEU rates for instance, indicated by the technology provider must be annotated to each flip-flop in the circuit and de-rated according to the role of the flip-flop in the design (see section 2.3.3)

$$\mathrm{SER}_{\mathrm{Seq}} = \sum_{\text{Flip-Flops}} \mathrm{FIT}_{\mathrm{SEU}} \cdot \prod_{\text{De-Rating}} \mathrm{DR} \tag{4.1}$$

Memory SBU/MCU data is tailored according to each instance specified and the impact of possible error management schemes (Error Correcting Code (ECC)) is integrated

$$\mathrm{SER}_{\mathrm{Memory}} = \sum_{Memories} \begin{cases} \mathrm{FIT}_{\mathrm{SBU}} \cdot \mathrm{Size} \cdot \mathrm{MDR} & \text{if no ECC} \\ \mathrm{FIT}_{\mathrm{MBU}} \cdot \mathrm{Size} \cdot \mathrm{MDR} & \text{if ECC} \end{cases} \tag{4.2}$$

The objective is to calculate at the Intellectual Property (IP) or design level the required overall reliability or functional safety metrics. While in some rare circumstances, the final deliverable is a number or a limited set of numbers, any realistic high-level metric will be dependent on a variety of parameters. Firstly, some parameters (such as voltage, temperature) inherited from the underlying technological layer will certainly impact the SEE fault error rate. It may still be feasible at this stage to provide indexed tables containing the necessary data for the possible combinations of a few given parameters. However, in more complex cases, this is not sufficient, as the design can be configured differently or used in various scenarios, affecting the exhibited failure rate. Complex ASICs can be configured to fulfil different function modes. Design settings (speed, number of active cores, buffer or cache sizes, memory modes) can be set according to each application requirements. Internal features (ECC, Error Detection Correction (EDC), safety mechanisms) can be activated or not with a direct impact on the design failure rate. CPU-based designs can show various failures with different manifestations and rate that depend on how the CPU cores are used by a given application. Additionally, the same design can have multiple physical implementations that can also impact reliability. As an example, the usage of packaging materials with different alpha emissivity rates will strongly impact the final SEU rate (from 1× to 1000×).

In conclusion, the set of parameters affecting the FuSa/Reliability metrics of a design can be large and impactful. The manufacturer will usually have difficulties in packaging and transmitting this information to the user. There are several options:

- A maximum, worst-case metric. This way, the actual failure rate is guaranteed to be lower than the indicated value. This approach can penalise some applications or systems and can lead to over-engineering when trying to manage the failure rate.
- A recommended utilization scenario. This scenario is intended to lead to a nominal, favourable error rate and is usually implemented through a “Safety Manual” where the manufacturer describes his vision on how the component should be used in order to fulfil safety goals and to lead to the desired failure rate

- An analytical method. The provider explicitly describes a function to calculate the actual failure rate according to the implementation. To give an example, when a system which includes memory instances is considered, then their contribution to the overall failure rate can be described similarly to the equation (4.2). However, this approach presents the drawbacks of disclosing to the user in a clear format, internal circuit information or raw technology data which can be critical pieces of IP or know-how.

Regardless of how the information is provided, the user of the design will use this component in its own system and will need to evaluate the failure rate for its own implementation. Moreover, a system integrator will typically use many individual components from various providers. Translating, adapting and integrating various reliability data is very challenging. Finally, the system integrator will have to deliver a final product with ideally, a clear set of metrics that can be compared to the requirements. However, even the system reliability metrics can depend on parameters that can vary according to the usage. The same exact product can show different failure rates when used in a data center, a car, an airplane or a satellite. The provider needs to express the metrics of its products according to a set of parameters and has to provide the user with a metric model with a set of parameters that are relevant for the given abstraction level.

The various collaborations between the providers and users of technology, components and systems need to be supported by adequate tools, preferably developed for reliability and functional safety efforts. While more specific tools exist, standard spreadsheet tools are widely deployed and used. Data import and export from these tools is inadequate and prone to errors.

Recent standardization efforts from IEEE [15] and Accellera [1] aims at defining an exchangeable interoperability format for functional safety analysis and functional safety verification activities. A format for Reliability exchanges has been proposed previously [24, 64], providing a solution to suppliers and consumers to exchange reliability information in a consistent fashion and to use this information to construct accurate reliability models. All these efforts stress the importance of building models hierarchically and according to the design abstraction level. Accordingly, a uniform, universal method to model functional safety or reliability metrics at any design stage or abstraction level and more importantly, combining the information available at the current level to benefit the next efforts in the pipeline would be opportune and useful.

Using machine learning techniques to build such a model has several advantages. Machine learning algorithms are known to efficiently learn even complex relationships. Models can be built from various types of input data, such as tables or functions. Therefore, it is suitable at any abstraction and hierarchical level. Several aspects can be combined in a single model, which makes it compact and easy to use by the user in the next design stage. Additionally, the representation of the Machine Learning model can be fundamentally different to the original representation and thus, obscure critical technology and design information.

4.3 Methodology

The proposed methodology is conceptually straightforward and easy to implement. The methodology relies on compact models build using Machine Learning (ML) regression models (see section 3.2). The models are trained using reference data provided by State-of-the-Art FuSa/Reliability assessment methods. The overall FuSa/Reliability metrics for the current level can be then calculated and used as reference data to train a compact Machine Learning model. The elaborated model can then fuel the analysis efforts required for the next design stage or upper hierarchical level or abstraction and can be shared with the corresponding engineer or user.

The methodology consists of three phases which are described in the following:

Phase I – Data collection Assuming that the current hierarchical level or design abstraction is a collection of elements (or components). Each element has one or multiple attached FuSa/Reliability metrics and models. These models can be analytical, data or Machine Learning based. Each individual metric i can have a collection of input parameters $\vec{p}_i$

$$\text{Metric}_i = f(p_{i,0}, p_{i,1}, p_{i,2}, \dots, p_{i,n(i)}). \tag{4.3}$$

Phase II - Integration At the current level, State-of-the-Art approaches will allow us to combine the existing data of the various elements in an overall, top-level FuSa/Reliability metrics

$$\text{Metric}_{\text{overall}} = \sum_i \text{Metric}_i = g(p_0, p_1, p_2, \dots, p_m). \tag{4.4}$$

This overall metric will depend on all of the parameters of the element metrics, including options, parameters and choices that are applicable at this design level. Obviously, the parameter list can be simplified by optimizing the parameters that are the same or equivalent. For instance, a supply voltage parameter can be applicable to several elements.

Phase III – Compact ML Model Elaboration The equations that composes the overall metric can be then exercised over the validity range of the parameters. The overall metric can also be a collection of data valid over a specific range. The collected data will be then used to train a machine learning model that will accept as inputs the aggregated set of parameters and will extend to the area covered by the available dataset. Once trained, the machine learning model is expected to have good if not perfect accuracy over the valid range and is an ideal surrogate or replacement for the discrete overall metric.

The main goal of the training is to create a model which accurately represents the given metrics function or collection of data. The model should be able to

accurately recall the trained values and depending on the learned metric, the model should also be able interpolate and extrapolate the data.

In comparison to classical machine learning applications this approach is a bit different. In classical applications the input parameter range is not always limited and known and for some cases more training data cannot be generated or gathered. For the proposed methodology the opposite is true. The limit of the desired input parameter range is known and, in most cases, it should be easier to generate more reference data for the training. In this way the data to train the model can be increased to improve the accuracy until a specified target is met.

Phase IV – Packaging and Reuse The elaborated machine learning model can be provided to the next user that can start applying this methodology on the next hierarchical level, flow stage or design abstraction level.

The presented methodology shows distinct advantages. The approach is uniform regardless the location in the design flow or design hierarchy. The machine learning models are compact, the training is not computationally intensive and implementations are available in a variety of language and programming frameworks, usually without additional licenses required. In many cases the trained machine learning model will hide or obscure sensitive design or technology information.

4.4 Demonstration on an Example

In this section the proposed methodology is exercised on a practical example. The example is addressing the calculation of a system's SER. For the purpose of the demonstration, simple and naive equations are used for modelling the error rates and aggregating the contribution of the various elements. In real applications any method can be used to generate the reference data.

Firstly, the following functions are introduced for the calculation of the various Soft Error Rates (indicated in FITs/MegaBit or FITs/MegaCell):

- A Flip-Flop has a 100 FITs/Mb at the nominal 1.2 V supply voltage (V_{dd}). The SER decreases at higher voltages and increases at lower.

$$\mathrm{SER}_{\mathrm{Seq}}(V_{dd}) = 100 \cdot (1 + (1.2 - V_{dd})) \tag{4.5}$$

- A combinational cell can exhibit a spectrum of SETs with decreasing event rate for larger, longer events. We will limit the minimal PW at 10 ps which is the shortest transient that the selected technological process can propagate.

$$\mathrm{SER}_{\mathrm{Comb}}(V_{dd}, \mathrm{PW}) = 50 \cdot (1 + (1.2 - V_{dd})) \cdot \frac{50}{\mathrm{PW}} \tag{4.6}$$

PW is the Pulse Width (in ps).

The presented models for the error rate of the sequential and combinational logic are analytical models (simple equations). They can also be delivered as different models, even machine learning models accepting a voltage and a pulse width parameter.

The Soft Error Rate for natural working environments is dominated by the contribution of SEEs caused by alpha particles emitted by impurities in the packaging materials and neutrons caused by the interaction of high-energy particles with the atmosphere. Both contributions depend on a large number of parameters and it can be difficult to provide a model that integrates the effect of all the parameters.

As an illustration, the neutron-induced SEE rate depends on many factors, including physical location, altitude, solar activity, shielding, etc. Following the approach from [39] the calculation is focused on altitude and cutoff (dependent on the location), ignoring solar modulation. In this case, the actual Neutron Flux (NF) at a given location can be expressed as

$$\mathrm{NF} = \mathrm{NF}_{\mathrm{ref}} \cdot \mathrm{GRF} \cdot e^{-\frac{A - A_{\mathrm{ref}}}{L}} \tag{4.7}$$

where $\mathrm{NF}_{\mathrm{ref}}$ is the neutron flux at the reference location (New York City, sea-level equals 14 n/sq · cm/h). L is the flux attenuation length for neutrons in the atmosphere ($\sim$148 g/cm^2). Finally, A is the areal density of the location of interest, A_{ref} is the real density of the reference location

$$A = 1033 - e^{-0.03813 \cdot (\frac{a}{1000}) - 0.00014 \cdot (\frac{a}{1000})^2 + 6.4 \cdot 10^{-7} (\cdot \frac{a}{1000})^3} \tag{4.8}$$

While these factors can be described analytically and integrated in the various models, the GRF represents the Geomagnetic Rigidity Factor and varies according to the geographical position. As an example, values of geomagnetic vertical cutoff rigidity used to calculate the relative neutron flux were provided by the Aerospace Medical Research Division of the Federal Aviation Administration's Civil Aerospace Medical Institute. The cutoff data were generated by Shea and Smart using the International Geomagnetic Reference Field for 1995 [69, 7, 54]. Therefore, the actual Geomagnetic Rigidity Factor (GRF) values can only be provided as a table of data indexed according the longitude and latitude. This causes a number of issues, including the need to interpolate between the available sparse, low granularity data and the difficulty to integrate tabular data in a compact model.

Machine learning models can cope very efficiently with these difficulties. Firstly, the training can be done on any type of data, analytical, tabular with any type of data representation for the input parameters: linear (for the altitude) or specific (geographical coordinates). Moreover, it will also be able, provided that an adequate model is used, to interpolate GRF data between the locations provided in the tables, allowing the approximate calculation of the GRF for any location.

The neutron-induced error rate is provided as base value for the reference setting (New-York, sea-level) that needs to be multiplied by an acceleration factor calculated according to the actual location and altitude. To illustrate, the table 4.1 shows the acceleration factors for a selection of altitudes (table 4.1a) and location (table 4.1b). The final

acceleration factor can be calculated by multiplying the appropriate location factor with the desired altitude factor. This part fulfils the *Phase I* of the proposed methodology.

Table 4.1: Neutron Flux Acceleration Factors

(a) Altitude-Depended Neutron Flux

Altitude (feet)	Altitude (m)	Neutron Flux [n/sq · cm/h]	Neutron Flux [n/sq · cm/s]	Neutron Flux (relative to sea level)
0	0	14.0	0.003 889	1.0
0	0	14.0	0.003 889	1.0
1000	304.8	18.2	0.005 056	1.3
2000	609.6	23.4	0.0065	1.7
5000	1524	47.6	0.013 222	3.4
10 000	3048	134.6	0.037 389	9.6
20 000	6096	668.5	0.185 694	47.8
30 000	9144	2001.1	0.555 861	142.9
35 000	10 668	2993.2	0.831 444	213.8
40 000	12 192	4147.0	1.151 944	296.2

(b) Location-Depended Neutron Flux

Location	Neutron Flux (relative to sea level)
Milpitas	0.92
Colorado Springs	4.42
Bangalore	1.02
Beijing	0.72
Grenoble, France	1.24

In *Phase II*, a simple de-rating approach is assumed to calculate the overall SER of an ASIC with 1 Mbit of flip-flops and 10 Mbit of combinational cells. The overall error rate $\mathrm{SER}_{\mathrm{ASIC}}$ can be computed as the de-rated contribution of each individual element:

$$\mathrm{SER}_{\mathrm{ASIC}} = \sum_{\text{instance}} \mathrm{SER}_{\mathrm{SEU|SET}} \cdot \prod_{\mathrm{LDR,TDR,FDR}} \mathrm{DR} \tag{4.9}$$

The applicable de-ratings are Logical De-Rating (LDR), Temporal De-Rating (TDR), Functional De-Rating (FDR) (see section 2.2). The de-rating factors can be expressed

Table 4.2: Generated SER Values

Sample	Parameters				SER [FIT]
	Voltage [V]	Environment	Frequency [MHz]	Location, Altitude	
0	0.8	Neutrons	0.1	NYC, sea-level	238.55
1	0.8	Neutrons	0.15	NYC, sea-level	353.24
			...		
1129	1.5	Neutrons	1.95	N/A	2241.00
1130	1.5	Neutrons	2	N/A	2241.00
1131	0.8	Alpha	0.1	NYC, sea-level	238.55
1132	0.8	Alpha	0.15	NYC, sea-level	353.24
			...		
2260	1.5	Alpha	1.95	N/A	2241
2261	1.5	Alpha	2	N/A	2298.35
2262	0.8	Heavy-Ion	0.1	NYC, sea-level	238.55
2263	0.8	Heavy-Ion	0.15	NYC, sea-level	353.24
			...		
3391	1.5	Heavy-Ion	1.95	N/A	2241
3392	1.5	Heavy-Ion	2	N/A	2298.35

for instance, by the following simple equations or values:

$$\mathrm{TDR}_{\mathrm{Seq}}(\mathrm{Freq}) = \frac{\mathrm{Slack}}{\mathrm{Clock\ Period}} = 1 - \frac{\mathrm{Freq}}{2 \cdot 10^6} \tag{4.10}$$

$$\mathrm{TDR}_{\mathrm{Comb}}(\mathrm{PW}, \mathrm{Freq}) = \frac{\mathrm{PW}}{\mathrm{Clock\ Period}} = \mathrm{PW} \cdot \mathrm{Freq} \tag{4.11}$$

$$\mathrm{LDR} = 0.25 \tag{4.12}$$

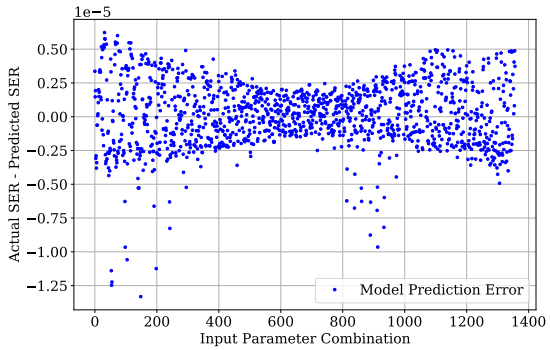
$$\mathrm{FDR} = 0.25 \tag{4.13}$$

Finally, the overall SER of the ASIC can be described as

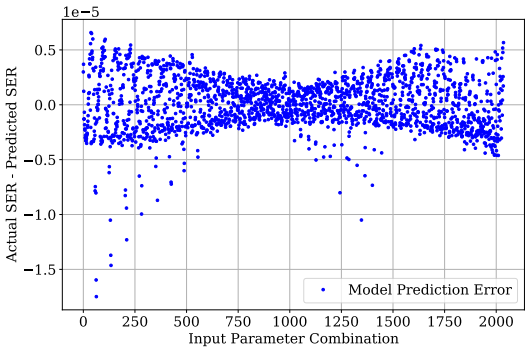
$$\begin{aligned}\mathrm{SER}_{\mathrm{ASIC}}(\mathrm{PW}, \mathrm{Freq}, V_{dd}) = \big(&1\,\mathrm{Mbit} \cdot \mathrm{SER}_{\mathrm{Seq}} \cdot \mathrm{TDR}_{\mathrm{Seq}} \\ &+10\,\mathrm{Mbit} \cdot \mathrm{SER}_{\mathrm{Comb}} \cdot \mathrm{TDR}_{\mathrm{Comb}}\big) \cdot \mathrm{LDR} \cdot \mathrm{FDR}\end{aligned} \tag{4.14}$$

This can be resolved to

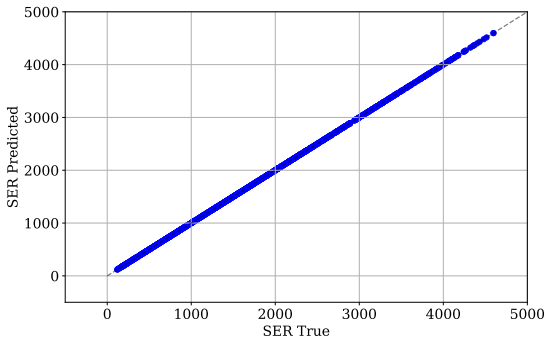
$$\begin{aligned}\mathrm{SER}_{\mathrm{ASIC}}(\mathrm{PW}, \mathrm{Freq}, V_{dd}) = \Bigg(&1\,\mathrm{Mbit} \cdot 100\,\mathrm{FIT/Mbit} \cdot \big(1 + \big(1.2 - V_{dd}\big)\big) \cdot \left(1 - \frac{\mathrm{Freq}}{2 \cdot 10^6}\right) \\ &+10\,\mathrm{Mbit} \cdot 50\,\mathrm{FIT/Mbit} \cdot \big(1 + \big(1.2 - V_{dd}\big)\big) \cdot \frac{50}{\mathrm{PW}} \cdot \mathrm{PW} \cdot \mathrm{Freq}\Bigg) \cdot 0.25 \cdot 0.25\end{aligned} \tag{4.15}$$



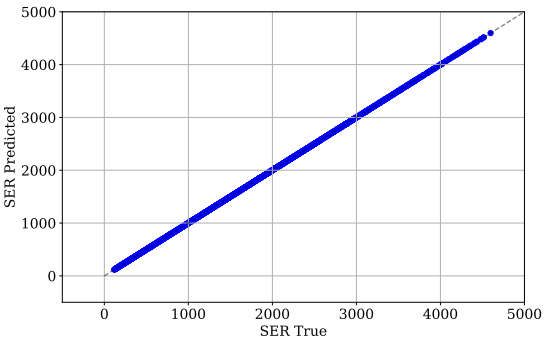
(a) Train Data Set: Model Error



(b) Test Data Set: Model Error



(c) Train Data Set: True vs Predicted Values



(d) Test Data Set: True vs Predicted Values

Figure 4.1: Results of predicting the train and test data set by using Ridge regression with polynomial kernel.

A final customization can be made to clarify the value or the value range for the Pulse Width parameter which is a technology attribute and may not make sense to or be fillable by the final user. Therefore, the ASIC provider, in agreement with the technology provider, may specify values for the PW according to the working environment. A neutron environment (ground applications) with a typical PW of 50 ps, Alpha particles with 10 ps and Heavy Ions with 100 ps could be considered.

From this example it can be noted that the overall SER equation can disclose unit technology data (FIT rates per Mbit, typical pulse widths, etc.), design structure (1 Mbit of flip-flops and 10 Mbit of combinational cells) or knowledge (such as calculation of de-rating factors). In the next phase, the training of the machine learning model, this sensitive information is obscured.

The next step, *Phase III*, is the machine learning model elaboration and consists in exercising the overall SER equation over the range of valid values for the environment, frequency and voltage. The results of this exploration are presented in table 4.2.

Phase III continues with the training of the machine learning models. The here presented work is evaluating the applicability of different machine learning models for

(a) Train Data Set: Model Error

(b) Test Data Set: Model Error

(c) Train Data Set: True vs Predicted Values

(d) Test Data Set: True vs Predicted Values

Figure 4.2: Results of predicting the train and test data set by using k-Nearest Neighbors regression.

the intended usage. Accordingly, several machine learning regression models have been evaluated:

- Linear Models (Linear and Ridge),
- Kernel Ridge Regressor (with Linear, Polynomial, RBF, Sigmoid Kernels),
- Decision Tree Models,
- Neighbors-based Models (K-Nearest Neighbors and Radius Neighbors),
- Support Vector Machine Models (Linear SVR, SVR with various kernels, NuSVR) and
- Multilayer Perceptron Neural Networks

All models have been trained with 60 % of the data set from table 4.2, the train data set. After the training, the models are exercised over the full permitted parameters

range. This allows, on the one hand, to measure the accuracy of the model to recall values which were already in the training data set. On the other hand, by testing the model with data not used for training it is evaluated if the model is able to interpolate and extrapolate the given data. The performance of the models is measured by comparing the predicted values against the reference values and calculate the following metrics:

- MAE - Mean Absolute Error,
- MAX - Max Absolute Error,
- RMSE – Root Mean Squared Error,
- EV - Explained Variance and
- R^2 - Coefficient of Determination

Table 4.3 presents the results for the most accurate and promising models.

The results show that the presented Machine Learning models are able to accurately recall the values from the train data set. The error metrics MAE, MAX, and RMSE are very low and the correlation like metrics EV and R2 are 1 for most of the models. Predicting the test data set shows similarly good results, which means that the models are also able to interpolate and extrapolate the given data. The k-Nearest Neighbors regression is able to perfectly recall the data the data used for training the model, due to the nature of its algorithm. However, predicting new values from the test data set, which were not used for training, shows the models weakness. The inter- and extrapolation capability is less good in comparison to other models.

Figure 4.1 shows the graphs representing the model prediction error for the train and test data set, as well as the correlation between the actual and predicted values when the Ridge Regression with polynomial Kernel is used. In comparison, the results for the k-Nearest Neighbors regression are shown in figure 4.2. The graphs show as well very clearly, that the k-Nearest Neighbors regression is perfectly able to recall the training data but less efficient when predicting the test data.

The results show that certain machine learning models are able to learn the given FuSa/Reliability metric function and it is a good approach for representing extensive reference data sets. Depending on the data or function other models might be less effective. Further, as seen in the example, they might only be appropriate to recall the data but not to inter- and extrapolate it. Other models might be less effective in general. This means for the given data several models need to be considered and evaluated.

Because of the limited and simple training data set, the training of the ML models is fast and does not require computationally intensive resources. A single-shot execution of a trained models is very fast for most models, with execution time inferior to the nanosecond.

Table 4.3: Top Machine Learning Model Performance

ML Model	Data Set	ML Model Error/Correlation Metrics				
		MAE	MAX	RMSE	EV	R2
Ridge Regression	Train	1.688×10^{-6}	1.199×10^{-5}	2.119×10^{-6}	1	1
w/ Polynomial Kernel	Test	1.695×10^{-6}	1.331×10^{-5}	2.106×10^{-6}	1	1
Ridge Regression	Train	2.716×10^{-5}	2.545×10^{-4}	3.530×10^{-5}	1	1
w/ RBF Kernel	Test	2.907×10^{-5}	6.609×10^{-4}	4.421×10^{-5}	1	1
k-Nearest Neighbors Regression	Train	0	0	0	1	1
	Test	21.91	195.2	30.98	0.99	0.99
Support Vector Regression	Train	1.787×10^{-2}	6.830×10^{-2}	2.233×10^{-2}	1	1
w/ Polynomial Kernel	Test	1.756×10^{-2}	6.243×10^{-2}	2.195×10^{-2}	1	1

4.5 Conclusion

In this chapter, a methodology has been proposed to help experts to approach the complexity of hierarchical modelling of reliability and functional safety metrics. The presented approach allows the use, elaboration and distribution of compact machine learning models in a uniform and systematic manner, minimizing both human and CPU efforts while maintaining high accuracy and fidelity. While the approach has been validated and seems to work effectively and efficiently on a modest example, further works will have to consider more complex systems with expanded sets of parameters and with more, non-linear fault models.

Finally, this approach can be used to integrate the effect of multiple failure mechanisms (such as Total Ionizing Dose (TID), circuit aging) that can contribute to an overall applicative failure rate. The effects of such mechanisms depend on generic (such as circuit structure, test-vector/workloads, Process, Voltage, Temperature (PVT), etc.) and specific (total and rate of received dose, age of the circuit, etc.). These trained machine learning models integrate a combination of Transient Faults/Soft Errors, TID and Aging Effects using the same fundamental platform of parameters plus a limited set of effect-specific parameters. The trained machine learning models will be able to quickly evaluate a large variety of effects, encapsulating very useful reliability and functional safety data in a compact and efficient solution that can be used and reused further down the design and manufacturing flow.

Chapter 5

Functional Failures Induced by Single-Event Transients in Clock Distribution Networks

5.1 Introduction

As discussed in chapter 2, the main contributor of to the overall event rate of a system are the Single-Event Upsets in the sequential logic and the Single-Event Transients in the combinational cells. However, due to technology scaling, lower supply voltages and higher operating frequencies, other circuit features such as the clock distribution network (CDN), reset circuitry, etc. become also more vulnerable to transient faults [21, 70, 80, 16] and could cause circuit-wide effects that are more difficult to mitigate and to correct. Indeed, clock buffers from the clock distribution networks have a high fan-out and very few masking mechanism; Single-Event Transients occurring in these cells can potentially reach many sequential cells and state elements and thus, significantly contribute to the overall functional failure rate.

So far, only few works studied the impact of SETs in clock networks. To determine the sensitivity of clock buffer cells to these events, some studies performed accelerated radiation tests of dedicated test chips [80, 48]. Other approaches computed a static failure rate by performing circuit simulation on the electrical-level and thus, obtaining the Electrical De-Rating per clock buffer, as well as the upset rate of the sequential logic due to SETs in the clock network. This upset rate was combined with the functional failure rate due to SEUs in the sequential logic obtained from a SEU fault injection campaign [18, 17]. However, their SET fault injection simulations used only static inputs and thus do not reflect any dynamic behaviour during the runtime of the circuit. Hence, [33] extended this method by injecting SETs in the clock distribution network during a dynamic electrical simulation and thus, obtaining the faulty latching activity of the sequential logic.

Nonetheless, the previous work did not analyse the subsequent impact of SETs on

the functional behaviour of the circuit and furthermore, they are all based on electrical simulations. Since the complexity of today's circuits is increasing, a dynamic simulation of the full circuit on the electrical level is not feasible.

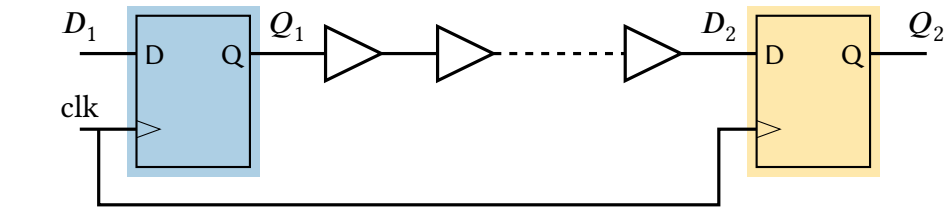
Contrary to the previous work, a new fault model is proposed which is based on logic-level simulation and thus, extends the classical physical and electrical analysis. The fault model is applicable for complex circuits and allows the calculation of a realistic functional failure rate for SETs in the clock distribution network. An initial methodology was published in [41] which proposed a fault model for the functional level and implements the effect of Single-Event Transients in the clock network on logic-level simulation. This methodology was then extended to also consider temporal masking effects. The proposed methods are evaluated by applying it on a practical example and performing a fault injection campaign.

5.2 Single-Event Effect Mechanisms with Regard to Clock Distribution Networks

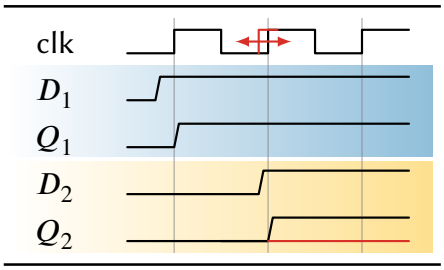
In chapter 2 it was described how energetic particles can affect combinational cells of a chip. This also includes the buffers in the clock tree [66] as well as the Phase Locked Loop (PLL) [72, 28]. The four main structural de-rating mechanisms, Electrical De-Rating (EDR), Temporal De-Rating (TDR), Logical De-Rating (LDR) and Functional De-Rating (FDR), as described in chapter 2.2, are not directly applicable for transients in the clock distribution network. For the propagation of a transient in the clock distribution network, the Logical De-Rating and Temporal De-Rating is limited. Potentially, an SET may be logically masked by a clock gating cell or an enable pin of a flip-flop. Temporal De-Rating is limited as the clock input of the flip-flop is by definition asynchronous.

Seifert et al. identifies two main effects due to transients in the clock network: radiation-induced jitter and radiation-induced race [66]. The effects are illustrated in figure 5.1. Jitter occurs if a transient causes the clock edge to move forward or backward. For the paths with the longest path delays, the most critical paths, the data just arrives close to the rising clock edge and thus, can cause a timing violation (figure 5.1b). A race condition occurs if a transient causes a flip-flop that is closed to become open allowing data to "race" through to the next stage (figure 5.1c).

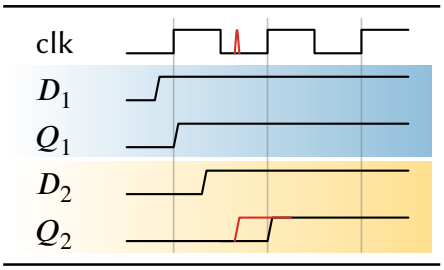
The methodology presented in this chapter aims to compute the error rate for functional failures of a circuit with regards to Single-Event Transients in the clock network. Therefore, the described radiation-induced effects are implemented in a fault model based on logic-level simulation which is presented in the following section.



(a) Example Circuit



(b) Clock Jitter



(c) Clock Race

Figure 5.1: Main effects caused by transients in the clock distribution network.

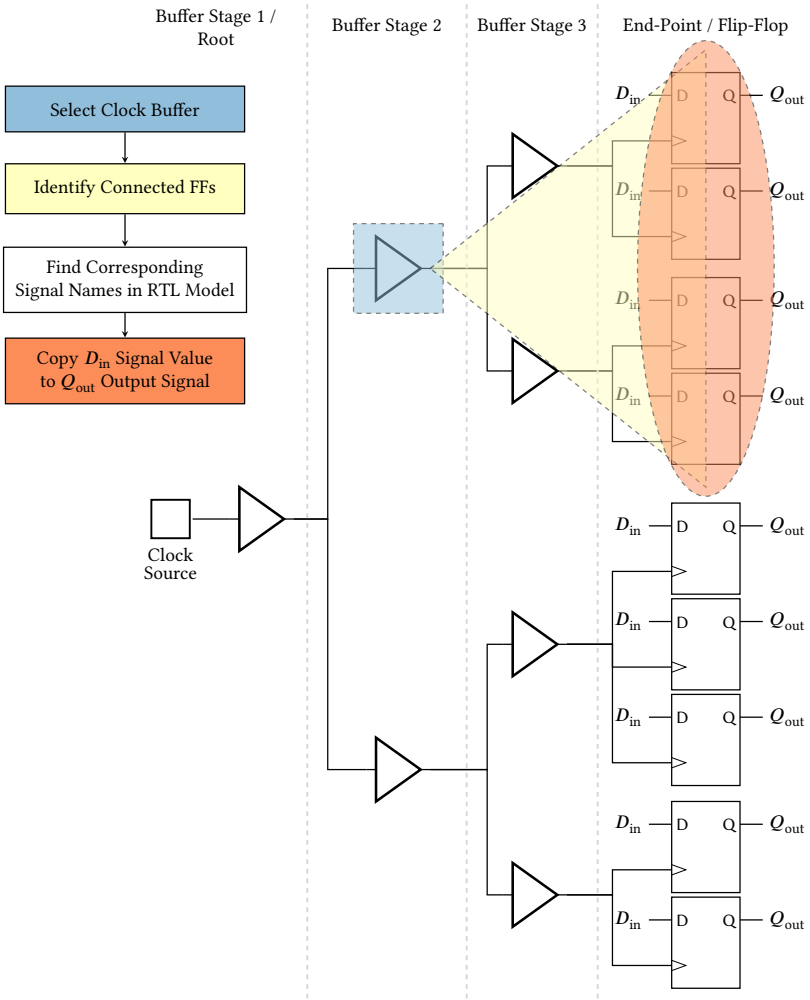
5.3 Methodology

To analyse how the functional behaviour is affected by SETs in the clock distribution network (CDN), the main radiation-induced effects are implemented in a fault model. In order to cope with the complexity of today's circuits the proposed fault model is based on logic-level simulation, which enables a faster analysis than simulations based on the electrical level. Additionally, the temporal masking mechanism is discussed and implemented in an extended fault model. By using the fault model in a fault simulation campaign, the functional failure rate for each clock buffer and the whole network can be calculated. Further, the vulnerability of the sequential logic in relation to these events can be computed.

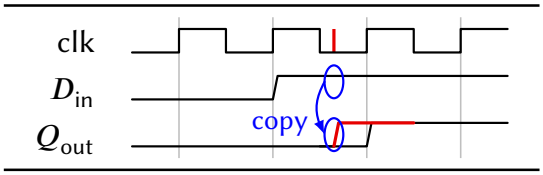
5.3.1 Fault Model for the Functional Level

The proposed fault model for the functional level implements the effect of Single-Event Transients in one of the clock buffers of the clock network. The model is illustrated in figure 5.2a. It is based on logic-level simulations and thus, only uses the Register-Transfer Level (RTL) description of a design. To emulate the SET in the clock network, first, a clock buffer is selected as injection target. Second, all flip-flops which are connected to the endpoint of the selected clock buffer are identified. Then, during the RTL simulation, for each identified flip-flop, the corresponding signal values at the flip-flop output are modified at the injection time. The SET induced clock pulse is imitated by copying the signal value from the flip-flop input signal D_{in} to its output signal Q_{out} as

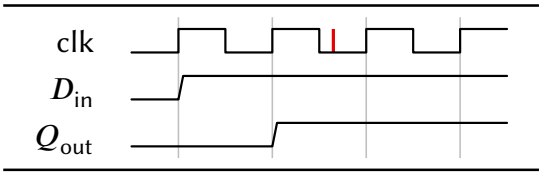
shown in figure 5.2b. Thus, only flip-flops which would have changed their state in the following clock cycle are impacted by the transient and others remain unchanged, as shown in figure 5.2c.



(a) Procedure to inject an SET into a clock buffer



(b) Affected flip-flop (state changed)



(c) Unaffected flip-flop (state unchanged)

Figure 5.2: Proposed fault model for Single-Event Transients in clock distribution networks based on logic level simulation.

The proposed fault model does not take any electrical or timing behaviour into account and thus, is representing a worst-case scenario. However, it can be combined with measured cross-sections of the clock buffer cells obtained during radiation experiments, as shown in [48], or Electrical De-Rating factors obtained from electrical level simulations (without taking the runtime behaviour into account) as described in [18].

Mapping of the RTL Signals to Clock Distribution Network

The proposed method relies on the RTL model of a design. Typically, these models do not provide a clock distribution network. The clock network is obtained by performing a clock network synthesis during the physical design stage of a chip.

In order to perform the fault injection on the RTL design, the clock network is extracted from the placed and routed circuit. For each clock buffer in the network the flip-flops connected at the endpoint are identified and mapped to the corresponding signal name of the RTL design. Thus, a list is created which associates each RTL signal with a buffer on each level of the clock network which is used in the presented fault model.

Fault Injection Simulation Campaign

With the previous described fault model, a logic-level simulation based fault injection campaign can be performed. Therefore, the RTL model of the considered design and a testbench is needed. The testbench allows to verify the correct behaviour of the circuit. This can be done, for example, by monitoring and recording all outputs of the circuit. The record can be used as the golden reference and any difference is considered as a functional failure.

In the fault injection campaign faults are injected into the clock buffers of the clock network at a random point in time according to the described fault model. During each fault injection the changed and unchanged flip-flops are captured and stored. After the injection, the simulation is continued. The circuit output is monitored during the whole simulation and compared to the golden reference. If, according to the monitored output, no failure on the functional level was noted, the injected fault was masked and the correct function is verified. If the functional behaviour is different to the reference, the fault is considered as a functional failure.

This method allows the computation of the Functional De-Rating factor for SETs in a clock buffer and the complete clock network. Further, by tracking the flip-flop changes which led to a functional failure the probability of the sequential logic element to cause a functional failure can be calculated. Thus, the flip-flops which are most probable to cause functional failures can be identified. This information can provide guidelines to the circuit designer to improve robustness of the clock distribution network. For example, techniques for selectively harden the most critical clock buffers are shown in [16] and [49].

5.3.2 Temporal De-Rating

As mentioned in section 5.2, due to the asynchronous behaviour of the clock network, temporal masking of an SET propagating along the clock tree is very limited. However, considering the individual flip-flops at the endpoint of the clock buffer, the SET can affect the flip-flop only within a certain timing window. This behaviour can be considered as Timing De-Rating and is illustrated in figure 5.3 and 5.4.

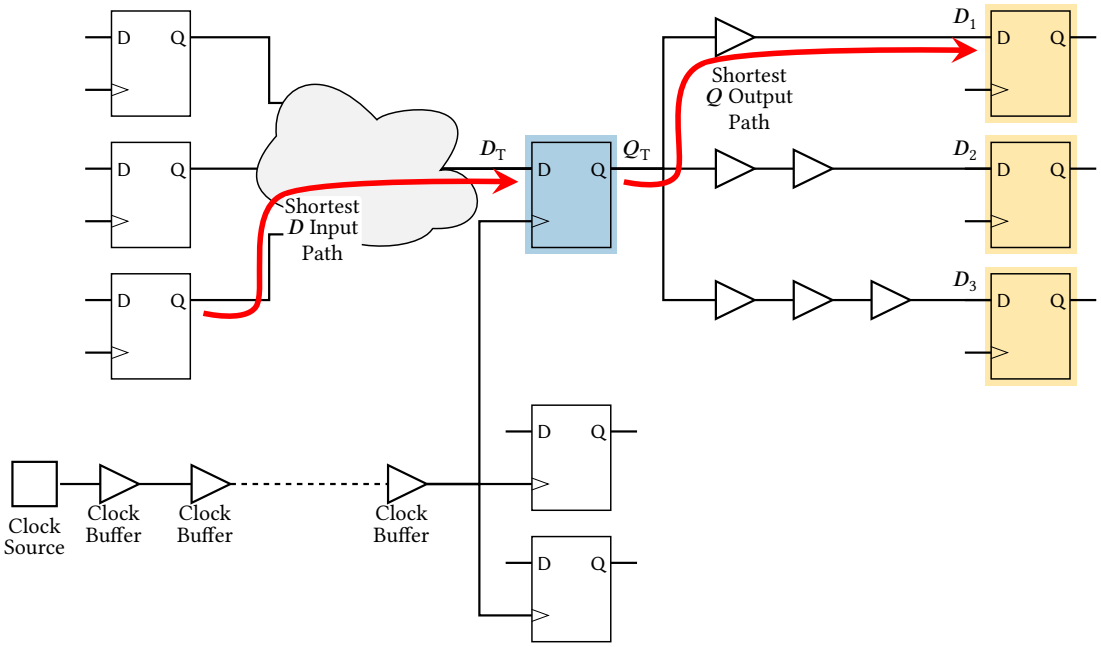


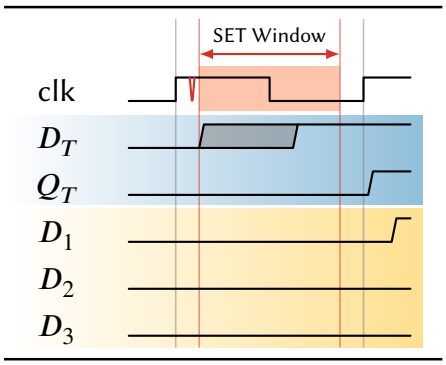
Figure 5.3: Simplified extract of a clock network and connected flip-flops.

The figure 5.3 shows a simplified extract of a clock network with some of the target flip-flops at the endpoint of the clock buffer. When looking at one flip-flop independently, an SET is only effective within a certain timing window. For one target flip-flop, marked in blue, the corresponding timing diagrams are shown figure 5.4.

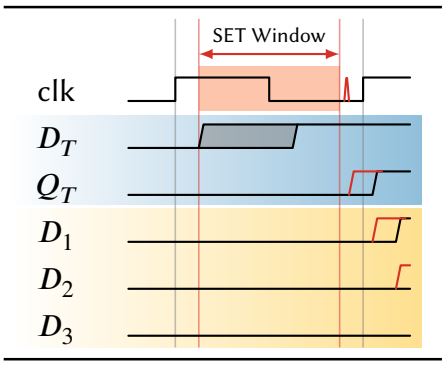
The timing diagrams show that, in case the SET occurs early within the clock cycle, the input D_T of the target flip-flop is still equal to its current state and thus, the SET is masked (figure 5.4a). If the SET occurs between than the shortest path delay $t_{D,\text{best}}$ and longest path delay $t_{D,\text{worst}}$ of the input D_T, the impact of the SET is uncertain. The input D_T can change several times due to the states of the flip-flops of the previous stage and the combinational logic. Thus, the SET may or may not affect the flip-flop. In case the SET occurs after the longest D_T input path delay, the input value of the flip-flop is settled and depending on the current state of the flip-flop the SET might cause the flip-flop to latch an erroneous value, as described in Section 5.3.1 (figure 5.4c). However, considering the downstream path of the target flip-flop, the SET has to occur early enough, in order for the erroneously latched value to propagate to the next stage

(figure 5.4b).

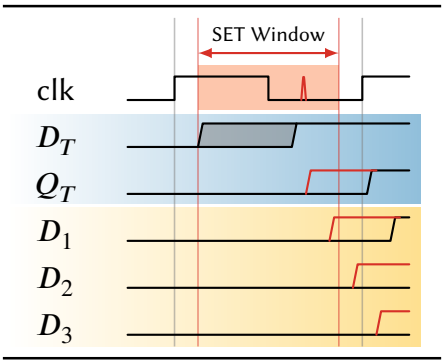
Taking this timing behaviour into account, an SET Timing Window can be defined. In case the SET occurs outside of this timing window it will definitely be temporally masked. The left side of the timing window depends on the shortest input path delay $t_{D,\text{best}}$ of the input D_T. To define the right side of the timing window, the assumption was made that the, by the SET, erroneously latched value has to propagate to at least one flip-flop in downstream path. Thus, the right side can be defined by the clock period T_{cyc} subtracted by the shortest path delay $t_{Q,\text{best}}$ at the output Q_T of the flop-flop.



(a) Masked SET due to shortest input D path delay $t_{D,\text{best}}$



(b) Masked SET due to shortest output Q path delay $t_{Q,\text{best}}$



(c) Not masked SET within SET Timing Window

Figure 5.4: Temporal masking mechanism for SETs in the clock distribution network.

Fault Injection Simulations

One possibility to analyse this temporal masking mechanism would be to inject faults into the post place and route gate-level netlist of the design in a back annotated timing simulation. However, for complex circuits the post place and route simulation can be very computationally intensive and time consuming. Therefore, the proposed fault

model based on logic-level simulations from Section 5.3.1, was extended with the described SET window. Therefore, the shortest path delays for the input and output of each flip-flop have to be extracted from the post place and route circuit. This is usually done within the normal design flow with a Static Timing Analysis (STA) tool. When injecting the SET, additionally to the injection time, the time within the clock cycle is given. According to the prior described fault model, the flip-flops connected to the endpoint of the target clock buffer are identified. Then, for each flip-flop the SET Timing Window is considered separately. The fault mechanism is only applied when the injection time falls inside the flip-flop's SET Timing Window.

5.4 Fault Injection Campaign

In this section the presented methodology and implemented fault model is shown on a practical example. Therefore, the circuit under test and the corresponding testbench is described. Afterwards, the functional failure rate for each clock buffer and for the complete network is computed.

5.4.1 Test Circuit, Testbench and Clock Distribution Network

For this case-study the Ethernet 10GE MAC Core from OpenCores is used, as in section 3.4. The circuit implements the Media Access Control (MAC) functions and consists of control logic, state machines, First-In First Outs (FIFOs) and memory interfaces. It is implemented at the RTL.

The corresponding testbench sends several packets via the 10GE MAC transmit packet interface. The transmit interface is looped-back to the receiver interface in the testbench. The frames are thus processed by the MAC receive engine and the testbench reads frames from the packet receive interface and prints out the results [6]. During the simulation all sent and received packets to and from the core are monitored and recorded. This record is used as the golden reference for the fault injection campaign.

The elaboration of the RTL design identifies 1234 flip-flops. The design was synthesised with Synopsys Design Compiler and Cadence Innovus was used for place and route, and clock tree synthesis (CTS). These steps were implemented by using the NanGate FreePDK45 Open Cell Library [74] and a target frequency of 3.2 ns. The resulting circuit consists of 1202 flip-flops and a clock tree with 5 levels and 33 clock buffers. The last level clock buffers have 21 to 76 endpoints. figure 5.5 depicts the resulting tree.

Due to logic optimisation the final circuit has fewer flip-flops than the original RTL design. In order to perform the fault injection on the logic-level the RTL signal names have been mapped to the corresponding flip-flops in the gate-level netlist according to their names. Thus, a list was obtained which maps the flip-flops from the RTL design to the clock buffers of the clock distribution network.

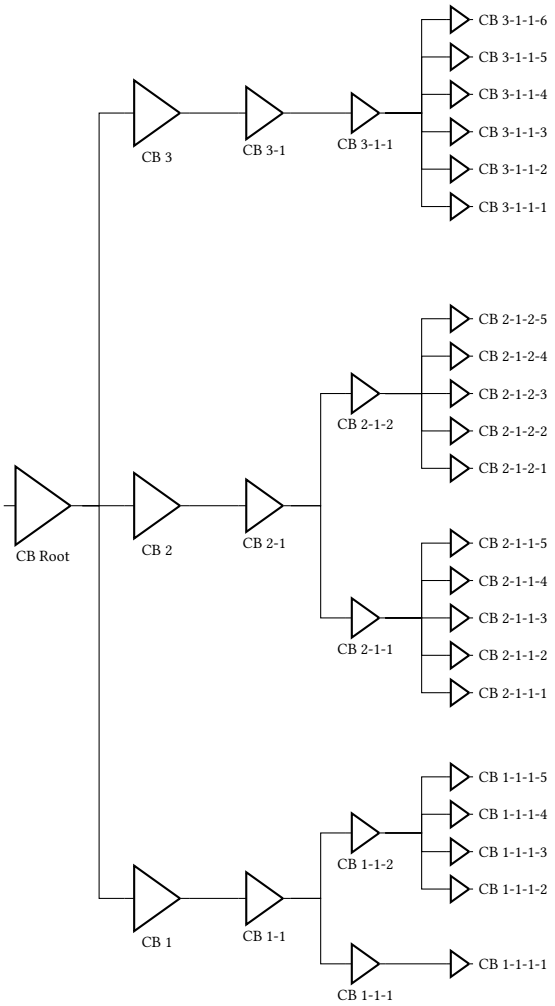


Figure 5.5: Obtained clock tree after clock tree synthesis.

5.4.2 Clock-SET Fault Injection Campaign

A fault injection campaign was performed to analyse the functional failure rate of SETs in the clock distribution network (CDN). Therefore, the SETs were injected at 200 different clock cycles in each of the 33 clock buffer. The faults were injected only during the active phase of the simulation, when packets are sent and received through the user packet interface.

To take the temporal masking mechanism into account, additional fault injection campaigns have been performed. Therefore, faults were injected at the same 200 clock cycles as for the Functional De-Rating campaign. For each clock cycle 25 SETs were injected at different random times within the clock period. In order to consider the variation of the path delays at different operating conditions (Voltage, Temperature, etc.) the simulations have been performed for three different corner cases, provided by

the technology library:

- slow with Vdd=0.95 V and Tj=125.0 ℃
- typical with Vdd=1.1 V and Tj=25.0 ℃
- fast with Vdd=1.25 V and Tj=0.0 ℃

In order to obtain a reference for the Functional and Temporal De-Rating campaigns, fault injection campaigns were performed by using the post place and route gate-level netlist in a back annotated timing simulation. For the back annotated timing simulation the Standard Delay Format (SDF) files were generated by the Synopsys Static Timing Analysis Tool PrimeTime for the different operating conditions. As for the combined Functional and Temporal De-Rating campaigns, SETs have been injected at the exact same times, 25 per each of the 200 clock cycles.

The results for the campaigns are presented in table 5.1. The number of reached, changed and unchanged flip-flops are listed for each campaign. Further, the number of injections which lead to a functional failure are shown as well as the resulting Functional Failure Rate/Functional De-Rating (FDR). As expected, the Functional De-Rating analysis without the temporal masking effect shows the highest Functional Failure Rate (RTL wo/ TDR). The failure rate is lowered by 5 % to 17 % when the temporal masking mechanism is taken into account, depending on the operating conditions (RTL w/ TDR). The fault injection campaigns by using the post place and route netlist and injecting in the back annotated timing simulation show (GLN w/ SDF) similar results. The biggest difference can be seen at the slowest operating conditions.

Comparison of the Clock-SET Campaigns

Comparing the failure rates of the different campaigns, it was noted that the failure rates differ depending on the considered operating condition. This can be explained by the usually longer path delays for the slow corner and shorter path delays for the typical and fast corners. The distributions of the shortest input path delay $t_{D,\mathrm{best}}$ and the shortest output path delay $t_{Q,\mathrm{best}}$ for each flip-flop are shown in figure 5.6. In these histograms the slack of the shortest output path is shown which is calculated by subtracting the shortest output delay $t_{Q,\mathrm{best}}$ from the clock period T_{cyc}. In this way the shortest input path delay $t_{D,\mathrm{best}}$ reflects the left side of the SET Timing Window and the slack of the shortest output path $T_{\mathrm{cyc}} - t_{Q,\mathrm{best}}$ the right side. Further, the distribution of the length of the SET Timing Window $T_{\mathrm{SET\,TW}}$ is shown. It can be seen that the SET Timing Window is shorter for the slow corner and gets longer for the typical and fast operating conditions. This fact is illustrated for one flip-flop in figure 5.7.

In general, the time an SET can occur within the clock cycle can be assumed as uniform. Figure 5.8 shows that this is also the case for the performed fault injection campaigns. Thus, for longer SET Timing Windows higher failure rates can be expected and vice versa lower failure rates with shorter SET Timing Windows. This also leads to

Table 5.1: Results of the Clock-SET Fault Injection Campaigns with and without Temporal Masking

Fault Injection Campaign	Operating Condition	Number of Injections	Reached FFs	Reached FFs per Injection	Changed FFs	Changed FFs per Injection	Unchanged FFs	Unchanged FFs per Injection	Functional Failures	Functional Failure Rate
RTL wo/ TDR	-	6600	1234000	186.97	106065	16.07	1127935	170.90	4724	71.58 %
RTL w/ TDR	slow	165000	30850000	186.97	2042338	12.38	28807662	174.59	97928	59.35 %
RTL w/ TDR	typical	165000	30850000	186.97	2423165	14.69	28426835	172.28	110294	66.84 %
RTL w/ TDR	fast	165000	30850000	186.97	2486130	15.07	28363870	171.90	112471	68.16 %
GLN w/ SDF	slow	165000	30050000	182.12	2134051	12.93	27915949	169.19	90765	55.01 %
GLN w/ SDF	typical	165000	30050000	182.12	2469710	14.97	27580290	167.15	108473	65.74 %
GLN w/ SDF	fast	165000	30050000	182.12	2527826	15.32	27522174	166.80	111586	67.63 %

the assumption that a circuit operating at a higher frequency and thus, has lower slack, also has a higher clock-SET Temporal De-Rating, which results in a lower failure rate.

The Functional Failure Rates obtained by performing fault injection in the back annotated timing simulation by using the gate-level netlist are similar to the corresponding RTL fault injection campaigns with TDR. However, due to the more accurate timing simulation, the results obtained with the post place and route netlist are lower and more accurate. Nonetheless, especially for complex circuits a full post place and route fault injection campaign is very computationally intensive and probably not feasible. Thus, the proposed combined Functional and Temporal De-Rating approach are useful to compute failure rates for pessimistic worst-case scenario.

5.4.3 Methodology to Approximate the Clock-SET Temporal De-Rating

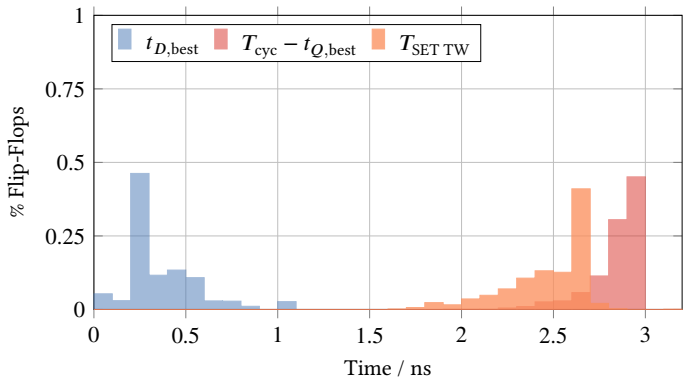
To evaluate the Temporal Masking effect and calculate a Temporal De-Rating in a fault injection simulation campaign additional simulations are required compared with the plain Functional De-Rating analysis. Hence, an estimation with a faster method might be preferable and sufficient in some cases. The observations and discussion from the previous section show that there is a strong dependency between the length of the SET Timing Window $T_{\text{SET TW}}$ and the Temporal De-Rating. Following this, an approach can be derived which takes the average length of the SET Timing Window $\overline{T}_{\text{SET TW}}$ of all flip-flops in the circuit. Normalizing this value to the clock period T_{cyc} gives a rough estimate of the Timing De-Rating factor for the full circuit:

$$\text{TDR}_{\text{Estimated}} = \frac{\overline{T}_{\text{SET TW}}}{T_{\text{cyc}}} \tag{5.1}$$

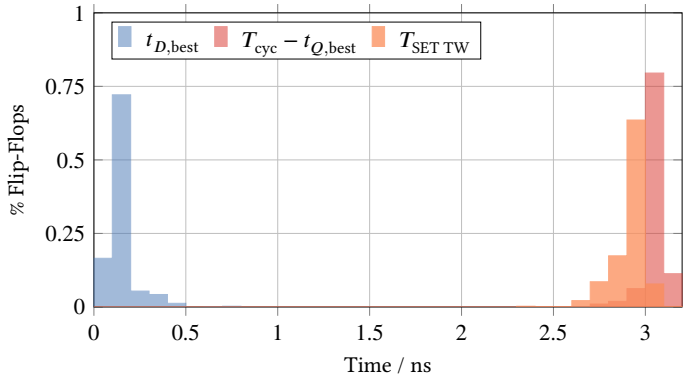
This method was applied to the performed campaigns and the results are listed in table 5.2. Therefore, the average length of each SET Timing Window was calculated and normalized to the used clock period T_{cyc} according to equation (5.1). The estimated TDR factor obtained in this way are then multiplied with the Functional De-Rating (FDR) value obtained from the plain Functional De-Rating simulations FDR = 71.58 %. The resulting estimation, shown in the table, is close to the computed values obtained from the exhaustive fault injection simulation campaigns (see table 5.1).

5.4.4 Results for Temporal Masking of SEUs in the Sequential Logic

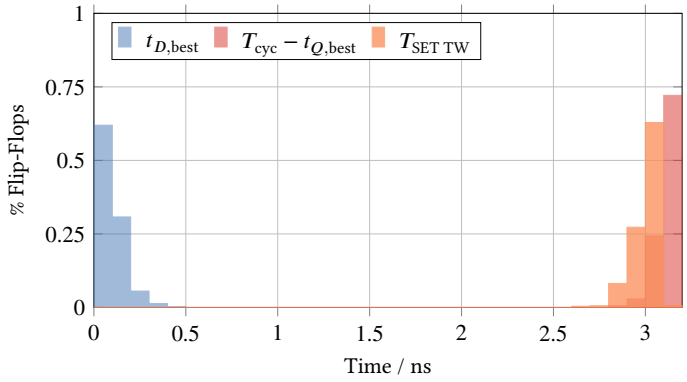
The functional failure rate caused by SEUs in the flip-flops is obtained by a classical full flat statistical fault injection campaign. The SEU is emulated by modifying the stored value of a flip-flop at a random point in time during the simulation. Similar to the SET



(a) Slow Corner



(b) Typical Corner



(c) Fast Corner

Figure 5.6: Distribution of the shortest input path delay $t_{D,\text{best}}$, the slack of the shortest output path and the SET Timing Window length $T_{\text{SET TW}}$. The slack of shortest output path is calculated as the difference between the clock period T_{cyc} and shortest output path delay $t_{Q,\text{best}}$.

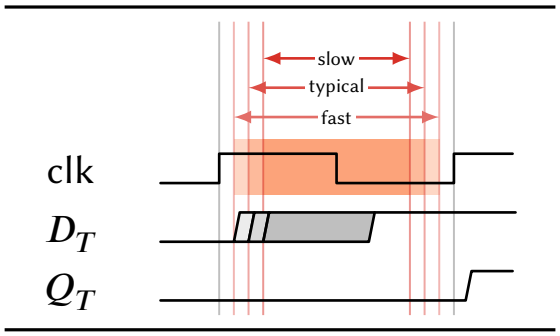


Figure 5.7: Length of the SET Timing Window $T_{\text{SET TW}}$ for different operating conditions.

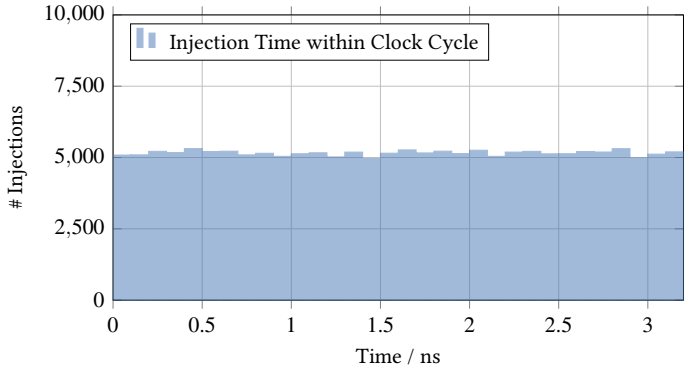


Figure 5.8: Distribution of the injection times within the clock cycle.

fault injection campaign, any difference in the send or received packages is considered as a functional failure of the application.

For the campaign 200 SEUs were injected in each of the 1234 flip-flops. Thereby, 53756 of the injected faults showed a functional failure, which corresponds to a Functional De-Rating factor of 21.78 %. Table 5.3 summarizes the overall results of the SEU fault injection campaign.

As described in section 2.2 also an SEU in a flip-flop can be masked due to timing behaviour. The SEU has to occur sufficiently early in the clock period in order not to be temporally masked. The erroneous value is then able to meet the setup time of one or more flip-flops in the downstream path. Intuitively, it is clear that as the available slack time on the paths increases, so does the probability of an SEU being captured. The Temporal De-Rating factor (TDR) for SEUs can be approximated by first, calculating the ratio of the slack t_{slack} to the clock period from each startpoint i to each of its endpoints j:

$$\text{TDR}_{\text{path}(i,j)} = \frac{t_{\text{slack,path}(i,j)}}{T_{\text{cyc}}} \tag{5.2}$$

After computing the TDR factors of each path, for each flip-flops its worst path value

Table 5.2: Estimated Timing De-Rating Factors

Operating Condition	Average SET Timing Window	Averaged $TDR_{Estimated}$	$FDR \times TDR_{Estimated}$
slow	2.472 ns	77.25 %	55.30 %
typical	2.922 ns	91.30 %	65.35 %
fast	3.001 ns	93.78 %	67.13 %

Table 5.3: Results of the SEU Fault Injection Campaign

	Total	Per Injection
Injection Targets (FFs)	1234	-
Number of Injections (SEU)	246800	-
Functional Failure	53756	21.78 %

(highest TDR) is assigned. This corresponds to the consideration that an SEU is propagating if the erroneous value has the time to propagate to at least one of the endpoints.

The Temporal De-Rating factors for each flip-flop were calculated for the considered operating conditions. Similarly, to the temporal analysis for the SETs in the clock distribution network, the path delay/slack values were extracted for the three considered operating conditions by using Synopsys PrimeTime. Table 5.4 shows the averaged TDR values for the full circuit and the weighted Functional Failure Rate.

5.4.5 Comparison and Discussion

To compare the functional failure rate due to SEUs in the sequential logic to the failure rate due to SETs in the clock network table 5.5 summarizes the average Functional and Temporal De-Rating factor per element. For the temporal masking the worst-case corner, the fast operating condition, was considered. It can be seen that the Functional De-Rating factor for the clock-SET events are about 3.5 times higher than for the SEUs in the sequential logic and the Temporal De-Rating factors are about the same for both events. However, the number of sequential elements is 40 times higher and thus, the SEUs in flip-flops are the leading contributor to the overall functional failure rate of the circuit.

Considering further physical effects the Functional De-Rating factor can be combined with a Failure in Time (FIT) rate obtained from a characterized standard cell library. In [19] FIT values for the NanGate FreePDK45 Open Cell Library [74] were obtained by using dedicated tools to calculate the event rate of cells. The average values for D-Flip-Flops and clock buffers were considered. These show that the FIT value for the sequential logic is about 2 times higher, which further lowers the effect of SETs

Table 5.4: Results of the Sequential Temporal De-Rating Analysis

Operating Condition	TDR	FDR × TDR
slow	81.50 %	17.75 %
typical	90.66 %	19.75 %
fast	92.34 %	20.11 %

in the clock network.

Nonetheless, if a fully SEU hardened circuit is considered, the sensitivity of the sequential logic is lowered. Depending on the implementation of the hardened cells, the sensitivity is usually about one order of magnitude lower than the one of un-hardened cells. Taking this into account, the Functional De-Rating factor would be lowered by the same amount. Since the usual hardening techniques for the sequential logic is affecting the sensitivity of the clock network to clock-SETs, the functional failure rates due to SETs in the clock network become more relevant.

Table 5.5: Summary of the Functional Failure Rate Analysis of the 10GE MAC circuit

Event	Affected Element	Number of Elements	Average TDR	Average FDR	FIT	Weighted FIT
SEUs in Flip-Flops	Flip-Flop	1234	0.92	0.22	148.33	37047
SETs in Clock Network	Clock Buffer	33	0.93	0.72	72.0	1590

5.5 Conclusion

This chapter investigates how the impact of Single-Event Transients (SETs) in the clock distribution network on the functional behaviour of a circuit can be analysed. Therefore, a methodology and a fault model were presented which implement the main radiation-induced effects in clock networks. The method enables the computation of the functional failure rate in a logic-level simulation based on the register-transfer level of the design. Thus, a faster evaluation can be performed than by simulating on the electrical or gate level.

Further, a temporal masking mechanism for Single-Event Transients (SETs) in clock distribution networks was introduced. The temporal masking is based on the shortest input path delay of the flip-flops and the shortest output path delay, which are defining an SET Timing Window within the clock cycle. SETs occurring outside of this window are masked. The fault model was extended considering this temporal masking effect which allows to compute the functional failure rate weighted with temporal de-rating.

The approach was applied on a practical example. SETs were injected into the clock network of the circuit under test in a fault injection campaign. Thus, the functional

failure rates of the clock network and the individual clock buffers were determined. Further, the results of the extended fault model with temporal masking were verified with a back annotated timing simulation of the post place and route netlist. It was shown that the proposed temporal masking implementation is able to compute a pessimistic worst case.

Finally, the functional failure rate due to SETs in the clock network has been compared to SEUs in the sequential logic. The discussion has shown that the SETs in the clock network usually cause functional failures which results in a high functional failure rate. The comparison has also shown that the leading contribution to the overall failure rate of the circuit are from SEUs in the sequential logic. However, considering circuits which are only protected against SEUs, clock-SETs become more relevant.

Chapter 6

Conclusion

Faults induced by radiation are one of the main sources of failures in today's electronic devices. Besides the typical high-reliability applications, also common systems must be designed to consider the reliability impact of soft errors. The research community has studied the physics behind the phenomena in detail and offers solutions for each step of the design flow. However, the growth in complexity and number of transistors in a device, vastly increases the required efforts to analyse a system. Thus, the assessment of a modern complex design remains a challenging task. The methodologies presented in this thesis are addressing these problems, by proposing new solutions on the higher abstraction level. The thesis proposed new approaches to advance the fault analysis on the functional level.

6.1 Machine Learning Techniques for Functional Failure Analysis

The first contribution of this thesis explored the use of machine learning techniques for the functional fault analysis of complex circuits. In chapter 3 two different machine learning based approaches are proposed which aim to reduce the required efforts to determine the propagation probability of faults on a functional level. A feature set to characterise each individual sequential element in the circuit was developed. This feature set combines structural, synthesis and signal activity related attributes.

The first approach accelerates a fine-grained functional failure analysis by reducing the computational cost to determine the FDR factors of the circuit's sequential logic. The approach allows the prediction of FDR factors per individual instances. Several machine learning models were evaluated varying the training size and predicting different failure classes. It was shown that the cost of a fault injection campaign can be reduced by a factor of 2 up to 5 in comparison to a classical statistical fault injection campaign. The method was also shown to be applicable in early design phases, working on the RT-level of the design. A feature subset was identified which can be extracted from an

elaborated RTL description of the circuit. The results have shown that the impact on the prediction performance was marginal.

The second approach uses machine learning clustering to reduce the fault space by grouping the sequential elements of the circuit together which are expected to have a similar sensitivity to faults. This allows to perform fault injection campaigns more efficiently. The approach advances already existing clustering approaches in the way that it is more flexible and does not make any assumption of the circuit or its representation. The effectiveness of the grouping by different machine learning clustering algorithms was evaluated on a practical example and compared to a random and ideal solution. With the approach it was possible to reduce fault injection efforts by a factor of 5× to 20×.

6.2 Cross-Layer Reliability and Functional Safety Assessment Through Machine Learning

Once the reliability analysis of a design was performed, the information is usually provided by using compact models to the end user. Since the typical design flow is hierarchical and relies on assembling many individual technology elements, from standard cells to complete boards, the designer has to group simpler elements in more complex structures. Thereby, the designer has to manage the corresponding propagation of reliability and functional safety information through the hierarchy of the system, with the additional problems of IP confidentiality, possibility of reverse engineering, etc. To address these problems, the thesis contributes by presenting a machine learning based approach which is able to integrate the many individual models of a subsystem's elements in a single compact model that can be re-used and assembled further up in the hierarchy. In this way, the machine learning model allows the technology/IP/component/system provider to accompany their product with compact reliability and functional safety models which provide consistency, accuracy, and confidentiality and can be safely used by their users.

6.3 Functional Failures Induced by Single-Event Transients in Clock Distribution Networks

Chapter 5 analyses SETs in Clock Distribution Network (CDN) on a functional level. The thesis contributes by presenting a methodology and a fault model which implement the main radiation-induced effects in clock networks. The method enables the computation of the functional failure rate in a logic-level simulation based on the RTL description of the design. This allows a faster evaluation of faults on the functional behaviour than by simulating on the electrical level or gate-level. Further, the Temporal Masking effect for SETs in clock distribution networks is introduced. The Temporal

Masking is based on the shortest input path delay of the flip-flops and the shortest output path delay, which are defining an SET Timing Window within the clock cycle. The functional fault model was extended by including this Temporal Masking effect which allows to compute the functional failure rate weighted with Temporal De-Rating (TDR) of a circuit. The approach was applied in a practical example where SET were injected into the clock network of the circuit under test in a fault injection campaign. It was shown that the proposed Temporal Masking implementation is able to compute a pessimistic worst case.

6.4 Summary

Electronics play a critical role in all aspects of the everyday life. The complexity of integrated systems is increasing and creating a challenge for the reliability analysis. There is a great need for new techniques to evaluate complex systems which are able to scale with the increasing complexity. The analysis has to be shifted to higher abstraction levels to hide unnecessary details. The key contribution of this thesis is to provide new techniques to analyse faults on a functional level which use higher abstraction levels and thus, complement previous work on Soft Errors.

Bibliography

[1] Accellera Systems Initiative Inc. *Functional Safety Working Group*. English. White Paper. 2021.

[2] D. Alexandrescu and E. Costenaro. "Towards Optimized Functional Evaluation of SEE-Induced Failures in Complex Designs." In: *2012 IEEE 18th International On-Line Testing Symposium (IOLTS)*. June 2012, pp. 182–187. DOI: 10.1109/IOLTS.2012.6313869.

[3] D. Alexandrescu, E. Costenaro, and A. Evans. "State-Aware Single Event Analysis for Sequential Logic." In: *2013 IEEE 19th International On-Line Testing Symposium (IOLTS)*. July 2013, pp. 151–156. DOI: 10.1109/IOLTS.2013.6604067.

[4] Dan Alexandrescu et al. "Enabling Cross-Layer Reliability and Functional Safety Assessment through ML-Based Compact Models." In: *2020 IEEE 26th International Symposium on On-Line Testing and Robust System Design (IOLTS)*. July 2020, pp. 1–6. DOI: 10.1109/IOLTS50870.2020.9159750.

[5] E. Alpaydin and F. Bach. *Introduction to Machine Learning*. Adaptive Computation and Machine Learning Series. MIT Press, 2014. ISBN: 978-0-262-32575-2.

[6] Andre Tanguay. *10GE MAC Core Specification*. Jan. 2013.

[7] C. E. Barton et al. "International Geomagnetic Reference Field, 1995 Revision: International Association of Geomagnetism and Aeronomy (IAGA) Division V, Working Group 8." In: *Geophysical Journal International* 125.1 (Apr. 1996), pp. 318–321. ISSN: 1365-246X. DOI: 10.1111/j.1365-246X.1996.tb06553.x.

[8] R. Baumann. "The Impact of Technology Scaling on Soft Error Rate Performance and Limits to the Efficacy of Error Correction." In: *Digest. International Electron Devices Meeting,* Dec. 2002, pp. 329–332. DOI: 10.1109/IEDM.2002.1175845.

[9] R. C. Baumann. "Radiation-Induced Soft Errors in Advanced Semiconductor Technologies." In: *IEEE Transactions on Device and Materials Reliability* 5.3 (Sept. 2005), pp. 305–316. ISSN: 1530-4388. DOI: 10.1109/TDMR.2005.853449.

[10] M. Bellato et al. "Evaluating the Effects of SEUs Affecting the Configuration Memory of an SRAM-Based FPGA." In: *Automation and Test in Europe Conference and Exhibition Proceedings Design*. Vol. 1. Feb. 2004, 584–589 Vol.1. DOI: 10.1109/DATE.2004.1268908.

[11] James Bergstra and Yoshua Bengio. "Random Search for Hyper-Parameter Optimization." In: *J. Mach. Learn. Res.* 13 (Feb. 2012), pp. 281–305. ISSN: 1532-4435.

[12] Christopher Bishop. *Pattern Recognition and Machine Learning*. en. Information Science and Statistics. New York: Springer-Verlag, 2006. ISBN: 978-0-387-31073-2.

[13] Arkady Bramnik, Andrei Sherban, and Norbert Seifert. "Timing Vulnerability Factors of Sequential Elements in Modern Microprocessors." In: *2013 IEEE 19th International On-Line Testing Symposium (IOLTS)*. July 2013, pp. 55–60. DOI: 10.1109/IOLTS.2013.6604051.

[14] Anselm Breitenreiter et al. "Selective Fault Tolerance by Counting Gates with Controlling Value." In: *2019 IEEE 25th International Symposium on On-Line Testing and Robust System Design (IOLTS)*. July 2019, pp. 15–20. DOI: 10.1109/IOLTS.2019.8854380.

[15] C/DA - Design Automation. *Standard for Exchange/Interoperability Format for Functional Safety Analysis and Functional Safety Verification of IP, SoC and Mixed Signal ICs*. English. Standard P2851. IEEE Computer Society, 2019.

[16] S. Chellappa, L. T. Clark, and K. E. Holbert. "A 90-Nm Radiation Hardened Clock Spine." In: *IEEE Transactions on Nuclear Science* 59.4 (Aug. 2012), pp. 1020–1026. ISSN: 0018-9499. DOI: 10.1109/TNS.2012.2183647.

[17] R. Chipana et al. "SET Susceptibility Estimation of Clock Tree Networks from Layout Extraction." In: *2012 13th Latin American Test Workshop (LATW)*. Apr. 2012, pp. 1–6. DOI: 10.1109/LATW.2012.6261256.

[18] R. Chipana et al. "Soft-Error Probability Due to SET in Clock Tree Networks." In: *2012 IEEE Computer Society Annual Symposium on VLSI*. Aug. 2012, pp. 338–343. DOI: 10.1109/ISVLSI.2012.39.

[19] Enrico Costenaro et al. "A Practical Approach to Single Event Transient Analysis for Highly Complex Design." en. In: *Journal of Electronic Testing* 29.3 (June 2013), pp. 301–315. ISSN: 1573-0727. DOI: 10.1007/s10836-013-5385-9.

[20] A. B. Dhia, L. Naviner, and P. Matherat. "Evaluating CLB Designs under Multiple SETs in SRAM-Based FPGAs." In: *2013 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*. Oct. 2013, pp. 112–117. DOI: 10.1109/DFT.2013.6653592.

[21] P. E. Dodd and L. W. Massengill. "Basic Mechanisms and Modeling of Single-Event Upset in Digital Microelectronics." In: *IEEE Transactions on Nuclear Science* 50.3 (June 2003), pp. 583–602. ISSN: 0018-9499. DOI: 10.1109/TNS.2003.813129.

[22] A. Evans et al. "Clustering Techniques and Statistical Fault Injection for Selective Mitigation of SEUs in Flip-Flops." In: *International Symposium on Quality Electronic Design (ISQED)*. Mar. 2013, pp. 727–732. DOI: 10.1109/ISQED.2013.6523691.

[23] Adrian Evans. "Abstraction Techniques for Scalable Soft Error Analysis and Mitigation." en. PhD thesis. Université de Grenoble, June 2014.

[24] Adrian Evans et al. "RIIF - Reliability Information Interchange Format." In: *2012 IEEE 18th International On-Line Testing Symposium (IOLTS)*. June 2012, pp. 103–108. DOI: 10.1109/IOLTS.2012.6313849.

[25] Federal Aviation Administration. *DO-254 - Design Assurance Guidance for Airborne Electronic Hardware*. English. Standard DO-254. Washington, D.C., United States: U.S. Dept. of Transportation, Federal Aviation Administration, 2000.

[26] V. Ferlet-Cavrois et al. "Investigation of the Propagation Induced Pulse Broadening (PIPB) Effect on Single Event Transients in SOI and Bulk Inverter Chains." In: *IEEE Transactions on Nuclear Science* 55.6 (Dec. 2008), pp. 2842–2853. ISSN: 1558-1578. DOI: 10.1109/TNS.2008.2007724.

[27] V. Ferlet-Cavrois et al. "Large SET Duration Broadening in a Fully-Depleted SOI Technology—Mitigation With Body Contacts." In: *IEEE Transactions on Nuclear Science* 57.4 (Aug. 2010), pp. 1811–1819. ISSN: 1558-1578. DOI: 10.1109/TNS.2010.2048927.

[28] Tomohiro Fujita, SinNyoung Kim, and Hidetoshi Onodera. "Computer Simulation of Radiation-Induced Clock-Perturbation in Phase-Locked Loop with Analog Behavioral Model." In: *Fifteenth International Symposium on Quality Electronic Design*. Mar. 2014, pp. 230–235. DOI: 10.1109/ISQED.2014.6783330.

[29] M.J. Gadlage et al. "Single Event Transient Pulse Widths in Digital Microcircuits." In: *IEEE Transactions on Nuclear Science* 51.6 (Dec. 2004), pp. 3285–3290. ISSN: 1558-1578. DOI: 10.1109/TNS.2004.839174.

[30] Massoud Mokhtarpour Ghahroodi et al. "Timing Vulnerability Factors of Ultra Deep-Sub-Micron CMOS." In: *2011 Sixteenth IEEE European Test Symposium*. May 2011, pp. 202–202. DOI: 10.1109/ETS.2011.40.

[31] B. Gill, N. Seifert, and V. Zia. "Comparison of Alpha-Particle and Neutron-Induced Combinational and Sequential Logic Error Rates at the 32nm Technology Node." In: *2009 IEEE International Reliability Physics Symposium*. Apr. 2009, pp. 199–205. DOI: 10.1109/IRPS.2009.5173251.

[32] Masami Hane et al. "Synthetic Soft Error Rate Simulation Considering Neutron-Induced Single Event Transient from Transistor to LSI-Chip Level." In: *2008 International Conference on Simulation of Semiconductor Processes and Devices*. Sept. 2008, pp. 365–368. DOI: 10.1109/SISPAD.2008.4648313.

[33] P. Hao and S. Chen. "Single-Event Transient Susceptibility Analysis and Evaluation Methodology for Clock Distribution Network in the Integrated Circuit Working in Real Time." In: *IEEE Transactions on Device and Materials Reliability* 17.3 (Sept. 2017), pp. 539–548. ISSN: 1530-4388. DOI: 10.1109/TDMR.2017.2733218.

[34] Ryo Harada et al. "Neutron Induced Single Event Multiple Transients with Voltage Scaling and Body Biasing." In: *2011 International Reliability Physics Symposium*. Apr. 2011, pp. 3C.4.1–3C.4.5. DOI: 10.1109/IRPS.2011.5784485.

[35] Ray Heald. "How Cosmic Rays Cause Computer Downtime." In: *IEEE Rel. Soc. SCV Meeting*. 2005.

[36] "IEEE Standard for Information Technology - Local and Metropolitan Area Networks - Part 3: CSMA/CD Access Method and Physical Layer Specifications - Media Access Control (MAC) Parameters, Physical Layer, and Management Parameters for 10 Gb/s Operation." In: *IEEE Std 802.3ae-2002 (Amendment to IEEE Std 802.3-2002)* (Aug. 2002), pp. 1–544. DOI: 10.1109/IEEESTD.2002.94131.

[37] International Electrotechnical Commission. *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems*. en. Standard IEC 61508-1:2010. Geneva, Switzerland: International Electrotechnical Commission, 2010.

[38] ISO Central Secretary. *Road Vehicles — Functional Safety*. English. Standard ISO 26262-1:2018. Geneva, CH: International Organization for Standardization, 2018.

[39] JEDEC Solid State Technology Association. *Test Method for Beam Accelerated Soft Error Rate*. English. Standard JESD89-3A. Virginia, USA: JEDEC Solid State Technology Association, 2012.

[40] Ron Kohavi. "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection." In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*. IJCAI'95. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, pp. 1137–1143. ISBN: 978-1-55860-363-9.

[41] T. Lange et al. "Functional Failure Rate Due to Single-Event Transients in Clock Distribution Networks." In: *2019 14th International Conference on Design Technology of Integrated Systems In Nanoscale Era (DTIS)*. Apr. 2019, pp. 1–6. DOI: 10.1109/DTIS.2019.8735052.

[42] T. Lange et al. "On the Estimation of Complex Circuits Functional Failure Rate by Machine Learning Techniques." In: *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks – Supplemental Volume (DSN-S)*. June 2019, pp. 35–41. DOI: 10.1109/DSN-S.2019.00021.

[43] Thomas Lange et al. "Machine Learning Clustering Techniques for Selective Mitigation of Critical Design Features." In: *2020 IEEE 26th International Symposium on On-Line Testing and Robust System Design (IOLTS)*. July 2020, pp. 1–7. DOI: 10.1109/IOLTS50870.2020.9159751.

[44] Thomas Lange et al. "Machine Learning to Tackle the Challenges of Transient and Soft Errors in Complex Circuits." In: *2019 IEEE 25th International Symposium on On-Line Testing and Robust System Design (IOLTS)*. July 2019, pp. 7–14. DOI: 10.1109/IOLTS.2019.8854423.

[45] Yanmei Li, Dongmei Li, and Zhihua Wang. "A New Approach to Detect-Mitigate-Correct Radiation-Induced Faults for SRAM-Based FPGAs in Aerospace Application." In: *Proceedings of the IEEE 2000 National Aerospace and Electronics Conference. NAECON 2000. Engineering Tomorrow (Cat. No.00CH37093)*. 2000, pp. 588–594. DOI: 10.1109/NAECON.2000.894965.

[46] H. Liang et al. "A Methodology for Characterization of SET Propagation in SRAM-Based FPGAs." In: *IEEE Transactions on Nuclear Science* 63.6 (Dec. 2016), pp. 2985–2992. ISSN: 0018-9499. DOI: 10.1109/TNS.2016.2620165.

[47] N. N. Mahatme et al. "Comparison of Combinational and Sequential Error Rates for a Deep Submicron Process." In: *IEEE Transactions on Nuclear Science* 58.6 (Dec. 2011), pp. 2719–2725. ISSN: 1558-1578. DOI: 10.1109/TNS.2011.2171993.

[48] V. Malherbe et al. "Investigating the Single-Event-Transient Sensitivity of 65 Nm Clock Trees with Heavy Ion Irradiation and Monte-Carlo Simulation." In: *2016 IEEE International Reliability Physics Symposium (IRPS)*. Apr. 2016, SE-3-1-SE-3–5. DOI: 10.1109/IRPS.2016.7574639.

[49] A. Mallajosyula and P. Zarkesh-Ha. "A Robust Single Event Upset Hardened Clock Distribution Network." In: *2008 IEEE International Integrated Reliability Workshop Final Report*. Oct. 2008, pp. 121–124. DOI: 10.1109/IRWS.2008.4796101.

[50] M. Maniatakos and Y. Makris. "Workload-Driven Selective Hardening of Control State Elements in Modern Microprocessors." In: *2010 28th VLSI Test Symposium (VTS)*. Apr. 2010, pp. 159–164. DOI: 10.1109/VTS.2010.5469589.

[51] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012. ISBN: 978-0-262-01825-8.

[52] Gordon E. Moore. "Cramming More Components onto Integrated Circuits." In: *Electronics* 38.8 (Apr. 1965).

[53] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning Series. MIT Press, 2012. ISBN: 0-262-01802-0.

[54] *Neutron Flux Calculation*. http://www.seutest.com/cgi-bin/FluxCalculator.cgi.

[55] H. T. Nguyen and Y. Yagil. "A Systematic Approach to SER Estimation and Solutions." In: *2003 IEEE International Reliability Physics Symposium Proceedings, 2003. 41st Annual.* Mar. 2003, pp. 60–70. DOI: 10.1109/RELPHY.2003.1197722.

[56] H.T. Nguyen et al. "Chip-Level Soft Error Estimation Method." In: *IEEE Transactions on Device and Materials Reliability* 5.3 (Sept. 2005), pp. 365–381. ISSN: 1558-2574. DOI: 10.1109/TDMR.2005.858334.

[57] Michael Nicolaidis, ed. *Soft Errors in Modern Electronic Systems*. en. Frontiers in Electronic Testing. Springer US, 2011. ISBN: 978-1-4419-6992-7.

[58] P. S. Ostler et al. "SRAM FPGA Reliability Analysis for Harsh Radiation Environments." In: *IEEE Transactions on Nuclear Science* 56.6 (Dec. 2009), pp. 3519–3526. ISSN: 0018-9499. DOI: 10.1109/TNS.2009.2033381.

[59] F. Pedregosa et al. "Scikit-Learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[60] I. Polian, S. M. Reddy, and B. Becker. "Scalable Calculation of Logical Masking Effects for Selective Hardening against Soft Errors." In: *2008 IEEE Computer Society Annual Symposium on VLSI*. Apr. 2008, pp. 257–262. DOI: 10.1109/ISVLSI.2008.22.

[61] O. Ruano, J. A. Maestro, and P. Reviriego. "A Methodology for Automatic Insertion of Selective TMR in Digital Circuits Affected by SEUs." In: *IEEE Transactions on Nuclear Science* 56.4 (Aug. 2009), pp. 2091–2102. ISSN: 0018-9499. DOI: 10.1109/TNS.2009.2014563.

[62] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd. USA: Prentice Hall Press, 2009. ISBN: 978-0-13-604259-4.

[63] P. K. Samudrala, J. Ramos, and S. Katkoori. "Selective Triple Modular Redundancy (STMR) Based Single-Event Upset (SEU) Tolerant Synthesis for FPGAs." In: *IEEE Transactions on Nuclear Science* 51.5 (Oct. 2004), pp. 2957–2969. ISSN: 0018-9499. DOI: 10.1109/TNS.2004.834955.

[64] A. Savino et al. "RIIF-2: Toward the next Generation Reliability Information Interchange Format." In: *2016 IEEE 22nd International Symposium on On-Line Testing and Robust System Design (IOLTS)*. July 2016, pp. 173–178. DOI: 10.1109/IOLTS.2016.7604693.

[65] N. Seifert and N. Tam. "Timing Vulnerability Factors of Sequentials." In: *IEEE Transactions on Device and Materials Reliability* 4.3 (Sept. 2004), pp. 516–522. ISSN: 1530-4388. DOI: 10.1109/TDMR.2004.831993.

[66] N. Seifert et al. "Radiation-Induced Clock Jitter and Race." In: *2005 IEEE International Reliability Physics Symposium, 2005. Proceedings. 43rd Annual.* Apr. 2005, pp. 215–222. DOI: 10.1109/RELPHY.2005.1493087.

[67] N. Seifert et al. "Radiation-Induced Soft Error Rates of Advanced CMOS Bulk Devices." In: *2006 IEEE International Reliability Physics Symposium Proceedings.* Mar. 2006, pp. 217–225. DOI: 10.1109/RELPHY.2006.251220.

[68] F.W. Sexton. "Destructive Single-Event Effects in Semiconductor Devices and ICs." In: *IEEE Transactions on Nuclear Science* 50.3 (June 2003), pp. 603–621. ISSN: 1558-1578. DOI: 10.1109/TNS.2003.813137.

[69] M. A. Shea and D. F. Smart. *Tables of Asymptotic Directions and Vertical Cutoff Rigidities for a Five Degree by Fifteen Degree World Grid as Calculated Using the International Geomagnetic Reference Field for Epoch 1965.0.* en. Tech. rep. AD-A-012509. Air Force Cambridge Research Labs., L.G. Hanscom Field, Mass. (USA), July 1975. Chap. Technical Reports.

[70] P. Shivakumar et al. "Modeling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic." In: *Proceedings International Conference on Dependable Systems and Networks*. 2002, pp. 389–398. DOI: 10.1109/DSN.2002.1028924.

[71] Allan L. Silburt et al. "Design for Soft Error Resiliency in Internet Core Routers." In: *IEEE Transactions on Nuclear Science* 56.6 (Dec. 2009), pp. 3551–3555. ISSN: 1558-1578. DOI: 10.1109/TNS.2009.2033915.

[72] Santiago Sondon et al. "Diagnose of Radiation Induced Single Event Effects in a PLL Using a Heavy Ion Microbeam." In: *2013 14th Latin American Test Workshop - LATW*. Apr. 2013, pp. 1–5. DOI: 10.1109/LATW.2013.6562682.

[73] L. Sterpone et al. "An Analytical Model of the Propagation Induced Pulse Broadening (PIPB) Effects on Single Event Transient in Flash-Based FPGAs." In: *IEEE Transactions on Nuclear Science* 58.5 (Oct. 2011), pp. 2333–2340. ISSN: 1558-1578. DOI: 10.1109/TNS.2011.2161886.

[74] J. E. Stine et al. "FreePDK: An Open-Source Variation-Aware Design Kit." In: *2007 IEEE International Conference on Microelectronic Systems Education (MSE'07)*. June 2007, pp. 173–174. DOI: 10.1109/MSE.2007.44.

[75] Katsuhiko Tanaka et al. "Study on Influence of Device Structure Dimensions and Profiles on Charge Collection Current Causing SET Pulse Leading to Soft Errors in Logic Circuits." In: *2009 International Conference on Simulation of Semiconductor Processes and Devices*. Sept. 2009, pp. 1–4. DOI: 10.1109/SISPAD.2009.5290214.

[76] G. V. Trunk. "A Problem of Dimensionality: A Simple Example." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.3 (July 1979), pp. 306–307. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1979.4766926.

[77] M. G. Valderas et al. "Extensive SEU Impact Analysis of a PIC Microprocessor for Selective Hardening." In: *IEEE Transactions on Nuclear Science* 57.4 (Aug. 2010), pp. 1986–1991. DOI: 10.1109/TNS.2009.2039581.

[78] Miguel Vilchis et al. "A Real-Case Application of a Synergetic Design-Flow-Oriented SER Analysis." In: *2012 IEEE 18th International On-Line Testing Symposium (IOLTS)*. June 2012, pp. 43–48. DOI: 10.1109/IOLTS.2012.6313839.

[79] I. Wali et al. "A Low-Cost Reliability vs. Cost Trade-off Methodology to Selectively Harden Logic Circuits." en. In: *Journal of Electronic Testing* 33.1 (Feb. 2017), pp. 25–36. ISSN: 1573-0727. DOI: 10.1007/s10836-017-5640-6.

[80] L. Wissel et al. "Flip-Flop Upsets from Single-Event-Transients in 65 Nm Clock Circuits." In: *IEEE Transactions on Nuclear Science* 56.6 (Dec. 2009), pp. 3145–3151. ISSN: 0018-9499. DOI: 10.1109/TNS.2009.2033997.

[81] Y. Yu, B. Bastien, and B. W. Johnson. "A State of Research Review on Fault Injection Techniques and a Case Study." In: *Annual Reliability and Maintainability Symposium, 2005. Proceedings.* Jan. 2005, pp. 386–392. DOI: 10.1109/RAMS.2005.1408393.

[82] J. F. Ziegler and W. A. Lanford. "The Effect of Sea Level Cosmic Rays on Electronic Devices." In: *Journal of Applied Physics* 52.6 (June 1981), pp. 4305–4312. ISSN: 0021-8979. DOI: 10.1063/1.329243.

This Ph.D. thesis has been typeset by means of the TEX-system facilities. The typesetting engine was LuaLATEX. The document class was `toptesi`, by Claudio Beccari, with option `tipotesi=scudo`. This class is available in every up-to-date and complete TEX-system installation.