FPGA Fault Tolerance in Radiation Environments

Dissertation
for attaining the doctoral degree
of Natural Sciences

submitted to the Faculty of Computer Science and Mathematics
of the Johann Wolfgang Goethe University
in Frankfurt am Main, Germany

by
Jáno Gebelein
born in Greiz, Germany

Frankfurt am Main 2016
(D 30)

accepted by the Faculty of Computer Science and Mathematics of the

Johann Wolfgang Goethe University as a dissertation.

Dean:	Prof. Dr. Uwe Brinkschulte
Expert assessors:	Prof. Dr. Udo Kebschull
	Prof. Dr. Lars Hedrich
Date of disputation:	November 22, 2016

Abstract (German)

Der Einsatz flexibel programmierbarer Field Programmable Gate Arrays (FPGA) hat in den vergangenen Jahrzehnten zunehmend an Attraktivität gewonnen, da sich Leistung und Speicherdichte erhöht, die Bereitstellungskosten jedoch verringert haben. Ihre Verwendung in Anwendungsbereichen, welche durch das Auftreten ionisierender Strahlung charakterisiert sind, stellt jedoch in vielerlei Hinsicht immer noch eine Herausforderung dar. Hierbei stehen nicht nur hochspezialisierte Themengebiete wie Militär, Raumfahrt oder Kernforschung im Vordergrund, auch alltägliche Anwendungen aus Medizin und Avionik müssen spezielle Schutzvorkehrungen treffen, um die Auswirkungen von Strahlung zu reduzieren. Dabei kann die Ursache von ionisierender Strahlung sowohl künstlich geschaffen als auch natürlichen Ursprungs sein: Kosmische Strahlung mit solarer und galaktischer Zusammensetzung, terrestrische Untergrundstrahlung von natürlichen Radionukliden sowie nukleare Zerfallsprozesse aus Zeiten der allerersten Kernversuche und Katastrophen tragen ihren Anteil zur irdisch akkumulierten Strahlung bei. Spezielle geographische Areale, an denen die Erdmagnetfeldlinien zusammenlaufen bzw. die Strahlungsgürtel der Erdoberfläche sehr nahe kommen, werden von aeronautischen Fahrzeugen mit hoher Altitude gar komplett gemieden, da sich das dort lokal erhöhte Vorkommen an ionisierender Strahlung negativ auf die Zuverlässigkeit elektronischer Bauteile auswirkt. Dies betrifft vor allem den Bereich in und um die Südatlantische Anomalie (SAA). Während die Strahlung des ersten oberirdischen Kernwaffentests noch mehrere Satelliten irreparabel beschädigte, kann das heutige Wissen um diese Effekte sowie die Beobachtung und Prognostik der solaren Aktivität das Risiko für Luftfahrt- und Raumfahrttechnik jedoch um ein Vielfaches reduzieren. Chromosphärische Sonneneruptionen mit koronalem Masseauswurf, die, sofern sie die Erde erreichen, das lokale Magnetfeld überproportional stauchen, verursachen jedoch immer noch lokal auftretende Stromausfälle sowie Fehlfunktionen technischer Komponenten direkt exponierter Satelliten- und Kommunikationstechnik. Diese Fehler zu minimieren bzw. sie zu erkennen und automatisch zu beheben, gilt als internationaler Forschungsschwerpunkt und ist, bezogen auf Halbleiter, ein wesentlicher Teil dieser Arbeit.

Strahlungseffekte in FPGAs stellen dabei eine besondere Herausforderung dar: Die bei einigen dieser flexibel konfigurierbaren Chips verwendete Static Random-Access Memory (SRAM) Speicher-Technologie erreicht zwar Bestwerte bei Lese- und Schreibzugriffen unter 1 ns, weist jedoch konstruktionsbedingt einen signifikan-

ten Nachteil gegenüber ähnlich ausgerichteten Technologien auf: Sie benötigt erheblich mehr Fläche auf dem Silicium-Träger und kann unter dem Einfluss ionisierender Strahlung ihren Zustand verlieren. Trifft ein energiereiches Teilchen auf solch einen Halbleiter, so ionisiert es vorübergehend das kristalline Silicium entlang seiner Flugbahn und schafft auf diese Weise zusätzliche Ladungsträger, sogenannte Elektron-Loch-Paare. Diese streben zunächst wieder nach Rekombination, werden jedoch von elektrischen Feldern effektiv daran gehindert. Derartige Felder werden zum Beispiel von leitenden Transistoren auf diesem Silicium-Träger selbst generiert. Einige der freien Ladungsträger interagieren daraufhin mit dem Transistor und ändern seine elektrischen Eigenschaften zeitweise oder dauerhaft. Dieser Effekt wird in der Literatur als Single Event Effekt (SEE) bezeichnet. Im ungünstigsten Fall führt die Ladungstrennung zu einer Brücke zwischen Silicium und Spannungsversorgung, die sich durch eine Überspannung einbrennen kann und den Transistor und damit auch die zugehörige SRAM-Zelle dauerhaft funktionsunfähig macht. Dieses Problem lässt sich nur durch zusätzliche Isolation lösen, tritt in konventionellen SRAM-FPGAs aber nur bei sehr energiereichen Schwerionen auf.

Der weitaus häufigere Fall beschreibt jedoch die impulsartige Ausbreitung der generierten Ladung innerhalb der auf dem Chip gefertigten Schaltkreise, ausgehend von dem beeinträchtigten Transistor. Dieser spontan auftretende, temporäre SEE wird auch als Single Event Transient (SET) bezeichnet. Er beeinflusst das Zusammenspiel aller sechs Transistoren innerhalb konventioneller SRAM-Speicherzellen und kann in ungünstigen Konstellationen deren Zustand ändern. Tritt ein SET synchron zur Setup-/Haltephase eines getakteten Flipflops auf, so können weiterhin falsche Werte in Register übernommen werden. In diesen beiden genannten Fällen wandelt sich der temporäre SET in einen persistenten Single Event Upset (SEU). Die daraus resultierenden Bitfehler in den logischen Einheiten eines FPGAs werden auch Single Bit Upsets (SBU) genannt. Ergeben sich mehrere dieser Bitfehler aus einem einzigen SET, so liegt ein Multiple Bit Upset (MBU) vor. Ihr Auftreten nimmt mit den kontinuierlich verkleinerten Halbleiter-Fertigungsprozessen und dem damit verbundenen Herabsetzen von Kapazität und Spannung unweigerlich zu. Aber auch viele weitere Variablen haben Einfluss auf die Häufigkeit von Bitfehlern, so zum Beispiel der Dotierungsgrad des Siliciums oder Kontamination des Chip Packages mit Radionukliden. Die Hersteller von SRAM-FPGAs, wie zum Beispiel Xilinx, versuchen derartige Effekte jedoch unter Zuhilfenahme verschiedenster Design- und Fertigungstechniken zu relativieren.

Generell unterscheiden sich alle Halbleiter-Fertigungschargen hinsichtlich ihrer Strahlenfestigkeit aus den genannten Gründen voneinander, dies gilt selbst für Chips gleicher Serien. Die Qualifizierung einzelner Bauteile für den Einsatz in Anwendungsbereichen mit ionisierender Strahlung erfordert daher stetige Kontrolle, was ihre Anschaffungskosten signifikant erhöht und damit dem Einsatz größerer Stückzahlen entgegensteht. Für die Verwendung ungetesteter Komponenten ist jedoch eine erhebliche Toleranzschwelle einzuplanen. Ziel sollte es daher sein, einen Kompromiss aus Kosten und Nutzen zu finden. Dies kann erreicht werden, indem größere Bauteil-Mengen einer einzigen Fertigungscharge bevorratet werden, eine kleine Auswahl daraus qualifiziert wird und anschließend statistisch auf die Eigenschaften aller übrigen Bauteile geschlossen wird.

Um die Einsatzfähigkeit sowie Zuverlässigkeit ihrer Bauteile zu erhöhen, haben die Chip-Hersteller ihre Bemühungen zur Entwicklung spezieller strahlenharter FPGAs intensiviert. Da sich deren Leistungscharakteristik jedoch meist weit hinter denen aktueller Topmodelle wiederfindet und die Entwicklungskosten massiv in die Preisgestaltung einfließen, sind die Fertigungs- sowie Absatzzahlen entsprechend niedrig. Parallel dazu werden andere Technologien erforscht, die konventionellen SRAM-Zellen überlegen sind und weniger anfällig auf ionisierende Strahlung reagieren sollen. Vielversprechende Beispiele sind FRAM, MRAM, CRAM und RRAM. Um eigene Aussagen zu deren Strahlenhärte treffen zu können, wurden im Rahmen dieser Arbeit Strahltests mit den ersten kommerziell verfügbaren FRAM-Speichern durchgeführt. Einen anderen, vielversprechenden Ansatz verfolgt der Chip-Hersteller Xilinx mit Einführung seiner zwölf-Transistor SRAM-Zellen. Diese Entwicklung macht es unmöglich, dass ein einzelner, gestörter Transistor den Zustand einer kompletten SRAM-Zelle ändern kann. Aufgrund des hohen Anschaffungspreises von 100.000 USD stand jedoch keiner dieser FPGAs für eigene Tests zur Verfügung. Sobald diese Technologie ihren Weg in konventionelle Chips gefunden hat, vereinfacht sich der Einsatz von FPGAs in strahlungsintensiven Umgebunden erheblich.

Treten trotz der durch die Chip-Hersteller vorgenommenen Modifikationen auf der Hardware-Ebene Bitfehler auf, so müssen diese auf den darüberliegenden Konfigurations-, Firmware-, System- und Software-Ebenen erkannt und möglichst effektiv repariert werden. Diese Techniken bilden zusammengenommen das Konzept der systemweiten Fehlertoleranz, welches Inhalt dieser Arbeit ist.

Da sich alle Design-Ebenen bei FPGAs, mit Ausnahme der Hardware, vom Benutzer modifizieren lassen, stehen verschiedenste Techniken zur Verfügung, um das ge-

setzte Ziel zu erreichen. Die optimale Zusammensetzung der einzelnen Fehlerkorrekturverfahren kann dabei den systemweiten Mehraufwand eindämmen und den Zusatzkostenfaktor limitieren. Zusätzlich dazu muss gewährleistet werden, dass die „Time to Market“ nicht signifikant ansteigt, denn auch Verzögerungen in der Produktbereitstellung sind unweigerlich mit einer Kostenzunahme verknüpft.

Aufbauend auf die Hardware-Ebene definiert die Konfigurations-Ebene nahezu alle grundlegenden Funktionen des FPGAs, demnach seine logische Schaltung. Sie beinhaltet die gesamte Konfiguration aller Basiselemente, die sich im Chip befinden, einschließlich Flipflops, Look-up Tables (LUT), Block-RAM (BRAM) und Routing. Das Routing nimmt dabei einen wesentlichen Anteil ein, denn es regelt die variable Vernetzung aller Einzelkomponenten zu einem funktionellen Gesamtsystem. Eine Spezialfähigkeit von FPGAs des Herstellers Xilinx ist dabei die Fähigkeit, diese Konfiguration dynamisch zur Laufzeit auslesen sowie ändern zu können, ohne dabei die Funktion des übrigen Chips zu beeinflussen. Dieser Effekt lässt sich nutzen, um Zellen mit identifizierten SBUs im Hintergrund von einem externen Speichermedium nachzuladen und damit die entstandenen Fehler zu korrigieren. Dies erfolgt entweder durch einen extern angebundenen Controller oder, in neueren FPGA-Generationen, durch einen intern arbeitenden Mechanismus. Die Controller-Variante wurde konzeptionell in die Hardware der aktuellen SysCore-Entwicklungsplattform mit Version 3.1 integriert und steht jedem Nutzer zur Verfügung. Für dieses als „Konfigurations-Scrubbing“ bezeichnete Refresh eignen sich jedoch nur die statischen Teile der Konfiguration wie Routing, statische LUTs oder Block-ROM (BROM). Das Zurücksetzen eines zur Laufzeit bereits angepassten Speicherinhalts würde das gesamte System kompromittieren. Um dies zu verhindern, muss die Konfigurationsdatei daher zunächst von den dynamischen Inhalten befreit und in ein partielles Format überführt werden. Hierfür wurde ein Tool entwickelt, das auf die speziellen Anforderungen des FPGAs zugeschnitten ist. Die korrekte Funktion der für das Konfigurations-Scrubbing verwendeten Hardware-Komponenten wurde in Strahltests mehrfach erfolgreich verifiziert und stand fortan für die weitere Nutzung im Rahmen des fehlertoleranten Gesamtsystems zur Verfügung. Mit diesem Verfahren kann demnach der gesamte statische Teil der FPGA-Konfiguration zyklisch korrigiert werden. Dies allein genügt jedoch noch nicht, um Fehler vollständig zu unterbinden: Im Zeitraum zwischen zwei Scrubbing-Zyklen können SBU in der statischen Konfiguration weiterhin zu Fehlberechnungen im FPGA führen. Dieser Aspekt lässt sich auf der nächsthöheren Firmware-Ebene parallel zur Korrektur von SBUs in dynamischen Speicherzellen ausgleichen.

Fehlertolerantes FPGA-Firmware-Design dient primär dazu, den Datenfluss innerhalb des FPGAs abzusichern und damit ein deterministisches Verhalten des Chips zu gewährleisten. Dies beinhaltet die Absicherung von dynamischen Flipflops/Registern, verteiltem Speicher in LUTs sowie eingebettetem BRAM. Es ist prinzipiell mit Overhead verbunden und führt damit zu einem erhöhten Ressourcenbedarf sowie vermehrt Leistungseinbußen der betroffenen Systeme. Zum Erreichen dieses Ziels stehen verschiedene Redundanz-Techniken zur Verfügung, die dazu dienen, Signalpfade vielfach parallel zu implementieren, Verarbeitungsketten mehrfach zu durchlaufen oder durch den Einsatz von Paritätsbits die Datenintegrität selbst zu erhöhen. Diese Verfahren dienen zusätzlich dazu, einzelne Fehler, die durch Beeinträchtigungen in den darunterliegenden Strukturen der Konfigurations-Ebene entstanden sind und noch nicht durch einen Scrubbing-Zyklus korrigiert wurden, kurzzeitig zu umgehen. Eine Kumulierung von Fehlern bei Verwendung ohne Konfigurations-Scrubbing kann dabei jedoch nicht überbrückt werden und ist daher unbedingt zu vermeiden.

Die am häufigsten verwendeten Redundanztechniken stellen Dual Modular Redundancy (DMR) und Triple Modular Redundancy (TMR) dar. Bei DMR werden funktionelle Teile einer Schaltung zweifach implementiert und die berechneten Daten an zuvor definierten Signalstellen miteinander verglichen. Tritt dabei eine Diskrepanz auf, so kann die Schaltung durch Neuberechnung oder Fehlerkennzeichnung der Daten bzw. durch einen System-Reset definiert reagieren. Eine sofortige Fehlerkorrektur ist dabei jedoch nicht möglich. Dies erfordert mindestens eine dreifache Logik-Implementierung mittels TMR. Entlang der parallel betriebenen Signalwege müssen dann in möglichst kurzen Abständen sogenannte Voter eingefügt werden. Diese führen auf Basis aller synchron eintreffenden Daten einen bitweisen Mehrheitsentscheid aus, bevor sie einen Datenwert an die nachfolgende Verarbeitungsinstanz weiterreichen. Je geringer der Abstand zwischen diesen Votern gewählt wird, desto mehr zusätzliche Ressourcen sind zu deren Integration im FPGA notwendig, aber umso effektiver ist auch die Fehlerkorrekturleistung. Da der getroffene Mehrheitsentscheid jedoch keinen Einfluss auf die ihm vorangestellten Register-Eingänge hat und dort lokal auftretende Datenfehler nicht korrigiert werden können, sollte jeder Voter von einem zusätzlichen Feedback-Signal begleitet werden, welches ein Nachladen der korrekten Eingangsdaten erzwingt. Dieser vollständige Prozess wird vom Chip-Hersteller Xilinx auch als XTMR bezeichnet. Er kann im Vergleich mit dem Originaldesign zu einem sechsfach erhöhten Ressourcenbedarf führen. Zusätzlich verschlechtert sich durch die Integration der synchronisierten Voter

das globale Zeitverhalten der Schaltung und anfänglich definierte Verarbeitungsgeschwindigkeiten können nicht mehr annähernd erreicht werden, was einem Totalausfall gleichkommt. In dieser Situation bleibt nur die Überlegung, ob es genügt, ausschließlich kritische Teilbereiche des Gesamtsystems mit TMR/XTMR abzusichern oder ob ein DMR-Ansatz mit Fehlererkennung, jedoch ohne Fehlerkorrektur, nicht auch zweckmäßig ist. Hierbei sei jedoch erwähnt, dass der zusätzliche Ressourcenbedarf von DMR ohne eine Möglichkeit der aktiven Fehlerkorrektur die gesamte Fehleranfälligkeit der Schaltung, die sogenannte Cross-Section, erhöht und damit unter Umständen sogar einen gegenteiligen Effekt auslöst. Die entwickelte Schaltung ist daher durch Fehlerinjektion im Simulator, mit Lasern in Labor oder unter realen Strahlungsbedingungen mit dosimetrisch überwachten, ionisierenden Partikeln zu testen. Zahlreiche Anleitungen, Hinweise und eigens dafür entwickelte Tools finden sich in der vorliegenden Arbeit. Sie wurden aus Erfahrungswerten erstellt und geben einen Einblick in die Planung, Durchführung und Auswertung derartiger Versuche.

Ein ähnliches Verfahren der Fehlerkorrektur verfolgt das Konzept der temporalen Redundanz. Dabei werden die Berechnungen jedoch mehrfach auf demselben Schaltkreis zu unterschiedlichen Zeitpunkten durchgeführt, was zu erheblich niedrigerem Ressourcenbedarf führt. Alle dabei erhaltenen Ergebnisse unterliegen anschließend jedoch auch einem Mehrheitsentscheid, bevor sie an die nächste Verarbeitungsinstanz übergeben werden können.

Die Auswahl einer geeigneten Fehlertoleranztechnik auf Firmware-Ebene bleibt demnach letztendlich dem Entwickler überlassen. Nur er kennt die kritischen Pfade seines Designs und kann über die Notwendigkeit einer Absicherung entscheiden. Um Fehlertoleranztechniken in eine bereits vorhandene Logikschaltung zu integrieren, stehen dem Entwickler verschiedene universitäre sowie kommerzielle Tools von Xilinx, Mentor Graphics® und anderen Anbietern zur Verfügung. Diese wurden im Rahmen dieser Arbeit untersucht: Sie operieren alle auf Netzlisten-Ebene, was ihre Transparenz, Simulation und Dokumentation beim Anwender erheblich verkompliziert. Zudem wurde die Weiterentwicklung vieler dieser Tools bereits vor Jahren eingestellt.

Die Implementierung von Fehlertoleranz kann jedoch auch manuell in jeder konventionellen Hardwarebeschreibungssprache (HDL) erfolgen. Hierbei muss jedoch sichergestellt werden, dass bei der Hardwaresynthese keine Optimierungsoptionen gewählt werden, die das Design minimieren und damit alle redundant eingefügten Komponenten wieder entfernen. Auch die Mehrfachverwendung von logisch

äquivalenten Schaltungsteilen muss unterbunden werden, da sie Berechnungsfehler künstlich vervielfältigt. Dieses Verhalten kann, abhängig vom verwendeten Synthese-Tool, üblicherweise durch die Definition von speziellen Attributen erreicht werden.

Das manuelle Fehlertoleranzdesign schafft beim Entwickler maximale Freiheitsgrade, welche ein automatisiertes Tool auf Netzlisten-Ebene niemals erreicht. Durch das selektiv abgesicherte Design ergibt sich eine kleinere Konfigurationsdatei, damit eine geringere Cross-Section und es können günstigere FPGAs beschafft werden. Der große Nachteil liegt jedoch in der Komplexität sowie in der zusätzlich erforderlichen Entwicklungszeit, was sich nur in größeren Stückzahlen positiv auf den Kosten-/Nutzen-Faktor auswirkt. Um dieses manuelle Design einfacher zu gestalten, wurde hierfür ein umfangreiches Regelwerk erstellt und der vorliegenden Arbeit beigefügt. Unter Berücksichtigung dieser Regeln wurde weiterhin damit begonnen, eine halbautomatisierte Java-Anwendung zu entwickeln, welche den Firmware-Designer dabei unterstützt, Fehlertoleranz auf HDL-Ebene in seine Schaltung zu integrieren. Dabei wird ein gegebenes Design-Projekt eingelesen, in einem Baum abgebildet und kann fortan entsprechend den Nutzervorgaben modifiziert werden. Die Anwendung verfügt zusätzlich über einen Generator für Hamming-codierte Zustandsautomaten (FSM), der nutzungsabhängig einen kompletten Satz an Zuständen und Zustandsübergängen, wie sie nur durch einen SBU erreicht werden, erstellt. Damit ist es zu keiner Zeit möglich, die FSM zu kompromittieren. Die Effektivität dieser Methode wurde in einem Strahltest validiert und die Ergebnisse finden sich in der vorliegenden Arbeit.

Für die Absicherung von Daten aus größeren Speicherbereichen, wie zum Beispiel eingebettetem BRAM, stehen einigen FPGAs neben den Datenbits noch zusätzliche Paritätsbits zur Verfügung. Diese können dazu verwendet werden, um Informationsredundanz für den kompletten Speicherinhalt zu realisieren. Da jedoch auch diese Komponente den eingangs genannten Effekten ionisierender Strahlung unterliegt, treten in den dynamischen Daten der Speicherzellen mit zunehmendem Zeitrahmen unweigerlich Bitfehler auf. Durch die Belegung der Paritätsbits mit Hamming-Code, können diese SBU beim Auslesen von Daten effektiv korrigiert werden. In einigen FPGAs mit Hardware-Unterstützung wird dies vom BRAM selbst angeboten. Bei allen anderen FPGAs muss die Funktionalität jedoch fehlertolerant in die Schaltung des FPGAs integriert werden. Wird beim Auslesen eines Speicherwortes ein Bitfehler erkannt und korrigiert, so muss dieser Datenwert manuell wieder zurück an die korrekte Adresse geschrieben werden. Wird dies nicht durchgeführt,

so können Fehler akkumulieren und Daten irreversibel verloren gehen. Dasselbe gilt für Speicherbereiche, die lange Zeit ungenutzt bleiben. Um dies zu verhindern, muss der vollständige Speicherinhalt kontinuierlich im Hintergrund ausgelesen und jeder enthaltene Fehler separat korrigiert werden. Diese Funktion übernimmt ein im Rahmen dieser Arbeit entwickelter TMR Memory-Scrubber, der auf die Charakteristika eines Xilinx FPGAs zugeschnitten ist. Parallel dazu wurde dieses Prinzip auch auf einen kommerziell erhältlichen Mikrocontroller von Texas Instruments übertragen. Dort führen die gleichen Strahlungsursachen zur Kumulierung von Speicherfehlern. Die korrekte Funktion dieses Algorithmus wurde während eines Protonenstrahltests validiert. Ergebnisse und Auswertung finden sich in dieser Dissertationsschrift.

Ausgewählte Logik-Bereiche mit definierten Eigenschaften lassen sich beim Design von FPGA-Firmware zu größeren Makroblöcken, wie zum Beispiel einem Signalprozessor, zusammenfassen und dadurch einfacher handhaben. Diese Makroblöcke können nahezu beliebige Funktionen realisieren und demnach auch Fehlertoleranz enthalten. Die System-Ebene des FPGA-Designs, auf der solche Makroblöcke einfach miteinander verbunden werden, um daraus ein funktionelles Gesamtsystem zu erzeugen, ist daher hervorragend dafür geeignet, die Entwicklungszeit fehlertoleranter Systeme signifikant zu verkürzen. Der größte Vorteil ist sicherlich darin begründet, dass sich der FPGA-Entwickler nicht mit der korrekten Definition komplexer Signalredundanz befassen muss. Zudem sinkt bei der Verwendung von Standardblöcken die Wahrscheinlichkeit für Entwicklungsfehler massiv ab. Um auch die System-Ebene im Rahmen dieser Arbeit abzudecken, wurden verschiedene Standard-Blöcke erstellt. Sie beinhalten zum Beispiel eine MIPS-CPU sowie verschiedene Interfaces wie Ethernet oder RS232 und wurden weitestgehend in Strahltests validiert. Die Ergebnisse finden sich in der vorliegenden Arbeit.

Gerade FPGA-Designs, die von der komfortablen Einbindung eines Mikroprozessors profitieren, sind extrem darauf angewiesen, dass alle Instruktionen unverändert im Speicher vorliegen, deren serielle Abarbeitung deterministisch verläuft und die verwendeten Ein- und Ausgangsdaten keinen spontanen Modifikationen unterliegen. Wurde die CPU nicht vollständig verifiziert so ist es möglich, dass bestimmte durch SBU entstandene Kombinationen aus Instruktionen und Daten im ungünstigsten Fall einen undefinierten Zustand provozieren, der ohne vollständigen Reset nicht mehr verlassen werden kann. Sollen Fehler in den Recheneinheiten, der Ablaufsteuerung sowie den Nutzerdaten der CPU zusätzlich abgesichert werden, vor allem wenn die Hardware-Ebene keine dedizierten Paritätsbits im Datenspeicher

anbietet, so kann dies auf der nächsthöheren Software-Ebene mittels der Software Implemented Hardware Fault Tolerance (SIHFT) erfolgen. Eine zeitlich verschobene Wiederholung von Instruktionen vergleichbar mit dem Konzept temporaler Redundanz auf Hardware-Ebene, die mehrfache Datenhaltung ähnlich DMR/TMR sowie eine vollständig replizierte, parallele Programmausführung in mehreren Prozessorkernen oder virtuellen Maschinen stehen dabei als Fehlertoleranztechniken zur Auswahl. Analog den erläuterten Anforderungen an das Synthesetool auf Firmware-Ebene müssen jedoch auch hier spezielle Vorkehrungen getroffen werden, die eine Optimierung durch den Software-Compiler verhindern. Da eine manuelle Implementierung unverhältnismäßig ist, stehen verschiedene Tools zur Verfügung, die diese Aufgabe automatisiert übernehmen. Eine Auswahl dieser Programme wurde im Grundlagenkapitel der vorliegenden Arbeit analysiert.

Die vollständige Verarbeitungskette, angefangen bei der grundlegenden Hardware-Ebene, über die Konfigurations-, Firmware- und System-Ebenen bis hin zur Software-Fehlertoleranz, bildet das Konzept der systemweiten Fehlertoleranz. Ein erfolgreicher Einsatz von FPGAs in Anwendungsfeldern mit ionisierender Strahlung ist daher von dem korrekten Zusammenspiel vieler einzelner Fehlerkorrekturmechanismen auf unterschiedlichen Verarbeitungsebenen abhängig. Keine einzelne Ebene wäre in der Lage, unabhängig von allen anderen vergleichbare Leistungen zu erbringen. Das Gesamtsystem ist dabei jedoch nur so zuverlässig wie seine schwächste Komponente. Dies beginnt bereits bei der Auswahl einer geeigneten Hardware-Plattform, welche den über die gesamte Laufzeit hinweg erwarteten, kumulativen Ionisationseffekten standhält und dabei kontinuierlich eine stabile Spannungsversorgung gewährleistet. In diesem Zusammenhang wurden im Rahmen dieser Arbeit auch lineare und getaktete DC/DC-Wandler untersucht und für Stabilitätsprognosen über längere Zeit bestrahlt. Diese Ergebnisse flossen bereits in die Entwicklung der jüngsten SysCore-Entwicklungsplattform mit der Version 3.1 ein.

Zusammenfassend konnte gezeigt werden, dass neben der Verwendung von XTMR auch andere Herangehensweisen für ein fehlertolerantes Firmware-Design zur Verfügung stehen, die einen geringeren Bedarf an Logikressourcen im FPGA aufweisen. Dies basiert zum größten Teil auf dem Ansatz der selektiv implementierten Redundanz in Verbindung mit der Kombination von Fehlertoleranzmechanismen anderer Designebenen. Die konstruktionsbedingt nachteiligen Eigenschaften von SRAM-FPGAs beim Einsatz in verschieden stark strahlenden Umgebungen, wie beispielsweise an der GSI/FAIR in Darmstadt/Deutschland, konnten damit weitestge-

hend relativiert werden. Die größten Hürden ergeben sich hierbei im Umgang mit Fehlertoleranz in HDLs – dieser Prozess soll mit weiterführender Anwendungsunterstützung in Zukunft vereinfacht werden. Entwicklungsfortschritte in der Halbleiterfertigung, dem SRAM-Zelldesign sowie der Technologiesubstitution, wie sie in dieser Arbeit dargelegt wurden, können jedoch dazu führen, dass Fehlertoleranz in zukünftigen Anwendungen keiner Notwendigkeit mehr unterliegt.

Der überwiegende Teil der Ergebnisse innerhalb dieser Dissertationsschrift wurde im Rahmen von Konferenzen und Printmedien dem Fachpublikum zugänglich gemacht. Eine vollständige Übersicht dieser Veröffentlichungen kann der Publikationsliste entnommen werden.

Acknowledgements

Many people deserve special thanks for their support and contribution to this work. First of all, my sincere and deep gratitude goes to Prof. Dr. Udo Kebschull for his everlasting patience and enthusiasm towards the thesis topic. His continuous inspiration, encouragement and the precious time he invested in training, support and supervision enabled a continuous progress and added a bunch of ideas and thoughts. I would also like to thank my colleagues Norbert Abel, Heiko Engel, Sebastian Manz, Cruz Garcia and Frederik Grüll for their support and fruitful discussions during the research phase. Finally, but not conclusive, I would like to thank my wife, my family and my friends for their continuous help and unconditional support during all the years.

I am grateful to the German Federal Ministry of Education and Research (BMBF) for providing funding, especially for the practical proof of concept within the scope of this thesis. Being a part of the GSI/FAIR CBM collaboration has always been a great experience that I would never like to miss. The members' readiness to help with knowledge and wisdom will always remain a major aspect to remember. Meeting scientists and experts on various national and international conferences would not have been possible without this support - it opened many new perspectives and opportunities also for my personal progression.

Finally, I really appreciate the help of all the other unnamed people which cannot be listed here one by one, but which have contributed in one or the other topic - you may find yourself while reading through the sections and references.

"It would be illogical to assume that all conditions remain stable."

SPOCK

Contents

Abstract (German) . . . 3
Acknowledgements . . . 13
Contents . . . 17
List of Abbreviations . . . 23
List of Figures . . . 29
List of Tables . . . 31
List of Work Publications . . . 33
1 Introduction and Motivation . . . 37
1.1 Terrestrial Applications and Outer Space . . . 37
1.2 The Facility for Antiproton and Ion Research . . . 39
1.3 The Compressed Baryonic Matter Experiment . . . 41
1.3.1 Major Objectives and Physics Background . . . 41
1.3.2 Detector Concept and Experimental Setup . . . 42
1.3.3 Data Acquisition and First Level Event Selector . . . 45
1.4 Key Goals of this Study . . . 46
2 Background Analysis . . . 49
2.1 Architectures and Technologies . . . 49
2.1.1 CMOS Transistor Technology goes FinFET . . . 49
2.1.2 SOI - Silicon-on-Insulator . . . 52
2.1.3 Xilinx Triple-Oxide Technology . . . 53
2.1.4 SRAM - Static Random Access Memory . . . 56
2.1.5 FRAM - Ferroelectric Random Access Memory . . . 58
2.1.6 MRAM - Magnetoresistive Random Access Memory . . . 61
2.2 EDA - Electronic Design Automation . . . 63
2.2.1 Hardware Description Languages . . . 64
2.2.2 Synthesis and Simulation . . . 66
2.2.3 Logic Device Time to Market . . . 68
2.2.4 Programmable Hardware Evolution . . . 69
2.3 Field Programmable Gate Arrays . . . 71
2.3.1 Configuration, Routing and Logic Blocks . . . 72
2.3.2 Boolean Function Generators . . . 75
2.3.3 Flip-Flop Storage Elements . . . 75
2.3.4 On-Chip Block and Distributed Memory . . . 76

2.3.5 Input/Output Buffers . . . 77
2.3.6 Embedded Hard Blocks . . . 77
2.3.7 Half-Latches . . . 78
2.4 Ionizing Radiation . . . 79
2.4.1 Passage of Radiation through Matter . . . 80
2.4.2 Electron-Hole Pair Generation . . . 82
2.4.3 LET - Linear Energy Transfer . . . 83
2.4.4 Radiation from Solar Flares and Galactic Cosmic Rays . . . 84
2.4.5 Radiation in Low Earth Orbit . . . 85
2.4.6 Terrestrial Radiation on Earth . . . 86
2.4.7 Radiation in Particle Accelerators . . . 88
2.5 Radiation Effects in Semiconductors . . . 88
2.5.1 MOSFET and Radiation . . . 89
2.5.2 Cross-section and Weibull-Fit . . . 90
2.5.3 Radiation Effects in SRAM Cells . . . 92
2.5.4 SEE - Single Event Effects . . . 94
2.5.4.1 SET - Single Event Transients and PIPB - Propagation-Induced Pulse Broadening . . . 96
2.5.4.2 SEU - Single Event Upset and SBU/MBU - Single/Multiple Bit Upset . . . 98
2.5.4.3 SEFI - Single Event Functional Interrupt . . . 100
2.5.4.4 SEL - Single Event Latch-up . . . 101
2.5.4.5 SEB - Single Event Burnout . . . 103
2.5.4.6 SEGR - Single Event Gate-Rupture and SHE - Single Event Hard Error . . . 103
2.5.5 Cumulative Effects . . . 104
2.5.5.1 Lattice Displacement Damage and NIEL . . . 104
2.5.5.2 TID - Total Ionizing Dose . . . 106
2.5.6 TID Impact on Xilinx COTS SRAM FPGAs . . . 109
2.5.7 SEE Impact and Mitigation on Xilinx COTS SRAM FPGAs . . . 110
2.5.7.1 Signal Routing Impact . . . 112
2.5.7.2 Look-up Table and Flip-Flop Impact . . . 113
2.5.7.3 DCM and Clock Routing Impact . . . 114
2.5.7.4 I/O Buffer Impact . . . 115
2.5.7.5 BRAM On-Chip Memory Impact . . . 115
2.5.7.6 DSP - Digital Signal Processor Impact . . . 117

2.5.7.7 Housekeeping-Primitives Impact . . . 117
2.5.8 Radiation Impact on ASIC Semiconductors . . . 118
2.5.9 Radiation Impact on Hybrid Semiconductors . . . 119
2.5.10 ITAR - International Traffic in Arms Regulations . . . 119
2.6 Hardware Radiation Hardening . . . 123
2.6.1 Shielding Effects . . . 124
2.6.2 Radiation-Tolerant ASIC CMOS Design . . . 125
2.6.3 Radiation-Tolerant FPGAs . . . 128
2.6.3.1 Radiation-Tolerant Antifuse FPGAs . . . 128
2.6.3.2 Radiation-Tolerant Flash FPGAs . . . 130
2.6.3.3 Radiation-Tolerant SRAM FPGAs . . . 131
2.6.4 Radiation-Tolerant Micro-Controllers and Micro-Processors . . . 134
2.7 Firmware Fault Tolerance . . . 136
2.7.1 System and Device Redundancy . . . 138
2.7.2 Spatial Redundancy Techniques . . . 139
2.7.3 Temporal Redundancy . . . 144
2.7.4 Information Redundancy . . . 145
2.7.5 Combined Redundancy . . . 147
2.7.6 State Machine Encoding . . . 148
2.7.7 Dynamic Partial Reconfiguration and Scrubbing . . . 151
2.7.8 On-Chip Memory Scrubbing . . . 156
2.8 Software Fault Tolerance . . . 157
2.9 Automated Fault Tolerance Tools . . . 160
2.9.1 Mentor Graphics Precision Rad-Tolerant Tool . . . 161
2.9.2 Xilinx TMRTool . . . 162
2.9.3 BYU-LANL TMR Tool . . . 163
2.9.4 Politecnico di Torino RoRA Tool . . . 164
2.10 SEE Simulation Tools . . . 164
3 Approaching Fault Tolerance for FPGAs . . . 169
3.1 Device and Board Selection . . . 169
3.2 Static Configuration Scrubbing - Benefits and Limits . . . 171
3.3 Redundant Firmware Design . . . 172
3.4 Automatic Firmware Redundancy . . . 173
3.5 Semi-Automatic Fault Tolerance with Steering . . . 174
3.6 Dynamic Data Retention in Memory Arrays . . . 176
3.7 System-wide Fault Tolerance . . . 177

4 System Implementation Details . . . 179
4.1 Static Configuration Scrubbing . . . 179
4.1.1 SysCore and Xilinx SelectMAP Interface . . . 180
4.1.2 Xilinx SelectMAP Configuration Protocol . . . 182
4.1.3 Partial Bitfiles for Configuration Scrubbing . . . 183
4.1.4 Automatic Partial Bitfile Generator . . . 185
4.1.5 Prerequisites for Design Scrubbing . . . 186
4.2 Dynamic run-time Memory Scrubbing . . . 187
4.2.1 Xilinx Virtex FPGA . . . 188
4.2.2 Xilinx Spartan FPGA . . . 188
4.2.3 Texas Instruments TMS570 Microcontroller . . . 190
4.3 Fault-Tolerant System Design . . . 192
4.3.1 Logical Design Decisions . . . 193
4.3.2 Central Processing Unit . . . 195
4.3.2.1 Processor Pipeline . . . 196
4.3.2.2 Error Detection and Handling . . . 197
4.3.2.3 Program Counter and Register Bank . . . 197
4.3.2.4 System Bus . . . 198
4.3.3 Serial Communication Interface . . . 198
4.3.4 Ethernet Controller . . . 200
4.4 JFTToolkit - The Java Fault Tolerance Toolkit . . . 202
4.4.1 VHDL, Chomsky Hierarchy and the Parser Package . . . 203
4.4.2 Basic Working Algorithm and VHDL Example . . . 205
4.4.3 UML Package and Class Diagram . . . 213
4.4.4 The Hamming FSM Generator . . . 214
4.5 Guidelines for Fault-Tolerant VHDL Design . . . 215
5 Experiments and Results . . . 221
5.1 Radiation Experiments . . . 221
5.1.1 Radiation Monitoring . . . 222
5.1.2 FPGA X-ray Analysis . . . 223
5.1.3 SysCore DUT Platform . . . 225
5.1.4 Experimental Configuration . . . 226
5.1.5 TID Tool . . . 227
5.2 Dynamic Partial Reconfiguration Experiments . . . 228
5.3 Experimental Particle Flux Determination . . . 232

5.4 Fault-Tolerant Firmware . . . 233
5.4.1 Dynamic Memory Scrubbing . . . 234
5.4.2 Hamming FSMs . . . 236
5.4.3 Fault-Tolerant CPU . . . 238
5.5 Considerations for the CBM Experiment . . . 240
5.6 COTS Component Evaluation . . . 242
5.6.1 Lattice ECP2M FPGA . . . 242
5.6.2 FRAM Irradiation . . . 244
5.6.3 Power Regulators . . . 247
6 Conclusions and Outlook . . . 249
A Neutron MTBU for Xilinx FPGAs . . . 255
B Parbitgen conversion log . . . 259
Bibliography . . . 265

List of Abbreviations

ASIC	Application Specific Integrated Circuit
AST	Abstract Syntax-Tree
BGA	Ball Grid Array
BISER	Built-In Soft Error Resilience
BJT	Bipolar Junction Transistors
BNCT	Boron Neutron Capture Therapy
BRAM	Block Random Access Memory
CAD	Computer-Aided Design
CBM	Compressed Baryonic Matter
CCL	Commerce Control List
CED	Concurrent Error Detection
CFCSS	Control Flow Checking using Software Signature
DICE	Dual Interlocked Storage Cell
CLB	Configurable Logic Block
CME	Coronal Mass Ejection
CMOS	Complementary Metal-Oxide-Semiconductor
CMT	Clock Management Tile
COSY	Cooler Synchrotron
COTS	Commercial Off-The-Shelf
CPLD	Complex Programmable Logic Device
CPU	Central Processing Unit
CRAFT	CRiticAlity based Fault Tolerance
CRAM	Chalcogenide Random Access Memory
CRC	Cyclic Redundancy Check
DAQ	Data Acquisition
DCM	Distributed Clock Manager
DDTC	Directorate of Defense Trade Controls
DICE	Dual Interlocked Storage Cell
DMR	Dual Modular Redundancy

continued on next page

DPB	Data Processing Board
DPR	Dynamic Partial Reconfiguration
DPSET	Double-Pulse Single Event Transient
DRAM	Dynamic Random Access Memory
DRP	Dynamic Reconfigure Port
DUT	Device under Test
DSP	Digital Signal Processor
DTI	Deep Trench Isolation
DWC	Duplication With Comparison
EAR	Export Administration Regulation
EBNF	Extended Backus–Naur Form
ECAL	Electromagnetic Calorimeter
ECC	Error Correcting Code
ECCN	Export Control Classification Number
ECR	Export Control Reform Initiative
EDA	Electronic Design Automation
EDAC	Error Detection And Correction
ED^{4}I	Error Detection using Diverse Data and Duplicated Instructions
EDIF	Electronic Design Interchange Format
EDS	Error Detection Sequentials
EEPROM	Electrically Erasable Programmable Read Only Memory
EPROM	Erasable Programmable Read Only Memory
ELT	Enclosed Layout Transistors
EMAC	Ethernet Media Access Control
EUV	Extreme Ultra-Violet
FAIR	Facility for Antiproton and Ion Research
FAR	Frame Address Register
FD	Fully Depleted
FD-SOI	Fully Depleted Silicon-on-Insulator
FEE	Front End Electronic
FIFO	First In, First Out

continued on next page

FinFET	Fin Metal Oxide Semiconductor Field Effect Transistor
FIT	Failure In Time
FLES	First Level Event Selector
FPGA	Field Programmable Gate Array
FRAM	Ferroelectric Random Access Memory
FSD	Functional Specification Document
FSM	Finite-State Machine
GCR	Galactic Cosmic Ray
GEM	Gas Electron Multiplier
GEO	Geostationary Orbit
GLE	Ground Level Event
GMR	Giant Magnetoresistance
GPIO	General Purpose Input/Output
GPU	Graphics Processing Unit
GSI	Gesellschaft für Schwerionenforschung
GUI	Graphical User Interface
HDL	Hardware Description Language
HETA	Hybrid Error-detection Technique through Assertions
HIT	Heavy Ion Tolerant
HLS	High-Level Synthesis
HPC	High Performance Computing
HZE	High atomic number (Z) and Energy
IC	Integrated Circuit
ICAP	Internal Configuration Access Port
IGBT	Insulated-Gate Bipolar Transistor
IOB	Input/Output Buffer
IP	Intellectual Property
ISE®	Integrated Synthesis Environment
ITAR	International Traffic in Arms Regulations
LEO	Low Earth Orbit
LLC	Last Level Cache
LUT	Look-up Table

continued on next page

LVS	Limited Value Shipments
MAC	Media Access Control
MBU	Multiple Bit Upset
MGT	Multi-Gigabit Transceiver
MLC	Multi-Level Cell
MMCM	Mixed-Mode Clock Manager
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
MRAM	Magnetoresistive Random Access Memory
MRPC	Multigap Resistive Plate Chambers
MTBU	Mean Time Between Upsets
MTJ	Magnetic Tunnel Junction
MUCH	Muon Chamber
MVC	Model View Controller
MVD	Micro-Vertex Detector
NBT	Negative Bias Temperature
NIEL	Non-Ionizing Energy Loss
OCM	On-Chip Memory
OPB	On-Chip Peripheral Bus
PCB	Printed Circuit Board
PD-SOI	Partially Depleted Silicon-on-Insulatory
PHY	Physical Layer
PIP	Programmable Interconnect Points
PIPB	Propagation-Induced Pulse Broadening
PLA	Programmable Logic Array
PLD	Programmable Logic Device
PLL	Phase Locked Loop
POL	Point-of-Load
POR	Power-On Reset
PRAM	Phase-change Random Access Memory
PROFiT	Profile-Guided Fault Tolerance
PROM	Programmable Read Only Memory
PSD	Projectile Spectator Detector

continued on next page

PSM	Programmable Switch Matrix
QGP	Quark-Gluon Plasma
RAC	Residue Arithmetic Code
RadWG	Radiation Working Group
RAM	Random Access Memory
RHBD	Radiation-Hardened by Design
RICH	Ring Imaging Cherenkov Detector
ROM	Read Only Memory
RPC	Resistive Plate Chamber
RRAM	Resistive Random Access Memory
RTL	Register Transfer Level
SAA	South Atlantic Anomaly
SAF	Synthetic Anti-Ferromagnetic
SBU	Single Bit Upset
SCR	Silicon Controlled Rectifier
SEB	Single Event Burnout
SECDED	Single Error Correct / Double Error Detect
SEE	Single Event Effect
SEFI	Single Event Functional Interrupt
SEGR	Single Event Gate-Rupture
SEL	Single Event Latch-up
SEM	Soft Error Mitigation
SERT	Single Event Resistant Topology
SET	Single Event Transient
SEU	Single Event Upset
SHE	Single Event Hard Errors
SIHFT	Software Implemented Hardware Fault Tolerance
SIMD	Single-Instruction-Multiple-Data
SIRF	Single-event effects Immune Reconfigurable FPGA
SIS	Schwerionensynchrotron (Heavy-Ion-Synchrotron)
SLC	Single-Level Cell
SOB	Silicon-on-Bulk

continued on next page

SOC	System-on-Chip
SOI	Silicon-on-Insulator
SOS	Silicon-on-Sapphire
SPE	Solar Particle Event
SRAM	Static Random Access Memory
STI	Shallow Trench Isolation
STMR	Selective Triple Modular Redundancy
STS	Silicon Tracking System
STT	Spin-Transfer Torque
SWIFT	Software Implemented Fault Tolerance
TAS	Thermally Assisted Switching
TDC	Time-to-Digital Converter
TID	Total Ionizing Dose
TMR	Triple Modular Redundancy
TOF	Time-of-Flight
TRC	Tunable Replica Circuits
TRD	Transition Radiation Detector
TSOP	Thin Small-Outline Packages
UCF	User Constraints File
USML	United States Munitions List
VDBP	Variable Depth Bragg Peak
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
WAML	Wassenaar Arrangement Munitions List
XST	Xilinx Synthesis Tool

List of Figures

2.1 Basic cross-section of an SOB N-MOSFET. 50
2.2 Basic cross-section of PD and FD SOI N-MOSFET. 52
2.3 Depiction of SRAM cells composed of four or six CMOS transistors. . . 56
2.4 Depiction of a powered 6-transistor SRAM cell storing a logical value. . 56
2.5 Dielectric PZT crystal of an FRAM cell's capacitor. 58
2.6 FRAM 2T2C cell circuit including sense amplifier. 58
2.7 Ferromagnetic Layers of an MTJ cell. 60
2.8 Y-Diagram for hardware design. 65
2.9 Simplified sketch of PLA and LUT. 69
2.10 Simplified sketch of a Xilinx FPGA's components. 73
2.11 Simplified sketch of a Xilinx Series-7 CLB. 74
2.12 Simplified schematic of a Xilinx Half-Latch PMOS circuit. 78
2.13 SRAM SEU cross-section curves of Xilinx Virtex-4 FPGA devices. 91
2.14 Radiation sensitive regions within a 6-transistor SRAM cell. 92
2.15 Basic cross-section of an SOB N-MOSFET struck by ionizing particles. 94
2.16 SET latching into a permanent SEU. 97
2.17 CMOS parasitic npn/pnp BJT forming a thyristor. 101
2.18 Simplified displacement damage in crystal lattices. 105
2.19 Maximum TID of Xilinx FPGAs. 110
2.20 Depiction of a fault-tolerant XTMR design, designated for a Xilinx SRAM FPGA. 140
2.21 Depiction of temporal redundancy design techniques. 144
2.22 Depiction of SBU impact on state transition diagrams of Gray or Parity encoded FSMs. 150
2.23 Depiction of SBU impact on the state transition diagram of a Hamming encoded FSM. 150

3.1 Simplified comparison between Xilinx FPGA firmware designs with or without XTMR. 175

4.1 Signal diagram of FPGA configuration via SelectMAP. 183
4.2 Depiction of DMR and TMR dynamic memory scrubbers. 189
4.3 TMS570 memory scrubber flowchart. 190

4.4 Schematic concept of a fault-tolerant FPGA system. 192
4.5 Depiction of the fault-tolerant CPU with doubled pipeline. 194
4.6 Signal Simulation of the transmission reset by error injection into the serial module's DMR instances. 201
4.7 Screenshot of the JFTToolkit GUI. 202
4.8 JavaCC™ VHDL parser generation activity diagram. 203
4.9 UML package diagram of JFTToolkit. 212

5.1 Virtex-4 FPGA X-ray and copper contact plate views. 224
5.2 SysCore v2.0 and v3.1 FPGA DUT platforms. 225
5.3 Radiochromic film, irradiated with GeV proton particles. 227
5.4 Irradiation beam test setup and readout chain. 228
5.5 TID, SEU and range calculation tool screenshot. 229
5.6 Configuration upsets with and without scrubbing. 230
5.7 Proton irradiation results of the TMS570 memory scrubber. 234
5.8 Hamming FSM beam test results boxplot (median and quartiles). . . . 237
5.9 Current log of mikroElektronika MIKROE-1486 break-out board during proton irradiation. 245
5.10 PTH05010 switching mode power regulator beam test logs. 248

List of Tables

1.1 Upset rate of a conventional semiconductor memory during solar proton events. . . . 38
1.2 Expected surface LET and range of ions expected at FAIR (SIS100). . . . 39

2.1 Neutron Cross-section for Xilinx FPGAs. . . . 54
2.2 Comparison of FRAM, Flash, DRAM and SRAM technologies. . . . 60
2.3 Trapped proton energies and fluxes in LEO. . . . 86
2.4 Xilinx Kintex-7 325T neutron, proton and mixed hadron cross-sections. 92
2.5 Maximum TID of Xilinx FPGAs. . . . 109
2.6 SEU distribution in radiation-tolerant XQR Xilinx Virtex-4 LX/SX devices. . . . 112
2.7 EAR CCL ECCNs of selected FPGA series. . . . 122
2.8 Comparison of radiation-tolerant FPGAs. . . . 129
2.9 Overview of common FSM encoding schemes. . . . 149
2.10 Maximum configuration bandwidth in Xilinx Virtex FPGAs. . . . 152

3.1 Xilinx SEU Mitigation Strategy Selection Guide. . . . 177

4.1 FPGA module size requirements of non-fault-tolerant, DMR and TMR serial interfaces. . . . 200
4.2 VHDL code comparison of a non-fault-tolerant, DMR and TMR design (Architecture Declarative Part). . . . 206
4.3 VHDL code comparison of a non-fault-tolerant, DMR and TMR design (Architecture Statement Part). . . . 207
4.4 Comparison of the resource consumption for redundancy designs. . . 208

5.1 Hamming FSM beam test data taken from three different designs. . . . 236
5.2 FLUKA Monte-Carlo simulation results for the CBM detectors. . . . 240
5.3 Accumulated FLUKA Monte-Carlo simulation results for the CBM detectors. . . . 241
5.4 MSP-EXP430FR5739 logfile excerpt from 20140806-0305. . . . 246

List of Work Publications

The following publications partially or fully include the author's ideas about topics discussed in this thesis. They have been released in primary or concurring authorship during the period of this work and may contain similar wording or reasoning.

José Antonio Lucio Martínez, Jano Gebelein, Udo Kebschull: "*A low cost fault tolerant commercial off-the-shelf MCU adapted with RTEMS/EPICS for controls and instrumentation*" in CBM Progress Report 2015 - DAQ and Online Event Selection, page 111, Darmstadt, Germany, 2016

Jano Gebelein, Sebastian Manz, Heiko Engel, Norbert Abel, Udo Kebschull: "*Smart Module Redundancy - approaching cost efficient radiation tolerance*" in Transforming Reconfigurable Systems: A Festschrift Celebrating the 60th Birthday of Professor Peter Cheung, Imperial College Press, ISBN 9781783266968, London, UK, April, 2015

Jano Gebelein, Christian Stüllein, Udo Kebschull: "*Evaluation of FRAM for use in Radiation Environments*" in CBM Progress Report 2014 - DAQ and Online Event Selection, page 107, Darmstadt, Germany, 2015

Andrei Oancea, Christian Stüllein, Jano Gebelein, Sebastian Manz, Udo Kebschull: "*Implementation and test of a flash-free configuration upset mitigation strategy for the CBM ToF ROB FPGA*" in CBM Progress Report 2014 - DAQ and Online Event Selection, page 105, Darmstadt, Germany, 2015

Christian Stüllein, Andrei Oancea, Jano Gebelein, Sebastian Manz, Udo Kebschull: "*Design, assembly and test of a positioning system for beam tests*" in CBM Progress Report 2014 - DAQ and Online Event Selection, page 106, Darmstadt, Germany, 2015

Andrei Oancea, Christian Stüllein, Sebastian Manz, Jano Gebelein, Udo Kebschull: "*A viable on-chip FPGA configuration memory scrubbing approach for CBM-ToF*" in DPG Conference, HK 59.5, Heidelberg, Germany, March 2015

Christian Stüllein, Andrei Oancea, Jano Gebelein, Sebastian Manz, Udo Kebschull: "*SRAM-Detektor gestützte Positionierung für Elektronik-Strahltests*" in DPG Conference, HK 69.5, Heidelberg, Germany, March 2015

Andrei Oancea, Christian Stüllein, Jano Gebelein, Udo Kebschull: "*A Resilient, Flash-Free Soft Error Mitigation Concept for the CBM-ToF Read-Out Chain via GBT-SCA*" in FPL 2015 Conference, London, UK, 2. - 4. September 2015

Jano Gebelein, Gerhard May, Udo Kebschull: "*SysCore v3.1 – A universal Read Out Controller and Data Processing Board*" in CBM Progress Report 2013 - DAQ and Online Event Selection, page 88, Darmstadt, Germany, 2014

Andrei Oancea, Jano Gebelein, Sebastian Manz, Udo Kebschull: "*Firmware Development for the SysCore v3.1 Configuration Controller*" in CBM Progress Report 2013 - DAQ and Online Event Selection, page 89, Darmstadt, Germany, 2014

Sebastian Manz, Jano Gebelein, Udo Kebschull, Andrei Oancea, Sven Schatral: "*GET4-ROC - Research and Development in 2013*" in CBM Progress Report 2013 - DAQ and Online Event Selection, page 90, Darmstadt, Germany, 2014

Ralph Erdmann, Jano Gebelein, Udo Kebschull: "*Beamtest results of the Lattice LFE2M20E FPGA*" in RADECS 2013 Conference Data Workshop, Oxford, United Kingdom, 23. September - 27. September 2013

Sebastian Manz, Jano Gebelein, Andrei Oancea, Heiko Engel, Udo Kebschull: "*ToF-ROC FPGA Irradiation Tests 2012*" in CBM Progress Report 2012 - Time-of-Flight Detectors, page 69, Darmstadt, Germany, 2013

Jano Gebelein, Dirk Gottschalk, Gerhard May, Udo Kebschull: "*SysCore3 – A universal Read-Out Controller and Data Processing Board*" in CBM Progress Report 2012 - DAQ and Online Event Selection, page 87, Darmstadt, Germany, 2013

Andrei Oancea, Sebastian Manz, Heiko Engel, Jano Gebelein, Udo Kebschull: "*Entwicklung einer fehlertoleranten Konfigurations- und Scrubbing-Kette für den CBM Read-Out Controller (ROC)*" in DPG Conference, Dresden, Germany, HK 34.5, 5. March 2013

Jano Gebelein, Udo Kebschull: "*Investigation of SRAM FPGA based Hamming FSM encoding in beam test*" in RADECS 2012 Conference, Biarritz, France, 24. September - 28. September 2012

Jano Gebelein, Udo Kebschull: "*SRAM FPGA Finite State Machines in Particle Physics Experiments: Beamtest Results*" in DPG Conference, Mainz, Germany, HK 12.4, 19. March 2012

Norbert Abel, Cruz Garcia, Jano Gebelein, Sebastian Manz, Udo Kebschull: "*SysCore3 - a new board for the Universal ROC*" in CBM Progress Report 2011 - DAQ and Online Event Selection, page 63, Darmstadt, Germany, 2012

Norbert Abel, Cruz Garcia, Jano Gebelein, Sebastian Manz, Udo Kebschull: "*SysCore3 - a new board for the Universal ROC*" in CBM Scientific Report 2011 - page 60, Darmstadt, Germany, 2012

Jano Gebelein, Udo Kebschull: "*FPGA fault tolerance in radiation susceptible environments*" in CBM Progress Report 2011 - DAQ and Online Event Selection, page 64, Darmstadt, Germany, 2012

Jano Gebelein, Udo Kebschull: "*FPGA fault tolerance in radiation susceptible environments*" in CBM Scientific Report 2011 - page 62, Darmstadt, Germany, 2012

Jano Gebelein, Udo Kebschull: "*A three-dimensional FPGA array beam detector for ionizing radiation experiments*" in RADECS 2011 Conference, Seville, Spain, 19. September - 23. September 2011

Norbert Abel, Heiko Engel, Jano Gebelein, Dirk Gottschalk, Sebastian Manz, Andrei Oancea, Udo Kebschull: "*Radiation tolerance of the Universal Read Out Controller*" in CBM Progress Report 2010 - DAQ and Online Event Selection, page 62, Darmstadt, Germany, 2011

Jano Gebelein, Heiko Engel, Udo Kebschull: "*FPGA fault tolerance in radiation susceptible environments*" in RADECS 2010 Conference, Längenfeld, Austria, 20. September - 24. September 2010

Jano Gebelein, Heiko Engel, Udo Kebschull: "*FPGA Fault Tolerance in Particle Physics Experiments*" in DPG Conference, Bonn, Germany, HK 10.2, 15. March 2010

Heiko Engel, Jano Gebelein, Udo Kebschull: "*Development of a Radiation Tolerant Softcore CPU for SRAM based FPGAs*" in DPG Conference, Bonn, Germany, HK 10.3, 15. March 2010

Jano Gebelein, Heiko Engel, Udo Kebschull: "*FPGA Fault Tolerance in Particle Physics Experiments*" in it - Information Technology (Oldenbourg Wissenschaftsverlag), Jahrgang 52 (2010) Heft 4 'Test und Zuverlässigkeit nanoelektronischer Systeme'

Sebastian Manz, Norbert Abel, Jano Gebelein, Udo Kebschull: "*An Universal Read-Out Controller*" in Journal of Instrumentation (JINST), Volume 5, Issue Num 11, Nov. 2010, ISSN 1748-0221, doi:10.1088/1748-0221/5/11/C11017

Jano Gebelein, Heiko Engel, Udo Kebschull: "*Verbesserung der Strahlentoleranz von FPGAs für Experimente der Hochenergiephysik*" in ZuE 2009 (Zuverlässigkeit und Entwurf) 3. GMM/GI/ITG-Symposium, Stuttgart, Germany, 21. - 23. September 2009

Jano Gebelein, Heiko Engel, Udo Kebschull: "*An approach to system-wide fault tolerance for FPGAs*" in FPL 2009 Conference, Prague, Czech Republic, 31. August - 2. September 2009

Jano Gebelein, Norbert Abel, Udo Kebschull: "*Fault-tolerant Logics for FPGA Linux*" in DPG Conference, Bochum, Germany, HK 53.8, 18. March 2009

Norbert Abel, Jano Gebelein, Udo Kebschull: "*Dynamical Partial Reconfiguration for Data Acquisition*" in DPG Conference, Bochum, Germany, HK 53.7, 18. March 2009

1 Introduction and Motivation

1.1 Terrestrial Applications and Outer Space

Time to market is nowadays a major pillar even in companies that provide highly specialized hardware components for various markets. But hardware development is a time consuming process mostly requiring multiple iterations before a product can be declared as final. Therefore, more and more parts of hardware became configurable and furthermore reconfigurable by so called firmware, even after the device had been released. This enabled frequently short time to market intervals by the price of incomplete products, but confirms one major requirement: reconfigurable hardware. This opportunity opened the gates for changing requirements. Therefore, even in the field of medical applications, basic physics research or the exploration of outer space, where ASICs were a familiar and well known factor to get calculations done, reconfigurable devices have stepped in and taken their place. But major advantages such as variability, fail-save operation and price-to-market go hand in hand with some drawbacks: Increased susceptibility to radiation, increased power consumption or even political availability on the global market. Especially radiation tolerance became a vital part for space industry when leaving the protective shield of Earth's atmosphere, crossing the Van Allen belts and heading towards outer space. Recently scheduled space missions to Jupiter's moon Europa have to deal with the Jovian synchrotron radiation belt doses [1] and therefore challenge avionics in ways never done before.

Conventional aircraft have to take care of electronic failures, especially those caused by the South Atlantic Anomaly (SAA). Within this area, Earth's Van Allen proton belt approaches closer to the surface due to displacement of the geomagnetic dipole axis [2] and hence drastically increases the generation of device upsets [3, 4]. As reported by NASA, even the SpaceX CRS-1 Dragon spacecraft recently experienced a single event effect in its trunk remote subsystem while being attached to the International Space Station which passed the SAA [5]. In addition, space weather influences nowadays flight plans. There have been many solar events recorded in history, which stressed electronic components more than usual. Assuming a flight in high altitude as well as latitude nearby Earth's magnetic poles, they even exceeded medical limits defined for radiation exposure of human bodies. For instance, when taking into account a conventional trip from London to Los Angeles at 12 km alti-

Solar Event	Neutron Flux $[cm^{-2} \cdot s^{-1}]$ 17 km / 12 km	Upset Rate $[h^{-1}]$ 17 km / 12 km	MTBU $[s]$ 17 km / 12 km
23.02.1956	2893 / 1113	582 / 247	6.2 / 14.6
29.09.1989	487 / 191	98 / 42	37 / 85.0
24.10.1989	80 / 31	16 / 7.0	224 / 517
GCR	9.3 / 5.8	1.8 / 1.2	2005 / 2935

Table 1.1: Upset rate as well as Mean Time Between Upsets (MTBU) of a conventional semiconductor memory during solar proton events while flying at high latitude. All values are indicated for altitudes of 17 km as well as 12 km, assuming a proton cross-section of $5 \cdot 10^{-14}$ $cm^2 \cdot bit^{-1}$ [7]. GCR is indicated for comparison.

tude, the proton stress events in February 1956 or September 1989 [6] would have exceeded 1 mSv [7], the annual artificial supplementary dose limit of a public person without radiation surveillance [8]. Table 1.1 gives a short overview of bit error rates, also called upset rates, in a conventional semiconductor memory, caused by these solar events in comparison to the Galactic Cosmic Ray (GCR).

Artificially generated radiation impact due to historical nuclear events comprises another significant source of semiconductor stress within affected regions on Earth. It even led to recent reliability studies of leading chip manufacturers to certify device characteristics [9]. Especially in the beginning of the atomic age, first nuclear tests as well as atomic disasters lasted for decades due to the long half-life period of some emitted particles. The Starfish Prime Event, a 1.4 megatons nuclear warhead detonated on July 9th 1962 at high altitude of 325 km, not only caused an electromagnetic pulse as well as widely visible aurora borealis nearby. It furthermore emitted enough X-ray radiation to cause ionization within large regions of the upper atmosphere. The hence emitted electrons finally formed a completely new artificial electron radiation belt within the magnetosphere which existed for a decade and finally caused at least seven satellites to fail because of total dose lifetime degradation [10] – Ariel-1, TRAAC, TRANSIT-4B, Telstar, and others. In addition, several nuclear disasters such as Chernobyl (1986), Three Mile Island (1979), Fukushima (2011), and many less critical ones partially emitted radiation with half-life periods beyond of 30 years to environment as well as atmosphere which can still today be monitored and have impact on semiconductor electronics.

In addition, several medical applications make use of ionizing radiation. Beside of the well known conventional X-ray medical imaging and the further improved

Ion	Energy $[GeV/amu]$	LET $[MeV \cdot cm^2 \cdot mg^{-1}]$	Range in Si $[mm]$
H	29	$2.25 \cdot 10^{-3}$	$6.20 \cdot 10^{4}$
He	14	$8.17 \cdot 10^{-3}$	$3.16 \cdot 10^{4}$
Li	12	$1.80 \cdot 10^{-2}$	$2.13 \cdot 10^{4}$
Ne	14	$2.04 \cdot 10^{-1}$	$6.31 \cdot 10^{3}$
Fe	13	1.37	$2.44 \cdot 10^{3}$
Kr	12	2.59	$1.77 \cdot 10^{3}$
Pb	11	13.3	$7.80 \cdot 10^{2}$
U^{28+}	2.7	16.7	$7.09 \cdot 10^{2}$

Table 1.2: Expected surface LET and range of ions expected at FAIR (SIS100), taken from [12].

computer aided tomography, radiotherapy became a major field of radiation application. Conventional cancer therapy penetrates extensive cell regions using X-ray waves, BNCT penetrates boron-targeted cells by the use of neutrons [11] and even modern ionization therapy penetrates focused cell regions by the use of carbon heavy ions. Therefore, especially human application requires a high grade of reliability and radiation hardness of all used electronic components to prevent machine failure and therefore possible radiation injuries.

Beside of all this, scientific fundamental research as for instance is achieved in nowadays particle accelerators due to nuclear collisions, feature the highest fluences of the most diverse spectra of all: α, β, γ, X-ray, proton, neutron, heavy ion and many more. Therefore, electronic components and devices which are going to be used within this field of application demand special requirements if constant failure is not an option.

Finally, all of the above mentioned application scenarios have one major thing in common: If they want to benefit from the numerous advantages coming with currently available high-performance FPGA devices, they have to deal with the arising issues due to radiation impact on conventional semiconductor electronics.

1.2 The Facility for Antiproton and Ion Research

FAIR, the Facility for Antiproton and Ion Research is an international institution for basic physics particle acceleration and collision experiments, whose main facil-

ity is located in Darmstadt, Germany. It has been officially recommended by the Nuclear Physics European Collaboration Committee (NuPECC) to enhance and advance nuclear physics in Europe for at least the next decade [13]. FAIR should act as a universal facility offering a widespread scientific program for use in atom, plasma, antimatter, particle, hadron and nuclear physics, applied physics as well as nuclear astrophysics, material science, biology and biomedicine. It should find answers to actual fundamental aspects about structure and evolution of matter, such as the origin of hadron mass, properties of cosmic matter under extreme dense and temperature conditions, the composition of matter in the early universe, and how it evolved to form elements [14]. 14 experiments currently apply for these challenges, united within 4 major pillars at the FAIR:

- **APPA** – Atomic, Plasma Physics and Applications – Gas of electrically charged particles will be created and investigated at high pressure and low temperatures. Furthermore, heavy ions will be used to conduct research on the impact of cosmic radiation on biological cells as well as technological materials.

- **CBM/HADES** – Compressed Baryonic Matter / High Acceptance Di-Electron Spectrometer – Nucleus-nucleus collisions will be used to create a superdense baryonic matter at cold temperature. This allows investigation of the quark matter phase diagram in the formerly unknown regions to learn more about the point of baryon phase transition from the dense hadronic to a new color superconductivity phase. Furthermore, the exact densities at which Quark-Gluon Plasma (QGP) phase changes take place have to be explored.

- **NuSTAR** – Nuclear Structure, Astrophysics and Reactions – For investigation, rare radioactive isotopes with very short-lived nuclei will be spatially separated and identified within few hundred nanoseconds by a so called Superconducting Fragment Separator.

- **PANDA** – antiProton ANnihilation at DArmstadt – Antiprotons will be collided with a fixed target. This proton-antiproton collisions will be recorded within a versatile detector to draw conclusions from the charged particles trajectory, energy as well as momentum. Although an international review committee rated PANDA in February 2015 to be not cost-effective [15], the FAIR council decided end of September 2015 to build it anyway, maybe due to the massive international nuclear physics community support [16].

Further details about the mentioned CBM detector are summarized in the following section 1.3.

1.3 The Compressed Baryonic Matter Experiment

As mentioned in section 1.2, the CBM experiment is one of the four major pillars of the FAIR. Since it has to share the experimental cave with the HADES experiment, all components have to be optimized not only in function, but also in size. In principle, CBM will investigate properties as well as behavior of dense, compressed baryonic matter at low temperature as it may exist in cosmic scale within the center of neutron stars or within the core of supernovae.

1.3.1 Major Objectives and Physics Background

Modern heavy ion particle accelerators, such as the SIS-100/300 at the upcoming FAIR, allow the creation of strongly interacting and super-dense baryonic matter by collision of nuclei on the atomic scale. Baryons are a class of particles built of three quarks, such as nucleons (protons, neutrons) or some unstable strange particles. Baryon density as well as temperature thereby nearly completely depend on the particle beam energy. This interacting matter contains nearly as much particles as anti-particles and therefore likely reproduces the condition when elements in the universe began to form. If temperature as well as density are raised, the baryons start to show baryon resonance and subsequently quickly decay into pions and nucleons, creating hadronic matter. Hadrons include all baryons and hyperons (three quarks) as well as mesons and quarkonia (one quark and one antiquark). If a specific temperature is exceeded, the hardrons start to melt and transit into the QGP phase. Vice versa, if the temperature remains cold and only density increases, the highly compressed cold nuclear matter may correlate and transit into a completely new color superconductivity phase [17], with analogies to known superconducting metals. In both cases, the density threshold for the phase change is completely unknown and the experimental approval would be a breakthrough in the understanding of strongly interacting matter. Systematic investigations of heavy ion interactions, starting from 1 AGeV up to multiples of 10 AGeV, are planned to gather enough statistically significant results that allow to close this gap within the phase diagram of strongly interacting matter. Further details regarding the physics background of CBM can be found in [18].

1.3.2 Detector Concept and Experimental Setup

To reach the well defined goals of the CBM experiment, multiple detectors are necessary to capture generated particle tracks and properties during the short moments of nuclei collision. The gathered and recorded data is subsequently reduced by the amount of already known physics. If the remaining data afterwards shows characteristic properties and exceeds a statistically significant threshold, the exploration of new physics might be within reach. CBM is currently planned to be setup in an initial start version with some detectors offering only reduced functionality and others extended to full capability. They include micrometer precise track vertex reconstruction within a magnetic field by using silicon pixel and strip detectors as well as an identification system for electrons composed of a Ring Imaging Cherenkov detector and a 3-layered Transition Radiation Detector. Furthermore they feature identification of spontaneous decaying muon particles as well as several time-of-flight measurements. All of the detectors as well as the connected semiconductor readout electronics need to meet very strict requirements regarding precision, speed and of course also radiation hardness to record about 1000 charged particles interacting with a speed of up to 10 MHz for central Au/Au collisions at 25 AGeV. In detail, CBM will be composed of the following detectors [18]:

- **Dipole Magnet** [19] – The dipole magnet provides the magnetic field with large acceptance for MVD and STS. It operates at a current of 686 A and provides a maximum magnetic flux density of 3.25 T, comparable to a medium magnetic resonance scanner in medicine. To be able to obtain the required particle momentum resolution, the magnet has a rigidity of about 1 Teslameter.

- **Micro-Vertex Detector (MVD)** – High resolution decay vertex reconstruction of charged particles within a magnetic field is done via four thin and fine grained layers of silicon pixel detectors. Low thickness of less than 500 µm is inevitable to reduce multiple particle scattering, while a high spatial resolution of less than 5 µm is necessary to reliably detect single particle positions. Both requirements can be achieved by the use of low voltage CMOS Monolithic Active Pixel Sensors (MAPS) [20] whose single pixel size are between 18 µm · 18 µm and 20 µm · 40 µm [19].

- **Silicon Tracking System (STS)** [21] – Charged particle trajectories within the magnetic field are reconstructed by the use of 1000 double-sided ultra-low-mass silicon strip detectors, arranged in 8 layers. Each detector features 2048

silicon strip sensors in lengths between 20 and 60 mm and a thickness of 300 µm. This finally requires to readout about 2 million sensors in total without adding too much weight and material due to cabling.

- **Ring Imaging Cherenkov Detector (RICH)** [22] – The identification of electrons as well as the suppression of pions which are created due to the baryon decay are a fundamental requirement for CBM. Therefore, two multi-anode photo multiplier planes recognize the photons that are emitted when charged particles pass the gaseous CO_2 system within the detector. A preceding glass mirror of 3 m radius and 6 mm thickness for each plane assists in focusing the photons onto both relatively small sensitive areas of about 1 m^2 or 55000 pixels respectively. This high granularity and the high number of recorded circular ring image events allows to reach the required electron identification and pion suppression rates for CBM.

- **Muon Chamber Detector (MUCH)** [23] – Low-momentum muon track identification within a dense heavy ion collision environment is performed via a sliced, multi-layered hadron absorber system with detection chambers sandwiched in between. The final MUCH version will consist of 6 separate hadron absorption layers made of iron with varying thickness and size to prevent muon stopping within just a single thick block. The high resolution gas chambers behind each iron slice feature triplets of detector planes which results in a total number of 18 layers based on different technologies. The two initial stations are Gas Electron Multipliers (GEM), the subsequent two are made of straw tube detectors and the fifth station is based on hybrid GEM+Micromegas technology. To increase the overall sensitive area, the last station will involve the TRD detector, since it is located directly behind the muon detector. This full system requires to readout a total number of about half a million data channels within a radiation field for the proposed event rate of 0.04 hits/cm^2 and 0.4 MHz/cm^2 respectively for central 10 MHz Au/Au interaction rate at 35 AGeV.

- **Transition Radiation Detector (TRD)** – Three major stations offering about 500 m^2 of sensitive area in total are used for additional particle tracking as well as electron identification via transition radiation of charged particles at the borders of the radiator material. This will be achieved by the use of fast gas system detectors, based on MWPC and GEM technology, each one offering a sensitive cell pad area of 1 or 2 cm^2 [24], capable of recording event rates up

to 100 kHz/cm^2 within 10 MHz central Au/Au particle collisions at 25 AGeV. The total number of channels which need to be synchronously readout and processed within radiation-hard ASICs, is about one million for the final TRD.

- **Time-of-Flight Detector (TOF)** [25] – For pion, kaon as well as proton identification it is necessary to get information about each particle's arrival time and to correlate them with the track data gathered by the STS detector to finally calculate the total time of flight. Therefore, a large 12 m · 9 m wall of Multigap Resistive Plate Chamber (MRPC) strip counters of variable length will be used. They are made of stacked low-resistivity glass, separated by thin gas gaps with a high electric field [26], and can detect the signals generated by the occurring electron avalanches. The plate chamber sizes differ between inner and outer regions due to the varying particle densities, but especially the inner chamber pads measuring 5 cm^2 have to deal with event rates up to 20 kHz/cm^2 and a time resolution of 80 ps for central Au/Au collisions at 25 AGeV. TOF finally requires a total number of 60000 channels to be read out and processed by radiation-hard electronics. In addition, the required data reduction will be achieved by the use of conventional COTS FPGAs which are located within the ionizing radiation field as well [27].

- **Electromagnetic Calorimeter (ECAL)** – To measure direct photons as well as photons decaying from neutral mesons, a common 'shashlik' type calorimeter is adapted for CBM. The modular system can be arranged in wall or tower shape related to the beam line. Each of the about 100 modules is composed of multiple 1 mm thin sliced lead and scintillator layers featuring energy loss measurement of the electron-photon showers. This enables detection of approximately 80 photons in total for a central Au/Au collision at 25 AGeV [18]. The current ECAL design is composed of 1088 modules which have a total number of 4352 channels.

- **Projectile Spectator Detector (PSD)** [28] – Heavy ion collision centrality as well as event plane reconstruction are important to make specific statements about collective particle flow but without taking the collision particles themselves into account. Therefore, the non-interacting nucleons as well as fragments of the projectile nucleus will be measured by the use of a calorimeter composed of 44 modules surrounding the beam spot. Each module is made of 60 sandwiched lead/scintillator layers with just 20 cm · 20 cm in size. Wavelength shifting fibers afterwards transport the generated scintillation light

for signal processing to Multi-Avalanche Photo-Diodes (MAPD). The required Time-to-Digital Converter (TDC) resolution is 1 ns but the total number of 500 detector channels is relatively low.

With regard to the limited spatial capabilities within the CBM cave, it has been decided to install a large crane, which is able to exchange the RICH and MUCH detectors against each other and therefore enables the configuration of different detector setups according to [29]:

- Bulk hadrons, multi-strange hyperons and open charm measurements: Magnet, MVD, STS, TRD, TOF, PSD
- Di-electrons measurements: Magnet, MVD, STS, RICH, TRD, TOF, PSD
- Charmonium measurements: Magnet, MVD, STS, MUCH, TRD, TOF, PSD

Since all of the detector systems are operated within a radiation environment, high standards are defined for all components. Especially the readout electronics for the indicated millions of analog channels are mostly located closely nearby the beam line and therefore highly vulnerable to degradation effects. This defines the necessity of radiation-tolerant electronics design for all exposed components and furthermore prohibits the use of conventional doped optic fibers and diodes normally used for high speed data transfers. It involves major parts of the data acquisition (DAQ) system, which allows all experimental data to be collected, pre-processed and forwarded from the self-triggered CBM detector system to the First Level Event Selector (FLES).

1.3.3 Data Acquisition and First Level Event Selector

Development and construction of a sufficient and reliable DAQ system as complex as for the CBM detector is a major challenge on the way to new physics experiments. It has to efficiently merge all different detector systems and subsystems under a global context which afterwards is used to forward the information with high speed to a data center that stores it on hard-drives for later analysis. Quite often, the detector systems decide for technology upgrades with higher bandwidths and lower latency which also results in major adaptions of the DAQ chain. Therefore, many things have changed since the first conceptional thoughts about a common DAQ system for CBM have been written down [30, 31]. Even the most central deterministic latency data transfer, clock distribution and time synchronization protocol 'CBMnet' [32, 33] recently had to be completely replaced [34] because it was not able to fulfill all basic requirements anymore. This was a critical decision, as some of the detector Front End Electronic (FEE) ASICs already implemented hard-wired

circuits 'speaking' the former protocol and now have to be redesigned or, if this is not possible anymore, supplemented by additional converter chips.

The newly conceptualized readout chain now promises successful transmission of measurements taken from the expected high reaction rates during particle collisions while offering the required back channels for configuration and control flows such as FPGA firmware, actuator control or back pressure notifications. Since CBM is a self-triggered system, every FEE component has to stay in sync with the system's global time to be able to generate correct time stamps for all captured events. This amount of generated data, about 2 TB/s in total, subsequently has to be buffered and transferred to a data processing layer located in a concrete-shielded area within the CBM cave. This is handled by using radiation-hard optic diodes as well as radiation characterized optic fibers with 4.8 Gbps per channel. The Data Processing Boards (DPB) in turn combine multiple input channels and pre-process the incoming data to reduce the outgoing bandwidth to a maximum of about 1 TB/s. This initial real-time analysis includes background suppression, physical relevance checks, cluster analysis as well as feature extraction. The chosen events are subsequently sent to the FLES for online event selection with thousands of processor cores and graphics cards. This reduces the amount of collision data to about 1 GB/s, designated for storage on conventional hard-drives for offline analysis [35]. The geographical distance between CBM cave and the FLES building "Green-Cube" is about 700 m and will be overcome by the use of a conventional 10 Gbps optic fibers network.

As the proposed Au/Au collisions result in an event rate of 10 MHz within the FEE, data reduction is a crucial requirement of the whole self-triggered DAQ system. Up to now, it is impossible to record the whole bunch of event information onto hard-drives while staying within budget. Therefore, the major goal is to perform data reduction as close as possible nearby the detector systems, which in turn requires radiation tolerance of readout ASICs and fault tolerance for FPGA designs (see sections 2.6 and 2.7 for further details).

1.4 Key Goals of this Study

Based on the requirements of the FAIR CBM detector readout chain and the hereby taken decision to utilize COTS FPGA devices to efficiently pre-process detector data in areas with ionizing radiation, the necessity of fault-tolerant system design raised. Therefore, radiation effects of the proposed FPGA types as well as alternative devices had to be investigated and mitigation strategies had to be analyzed. Considering spatial overhead and timing interference, an estimation of the

additionally introduced but necessary fault tolerance techniques had to be given. A possible simplification of the existing manual and automated solutions should be taken into consideration, as manual design was known to be very complex and most of the tools back then were known to operate only on design layers where user intervention has become close to impossible. Particle accelerator radiation tests should be performed to provide a proof of principle to finally be able to easily utilize fault-tolerant designed FPGAs for CBM.

According to these plans, the following chapters are segmented to provide a mostly linear approach towards reaching the major goals. Starting with the basics about semiconductor technologies which have been taken into consideration in section 2.1 as well as the main FPGA characteristics in 2.3, the effects of ionizing radiation and their impact on semiconductors are indicated in sections 2.4 and 2.5. Subsequently, radiation mitigation, firmware fault tolerance as well as software fault tolerance are depicted in sections 2.6, 2.7 and 2.8. The basics section is finalized with a short overview of existing fault tolerance as well as error simulation tools in sections 2.9 and 2.10. The subsequent sections in chapter 3 depict the approach towards fault tolerance for FPGAs. Following this pathway, the necessary milestones which have to be taken into consideration when designing a fault-tolerant system are shown in chapter 4. Starting with the FPGA configuration and memory refresh sections 4.1 and 4.2, some functional modules have been realized by manual design in section 4.3. The subsequent automated approach can be found in section 4.4, round up by the guidelines for fault-tolerant VHDL design in 4.5. Experimental setups and validation in particle accelerator beamtime experiments are indicated in chapter 5. In this context, test platform assembly and radiation monitoring are shown among others in section 5.1. Partial device reconfiguration and the fault-tolerant firmware improvements are analyzed in sections 5.2 and 5.4. In addition, some COTS components relevant for CBM have been irradiated and tested in section 5.6, round up by a semiconductor feasibility study focusing on the expected radiation levels and usage scenarios in CBM that can be found in section 5.5. Finally, chapter 6 summarizes the achieved work and concludes with a short outlook.

As one of the major concepts of this work was to impart knowledge to a wide audience and therefore to keep the complexity at a reasonable level, some concepts may provide a more detailed description to get sure that the necessary theoretical background is at everybody's disposal.

2 Background Analysis

This chapter represents the current state-of-the-art in the fields of semiconductor architectures, ionizing radiation effects, radiation mitigation methods, fault tolerance techniques and a selection of existing fault tolerance tools which in total are related to FPGA devices. It tries to constitute the technologically advancing development but also includes latest research in the field of semiconductor devices used in radiation environments.

2.1 Architectures and Technologies

The following sections contain a basic overview of technologically relevant architectures and manufacturing processes which the author got in contact with during his research on the proposed topic of this thesis. It makes no claims to completeness but contains the most important facts about current and future technologies which have to deal with several radiation effects.

2.1.1 CMOS Transistor Technology goes FinFET

Complementary Metal Oxide Semiconductor (CMOS) technology is nowadays a well established standard for manufacturing metal oxide semiconductor field effect transistor (MOSFET) circuits on the basis of silicon. It is based on the combination of pull-up PMOS 'pnp'-junction as well as pull-down NMOS 'npn'-junction transistors to create simple logic circuits such as inverters, logical 'OR' or logical 'AND'. PMOS and NMOS are composed of positively as well as negatively doped semiconductor materials which are created by introducing foreign atoms within the silicon lattice. This process requires multiple cycles of photo-lithographic masking and etching, combined with the application of several metal layers for routing connectivity between transistors, to subsequently define the overall behavior of the final circuit. Since doped silicon, especially beneath powered transistors, is susceptible to ionizing radiation effects as shown later-on in section 2.5, it has to be clearly understood, how CMOS works and how its latest improvements affect this process.

As illustrated in figure 2.1, an NMOS transistor is basically build of source, drain and bulk contacts as well as a gate electrode. The gate itself is clearly separated from the doped bulk silicon by a thin insulating gate oxide, normally made of silicon dioxide, silicon nitride, or silicon oxynitride. It forms a capacitor between gate and bulk

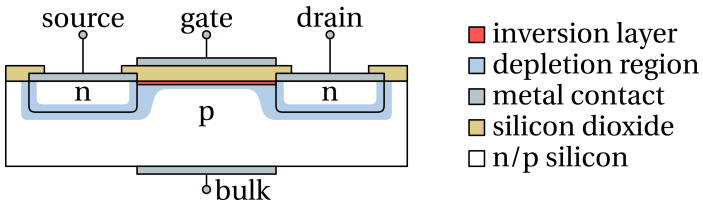


Figure 2.1: Basic cross-section of an SOB N-MOSFET. P-doped (electron holes) and N-doped (free electrons) silicon areas form a depletion region along their junction area without conductance. Powering the gate electrode also shifts the conduction band of the p-doped silicon until the introduced electrons form a conducting inversion layer below the gate oxide. This newly formed inversion layer now connects source and drain electrodes.

silicon substrate. The slightly doped p-type silicon is manufactured in a way that it offers few electron holes in the material's valence band whereas the n-doped silicon has an excess of electrons to ensure the transistor's proper conductivity. Powering the gate electrode with a positive voltage generates an electric field, which causes free electrons to combine with the substrate holes along the p/n junction, forming the so called depletion region without conductance (except of leakage current). Increasing this voltage continuously shifts the energy bands of the bulk silicon, which results in electrons populating the conduction band. If this introduction of electrons within the substrate continues and if a material-dependent threshold voltage is exceeded, no holes are available for recombination anymore. This forms a conductive inversion layer along the oxide junction within the depletion zone, which electrically connects source and drain electrodes. This effect of the doped silicon's shifted Fermi level allows current to flow between source and drain, as long as it does not exceed the maximum value which increases resistance level and counteracts with the creation of the inversion layer. Minor changes within the balanced electron-hole-pair ratio therefore can lead to malfunction of the whole MOSFET. The gate oxide itself has to be well defined, considering operation voltage and temperature of the transistor. Its diameter mainly defines the amount of electrons which will leak through the gate: The thinner, the worse. Moreover, this effect, together with the mentioned leakage of electrons from source to drain, increases with the circuit's operation voltage.

Since the CMOS process itself is geometrically scalable very well, a significant performance update could be reached whenever the architecture was shrunk to a

smaller level on the nanometer scale. As a result, gates became thinner, the number of electrically trapped holes and electrons within the p- and n-doped silicon got smaller, and transistor switching speed increased, but in consequence, leakage current at the gate electrode significantly increased [36]. Therefore, the MOSFET operating voltage had to be reduced. While moving to fully depleted (FD) transistors, where source and drain are separated only by a very small region of silicon that acts as the conduction layer, recent architecture scaling introduced 16 nm TSMC FinFET+ 3D transistor technology for Xilinx Kintex and Virtex UltraScale+ SRAM FPGAs [37], 14 nm Intel Tri-Gate FinFET technology for Altera/Intel Stratix 10 SRAM FPGAs [38], 20 nm for Altera/Intel Arria 10 SRAM FPGAs [39], 22 nm for Tabula ABAX2P 3PLD SRAM FPGAs [40] and 22 nm for Achronix Speedster22i SRAM FPGAs [41]. In FinFET technology, the gate electrode of a traditional planar MOSFET was rotated vertically by 90 degrees and thinned to just a small fin which is surrounded by the gate and in between the gate oxide. This allowed a massively shrunk horizontal gate size due to the extension in height, but also influenced doping requirements [42, 43], as the whole fin itself now forms the conducting inversion layer below the gate oxide. But FinFET also enables new possibilities, e.g by combining multiple fins to a single logical transistor to increase the total drive strength for higher performance and reliability as done for Intel's Tri-Gate technology processors [44]. In contrast, lower fan-outs complicate ASIC and FPGA timing closure for synthesized logic when trying to maximize the improved operating frequencies coming with FinFET. Therefore, new physical design patterns for FPGA SRAM cells have to be used [45]. Finally, there are many issues to solve, starting first of all with the most basic lithography processes required to manufacture FinFETs, which currently require design patterns for separate masks or the expensive use of Extreme Ultra-Violet (EUV) wavelength light sources to print MOS circuits.

By following Moore's law of doubling the number of transistors of an integrated circuit approximately every two years, CMOS down-scaling will reach atomic scale sooner or later. Hence, physical quantum effects will start to play a major role. Moreover, electrical cross-talk between dense conducting layers will increase and in consequence, the number of errors which have to be handled within a circuit will rise. Alternatives, which try to solve these problems beyond nanometer scale can be found among others in the atom-scale quantum-dot research [46] as well as in the removal of the conventional doping material itself by using nanowire field effect transistors in combination with SOI [47].

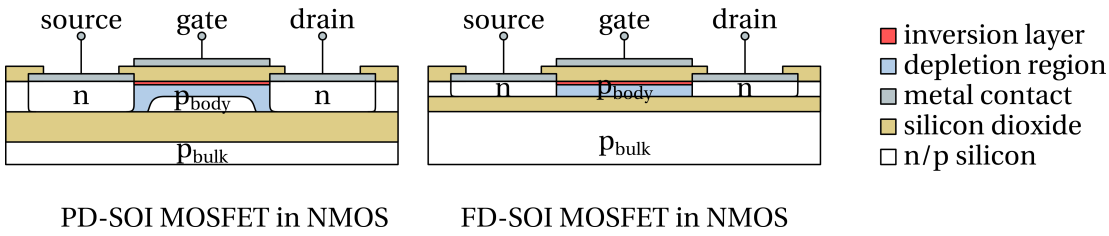


Figure 2.2: Basic cross-section of PD (left) and FD (right) SOI N-MOSFET. The bulk silicon carrier is separated from the MOSFET by an additionally introduced insulator material. This enables FD MOSFET design, where the top silicon layer as well as the insulator itself are significantly thinned in comparison to conventional PD design. In consequence, the partially depleted body region becomes fully depleted.

2.1.2 SOI - Silicon-on-Insulator

In contrast to conventional CMOS transistors, which are manufactured in SOB directly on bulk silicon as shown in section 2.1.1, the silicon carrier of SOI devices is separated from the MOSFET by an additionally introduced insulator material, for example an oxide or sapphire, forming a silicon-insulator-silicon stack as shown in figure 2.2. This insulation layer encapsulates the top silicon and hinders electrical connection to the bulk. In consequence, source, drain and the partially depleted (PD) body region become unbiased and hence floating. This enables charge to be stored within the body region and leads to modified capacitance, varying threshold voltage, and history effects where delay times between first and second switching can be observed [48]. To overcome these effects, the top silicon layer of SOI devices can be significantly thinned in comparison to conventional MOSFET design, resulting in the body region to become fully depleted (FD) of mobile charges [49] without showing history effects anymore. The insulating layer itself can be kept thin and due to the top silicon thinning, the channel needs to be lightly doped only or it can even remain completely undoped, reducing the hot carrier effect [50]. This in turn as well as the smaller sensitive volume reduces radiation impact due to ionizing particles [51].

In general, PD and FD SOI MOSFET entails many positive effects in comparison to SOB. Capacitance between gate and bulk silicon substrate is reduced and therefore the number of carriers required for transistor switching is decreased. This improves switching speed and therefore the overall operating frequency. Electrical insulation from the bulk as well as the fully depleted body region reduce leakage current and therefore the overall power consumption.

Another prominent insulator material to benefit from the SOI advantages and furthermore improve device radiation characteristics is sapphire. It is solely utilized as carrier material for so called Silicon-on-Sapphire (SOS) SOI devices. Due to its ideal insulation properties, penetrating ionizing particles can be effectively hindered from influencing powered circuits nearby [52]. One famous example for an SOS device is the 8 bit Microprocessor RCA1802 which has been used multiple times in the Galileo spacecraft [53].

A recent high performance semiconductor device using SOI is the 4 GHz 22 nm IBM POWER8 CPU [54]. Even FinFETs can be manufactured on SOI substrates instead of SOB, but currently cope with some major disadvantages such as higher wafer cost and lower yield. They also suffer from poor heat dissipation characteristics and therefore Negative Bias Temperature (NBT) stress [55]. Furthermore, the radiation impact on FinFETs caused by ionizing particles differs slightly but has to be considered for radiation environments [56, 57, 58].

2.1.3 Xilinx Triple-Oxide Technology

CMOS manufacturing lithography, and therefore transistor size, significantly shrunk over the last decades to increase wafer yield while reducing total costs. Transistor source, drain and gate oxide structures were also minimized to increase switching speed and to limit power consumption and therefore heat. But thinner silicon dioxide gate structures cannot be operated with common voltage levels, otherwise degradation and power leakage, which depend on temperature, would speed up significantly and the device will become unusable way before its proposed lifetime has been reached. Therefore, the maximum transistor operation voltage had to decrease in parallel to the CMOS process itself. Xilinx Virtex-4 (90 nm) FPGAs, for example, are operated with an internal core voltage of 1.2 V, Virtex-5 (65 nm) with 1.0 V and Virtex-7 (28 nm) requires even less down to 0.9 V [60]. On the other hand, chip-externally interfaced components are operated at voltage levels of up to 5.0 V and mostly do not require highest transfer speeds. To mitigate degradation and power leakage effects due to these voltages, gate silicon dioxide of the corresponding input/output pin transistors have to remain at a reasonable thick level. Additional effects like an increased current leakage of thin in comparison to thick gate dioxide transistors may justify any manufacturing process in between, where switching speed is not necessary.

Due to these requirements and issues regarding performance, powering and size, chip vendor Xilinx manufactures its in-house Virtex FPGA devices in a special triple

Technology Node	Product Family	Config. Memory $[\text{cm}^2 \cdot \text{bit}^{-1}]$	Block Memory $[\text{cm}^2 \cdot \text{bit}^{-1}]$	Error
250 nm	Virtex	$9.90 \cdot 10^{-15}$	$9.90 \cdot 10^{-15}$	±18%
180 nm	Virtex-E	$1.12 \cdot 10^{-14}$	$1.12 \cdot 10^{-14}$	±18%
150 nm	Virtex-II	$2.56 \cdot 10^{-14}$	$2.64 \cdot 10^{-14}$	±18%
130 nm	Virtex-II Pro (dual-oxide)	$2.74 \cdot 10^{-14}$	$3.91 \cdot 10^{-14}$	±18%
90 nm	Virtex-4 (triple-oxide)	$1.55 \cdot 10^{-14}$	$2.74 \cdot 10^{-14}$	±18%
90 nm	Spartan-3	$2.40 \cdot 10^{-14}$	$3.48 \cdot 10^{-14}$	±18%
90 nm	Spartan-3E Spartan-3A	$1.31 \cdot 10^{-14}$	$2.73 \cdot 10^{-14}$	±18%
65 nm	Virtex-5 (triple-oxide)	$6.70 \cdot 10^{-15}$	$3.96 \cdot 10^{-14}$	±18%
45 nm	Spartan-6 (dual-oxide)	$1.00 \cdot 10^{-14}$	$2.20 \cdot 10^{-14}$	±18%
40 nm	Virtex-6 (triple-oxide)	$1.26 \cdot 10^{-14}$	$1.14 \cdot 10^{-14}$	±18%
28 nm	7 Series FPGAs (triple-oxide)	$6.99 \cdot 10^{-15}$	$6.32 \cdot 10^{-15}$	±18%

Table 2.1: Neutron cross-section for Xilinx FPGAs. All data has been taken from table 1-17 in [59].

oxide process [61]. But the cheaper Spartan series was never offered such an option and it remained with a dual-oxide process, missing the thin oxide layer. Nowadays, each modern Xilinx FPGA is a combination of three different gate silicon dioxide thicknesses, which have to interact precisely to gather the overall designated performance and power savings. The decision about which thickness for which feature had been made for time critical processing elements, basic configuration cells, and the chip's external input/output connectors.

All performance critical transistors within an FPGA, that provide the basis for logic flip-flops, interconnect buffers as well as on-chip embedded block memory, are done in the thinnest available gate oxide with lowest threshold voltage and shortest channel length [61]. It enables the use of high density transistors with increased switching speed at lower voltage – at the price of high leakage. Thin oxide also changes the cross-section and therefore susceptibility to radiation if not considered

and correctly mitigated during chip design phase by modifying doping, materials, isolation or by adding circuit redundancy, guard rings or enclosed layout transistors (ELT) [62]. As seen in table 2.1, such a change happened while moving forward to 45 nm.

The second layer, introduced with the Virtex-4 devices [63], is manufactured in a medium size gate oxide thickness and is used for all FPGA configuration memory transistors as well as the interconnect pass gates. The configuration memory is accessed at chip initialization phase or reconfiguration phases only. Therefore, it does not need to be that fast in comparison to the processing elements within the chip. This significantly decreases power consumption [63] and therefore heat dissipation. Nevertheless, as its manufacturing process is slightly bigger, susceptibility to radiation is also different and lacks of exactly the same problems at a different scale. A significant change in the mid-oxide process, where neutron cross-section was decreased at the order of a magnitude, can be seen in table 2.1 while stepping to 65 nm. But the subsequent increase of series 6 devices can be seen temporal due to a major change in the manufacturing process and had been fixed again for newer series 7 devices. A popular design technique to reduce the sensitivity of a memory cell even if the CMOS process size is shrinking lies in the maximization of the capacitive load where speed is not critical [64].

The third and last triple-oxide CMOS layer is manufactured in a thick gate dioxide. This is due to the requirement of handling higher input/output voltages up to 5 V, for example LVCMOS25 or LVCMOS33. It retains compatibility to conventional Printed Circuit Board (PCB) design and connectivity of external buses and interfaces. Exact radiation susceptibility of this layer is not included in the reliability reports, but has been proven to be relatively uncritical [65].

Due to the triple- and double-oxide manufacturing processes, Virtex and Spartan devices of the same generation always show different susceptibility results. This behavior continued while moving to series 7 where the Spartan product line was replaced by Artix and Kintex chips, which are both manufactured in the same process and therefore share the same cross-section indicated in table 2.1. Virtex-7 chips are slightly different and use a highly purified boron for the inter-layer isolation of the copper process, which does not add susceptibility while decomposing to alpha particles when trapping low energy thermal neutrons within the device material itself. This results in about 20% lower cross-sections (see [66, 67, 68]) and is not indicated in table 2.1.

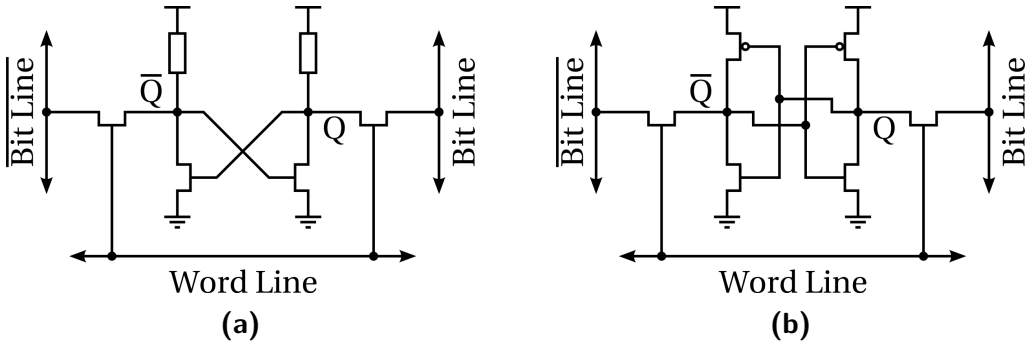


Figure 2.3: Depiction of SRAM cells composed of (a) four or (b) six CMOS transistors. Manufacturing of the resistor cell (a) is more compact but also more complex due to an additionally required highly resistive polysilicon layer that offers the pull-up functionality.

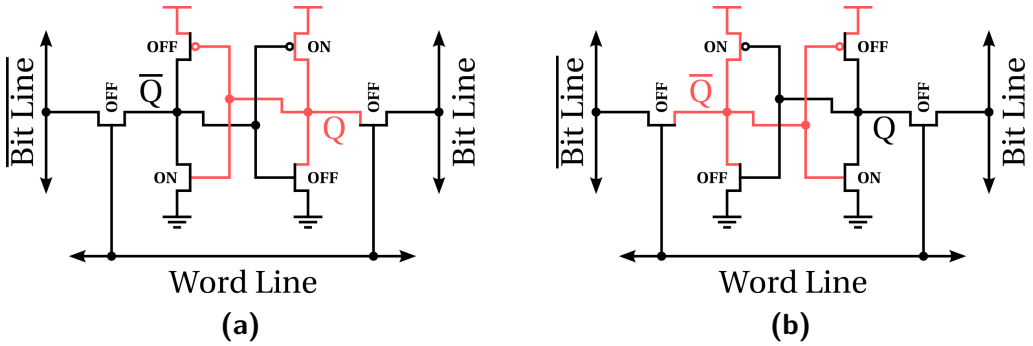


Figure 2.4: Depiction of a powered 6-transistor SRAM cell storing a logical value within its cross-coupled inverters as long as the system stays powered. While word line and bit lines are unpowered, the SRAM cell can hold two different values in a stable condition (a) or (b). Red wires indicate powered circuits and black wires indicate unpowered circuits.

2.1.4 SRAM - Static Random Access Memory

Static Random Access Memory (SRAM) is a volatile memory which stores bit information based on a bi-stable latching circuit. Plenty of SRAM cell design approaches are available in literature but de facto only few have made it into current devices. A four-transistor resistor cell is depicted in figure 2.3a. The area consumption of such cells is fairly low, but manufacturing requires a highly resistive polysilicon layer to be added to the CMOS process which offers the necessary pull-up resistor functionality. Due to the high resistance, the cell is also sensitive to noise and soft errors and access speed is slow. Furthermore, constant standby current of the NMOS pull-down transistors is a major drawback [69] which, in contrast, is negligible in transistor-only cells. Therefore, the resistors have been replaced by PMOS transistors and nowadays COTS SRAM devices mostly use cells composed of six to twelve conventional

CMOS transistors. FPGA vendor Xilinx for example has decided for 6-transistor cells within conventional COST devices as shown in figure 2.3b, as well as 12-transistor cells in its specialized radiation-hardened chips (see section 2.6.3). This increased complexity obviously requires extensive size and therefore adds a significant price value, but in turn provides additional benefits such as reduced power consumption, less leakage current, increased reliability or radiation hardness. More economic approaches for very high density SRAM devices are based on loadless 4-transistor cells as explained in references [70, 71].

The bit information within a 6-transistor SRAM cell is stored in two cross-coupled inverters composed of 4 transistors. Each inverter contributes to keep the other inverter's value alive. The two additional transistors are required to provide access control via the word line. This design does not make SRAM the densest storage architecture, since conventional DRAM can store even more bits on the same die size by using voltage level capacitors, but it enables very fast read and write operations of only few nanoseconds and even less in recent FinFET SRAMs [72]. Furthermore, there is no need for continuous refresh as known from DRAM and its capacitor current leakage.

As long as the signal strength of both Q and $\overline{Q}$ depicted in figures 2.4 does not exceed the threshold limit of the inverter system, the actually stored configuration value of an SRAM cell remains untouched, assuming the system stays powered. Vice versa, if both exceed this limit, the cell inverters get programmed with the recently set values. This usually is the case when the word line is selected and both bit lines are driven strong with the designated bit values during a cell write request. In case of a read request, both bit lines will be pre-charged weakly to accept the SRAM cell's currently stored values Q and $\overline{Q}$ as soon as the word line is selected. The concept of cross-coupled inverters had been chosen to reduce the overall susceptibility to signal noise. A failure within this system may lead to incorrectly stored states and therefore upsets within the SRAM cell. More information regarding such effects can be found in section 2.5.3.

Since SRAM is a performant memory type which can be easily manufactured in CMOS architecture, it has become a very prominent candidate in many of today's CMOS devices, such as Microprocessors from Intel, FPGAs from Xilinx and Altera/Intel or even standalone memory chips from Micron, Cypress and others. At least the 18-core Intel Xeon E5 v3 Haswell-EP server CPU contains 45 MB of SRAM memory to hold instructions and data within a fast on-chip L3 Last Level Cache (LLC). But recent research has indicated some problems with SRAM operation voltage, which

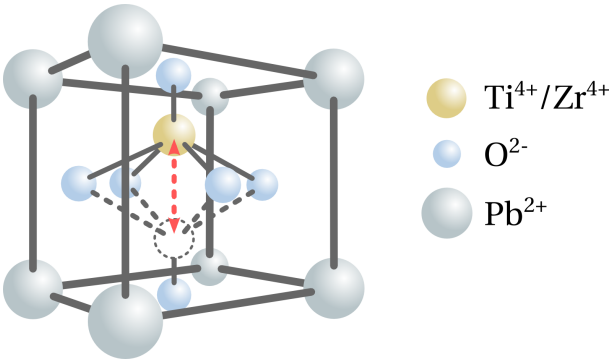


Figure 2.5: Dielectric PZT crystal of an FRAM cell's capacitor, according to [77]. To represent logic '0' and '1' states, mobile Zirconium (Zr) / Titan (Ti) atoms can be shifted between two semi-permanent electric dipoles within a Lead (Pb) / Oxygen (O) crystal lattice by applying an electric field.

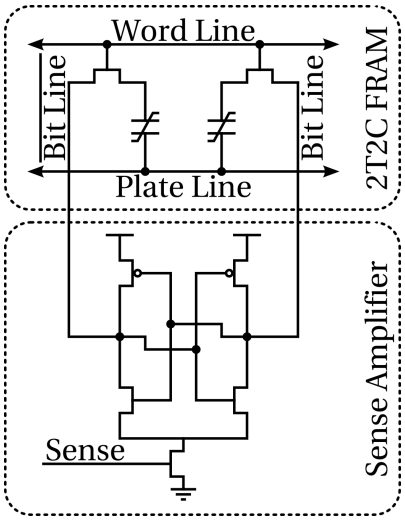


Figure 2.6: FRAM cell circuit composed of two transistors, 2 capacitors (2T2C) and an additional sense amplifier according to [78]. The cell is able to hold a single bit by two differentially inverted signals. Both bit lines are connected to a sense amplifier in common 4-transistor CMOS design, which is required to readout the cell by the use of cross-coupled inverters.

does not scale in the same way as logic structures do [73]. Furthermore, the increased die size of SRAM in comparison to conventional DRAM becomes a major issue, since on-chip memory demand is currently exploding and SRAM already occupies the biggest amount of CPU die [74] - according to Intel up to 50% [75]. Therefore, development has been taken towards an integration of so called embedded DRAM (eDRAM) paired with a conventional chip within a single package to fit at first recent memory needs in Intel Crystalwell GPU devices. More details comparing the different technologies can be found in [76].

2.1.5 FRAM - Ferroelectric Random Access Memory

Ferroelectric Random Access Memory (FRAM), also referenced in literature as FeRAM [79], is a non-volatile, low power memory with high read/write speed and endurance that is known for many years [80]. It stores bit information in semi-permanent electric dipoles formed within a dielectric crystal cell (dielectric con-

stant > 1000) by reversible spontaneous electrical polarization [77]. Each storage cell is composed of either two transistors and two capacitors (2T2C), which provide reliability by differential signals [81] as shown in figure 2.6, or one transistor and one capacitor (1T1C), which require less power and area [82]. The MOSFET is built in a conventional CMOS process, while the capacitor typically uses ferroelectric SBT ($SrBi_2Ta_2O_9$) or PZT ($Pb(Zr,Ti)O_3$) material as depicted in figure 2.5. At least PZT can easily be added to a conventional CMOS process by insertion of two additional mask layers between substrate contact and metal layer [77].

Writing or 'charging' an FRAM capacitor, despite of its current state, simply requires application of an electric field across the ferroelectric layer. Depending on the field polarity, the inner atoms will move to one of two semi-permanent electric dipole states and the corresponding logic representation will change to '1' or '0'. In contrast, reading a cell will force the inner atoms to either constant '0' or '1' (predefined). If the opposite state is currently stored within the cell, the subsequent dipole change will cause a short current pulse at the output, which is sensed by a connected latch-type amplifier. Therefore, reading a cell is 'draining' the current state. The memory controller has to ensure, that in case of a reading state inversion, the initial value has to be rewritten immediately, which obviously requires additional power. This behavior is well known from DRAM, but in contrast, an FRAM cell does not need to be continuously refreshed in standby mode and therefore saves a lot of power during operation.

The core PZT cell is announced to be insensitive against magnetic fields as well as irradiation and does not require radiation-critical charge pumps such as conventional flash memory. Neutron irradiation tests according to JEDEC JESD-89A have proven stability with only 0.051 FIT/Mbit for unpowered passive irradiation as well as 0.16 FIT/Mbit for dynamic readout in beam over 130 h in total [77]. Proton irradiation has been additionally investigated within this thesis' section 5.6.2.

FRAM is a prominent candidate to replace current Flash memory technology. It should also be considered in low power DRAM scenarios as soon as single cell size decreases, memory capacity increases and if speed does not have to be at the edge. Even conventional 6-transistor SRAM cells have been extended with ferroelectric capacitors to meet the requirement of speed *and* non-volatility [83, 84]. This results in 6T4C cells which are capable of performing an additional 'store to FRAM' command beside of the regular read/write operations at full SRAM speed. Finally, the current FRAM specifications shown in table 2.2 attest suitable competence with the additional benefit of lowest power consumption and non-volatility. Furthermore, FRAM

Specification	FRAM	Flash	DRAM	SRAM
data retention	non-volatile	non-volatile	volatile	volatile
cell structure	2T2C, 1T1C	1T	1T1C	6T
read time	110 ns	<120 ns	70 ns	1 ns [86]
write time	180 ns	1 s per sector	70 ns	<1 ns [86]
standby current	5 µA	5 µA	1000 µA	7 µA
read/write current	4 mA	12-35 mA	80 mA	40 mA
single bit read/write	yes	no	no	yes
max endurance	PZT: $10^8 - 10^{12}$ SBT: $> 10^{12}$	100.000	∞	∞

Table 2.2: Comparison of FRAM, Flash, DRAM and SRAM technologies according to [84] with minor updates where indicated.

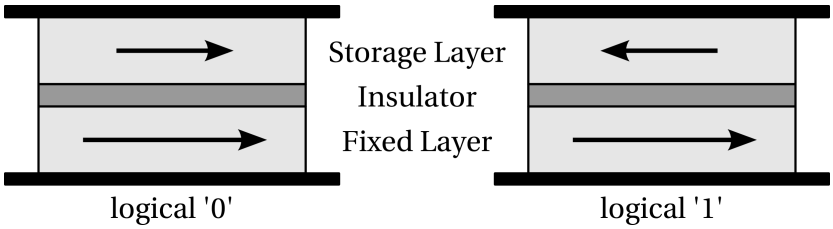


Figure 2.7: Ferromagnetic Layers of an MTJ cell. If fixed and storage layer are aligned in the same direction, a logical '0' is stored, otherwise, if they are directed opposite, a logical '1' is stored. The magnetization of the storage layer is lower in comparison to the fixed layer. Electrons are able to tunnel through the insulation layer.

may reduce or even eliminate the necessity for a combination of different memory types, such as DRAM and Flash, within a single device, which is a major step towards cost reduction. The manufacturing of larger cell arrays can provide additional benefit in the overall ratio of cells per uni area, as a size reduction of about 60% seems feasible only by chaining and amplifier reuse [85].

Current devices with plain memory arrays in BGA/TSOP package are available with a maximum size of 4 Mbit, for example the Cypress FM22L series. There are also few microcontrollers which feature up to 1 Mbit of FRAM, for example the Texas Instruments MSP430FR devices.

2.1.6 MRAM - Magnetoresistive Random Access Memory

Magnetoresistive Random Access Memory (MRAM) is a non-volatile, high endurance memory that stores bit information on the basis of electromagnetism instead of using conventional electric charge. Therefore, every storage cell is built of a Magnetic Tunnel Junction (MTJ) element, which in principle is a stack-up of two ferromagnetic layers, separated by a thin Synthetic Anti-Ferromagnetic (SAF) insulator. The magnetization of the first layer is statically charged and fixed, while for the second layer it can be varied between two stable positions and therefore be used to store a logical value as seen in figure 2.7. Since the insulation layer is only few nm in height, electrons are able to tunnel through it. Therefore, a cell-datum can be readout by utilization of the Giant Magnetoresistance (GMR) effect. This quantum mechanical effect causes the resistance value of the MTJ element to be different depending on the magnetic alignment of fixed and storage layer. Low resistance (logical '0') means that the magnetization of both layers share the same direction, while high resistance (logical '1') means that both layers are aligned opposite. Writing an MTJ cell means to re-define the storage layer's magnetic alignment by application of a short magnetic field toggle pulse. As it is only possible to pulse a swap of the magnetic field in conventional toggle MRAMs, a read operation is required before each write to determine the current cell condition.

Ongoing development in the field of MRAM technology lead to further advanced Spin-Transfer Torque (STT) MRAM. It is based on the quantum effect, that the magnetic orientation of an MTJ layer can be modified by applying a quantum spin-polarized current. Such current can be created by passing regular, unpolarized charge carriers, such as electrons, through a polarized magnetic field. Within the MTJ element, this is done by using the fixed magnetization layer. Afterwards, this so created spin-polarized current is able to interact with a second magnetic field of lesser magnitude to change its magnetic orientation. In MTJ elements, this is usually the storage layer. Reading an STT-MRAM cell-datum can be performed identically to a regular toggle MRAM cell, but writing requires application of spin-polarized current. STT-MRAM offers a better CMOS scalability than regular MRAM, but it still requires a relatively high current to reset the magnetic polarization of the storage layer.

Another technology which tries to solve this problem is the Thermally Assisted Switching (TAS) MRAM. In difference to STT, the MTJ gets temporarily heated by a short current pulse on write request to easily release the magnetic polarization of

the storage layer. The subsequent re-polarization requires less power in comparison to a conventional MRAM cell [87].

Since MRAM uses magnetism to store information, it is highly vulnerable to magnetic field interference. Therefore, MRAM devices have to be shielded against conventional field intensities by applying a copper plate at the bottom as well as a thick lid on top. Freescale MR2A16A is one of the devices following this recommendation.

MRAM offers some benefits when used in radiation environments, as the MTJ cell is unsusceptible against SEUs by design. Radiation tests with multiple 90 nm CMOS architecture Freescale MR2A16A 4 Mbit MRAM devices [88] have certified SEU immunity of the MRAM cell itself, but SELs had been observed at an LET of 7 $\text{MeV} \cdot \text{cm}^2 \cdot \text{mg}^{-1}$. Moreover, a TID limit of 45 krad had been determined before first errors occurred while irradiating with a ^{60}Co source. A second test used Everspin (formerly Freescale) MR0A08B 1 Mbit MRAM devices under test (DUT), manufactured in 130 nm CMOS architecture [89]. Featuring the same MTJ cell toggle mechanism, no SEL had been observed until the test ended with an LET of 84 $\text{MeV} \cdot \text{cm}^2 \cdot \text{mg}^{-1}$. TID functional failures started at 75 krad for Everspin PR0A08BCYS35 devices. A third irradiation test investigated a specifically radiation-hardened Everspin UT8MR2M8 16 Mbit MRAM DUT [90]. Due to this process, SEL LET could be increased from 17 $\text{MeV} \cdot \text{cm}^2 \cdot \text{mg}^{-1}$ to 112 $\text{MeV} \cdot \text{cm}^2 \cdot \text{mg}^{-1}$ and TID from 75 krad to 1 Mrad. First SEFIs occured at an LET of 29.5 $\text{MeV} \cdot \text{cm}^2 \cdot \text{mg}^{-1}$. Another test irradiated an unspecified 4 Mbit toggle MRAM for automotive applications, manufactured in 180 nm CMOS, with neutron particles up to 80 MeV as well as ^{252}Cf He^{2+} alpha particles according to JEDEC JESD89 [91]. The neutron fluence of 10^9 n/cm^2 over a time frame of 8 hours did not cause any SEU or SEL within the device and the α particles with a fluence of $1.7 \cdot 10^4$ s^{-1} did not indicate any SEU.

Due to the coherent requirement for non-volatile, high speed and high density System-on-Chip (SOC) devices, MRAM utilization for large-scale, reconfigurable memory arrays is inevitable. Therefore, next steps lead toward manufacturing of commercially available and well established FPGAs, which currently base on conventional Flash- or SRAM technology.

[92] shows first research towards this direction. A working micro-chip using MRAM to build up a basic LUT has been designed and manufactured in 350 nm CMOS technology. This work has been continued in [93, 94] to describe a novel TAS-MRAM FPGA architecture combined with conventional DRAM to benefit from the overall radiation hardness offered by the MTJ cells and the speed of DRAM. Therefore, it proposes to replace all configuration memory cells with MRAM cells,

whereas the memory cells and switching transistors themselves remain in DRAM. These DRAM cells are afterwards periodically refreshed from the MRAM cells located nearby. By utilizing this approach, it is possible to prevent SEUs from occurring in the configuration memory of an FPGA device. Subsequent prototypes with routing interconnects and LUT configurations have been designed in 130 nm CMOS process and successfully tested.

Since MRAM manufacturing can be easily combined with the well established CMOS process by placing the MTJ cells in about 5 additional masks above the regular CMOS structures, cell density is high, production yield is good and total costs are low. Due to its characteristics of non-volatility (10 years), low power, high endurance (10^6 to 10^{12} cycles), high density and improved switching speed (30 ns read) [92], MRAM is a promising candidate to replace conventional Flash- or SRAM-based memories, including FPGAs, while offering SEU radiation hardness by design.

2.2 EDA - Electronic Design Automation

In the early days of integrated circuit and printed circuit board design, hardware developers manually sketched and layouted signal paths as well as logical elements in a completely graphical manner before the first practically useful evolution in Electronic Design Automation (EDA) took place. These tools provided computer-aided design (CAD) for example to generate all required input data for the Gerber photoplotter machines of that time. The 1980's can be seen as birth of commercial EDA with a significant increase in professional routing algorithms and automated physical design tools due to the increasing number of transistors becoming available within a single chip [95]. This was also the time when major EDA companies such as Mentor Graphics® and Valid Logic Systems (now Cadence) were founded and major EDA conferences with industrial exhibitions have grown. Prominent examples are DAC (already established in 1964), ICCAD (established in 1983 [95]) and later EURO-DAC (1992-1997) [96] and DATE (established in 1998 [97]).

This success led to nowadays modern EDA tools, which are able to translate abstract hardware models, composed in standardized description languages, into a programming mask of highly modular hardware circuits and cells of a designated architecture (see section 2.1). These cells can afterwards represent variable logic functions, ranging from a basic AND gate up to a complex function with multiple inputs. The underlying technology has therefore become secondary and can even be exchanged while keeping the design description.

2.2.1 Hardware Description Languages

The success of EDA tools mostly depends on the availability of a standardized, powerful and commonly available technique to abstract and describe hardware circuits as well as logic primitives independently from the designated architecture or underlying platform. It should furthermore be human readable and offer structures to combine and automate recurring tasks. Preferably, it also supports libraries to conserve and reuse well established functionality. While the first automated tools were more or less specialized scripts to ease the development of in-house devices in different companies, the development of a first official standard for description, simulation and documentation of Very High Speed Integrated Circuits (VHSIC) started in 1980 [98]. It succeeded in 1987 with a first release of the VHSIC Hardware Description Language (VHDL) IEEE standard 1076-1987 [99] which had been revised later on by extending or removing syntactic and semantic constructs for IEEE standard 1076-1993 [100], adding protected types in 1076-2000 [101], changing buffer port rules in 1076-2002 [102] and adding major enhancements regarding functionality and usability in the current release 1076-2008 [103]. But even this latest version cannot be considered to be final, because an active IEEE standard has to be revised every 10 years according to the guidelines [104]. Since VHDL handles digital circuits only, IEEE standard 1076.1 [105, 106] has been created in parallel for analog and mixed signal (VHDL-AMS) circuits. It simply embeds VHDL as a subset. Beside of this strand, a second HDL, Verilog, had been created and opened to public in 1995 as IEEE standard 1364-1995 [107]. It has undergone similar improvements in standards 1364-2001 [108] or 1364-2005 [109] and is nowadays supported by the EDA tools in the same way as VHDL. As this work mostly focuses on VHDL, Verilog is only marginally referred in some of the following chapters.

All of these standards define the basic tasks of an HDL in providing a formal description syntax for electronic circuits that enables automatic analysis, processing and simulation by the use of EDA tools. This modeling can be done in different perspectives/domains: System behavior, component structure and device geometry. Each domain covers five levels of abstraction, ranging from the outermost abstract system level to the innermost detailed circuit level, as seen in figure 2.8. This famous Y-diagram, known as *Gajski-Diagram*, was created in 1983 by Daniel D. Gajski and Robert Kuhn [110] and had been further improved in 1985 by Robert Walker and Donald Thomas [111]. By following the rings from the outer to the inner level, it is possible to particularize a given hardware description until it reaches the required physical silicon layer of a digital device. While starting from a behavioral system

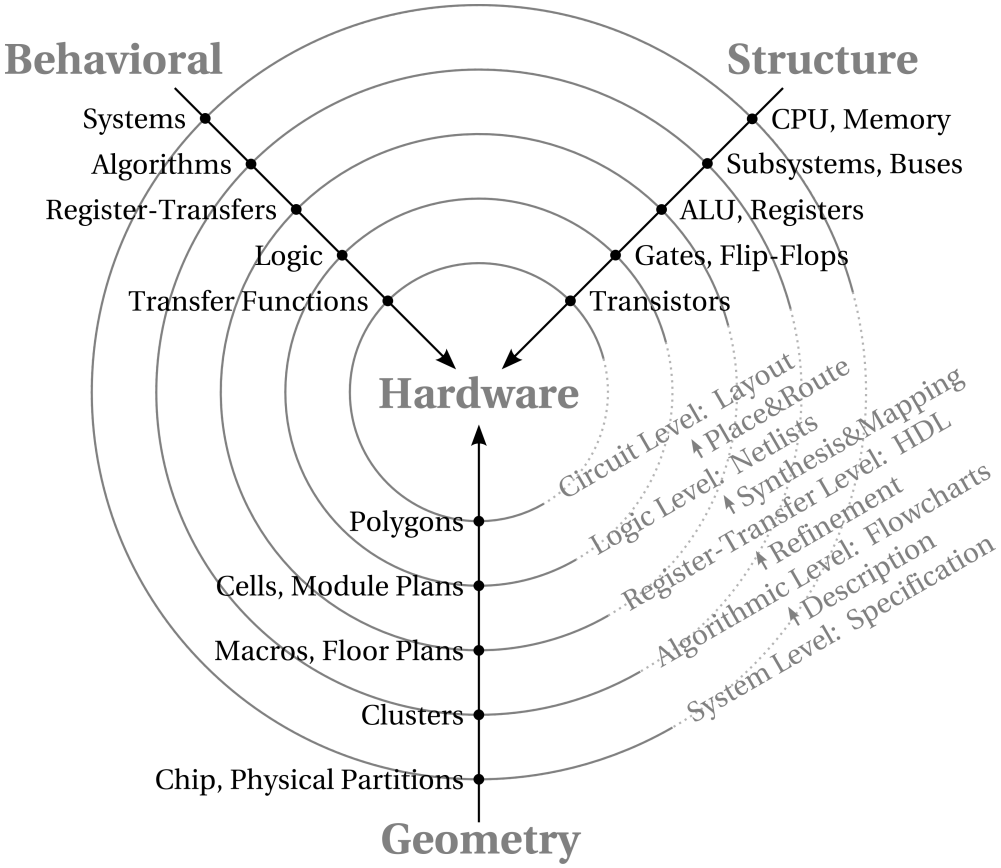


Figure 2.8: Y-Diagram, reflecting the three perspectives/domains of modern hardware design as well as the five different abstraction levels/rings, according to Gajski/Kuhn [110] and Walker/Thomas [111]. It has been extended by the general design tasks when proceeding from the outermost behavioral domain's system level to the innermost geometry's domain circuit level (light gray). Each outer level provides higher degree of abstraction than all inner levels.

level description, which represents a generally written or sketched system specification, the subsequent characterization step requires manual creation of basic algorithms on the algorithmic level. Refinement of these algorithms can be done on the register-transfer level by using VHDL or Verilog. If a high level language such as SystemC [112], MaxJ [113] or Xilinx HLS [114] has been used to define all algorithms, this step is skipped. Otherwise, the description needs to be done manually. The so created HDL code is afterwards automatically synthesized to device unspecific netlists and in a subsequent mapping step to device-specific netlists on the logic level. As this step requires preparation of the logic design by the use of device-specific primitives such as gates and flip-flops, the description domain switches to the structural

representation. Finally, a chip vendor or fab specific place and route tool automatically tries to fit all mapped elements into the target device's masks and layouts. This process is done in the geometry domain's circuit level as it requires exact knowledge of analog signal levels and timings.

While looking at this common procedure of designing hardware circuits, at least three of the five abstraction levels on the behavioral domain can be efficiently covered by VHDL [115]: The algorithmic level, as long as no circuit structures are enforced, the register-transfer level, which indicates the formal circuit structure by combining registers and adding timing signals, and finally the logic level, which describes a circuit design by a combination of boolean equations as well as timing information. More detailed abstraction levels are covered by the EDA tools. As a result, few knowledge about VHDL and a set of EDA tools are sufficient to describe hardware circuits. To furthermore ease the design process, VHDL supports the combination of multiple abstraction levels, for example the addition of boolean equations from the logic level to a description on register-transfer level.

2.2.2 Synthesis and Simulation

Both VHDL and Verilog language standards [103, 109] define a comprehensive amount of instructions for extensive logical circuit simulation, including language constructs for function calls and recursion. But only a subset of them can actually be used by EDA tools to synthesize netlists for physical hardware. Well known instructions that are suitable only for simulation purposes are for example the 'after' delay command in VHDL and the '$display' command in Verilog. Exactly the same situation is actually present for the VHDL-AMS and Verilog-AMS extensions due to missing EDA support. It should furthermore be added that EDA tools normally bring a whole tool chain to process specific tasks on independent abstraction levels as explained in section 2.2.1 and depicted in figure 2.8. On the one hand, this offers a possibility to analyze, simulate and maybe manipulate the generated output data of every single stage before it is forwarded to the next abstraction level. But on the other hand, this complicates the process of information transfer through the whole tool chain since every tool instance requires its own configuration options which may collide with previous ones and can finally result in unwanted behavior. Just to give a practical example, the registers which are fed from output signals designated to leave a device may work perfectly when defined and simulated on register-transfer level, but as soon as they are mapped to physical circuits, they can be placed all over the device, resulting in quasi non-deterministic output timing behavior be-

tween multiple translation runs. Therefore, a detailed mapping specification of the designated output buffer is required already on register-transfer level and has to be forwarded accordingly to the correct tool. To understand this process, the following section describes the tool chain used in this work from chip vendor Xilinx – the Integrated Synthesis Environment (ISE®) [116].

Starting from the behavioral description model given by statements and libraries in an HDL such as VHDL or Verilog (but not mixed), the Xilinx ISE® software generates a gate level netlist in the Electronic Design Interchange Format (EDIF) or as vendor-specific Native Generic Circuit (NGC) file [116]. This gate level description is at first independent from the underlying architecture and can therefore be shared and embedded across other designs, even between different vendors. In a second step, ISE® translates the netlist and merges user-given hardware constraints, such as timing or floorplan information, to a vendor-specific Native Generic Database (NGD) file. This file describes the whole logic design by the use of Xilinx primitives. The subsequent mapping cycle takes this NGD file and maps it into the physical device elements, such as gates or flip-flops. It creates another vendor-specific Native Circuit Description (NCD) file. This file now contains a list of how much input/output pins, flip-flops or embedded processors to take and how to generally connect them, for example: Take pin X and connect it to a D-type flip-flop, this flip-flop afterwards needs to be wired with a processor's overflow output, and so on. It does not contain location details about which flip-flops or wires are taken exactly from the designated device architecture. These decisions are made in the subsequent place and route design step where the tools are taking the current NCD file for input and add exact coordinates for all required component's within the chip layout. Therefore, an initial placement phase assumes a suitable allocation of all components, normally filling the device from the upper left area according to the entries taken from the netlist. The subsequent routing phase tries to wire this configuration whereas satisfying the given timing specification and user constraints. If no valid placement was assumed, the configuration is refined in multiple iterations or finally discarded. This processing can take multiple hours for larger designs which occupy most of a chip's surface, since the place and route computing problem is well known to be NP-complete [117]. Finally, the successfully routed design is converted by the bitgen tool into a binary, linear chain format (BIN) which is understood by the targeted device's programming interface.

Beside of the behavioral HDL description on register-transfer level, nearly all of the intermediately generated files from synthesis, mapping as well as place and

route can be individually used as input for internal or external logic simulation tools (ISim, ModelSim®, ...) to perform behavioral, post-synthesis or post-place&route timing simulation [116]. Particularly timing simulation for a successfully routed design can help to identify worst case temperature and aging scenarios [118] where even a working design may spontaneously fail because of additionally added delay in a critical signal path. All of these steps are an essential process for the validation of design and tooling results to identify and isolate design issues as soon as they appear. It allows immediate HDL adaption and synthesis re-initialization to finally keep an overall reasonable time to market.

2.2.3 Logic Device Time to Market

Due to the rapid development of integrated circuits and shrinking of the lithographic manufacturing processes, the number of transistors per device increased significantly, following Moore's law of doubling the number of transistors of an integrated circuit approximately every two years. EDA tools had to follow this evolution. While, in the beginning, it was quite easy to manually find an optimum solution for arrangement or placement of transistors and signal interconnects, nowadays this has become impossible even for CAD tools. Particularly when talking about the efficient allocation of primitives and wiring resources within FPGAs, multiple NP-hard routing problems such as the Traveling-Salesman-Problem (TSP) [119] and other NP-complete problems such as the path-length limited decision version of the TSP [120] have to be solved within a reasonable time frame. This effect causes a major drawback in classic EDA, since devices have grown faster than they could be effectively filled [121], finally reducing the overall time to market for these additional resources. As it is unforeseeable that these problems will be completely solved in time, even by the use of genetic algorithms [122] or cross-entropy methods [123], the currently followed optimization strategy is to separate the overall device into smaller areas which can be optimized individually and which are afterwards combined via well defined high speed interfaces. Reuse of already optimized and standardized, maybe embedded, circuit modules is also a viable option. It acts as basis for the proposed approach of using additionally available resources to decrease a system's overall susceptibility to soft errors by introducing fault tolerance while mitigating the time to market (see chapter 3).

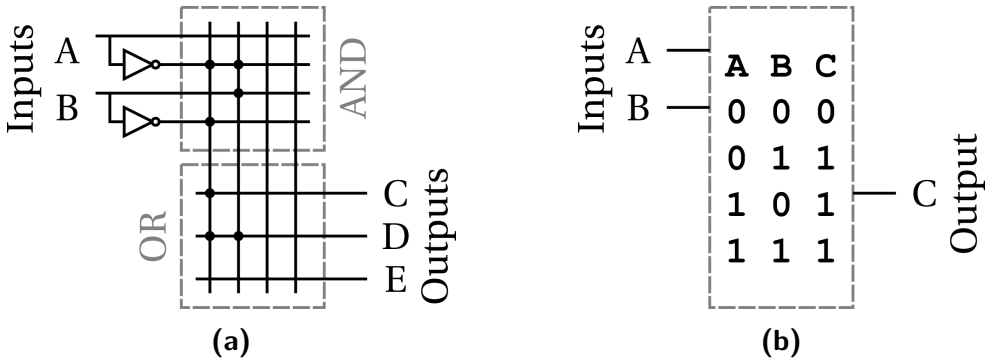


Figure 2.9: Simplified sketch of (a) Programmable Logic Array (PLA) and (b) Look-up Table (LUT). While the PLA can implement simple combinatorial logic functions by linking AND and OR arrays only, an LUT is able to represent any programmed logic function represented in a logic truth table.

2.2.4 Programmable Hardware Evolution

The history of programmable hardware goes back to 1958 with the invention of the first integrated circuit by Jack S. Kilby [124]. This milestone started a success of subsequent computer chips which got more compact, more powerful and more efficient, but also more complicated over time. Beside all technological benefits, this whole set of integrated circuits had one major characteristic in common: Logic functions and wiring were fixed. Even dynamic data within memory storage elements could only operate on the finally defined circuits. The first commercially available 4 bit Intel 4004 microprocessor chip [125] released in 1971 followed this principle and therefore could only implement a small set of given instructions [126]. This issue complicated the development of an optimal set of features within the fixed architecture's integrated circuit. Even today, recent microprocessor developers have to decide whether to implement a faster but more efficient *reduced instruction set* or a slower but widely functional *complete instruction set*. Nevertheless, these microcontrollers and microprocessors managed to be omnipresent in nearly everybody's daily life as the manufacturing process has massively scaled down and pricing became low, which finally enabled them to improve the technological benefit of a very large variety of devices in the COTS mass market.

Physically programmable hardware circuits are following a different strategy and have risen based on the well established integrated circuit technology. They mainly focus on the user-definition of interconnecting routing between these standard circuit elements to implement a designated logic function directly in hardware. Therefore, the first devices used a basic programmable AND gate array combined with a

programmable OR gate array to describe a simple binary output function as logical sum of signal input products within a so called Programmable Logic Array (PLA) as sketched in figure 2.9a. Unfortunately, these PLAs could only be programmed in factory during the chip manufacturing process by altering the metal layer since they did not include any programmable storage elements. Furthermore, they do not provide any feedback logic or storage elements to build up sequential logic.

Starting with the first non-volatile Programmable Read Only Memory (PROM), that offered one time programming by high voltage vaporization of metal contacts, storage of user-customized digital values, even after the chip packaging process, became possible. Moving forward to Erasable Programmable Read Only Memory (EPROM) in the 1970s and to Electrically Erasable Programmable Read Only Memory (EEPROM) in the 1980s added the possibility to erase previously stored values due to the application of ultraviolet radiation or electricity. This made the factory programmable memory field programmable. All of these technologies founded the basis for the first non-volatile Programmable Logic Devices (PLD) and later Complex Programmable Logic Devices (CPLD). Still today, CPLDs offer programmable AND/OR arrays to perform logic functions, but the main building blocks are macro cells. These macro cells can be programmed to perform disjunctive normal form combinatorial or sequential logic together with feedback connections and also flip-flops [127]. As the input to output signal delay of CPLDs is quite slow and nearly constant due to the AND/OR switching matrix and due to the limitations of the underlying internal routing resources, a completely new and complex class of PLD based on gate array technology had been developed and viably commercialized by Xilinx in 1985 [128]. This so called Field Programmable Gate Array (FPGA) device offered a dense grid of flip-flops and Look-up tables (LUTs) instead of PLAs (figure 2.9b). LUTs can be easily programmed by the use of truth tables to represent any logic function. This utilizes the available resources much better than in former PLA devices. A dense grid of configurable routing wires additionally ensures that every component can be easily reached by all the others. Nowadays, FPGAs contain many additional functional blocks, such as embedded memory, digital signal units and even processor cores beside of the regular resources. Due to the high amount of logic cells within recent FPGAs, the synthesis as well as the routing process of a hardware description language such as VHDL (see section 2.2.1) has become a challenging field of research. More information on FPGAs can be found in section 2.3.

At this point it is important to clearly distinguish between fixed microcontroller/microprocessor circuits operating sequentially on data that is stored in a

memory element and dynamic hardware circuits operating in parallel on data that is stored in LUT and routing memory elements. Up to now, combinations of both worlds are rare and working examples using co-processor cards can mostly be found in special applications such as High Performance Computing (HPC) for particle accelerators [129]. Another approach can be found in recent FPGA devices from Xilinx and Altera/Intel. The Xilinx Virtex-4 FX chip series embedded up to two separate IBM PowerPC 405 RISC processor blocks for 450 MHz operation [130] while the Virtex-5 FX series offered up to two IBM PowerPC 440 RISC processor blocks at 550 MHz [131]. Latest Xilinx Zynq as well as Altera/Intel Arria 10 devices have decided for embedded dual core ARM Cortex-A9 processor blocks for 1 GHz operation frequency [132] and 1.5 GHz [133] respectively. Recent development in the field of co-processor cards has focused on the integration of large scale static Graphics Processing Units (GPU). Therefore, from this point of view, programmable hardware evolution has just begun.

2.3 Field Programmable Gate Arrays

An Application Specific Integrated Circuit (ASIC) is the product of choice when it comes to mass production of highly efficient integrated circuits. A high device yield significantly compensates the development and manufacturing costs within the whole chain of ASIC silicon mask design, waver production, chip bonding and packaging. But due to construction, all device features are fixed within silicon and cannot be altered anymore, maybe except of some tentatively implemented tuning variables. This frequently leads to the disposal of whole chip batches if devices have not carefully been tested and simulated in advance or if the device functionality needs to be altered. In contrast, an FPGA does not possess such a disadvantage, as its logical elements as well as interconnections and therefore the logic functions can be altered even at runtime with a user-specific design. This easily allows to implement scenario-specific, user-designed hardware circuits without the overhead of ASIC production, but at the price of increased asset costs. For smaller projects with low quantities, the general use of FPGAs can even be a permanent option to keep production costs at a tenable level and if device performance or firmware updates during the product lifecycle are part of the Functional Specification Document (FSD). Prototyping of modern hardware components and interfaces nowadays requires high performance and configurable emulation systems which do not lack of flexibility as well as speed. Therefore, universally configurable arrays of memory cells as available in FPGAs based on SRAM technology (see section 2.1.4) became

very popular in this field of applications, for example CPU and interface emulation [134]. But an increasing cell density and therefore lower cost combined with the benefits of modern EDA (see section 2.2) summed up to make FPGAs more popular in many fields of modern information technology to solve sophisticated problems in various application scenarios. Performance improvements of about one to two orders of magnitude in comparison to conventional processor-based approaches can be achieved, which resulted in breakthroughs for many computational problems, such as DNA sequence matching, signal processing, emulation, and cryptography [135]. As a logical consequence, Intel Corporation recently announced a hybrid FPGA coprocessor embedded within a conventional server CPU that fits into a standard Xeon E5 LGA socket. This entails a significant improvement in comparison to its formerly announced Stellarton devices, which simply combined a consumer CPU and FPGA via an external PCIe lane [136], but follows exactly the same aim: The combination of configurable circuit cells and static structures to perform a varying and programmable set of functions, only dependent from the specific usage scenario as exemplified for an FPGA coprocessor LZ data compressor in [137]. A similar, but inverse approach is followed by the FPGA manufacturers themselves by embedding full micro-controllers (PowerPC/ARM) within their regular devices to benefit from the highly optimized instruction sets on the one side and save a great number of device resources on the other side. This feature was, among others, introduced as a lot of FPGA firmware designs embedded a soft core CPU component by occupying valuable logic cells. But processor cores are only an excerpt of the various possibilities coming with nowadays FPGAs. The following paragraphs about FPGA internals shall give a basic overview of how FPGAs are logically structured to provide a solid basis for later risk assessment and discussion.

2.3.1 Configuration, Routing and Logic Blocks

A classic standalone packaged FPGA semiconductor device from chip vendor Xilinx is depicted in figure 2.10. As clearly visible, it consists of multiple logical elements, which are interconnected by a chip wide Programmable Switching Matrix (PSM) made of Programmable Interconnect Points (PIP) [139]. Each PIP is a simple transistor that can be configured by an adjacent SRAM cell to connect or disconnect two wires [140]. The PSM is necessary to provide a flexible way for communication between all external and internal components. In addition, the globally distributed synchronous clock signals within the chip are routed via a second, dedicated timing network to guarantee minimal latency and skew and to prevent interference with

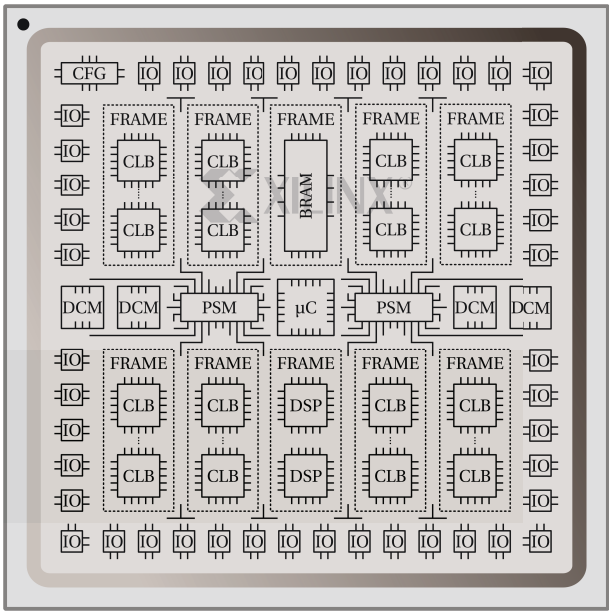


Figure 2.10: Simplified sketch of the logical components within a Xilinx FPGA, inspired by [138]. Two major configuration frame rows (upper half and lower half), containing general Configurable Logic Block (CLB), Block Random Access Memory (BRAM) or Digital Signal Processor (DSP) slices, are round up by several specialized entities, such as Distributed Clock Managers (DCM) / Mixed-Mode Clock Managers (MMCM) or Microcontrollers (PowerPC/ARM). A Configuration Interface (CFG) as well as Input/Output Buffers (IO) of different speeds allow device programming and data communication. All elements are connected by a dense Programmable Switch Matrix (PSM).

signal delays from the data logic. Finally, this global clock network is accompanied by several local clock regions, which allow the FPGA to synchronously run multiple clocks in multiple design regions in parallel. These clock nets are fed by specialized Distributed Clock Managers (DCM) or Mixed-Mode Clock Managers (MMCM) in more recent devices, both providing Phase Locked Loops (PLL), and are themselves connected to external clock pins. Hence, only the amount of SRAM configuration cells for the PSM represents a significant part of the whole device configuration bitstream. But this bitstream contains considerably more information - it is used to completely initialize all logical components of the FPGA and, at least for devices from chip vendor Xilinx, it offers multiple device-specific configuration commands for the FPGA-internal state machines and operational registers, for example a built-in CRC to find transmission errors during the configuration process [141]. The direct correlation of configuration bits and physical device primitives is not publicly available, but under investigation [142].

The configuration bitstream can also be read back from the device. Starting with series 7 FPGAs, the total configuration CRC as well as additional frame-based Error Correcting Codes (ECC) can be calculated during this process by the use of embedded ASICs to take notice of spontaneous modifications within the device [143]. This had to be done in FPGA firmware logic for elder device versions [144]. The physical

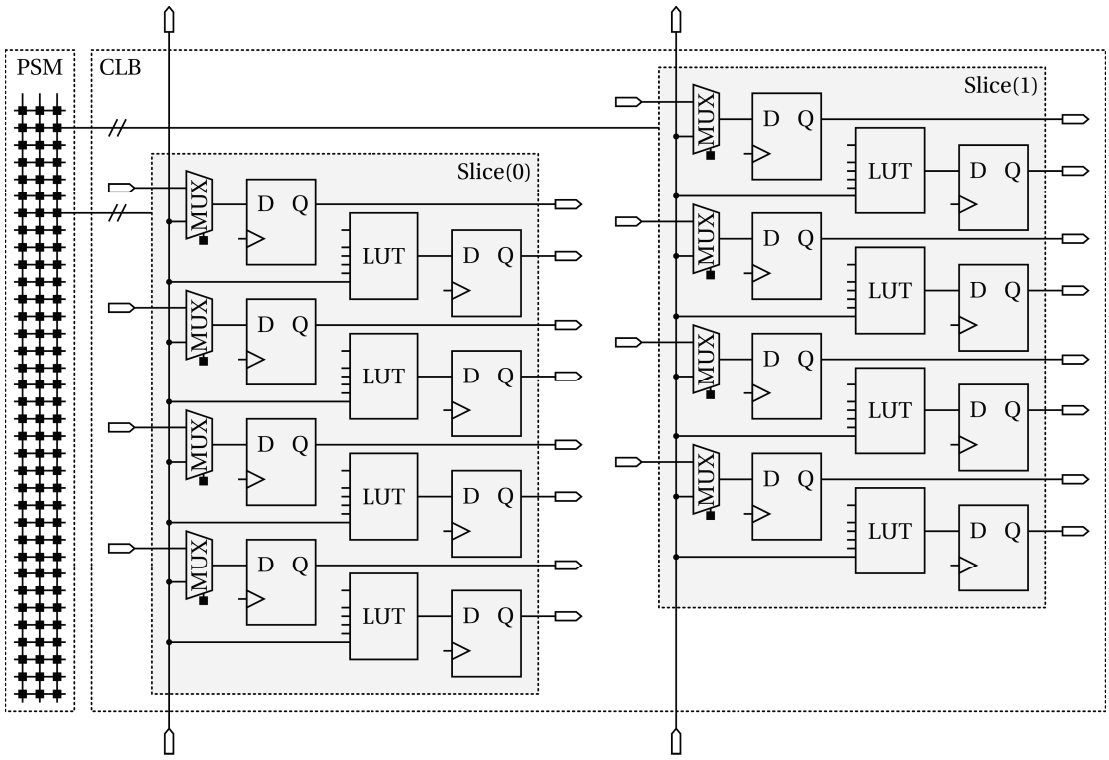


Figure 2.11: Simplified sketch of the logical components within a Xilinx Series-7 FPGA's Configurable Logic Block (CLB), based on [146]. Two slices are present within each CLB, composed of several basic device primitives, such as Multiplexers (MUX), Look-up Tables (LUT) or flip-flops which act as logic function generators and storage elements. Their number and availability differs between logic-optimized SLICELs and memory-optimized SLICEMs. All elements are connected to the dense Programmable Switch Matrix (PSM) and offer multiple configurable interfaces as well as carry logic.

FPGA resources themselves and hereby the bitstream programming data are organized in multiple frames of fixed length (started with Virtex-4 devices [145]), which are split into an upper and lower half of the device and which contain the initial configuration for Configurable Logic Blocks (CLB) or Block Random Access Memory (BRAM). A basic insight of such a CLB in Series-7 FPGAs is depicted in figure 2.11.

As defined for the very first Virtex devices [147] from FPGA-inventor Xilinx [148], a CLB consists of two slices which house the logic function generators, multiplexers, carry logic as well as the storage elements that are required to realize a clock synchronized circuit. While the first Virtex offered only two LUT and D-type flip-flops per slice [147], modern FPGAs such as the Virtex-7 feature four LUTs and eight flip-flops [146] to increase flexibility. The additionally implemented carry logic enables

an efficient interconnection between multiple CLBs by supporting basic arithmetic functions such as comparators or adders which cannot be handled in an embedded hard block Digital Signal Processor (DSP). Some of the CLB slices, called *SLICEL*, are available with logic function generators only, while others, so called *SLICEM*, offer additional distributed memory and 32 bit register data shifting options [146].

2.3.2 Boolean Function Generators

The boolean functions in FPGA designs are stored as truth tables in LUTs. Nowadays LUTs in Xilinx FPGAs consist of at least 6 independent input and two independent output signals [146]. This enables the EDA tools to easily realize any arbitrarily defined boolean function, starting from simple inverters and ending in complex terms with 6 different variables. The second output even enables the use of two five-input functions in a single LUT as long as the input variables are identical. Even more complex boolean functions with up to 8 inputs can be efficiently realized within a single slice by simply combining and multiplexing the outputs of all LUTs. The use of 9 and more variables requires a combination of multiple slices via the PSM. Calculated LUT-results can be directly fed into the slice D-type flip-flops to form clocked feedback loops or to be available for further processing on clock edges.

2.3.3 Flip-Flop Storage Elements

Beside of the SRAM cells (see section 2.1.4) used for LUT, MUX or PIP configuration, Xilinx FPGAs make use of additional flip-flops, directly embedded within the CMOS process as explained in section 2.1.3. In comparison to SRAM cells, flip-flops are occupying a larger area in silicon due to the significantly higher number of transistors, required to implement the extended set, reset and synchronization features. While configured in the same way as LUTs, but with initial and reset values, these flip-flops can be driven either as synchronously triggered D-type flip-flops (FD) or as asynchronously level-driven latches (LD) [146]. But especially the LD-type is very critical for usage in radiation environments as shown in section 2.5.4.1, since temporarily upset signals can be permanently acquired. All flip-flops within a slice have one thing in common: They can be collaboratively switched on or off to preserve current data even in synchronous logic [146] with changing input data, which is a convenient feature when building pipelines [149]. While a solely used flip-flop represents only a single bit register, the combination of all eight D-type flip-flops within a slice forms an 8 bit wide register. The supported combination with flip-flops from other slices even creates word lengths that are common for example in recent micro-

processors. If larger memories are necessary, chip vendor Xilinx provides specialized embedded storage elements as explained in the next section 2.3.4.

2.3.4 On-Chip Block and Distributed Memory

Using FPGA flip-flops to store larger amounts of plain data is certainly possible, but depletes the device's totally available resources very fast and would significantly impact the cost-benefit balance. Even a modern FPGA such as the Virtex-7 XC7V2000T features only 2.4 Million flip-flops in 152,700 CLBs [60]. Therefore, chip vendor Xilinx started to embed dense memory arrays within its FPGAs, so called Block Random Access Memory (BRAM). The physical organization of this memory is realized in configuration column frames as depicted in figure 2.10, similar to CLB frames. For easy handling, its initial configuration can be integrated into the conventional bitstream file by the use of external tools such as 'data2mem'. As BRAM occupies valuable area on the chip die, which otherwise could be used for CLBs, there are FPGAs available with less CLBs but more BRAM, such as the Virtex-7 XC7VX1140T that offers 67.7 Mbit of BRAM but only 89,000 CLBs [60]. As BRAM is an embedded hard-wired device primitive with plenty of ASICs under the hood, it can offer additional functionality such as a fully synchronous dual-port read/write interface [147], configurable storage word width or ECC functionality, which otherwise needs to be manually implemented when using basic flip-flops for storage. While the first feature is convenient for building efficient First In First Out (FIFO) data buffers with pseudo-parallel read and write operations, the latter one derives a significant benefit when using an FPGA in radiation environments where bits can be upset (see section 2.5.3). The BRAM ECC feature had been introduced with the Virtex-4 FPGA [150] and uses hard-wired circuits without the requirement for additional logic resources such as LUTs or flip-flops. Unfortunately, the ECC value for a designated data word is only calculated when it is read back. Therefore, random error accumulation within an embedded memory block is not sufficiently safeguarded.

If flip-flop storage and BRAM are both insufficient to fulfill an application's memory requirements, Xilinx FPGAs additionally offer, as previously mentioned, the possibility to configure the LUTs of SLICEMs as distributed, synchronous RAM. But given that a CLB in modern FPGAs contains either two SLICELs (0 LUT RAMs) or one of each SLICEL and SLICEM (4 LUT RAMs) [146], the total amount of such elements is limited. Therefore, even one of the biggest Virtex-7 FPGAs, the XC7V2000T, provides only 21.6 Mbit of distributed memory [60]. The advantage of LUT RAM

is that its size and the number of access ports can easily be varied by combining multiple SLICEM LUTs to configurations such as a $32 \cdot 1$ bit single-port, a $64 \cdot 1$ bit quad-port, a $32 \cdot 6$ bit dual-port, or to a $256 \cdot 1$ bit single-port RAM [146]. But the biggest disadvantage of LUT RAM can be found in the kind of storage itself. Since LUT RAM does not have native ECC support and since a configuration refresh would destroy its dynamic content if not used in ROM mode (see sections 4.2 and 4.1), it is not recommended to use distributed LUT RAM in radiation environments.

2.3.5 Input/Output Buffers

To retain a wide pin compatibility with different kinds of auxiliary components and interfaces available for modern PCBs, Xilinx FPGAs support multiple input and output voltages in parallel. They range from 1.0 V to 3.3 V for modern devices at features sizes of only few nm [151] and are bundled in independent voltage banks. Therefore, it is easy to connect nearly every component that follows the electrical signaling standards. The biggest chip package available for Series-7 offers about 1900 connection pins, whereas about 1200 are available for General Purpose Input/Output (GPIO) as well as high speed MGT/GTP/GTX Gigabit Transceivers. To ensure correct input data sampling and stable data output, Input/Output Buffers (IOB) are integrated within the FPGA silicon chip's I/O cells. Additional features include passive pull-up or pull-down resistance, tri-state handling, output slew rate control, and input delay switching. Series-7 even includes four shallow FIFOs located in each of the I/O banks [152]. This increases complexity and functionality but also adds more and more CMOS transistors to the IOBs, which in turn are impacted by radiation.

2.3.6 Embedded Hard Blocks

Beside of the CLBs, there can be found several hard-wired logic components in modern FPGAs, that are directly manufactured as ASICs to cover specialized tasks more efficient than plain user logic could do or even enable functionality which cannot be implemented with conventional logic cells, such as high speed Gigabit Transceivers. This increases not only calculation speed and reduces power consumption, it also preserves valuable logic cells for parallelizable tasks. The previously mentioned DCMs and BRAMs are only few examples for embedded macro blocks as seen in figure 2.10. Ethernet MAC interfaces, PCI-Express endpoints and not to forget DSPs are further available hard-cores. Especially DSPs are very prominent to realize fully parallel signal processing algorithms. They are organized in

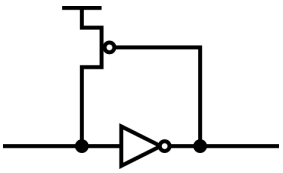


Figure 2.12: Simplified schematic of a Xilinx Half-Latch PMOS circuit used to locally provide a weak pull-up logic signal to uninitialized primitives within former Virtex FPGAs.

slice columns within the FPGA, similar to CLBs and BRAMs, whereas two DSP slices match the height of five CLBs or one BRAM [153]. Just to give an example, the Virtex-7 XC7VX980T offers 3600 DSP slices whereas each one contains a pre-adder, a $25 \cdot 18$ bit multiplier, an adder and a 48 bit accumulator element [60] to efficiently perform arithmetic, shift, multiplex, compare, logic, pattern detection, and counter operations [153]. It even supports cascading of multiple DSPs for complex arithmetic functions. As DSPs entail multiple storage cells for arithmetic operands and calculation results, for example in the accumulator, they are susceptible to radiation effects, but handling seems quite difficult since the fixed logic circuits cannot be altered anymore.

2.3.7 Half-Latches

The first Xilinx Virtex FPGAs utilized a simple mechanism to constantly feed the input pins of its various device primitives with stable logic '0' or '1' signals and thus prevent floating in case they had not been initialized within the user design. This mechanism also known from CoolRunner devices was realized by an inverter in conjunction with a PMOS transistor and is called 'half-latch' or 'keeper' [154], depicted in figure 2.12. If the half-latch has been initialized with a logic '1', the corresponding pull-up drives a constant value. But this signal possesses a weak character only, since it can be easily overwritten by every other connected signal which satisfies the transistor's switching requirements. Therefore, a half-latch acts completely transparent in case of a connected signal and furthermore always keeps its latest value. Due to this method, Xilinx was able to prevent the occupation of expensive LUT just for configuration of unused logic. Half-latches are spread across the whole device, as they are locally required by inputs of various primitives, such as CLBs and IOBs. This optimizes the overall routing network, keeps wires short and hence the implementation tools make heavy use of it.

Unfortunately, half-latches are susceptible to radiation effects, such as logic upsets and transients. Especially those which drive the inputs of multiplexers have been denoted as the most critical ones [155], as modifications can directly impact the routing behavior of a logic design. But given that half-latches can be initialized

by a full device configuration only and direct manipulation is currently not supported [156], error detection via read-back as well as error correction via dynamic partial reconfiguration are impossible. Therefore, the half-latch usage in error-prone applications has been ranked critical and mitigation tools such as XTMRTool (see section 2.9.2) are removing half-latches from the final bitstreams. A circuit-redesign of the Xilinx Virtex FPGA series, subsequent to Virtex-II, dramatically reduced this susceptibility issue and therefore, recent devices do not require half-latch extraction anymore [157].

2.4 Ionizing Radiation

An electrically neutral atom consisting of an equal number of protons and electrons is able to gain or loose electrons due to excitation and therefore gets electrically charged – ionized. Excitation shifts electrons to a higher energetic level while ionization ejects them from their current position by the use of Coulomb attraction, generating freely available negative charge carriers while leaving positive holes at the former positions. This process of electron-hole pair generation will be described in section 2.4.2. The ionization process can either result in a positively charged cation or negatively charged anion in all phases of matter and antimatter. A highly ionized particle mixture with ions and the separated electrons accordingly forms a plasma phase.

Radiation that is able to ionize matter (ionizing radiation) therefore carries sufficient energy to remove electrons from atoms and therefore create a dense plasma which is able to cause various effects within matter, for example within semiconductor circuits (see section 2.5). Since ionizing radiation can be caused by multiple sources, it can be clearly distinguished between:

- γ, X-ray and higher ultraviolet photon irradiation which starts at few nanometer wavelength of the electromagnetic spectrum and causes direct ionization
- α, β (electron, positron), proton, heavy ion and other subatomic and secondary particle irradiation which causes direct ionization
- neutron irradiation that causes indirect effects by generated secondary particle ionization

The following sections now aim to give some basic information regarding charge generation due to excitation and ionization processes as well as the hereby transferred energy to the affected material. In addition, some sources of radiation are examined to clarify their relevance for semiconductor usage in today's various application scenarios.

2.4.1 Passage of Radiation through Matter

Photon as well as particle irradiation traversing matter are interacting with the target material in different ways, depending on radiation type, energy as well as the affected material itself. This happens either directly by excitation and ionization in case of charged particles and electromagnetically interacting photons or indirectly in case of nuclear effects.

Electrically neutral photons with only few nanometer wavelength and below, namely γ, X-ray and higher ultraviolet photons, are able to directly interact with matter by multiple electromagnetic processes. They vary with incident angle as well as energy. Low energy photons below the electron binding energy underlie the *photoelectric absorption* effect. Hereby, the photon energy is completely transferred to an electron while eliminating the photon itself [158]. The interaction type for medium energy photons is the *Compton effect*, which describes the energy transfer from photon to electron while increasing the wavelength of the photon and scattering it to a new direction for further interactions [159]. The excited electron from both types of interactions can subsequently be ejected from its position, generating an electron-hole pair (see section 2.4.2). Finally, for higher photon energies above 30 MeV, the primary interaction with matter is *pair production* [160]. Hereby, the penetrating photon within the electromagnetic field of the nucleus is converted to a particle-antiparticle pair, most likely electron and positron [160]. Both in turn interact with matter while losing energy and finally collide in an annihilation process while emitting new secondary photons with lower energies.

Alpha particles are simple Helium nucleons consisting of two protons and two neutrons without any orbital electrons. They are emitted in nature, for example, by unstable heavy nuclei isotopes such as ^{210}Po, ^{241}Am or ^{242}Cm [161]. While being two times positively charged, α particles exert considerable ionization impact on a traversed target material. They interact with either nuclei by deflection and scattering effects or with electrons by excitation and ionization effects [158]. This interaction slows down the α particle along its path until it is either leaving the target material at lower energy or getting completely stuck while finally capturing two electrons to form a conventional helium atom. Low energy α particles have a short penetration depth only and are usually created from radioactive decaying processes. Therefore, a simple sheet of paper can easily stop them [158]. In contrast, high energy α particles can mostly be found in cosmic radiation (see section 2.4.4) and they are able to travel for example 1.7 m in silicon at 1 GeV [162]. Given this background, the quickly passing high energy particles are generating less ionization damage in a critical volume

than the low energy ones [158]. This is especially critical as soon as a long-living, low-energy α emitter is deposited directly within the irradiated volume itself.

Beta particles are high energy β^- electrons or β^+ positrons, mostly generated by radioactive beta decay, for example from ^{40}K. They can interact with either nuclei by deflection effects or with electrons by excitation and ionization effects [158]. Deflection of electrons along a nuclei's Coulomb fields slowly degrades their energy while emitting electromagnetic radiation (X-ray photons), known as Bremsstrahlung [161], which subsequently contributes to the total ionization effect. In contrast to α radiation, β^- particles underlie Coulomb repulsion when passing other electrons. At sufficiently high energy, these encounters frequently lead to ionization [158], which results in slowed down β^- particles, ejected electrons as well as the remaining electron holes. As soon as a β^- particle spent all of its energy, it gets stuck by combining with a positively charged ion or remains a free electron [158]. In contrast, β^+ positrons will most likely combine with electrons while emitting secondary photons as mentioned previously in this section.

Positively charged protons p^+ as well as heavy ions with more than two protons that traverse a target material are able to directly interact with either nucleons by fission, deflection and scattering effects or with electrons by excitation and ionization effects in the same way as α particles but on a different scale. These processes will slow down the particles until they leave the target material or get stuck after losing sufficient energy. In the last case, they capture local electrons and form additional, neutral atoms within the material, for example hydrogen in case of protons. Impacting particles are furthermore able to fill local atom vacancies within crystal lattices created by scattering effects as described in section 2.5.5.1. Highly charged ions at high energies with at least three protons but without any electrons are referred to as High atomic number Z and Energy (HZE) ions [163] – they can predominantly be found in GCRs but only with rare probability.

Neutrons are electrically neutral. Therefore, they are unable to use Coulomb interaction to directly ionize atoms. But along with all other particles traversing matter, they have a specific probability to collide with electrons or nuclei while slowing down. In case of hydrogen atoms, present in a penetrated matter, a neutron collision can generate secondary, high energy recoil protons [158], which subsequently contribute to the ionization process by direct interaction with the target material as explained above. This process certainly requires a sufficiently high neutron energy. In case of bigger nuclei collisions, neutrons interact by elastic and inelastic scattering effects.

Detailed information regarding the passage of particles through matter can be found in [164], pages 398 ff. The corresponding cross-sections are available in [165].

2.4.2 Electron-Hole Pair Generation

The complex quantum mechanics explaining the conductivity in crystal lattices [166] and hence semiconductors can be simplified by using an electron band model [167]. This model basically describes the energy levels of atoms and their charge carriers, merged into two bands of valence and conduction as well as a separating band gap in between both. The valence band represents charge carriers that are bound to lattice atoms while the conduction band is characterized by freely available fast electrons. Fully occupied bands do not contribute to the overall conductivity, as they do not offer mobile charge carriers, therefore the electrons initially have to be moved from the valence band to the conduction band. This requires the excitation of an atom either by increasing its temperature, applying a sufficiently high electric field or by application of irradiation. Each shifted electron leaves a hole within the valence band which itself contributes to the overall conductivity by providing a vacancy to a nearby electron. It hereby slowly moves by electron hopping opposed to the direction of the conduction band's electrons. But this process of electron-hole pair generation requires excitation of the material dependent band gap energy. This energy is about >4 eV for insulators but only 1-4 eV for semiconductors, due to the artificial population with donators (n-doping) or acceptors (p-doping) within the energy range of the band gap. Donators are embedding excessive, negative electrons which are able to easily move to the conduction band on only little excitation, while acceptors are embedding positive holes, which capture electrons from the valence band and thus generate electron holes in it. Both doping processes enhance charge carrier generation, but also increase the susceptibility to ionizing radiation.

This ionizing radiation uses the mechanisms of energy transfer explained in section 2.4.1 to excite atoms within an irradiated material. This again shifts electrons from the valence band to the conduction band and therefore additionally boosts the electron-hole pair generation of mobile carriers. The external energy, required to create a single electron-hole pair in a silicon substrate is 3.61 eV ± 0.01 for α particle radiation [168, 169, 170] and 3.79 eV ± 0.01 for β electron radiation [168]. Since both values are temperature dependent, they are given for 300 Kelvin [171]. For photon, electron and proton radiation in silicon dioxide material as used in MOS devices, the amount of energy that is necessary to create an electron-hole pair is given with 17 eV

[170]. According to these low energy values, a highly charged photon or particle "can create thousands (up to millions) of electron-hole pairs" [170].

Both separated mobile charge carriers try to recombine immediately to bring the whole system back into equilibrium, but this process is effectively hindered by strong local electric fields as they are present in powered MOS devices. As soon as an electron from the conduction band loses enough energy to cross the band gap, it directly re-occupies a vacancy in the valence band by *radiative recombination* while emitting a new photon with at least the energy of the band gap. In the special case of doped semiconductors, the artificially added energy bands act as intermediate stages for an indirect gradual recombination of electrons and holes, since they require less energy loss for occupation. Therefore, doped atoms form recombination centers within the material [172]. This radiation-free *Shockley–Read-Hall recombination* [173] emits only thermal energy in form of lattice vibration / quantum mechanics phonons. It is not the only radiation-free energy transfer. The energy loss while moving from conduction to valence band, can also be completely transferred to another electron from the conduction band by direct *Auger recombination* [173]. This enables the electron to leave the crystal lattice and interact with other atoms or relax at its position by emitting lattice vibration. Detailed information regarding recombination can be found in [174].

2.4.3 LET - Linear Energy Transfer

As depicted in section 2.4.2, the generation of a single electron-hole pair requires a threshold energy to cross the material dependent band gap. This energy is lost by the ionizing particle along its path through the material – the bigger an ion's HZE or the denser the material, the more energy is transferred and the more electron-hole pairs are generated. This relation is proportional and defined for a single particle as Linear Energy Transfer (LET). It describes the amount of deposited energy within a material in dependence of the ionization energy, the material's density, as well as the distance traveled. It can be calculated by the formula:

$$\mathrm{LET} = -\frac{dE}{dx} \cdot \frac{1}{\rho} \left[\frac{\mathrm{MeV} \cdot \mathrm{cm}^2}{\mathrm{mg}} \right]$$

where $\frac{dE}{dx}$ is the stopping power, determined by the Bethe-formula which returns the ionization energy loss of a charged particle in matter per distance in $\left[\mathrm{MeV} \cdot \mathrm{cm}^{-1}\right]$ [169] (except of small mass electrons), and ρ is the target material's density in $\left[\mathrm{g} \cdot \mathrm{cm}^{-3}\right]$. The negative sign results from the energy loss as a matter of principle.

Since this differential calculation is nontrivial due to the continuous deceleration of particles within matter, several tools such as the LET Calculator [162], SRIM & TRIM [175] or FLUKA [176] provide simulation results after specification of material layers, particle and energy. To give an example: An LET of 100 MeV·cm^2·mg^{-1} within silicon generates electron-hole pairs equivalent to Q = 1 pC·µm^{-1} [177]. As not every single electron-hole pair separation shows an effect in MOS devices, a certain threshold value LET_{th} has to be exceeded before a critical charge Q_{crit} within the transistors can be accumulated and logical failures start to occur as explained in section 2.5.

It is important to note that with increasing energy of a specific ion, the stopping power and therefore the effective LET decreases [178]. In consequence, the stopping power increases as long as the ion is slowed down within the material. As soon as its energy is nearly depleted, it shows a characteristic maximum in energy-loss, the so called Bragg-Peak, and is afterwards captured completely. This effect is used for medical carbon ionization therapy as well as for radiation testing with the Variable Depth Bragg Peak (VDBP) method [179] which eliminates the requirement of device thinning for weaker particle test beam energies (see section 5.1).

Another consequence is, that for all objects which are not shaped like a perfect sphere, the radiation incident angle plays a major role for charge deposition, as it can lengthen or shorten the path of the ionizing particle and hence affect the effective LET. This has been demonstrated for Xilinx FPGAs to have cosine dependence for the tilt angle [64].

2.4.4 Radiation from Solar Flares and Galactic Cosmic Rays

One of the eldest and permanently present sources of ionizing radiation within the known universe, that is able to impact electronic devices in space as well as on Earth, is the Galactic Cosmic Ray (GCR) [180]. It is primarily composed of various higher atomic nuclei heavy ions such as *C*, *O* or *Fe* (1%), electrons (3%), alpha particle *He* nuclei with 2 protons and 2 neutrons (13%), as well as protons (83%) [181] with the highest energy of 10^{15} eV [182] to 10^{20} eV [183] ever observed. It has recently been expected to originate and to be accelerated from white-dwarf stars' supernovae explosions [184]. A detailed distribution of particles, energies and fluxes for the GCR can be found in [170] and [181]. Earth's atmosphere as well as the Van Allen belt capture and split some of these particles, but due to the high energies, which are currently impossible to shield in space, they present a significant risk to human and electronic health in space transportation and satellite applications.

In addition, one of the biggest sources of ionizing radiation within the Sol system is the sun itself. As it follows a regular cycle of minima and maxima, extreme space weather conditions can be observed about every 11 years [163]. Due to the fast moving electromagnetic fields on the sun's surface, electron rich solar flares are released and circular solar prominences are formed and erupted in Coronal Mass Ejections (CME). The latter emits a vast amount of X-ray photons, > 10 MeV protons (90-95%) as well as fewer rates of alpha particles and other heavy ions [170], summed up to a total mass of about 10^{15} g to 10^{17} g per CME [181]. At least the proton concentration exceeds GCR values by a factor of about 10^4 while heavy ions cannot reach more than 50% [170]. As soon as the X-ray radiation arrives at Earth, it progressively ionizes the ionosphere, resulting in limited short-wave radio communication, global radio noise or statistically increased cellular connection losses [185]. Protons as well as other plasmas are diverted by the magnetosphere to the poles and increase the local atmospheric ionization in about 80 km altitude, visible as well known auroras. In case of bigger solar events with proton energies above 500 MeV, this effect can even be measured at sea level as Ground Level Event (GLE). In addition, the solar plasma is able to distort Earth's magnetic field by causing magnetic storms. This effect deforms the magnetosphere towards Earth and shifts at least the Geostationary Orbit (GEO) (35,786 km altitude), including all geostationary satellites, into a higher radiation area as well as increases radiation doses for all lower orbit objects, including the Low Earth Orbit (LEO).

In consequence, even applications without a requirement for radiation safety can be influenced by GCR and solar events. Recent history has shown that major power outages due to geomagnetic storms with induction in transmission lines, the increased aging of solar panels [186] as well as the precision of navigation systems can all be attributed to radiation events in space. Therefore, space weather prediction for planes, satellites and space stations has become a major challenge in today's technological age to enable proper reaction on solar events. More information regarding space weather can be found in [187].

2.4.5 Radiation in Low Earth Orbit

LEO denotes an altitude of about 160 to 2000 km where nearly all of nowadays manned space flights take place and where the International Space Station (ISS) is currently located (400 km). Radiation within these orbits is characterized by trapped electrons, protons, and few low energy heavy ions [2]. Since LEOs are penetrated by the inner Van Allen radiation belt, they are dominated by trapped high energy

proton energy	altitude (60 ° inclination)	
[>MeV]	300 km LEO	500 km LEO
1	$2.056 \cdot 10^7$	$5.126 \cdot 10^7$
10	$1.191 \cdot 10^6$	$4.774 \cdot 10^6$
100	$2.276 \cdot 10^5$	$1.279 \cdot 10^6$
500	$7.257 \cdot 10^2$	$1.618 \cdot 10^4$

Table 2.3: Comparison of trapped proton energies and fluxes in different LEOs at a solar minimum cycle. Excerpt from table 1 in [2].

protons at densities of 10^3 to 10^5 cm^{-3} particles [187] that constantly interact with electronic equipment when passing by. Trapped proton energies in LEO are spread over a wide spectrum, starting with only a few keV and going up to 500 MeV or even more, while the number of low energy protons is always higher than the number of high energy protons [2]. Table 2.3 gives a basic example of what can be expected in LEO. A detailed table of trapped proton and electron fluxes as well as their total doses in LEO can be found in [2]. Particular attention should be paid to SAA transits, where the Van Allen belts approach closer to Earth's surface due to the 11° angular tilt of the geomagnetic dipole axis and the 500 km translational shift towards the Western Pacific Ocean [2]. Locations nearby this area intensify the radiation impact on electronic components, even for radiation-hardened devices [3, 4].

2.4.6 Terrestrial Radiation on Earth

Terrestrial radiation is a composition of different surface background, atmospheric interactions as well as aftermaths from nuclear events in human history. Its highest extent can be observed in mountainous regions, while at sea level, the rates drop significantly to about one tenth [188].

Natural surface background is predominantly dependent from presence of the radionuclides ^{232}Th thorium, ^{238}U uranium and ^{40}K potassium. They emit α and β particles as well as γ photons while decaying to other products, especially ^{222}Rn radon gas, and therefore cause ionization effects in biological systems as well as semiconductors. But in contrast to the widely spread cosmic radiation, its occurrence is locally constraint and can easily be mitigated by avoidance of such materials and gases.

Atmospheric radiation is caused by interaction of cosmic neutrons with gas molecules (heavy ions) within the upper layers of the atmosphere and the corresponding decaying processes. As the geomagnetic field shields Earth's surface from GCR and solar events, the radiation spectrum is mostly characterized by locally generated neutrons, X-rays and γ photons as well as electromagnetic radiation. Radiation levels vary with altitude and latitude because of the curved geomagnetic field lines and therefore show a minimum at equatorial regions while reaching the largest extent nearby the geomagnetic poles [189] as well as a significant increase at the SAA. The collisions within the atmosphere generate a wide range of direct and secondary particle showers from basically three major groups: meson showers, electromagnetic showers and nucleon cascades [190]. While unstable muons may reach the sea level but immediately decay, only electrons and positrons, which generate Bremsstrahlung due to their pair production (see section 2.4.1), as well as protons and neutrons, which directly and indirectly impact semiconductors (see section 2.5.5.1), have to be considered. The amount of neutrons with a reasonable LET exhibits the highest flux values at sea level (100-300 $m^{-2} \cdot s^{-1}$ in total [188], including 42 $m^{-2} \cdot s^{-1}$ >20 MeV [191]) and therefore implies the biggest cosmic radiation impact for semiconductors on Earth [192]. The Boron ^{10}B isotope for example, widely used as p-type dopant in silicon, gets unstable when irradiated with atmospheric or artificially generated neutrons since its neutron capture cross-section is 3-7 orders of magnitude higher than other applied isotopes [192]. It decays to 0.840 MeV to 1.014 MeV ^{7}Li atom, a 1.47 MeV ^{4}He alpha particle as well as a γ photon [192] which subsequently can interact within the semiconductor's material. As this process takes place directly within the silicon, it cannot be shielded in any way. In consequence, the Xilinx Virtex-7 FPGAs for example use only highly purified boron to prevent the artificial addition of device susceptibility. Similar effects can be observed from other decaying impurities in different ceramic packing [193] and molding compounds.

Since nuclear detonation experiments with radioactive fallout within Earth's atmosphere have ended, there is no significant global contribution to the terrestrial background radiation anymore. Only nuclear disasters such as Chernobyl (1986), Three Mile Island (1979) or Fukushima (2011) add locally constrained radiation constants to the overall values.

Summarizing all of these background effects, the places with the lowest terrestrial radiation can be found nearby coastal areas in equatorial regions without any sedimentary radiation burdens and far away from atomic disasters and the SAA. Any other regions suffer from the previously mentioned effects. Even particle accelera-

tors have to deal with atmospheric effects from cosmic radiation, since it can cause false positives within the highly sensitive detector arrays.

2.4.7 Radiation in Particle Accelerators

Similar to GCR, particle accelerators can generate nearly all kinds of photon, particle as well as neutron radiation with highest energy spectra but furthermore with significantly increased flux rates. Recent facilities such as CERN are even able to provide enough energy to create a dense QGP with energies up to 7 TeV from proton-proton collisions [183]. While mostly developed and built for physical research or medical applications, particle accelerators can also be used for radiation qualification of electronic components and systems. The accelerated testing of surface effects on semiconductors for example can perfectly be accomplished by the use of neutron beams as available from Los Alamos Neutron Science Center (LANSCE) with 600 MeV neutrons and a flux of $3{\cdot}10^5$ $cm^{-2}{\cdot}s^{-1}$ [194]. The Neutron Time Of Flight Facility at CERN (n_TOF) in comparison is able to provide 250 MeV neutrons with $1.5{\cdot}10^5$ $cm^{-2}{\cdot}s^{-1}$ [195]. In addition, CERN is currently able to generate 7 TeV protons as well as 2.76 TeV/amu lead ions [196, 197]. Other heavy elements are available at the SIS18 accelerator at GSI in Darmstadt. They offer all ions from C to U up to 2 GeV/amu at a maximum fluence of 10^{12} cm^{-2} [194]. As soon as the expansion of GSI to FAIR has been completed, this limit will be extended to 14 GeV/amu (see table 1.2).

2.5 Radiation Effects in Semiconductors

Recent news confirmed the failure of the Phobos-Grunt Mars mission due to charged particles in space. Apparently some electronic SRAM components on board of the probe, designated for landing on the mars moon Phobos, were not suitable for usage in ionizing radiation environments. The caused reset of operational components lead to undefined system behavior and the system's entered standby-state could not be ended [198]. Since news about semiconductors dealing with radiation are mostly related to satellite and space missions, it does not necessarily mean that there is no radiation impact on ground level. Cosmic rays are interacting with Earth's atmosphere while scattering plenty of secondary particles like protons, neutrons or electrons in addition to x-ray radiation. Its dose varies with altitude, solar cycles or the geomagnetic field and gets its biggest extent at the magnetic poles where charged particles are trapped and arise in auroras. Beside of the known ra-

diation effects on human bodies, especially in aircraft flights at high altitude during solar events along the pole regions [189], electronic semiconductors are widely exposed to cosmic as well as artificial radiation in multiple variations. Therefore, the following sections try to cover most of the currently understood temporary as well as permanent effects of ionizing radiation on semiconductor devices.

2.5.1 MOSFET and Radiation

CMOS microelectronics usage became very popular due to low pricing and simply due to the requirement of their processing powers for a variety of today's applications. But the MOSFET's susceptibility to radiation had quickly been discovered in its youth [199, 200, 201, 202]. A continuous shrinking of CMOS transistor gates and therefore the implicitly required reduction of operation voltage to reduce leakage current as explained in section 2.1.1 additionally increased the probability of ionizing radiation to successfully impact the circuit's electrical properties while passing by – in particular powered SOB MOSFETs are susceptible to radiation effects. Irradiating a semiconductor device creates electron-hole pairs within the manufactured oxide [170] as described in section 2.4.2. Under normal conditions, these pairs try to recombine immediately. The same applies to the electrons and holes within the sensitive gate silicon dioxide. But powering the transistor generates an electric field which enables the fast electrons to leave via the gate contact, traps the slow electron holes within the material and therefore hinders a subsequent recombination of both. This cumulative effect can be observed as long as an MOSFET is exposed to ionizing radiation by energy deposition and is referenced in literature as Total Ionizing Dose [203]. It will be explained in detail in section 2.5.5.2. In addition, the temporary track of electron-hole pairs generated by a penetrating particle along its path throughout the silicon is able to change a transistor's electrical properties – temporary or permanent. These effects are referenced in literature as Single Event Effects (SEE) [204] and will be described in detail in section 2.5.4. A so created SEE connection short can be established between the grounded substrate and the supply voltage, creating a parasitic sub-surface silicon thyristor or Silicon Controlled Rectifier (SCR) within the substrate of the CMOS transistor. Afterwards, this thyristor can be permanently activated due to large voltage spikes which permanently damages the transistor in a so called latch-up [205]. For SOI MOSFETs, this problem is implicitly solved, as the n/p regions have no contact to the substrate anymore due to the introduction of the intermediate insulating layer. But otherwise, SOI suffers from back-channel leakage current due to positive charge trapping in the buried oxide

[206] which contributes to the overall degradation. Even thermally caused shorts are possible due to limited heat dissipation characteristics and therefore Negative Bias Temperature (NBT) stress [55], especially in SOI MOSFETs. Penetrating non-ionizing particles can furthermore directly interact with the crystal lattice of a semiconductor by nuclear collision, even followed by a subsequent nuclear scattering, causing the so called displacement damage as described in section 2.5.5.1.

One of the first spacecrafts which operated a large number of 2400 CMOS devices in Earth orbits with intense radiation of up to $2.639{\cdot}10^5$ $p^+{\cdot}s^{-1}{\cdot}cm^{-2}$ above 5 MeV and $5.181{\cdot}10^5$ $e^-{\cdot}s^{-1}{\cdot}cm^{-2}$ above 0.5 MeV was the Atmosphere Explorer 51 (AE51) in 1973[207].

2.5.2 Cross-section and Weibull-Fit

The probability that a reaction between particles occurs is represented by the cross-section. In other words, cross-section is the effective area that a device represents to a particle. Therefore, every binary storage unit within an FPGA has a sensitive area for interactions, specified by its cross-section. The insensitive area in between is not of interest, as it does not contribute to the nuclear reaction. The bigger the active area, the more likelier an interaction, therefore the cross-section depends on the semiconductor's manufacturing process. The bigger the particle, the more likelier an interaction, therefore the cross-sections rises from neutrons to protons to heavy ions. Finally, a minimum deposited charge within the sensitive area, expressed by the particle's energy threshold LET_{th} (see section 2.4.3), has to be exceeded, before an interaction effect will become visible. Therefore, the cross-section for a semiconductor device normally comprises it's elementary unit, which is represented by a single bit. This allows calculation of the cross-section σ of a nuclear reaction within semiconductors according to [208] by using the equation:

$$\sigma = \frac{N_R}{\Phi \cdot N_S}$$

with N_R the number of reactions per unit time, Φ the beam particles per unit time per unit area (flux) and N_S the number of scattering centers. It results in unit area per scattering unit. Assuming error rate and particle flux of a particle beam test using a 90 nm Xilinx Virtex-4 SRAM FPGA DUT, it can be calculated to:

$$\sigma = \frac{0.446\ s^{-1}}{4 \cdot 10^6\ s^{-1} \cdot cm^{-2} \cdot 7200256\ bit} = 1.55 \cdot 10^{-14}\ cm^2 \cdot bit^{-1}$$

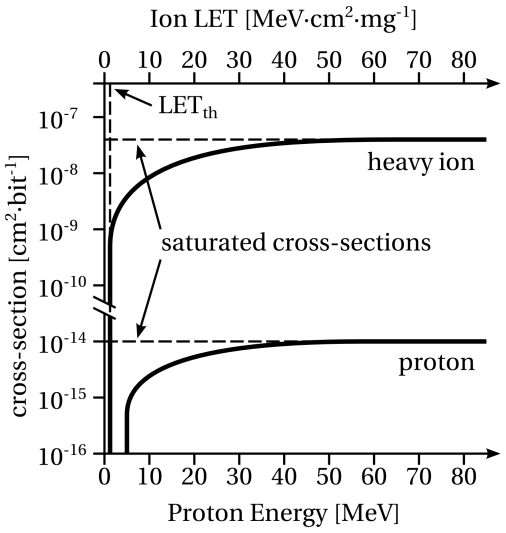


Figure 2.13: Simplified depiction of the SRAM SEU cross-section curves of a Xilinx Virtex-4 FPGA devices during proton and heavy ion irradiation. For heavy ions, the cross-section is given as a function of the LET to the material. The initial amount of energy that is required to show first bit upsets is clearly indicated by LET_{th}. For protons, the LET is too low [213]. Therefore, proton cross-section is shown as a function of particle energy. Both curves indicate a saturation in the cross-section as soon as the maximum radiation impact is reached. All experimental data for these curves have been taken from [214].

This is a common value for proton impact on such devices using high energy particles [209]. The sensitive area of a whole device can therefore be calculated by this cross-section multiplied with the total number of bits.

The accuracy of a device's overall cross-section can be further improved by calculating and summing up the separate cross-sections of single, double, triple, … bit upsets (see section 2.5.4.2), according to [210]. This results in the adapted formula:

$$\sigma = \sigma_{1\,\text{bit}} + \sigma_{2\,\text{bit}} + \cdots = \frac{N_{R\,1\,\text{bit}}}{\Phi \cdot N_S} + \frac{N_{R\,2\,\text{bit}}}{\Phi \cdot N_S} + \ldots$$

With exceeding LET_{th}, the bit cross-section rises. But this effect saturates at a certain point of higher LET values [211], since the total amount of sensitive areas that can be upset within a device is constant. This effect is exemplified in figure 2.13. Mathematically, the shape of this LET to cross-section curve can be described with a Weibull Fit [212]. As the maximum LET of protons is very low, for example 16 $\text{MeV} \cdot \text{cm}^2 \cdot \text{mg}^{-1}$ in silicon [213], the cross-section is usually shown as a function of particle energy that exhibits a similar saturation plateau. In conclusion, a device's cross-section for high energy particles solely depends on the kind of particle (neutron, proton, heavy ion) as well as its probability to hit a sensitive area within the device. In case of FPGAs this means to hit specific transistors within a powered SRAM cell (see section 2.1.4). Vice versa, a known cross-section can be used to calculate the SEE rate of a device (see section 5.1.1).

Nowadays, many cross-sections for a variety of devices can be found in literature. Regarding Xilinx SRAM FPGAs, neutron cross-sections can be found in [59, 215], proton cross-sections can be found in [215, 214, 209] and heavy ion cross-sections

Particle Beam	Config. Memory	Block Memory
LANSCE (WS Neutron)	$6.89 \cdot 10^{-15}$	$6.15 \cdot 10^{-15}$
TSL (180MeV Proton)	$8.29 \cdot 10^{-15}$	$8.19 \cdot 10^{-15}$
CERN H4 (HE Hadron)	$1.50 \cdot 10^{-14}$	$1.40 \cdot 10^{-14}$

Table 2.4: Xilinx Kintex-7 325T neutron, proton and mixed hadron cross-sections taken from [215]. All values are given in $\left[\text{cm}^2 \cdot \text{bit}^{-1}\right]$

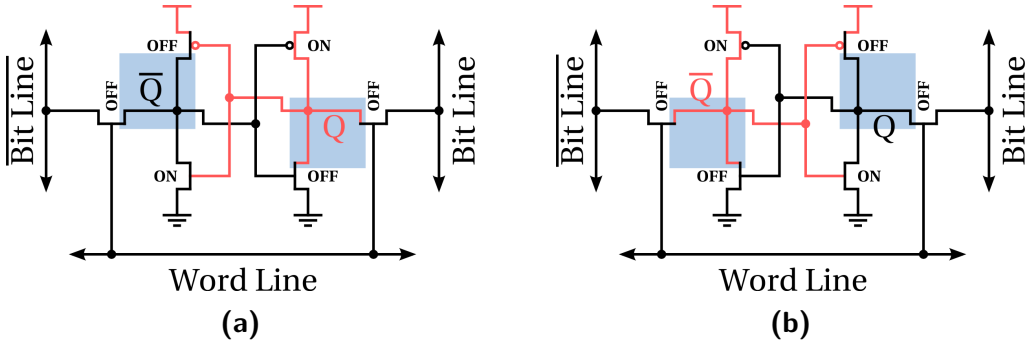


Figure 2.14: Radiation sensitive regions within a 6-transistor SRAM cell, according to [220], derived from [221]. High energy particles striking the depicted sensitive areas (blue) affect the electric potential of the transistors by transient impact. This can cause a spontaneous reversal of the cross-coupled inverter system and therefore can flip the SRAM cell's stored value. The process is based on the prerequisite that the affected transistors are in 'OFF' state [222] as depicted in subfigures (a) and (b) for both possible configurations.

can be found in [216, 217, 214, 156]. Every device series exhibits a different cross-section due to the modified manufacturing process, for example the Virtex-4 is nearly twice less susceptible than the Virtex-II device series [59, 214]. Even within a single device, cross-sections of the different FPGA-components vary by each other due to the triple-oxide thicknesses as described in section 2.1.3. Typical cross-sections for one of the latest 28 nm Kintex-7 325T device's configuration and block RAM can be found in table 2.4.

Beside of the conventional method to measure the cross-section, there are also theoretical models available which try to find a coherence between proton and heavy ion cross-sections, for example [218, 219].

2.5.3 Radiation Effects in SRAM Cells

As depicted in section 2.1.4, conventional SRAM cells are made of multi-MOSFET cross-coupled inverter systems. In combination with the fact that MOSFETs are af-

fected by ionizing particles (see section 2.5.1), it becomes apparent that conventional SRAM cells exhibit radiation effects. Whereas a single standalone transistor's state can easily be upset, the construction properties of the cross-coupled CMOS inverter system in SRAM cells indicates several benefits as well as drawbacks regarding upset susceptibility, as some transistors are contributing to a greater extent than others do. An ionizing particle striking one of the SRAM cell's powered transistors most likely affects either the p-doped channel region of an NMOS transistor or the p-doped drain region of a PMOS transistor, both currently idling in 'OFF' state [222] (see section 2.5.4 for more details). This impact immediately creates a short conductive glitch within the transistor's reverse biased p/n junction as soon as the particular material's threshold value has been exceeded, and therefore temporarily toggles the transistor's state to 'ON'. If this state contributes to one of the inverter pairs outputs, it can be propagated to the other one's input and subsequently lead to an inversion of the overall stored SRAM cell's configuration. Therefore, the resulting sensitive areas within an SRAM cell are depicted in figures 2.14a and 2.14b for both possible configurations, according to [220]. Beside of transistor type, doping, state and location, the critical amount of charge Q_{crit} that can trigger a cell configuration change, depends on many other device-specific properties, primarily defined by node capacitance, operating voltage, and strength of the feedback transistors [192]. Finally, the overall number of susceptible transistors of which an SRAM cell is built of has to be considered. The larger its number and the smaller Q_{crit}, the more impact will occur.

However, configuration flips may be the most obvious ionization impact in SRAM cells, but the mentioned MOSFET radiation effects from section 2.5.1 additionally modify transistor reaction characteristics and therefore directly impact an SRAM cell's switching speed, which insidiously causes inconsistencies between timing constraints. Since logical designs composed of such cells highly depend on well defined timing specifications, a whole system may fail due to a single additionally introduced delay within a critical design path. Such effects can also emerge from accelerated aging in device-hazardous environments [118]. The major advantage of reconfigurable SRAM in this case is, that compromised cells within bigger arrays can easily be restricted from a user design and therefore blocked from further usage. This of course requires a use-case which offers device reconfiguration.

As SRAM is used for registers and caches in modern microcontrollers and high performance processors [69], reliability is an urgent requirement even for operation within the natural radiation environment on Earth as well as higher altitude.

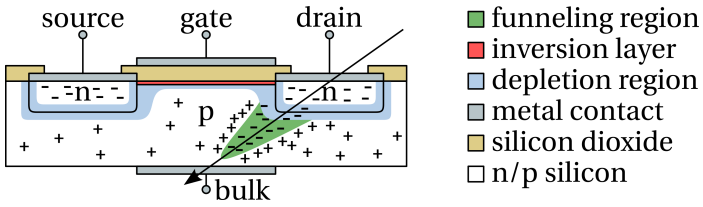


Figure 2.15: Basic cross-section of an SOB N-MOSFET stroke by ionizing particles. The non-conductive depletion region along the p/n junction is partially extended by the created electron-hole pairs in a funneling region. Free charge carriers are drifting to the drain electrode, creating a transient current spike, known as Single Event Effect.

Therefore, devices have to deal with alpha particles, emitted from natural radioactive isotopes within the chip package itself, as well as neutrons, generated due to cosmic ray high energy proton scattering on collision with atmospheric nucleons, as explained in section 2.4.6. To withstand or handle such effects, most manufacturers provide special features for their memories, such as additional parity recovery cells which are fed and validated by ECC circuitry in every access cycle. Another option is to increase Q_{crit} by modification of radiation critical cell specifications such as capacitive transistor load or by minimization of cell size and therefore source/drain regions. Beyond this, many other techniques are available and will be explained in section 2.6.

Finally, there is also an option to artificially cause radiation effects in SRAM cells for testing purpose without the usage of any ionizing radiation. Pulsed laser technology and its generated photocurrents within silicon substrate can be utilized to directly impact a really fine grained area of designated SRAM structures [223, 224]. Unfortunately, this method, similar to other optical approaches, ideally requires removal of all metal layers in-between the device's penetration surface and the examined circuit to minimize reflections, especially by wires required for basic transistor voltage biasing as well as bit line and word line routing.

2.5.4 SEE - Single Event Effects

High energy particles penetrating or traversing semiconductor devices are impacting the silicon material as well as its dopants by scattering electrons from atoms, thus creating a dense plasma of electron-hole pair charge carriers along their trail. As soon as such a particle crosses a significant area on the silicon die, which is occupied by MOSFET structures, it may produce adverse effects commonly known as

SEE. The particle trail itself measures only very little in size, varying with particles and energy, but given that the mask size of recent semiconductor structures as well as the corresponding threshold voltage is constantly shrinking, a transistor of currently about 14 nm gate size or less suffers significantly from such an impact. According to [225], scaling is even the main reason for SEE. SEEs have been initially reported as anomalies within JK flip-flops in space environments [226] and later-on confirmed to be radiation-induced alpha particle soft errors in dynamic memories caused by impurities within the device material itself [227]. Nowadays, it is well known that multiple types of particles (see section 2.4) are able to react with semiconductors in a way that causes SEEs.

The process of energy transfer from the high energy particle to the substrate by interaction with electrons and holes from lattice atoms slightly alters the particle's trajectory, but as long as no direct nucleus collision takes place (see section 2.5.5.1), it remains nearly linear. While passing by, it is able to eject plenty of electrons from their positions while leaving an equal number of corresponding holes within the material, thus creating a dense plasma of electron-hole pairs. These pairs try to recombine immediately as explained in section 2.4.2, but in case the ionization process takes place near a powered transistor's p/n junction, the local electric field separates electrons and holes by collecting the corresponding charge carriers [181] (electrons in NMOS and holes in PMOS). The total number of electron-hole pairs is known to be proportional to the LET, expressing the amount of energy deposited within the substrate, and therefore it is directly related to the ionizing particle's type, energy as well as the target material as stated in section 2.4.3. While holes within silicon crystals have to move by electron hopping, their speed is only about one-third that of electrons which are able to freely flow within the substrate [228], but nevertheless, both NMOS and PMOS transistors are contributing to SEE [222].

In consequence of the electron-hole pair generation, the conductive plasma extends the transistor's depletion region along the particle's trail deep into the substrate [229, 230], forming a so called field tunneling or funneling region as depicted in figure 2.15 for N-MOSFETs. This conduction tunnel expands the electric field further into the substrate, causing passage of a large number of generated charge carriers into the p/n junction [229]. Since the depletion region controls the transistor's electrical field, this process alters the electrostatic potential of the MOSFET by adding accumulated charge, but according to [231] it is limited to the same charge initially stored within the depletion layer. As soon as the transistor's threshold value has been exceeded, a transient current spike is generated on the gate electrode. The

spike lasts about hundreds of picoseconds and occurs right after the corresponding particle strike [181]. Only a few nanoseconds later, the electron-hole plasma collapses, the funneling region dissolves [229] and the depletion region normalizes by the regular carrier diffusion process along the p/n junction. Remaining mobile carriers will be transported by this process to the drain electrode and therefore finally contribute to the overall charge in a second, less prominent process. This takes about few hundreds of nanoseconds [181]. According to [229], the funneling process depends on substrate concentration, bias voltage, as well as particle energy. Therefore, using highly doped material or low electrode voltage are able to successfully mitigate at least the funneling effect [225]. Unfortunately, there are many of such effects within semiconductor devices. They can be found in [181].

Nowadays, SEEs are classified in two major categories, basically indicating the specific effect caused within semiconductor devices. The first category represents non-destructive '*soft*' errors which can change logic memories or device operation. They are called transients, upsets or functional interrupts and can mostly be mitigated by a clever system design utilizing fault tolerance techniques. As this kind of error is temporal, it can at least be reset by performing a power cycle. The second category contains destructive '*hard*' errors which are able to partially or completely destroy a physical device. Latch-up, burnout or gate-rupture errors belong to this category. As this type of error is permanent, it cannot be directly recovered. It may be possible to exclude the defective area from the whole device, but this approach mostly depends on the type of device as well as possible methods for intervention.

2.5.4.1 SET - Single Event Transients and PIPB - Propagation-Induced Pulse Broadening

According to the above-mentioned physical SEE generation process within MOSFETs, the emitted current spike on the drain electrode rises as transient impulse within the semiconductor circuit. The pulse width of this temporary glitch is usually about 100 to 200 picoseconds [177, 232], also measured in [233], and depends on multiple factors such as the incident particle, circuit process technology, node capacitance and driving strength of the transistor [234]. But it can easily extend to more than a nanosecond by dynamic pulse propagation within a chain of closely coupled CMOS inverters, so called Propagation-Induced Pulse Broadening (PIPB) [232]. Another effect seen in such inverter chains is called Double-Pulse SET (DPSET) and describes the process of cross-boundary charge sharing between separate PMOSFET n-well regions of serially struck inverters [235]. It leads to a sec-

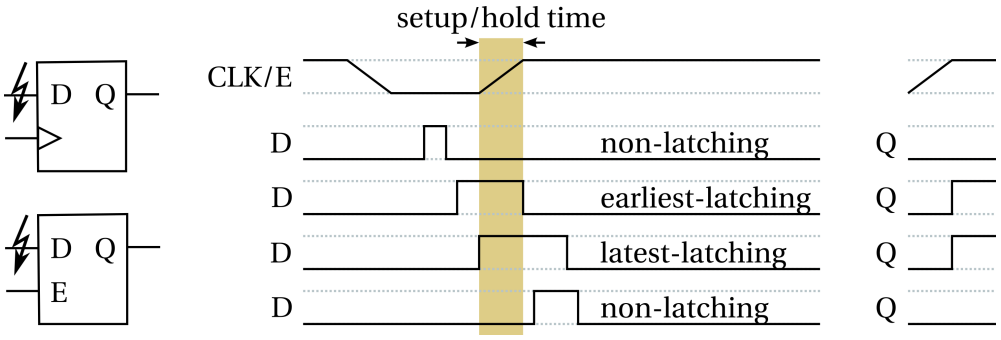


Figure 2.16: Simplified depiction of a clock sampled SET that becomes a permanent SEU supposing it appears in coincidence with the setup and hold times of a D-type flip-flop or a D-type latch switching into hold mode. To meet the latching condition, SET arrival can be either earliest or latest, according to [177]. D represents the input data signal, CLK/E the input enable signal and Q the latched output signal. $\overline{Q}$ has been omitted for simplification. Latches in transparent mode are sampling the data input immediately.

ondary transient pulse only from a single ion strike and is mainly angle-dependent [236]. A nearly similar correlation between n-wells and SET pulse width has been observed in [237]. As inverter chains are common in today's semiconductor devices with complex logic paths, this behavior contributes to the critical SET generation. In analog circuits, a transient pulse is able to immediately cause inadvertent oscillation, whereas in digital clock sampled circuits it needs to propagate into the logic elements by getting latched. The probability of this latching process is directly proportional to the mentioned SET pulse width and therefore a timing critical factor. While a narrow glitch may only create cross-talk unable to exceed the necessary sampling characteristics of device primitives, a wider shape can have serious consequences. A temporal SET in combinatorial logic arising at the data input of a D-type latch or flip-flop in coincidence with a regular sampling pulse can be converted into a permanent value known as Single Event Upset (SEU). This process is depicted in figure 2.16 for well defined overlaps. If this coincidence is not that accurate, it may result in random or even metastable output states by violating the flip-flop's setup or hold times. This clock-dependent sampling process in D-type flip-flops implies, that the latching sensitivity is also highly related to the system's operation frequency. An increased sampling frequency therefore results in a higher probability of an SET to be converted into an SEU. Transients which are approaching to transparent latches are sampled immediately. In contrast, an SET which does not get sampled will not show any effect. Unfortunately, this adds some difficulties to the characterization, simu-

lation as well as prediction of such effects [238]. Beside of the combinatorial logic, of course the global clock as well as asynchronous reset networks themselves can be affected by SETs. This will have direct consequences for the whole device and is depicted in section 2.5.7. Synchronous resets do not exhibit a special susceptibility, as they are built of regular combinatorial signals.

2.5.4.2 SEU - Single Event Upset and SBU/MBU - Single/Multiple Bit Upset

Irradiation of semiconductors is able to generate SETs as explained in section 2.5.4.1. If such a temporal SET exceeds the critical amount of charge required to get captured in a D-type latch, in a flip-flop or in the sensitive areas of a cross-coupled inverter circuit as depicted in section 2.5.3, it forms a so called Single Event Upset (SEU) which mostly becomes visible to the user as a Single Bit Upset (SBU) within a memory element. The SEU immediately overwrites the originally stored information but is not destructive and can therefore easily be fixed by refreshing or resetting the corresponding memory cell.

The SEU rate R_{SEU} for a device can be calculated by using the following formula:

$$R_{\mathrm{SEU}}\left[\frac{1}{\mathrm{s}}\right] = \Phi\left[\frac{1}{\mathrm{s}\cdot\mathrm{cm}^2}\right]\cdot\sigma\left[\frac{\mathrm{cm}^2}{\mathrm{bit}}\right]\cdot N_D\ [\mathrm{bit}]$$

whereas Φ is the particle flux which penetrates the substrate material, σ is the cross-section (see section 2.5.2) and N_D represents the total number of device bits. If Φ is given experimentally by the use of dosimeter devices, it has to be normalized to the device's covered area first. Therefore it is important to know the exact internal dimensions of a device as explained in section 5.1.2.

For really small SEU rates, as it can be found in technical documentations of commercial devices which just have to consider the environmental radiation field, manufacturers specify Failures In Time (FIT) per billion hours [64] to convey how many errors can confuse a device. This value can be calculated by using the formula:

$$\left[\frac{\mathrm{FIT}}{\mathrm{Mbit}}\right] = \Phi\left[\frac{1}{\mathrm{s}\cdot\mathrm{cm}^2}\right]\cdot\sigma\left[\frac{\mathrm{cm}^2}{\mathrm{bit}}\right]\cdot 3600\,[\mathrm{s}]\cdot 10^9\cdot 10^6$$

whereas Φ is the particle flux which penetrates the substrate material and σ is the cross-section (see section 2.5.2).

As SEUs can occur in any kind of semiconductor device which makes use of radiation susceptible CMOS latches, flip-flops or SRAM cells, it can affect for example FPGAs, microprocessors, state machine controllers or memory devices, leading to

defective data storage or even functional failures, so called Single Event Functional Interrupt (SEFI) as explained in section 2.5.4.3. In case of SRAM-based FPGAs, an SEU can even alter the internal device configuration that contains the internal routing information, leading to random logic circuit shorts and therefore unpredictable device behavior (see section 2.5.7). To give a practical example about the SEU configuration susceptibility of such FPGAs, recent data of the CERN ALICE detector can be used: Xilinx Virtex-II Pro SRAM FPGAs had been placed on the Readout-Controller Unit (RCU) boards which are responsible for data acquisition from the Time Projection Chambers (TPC). Since these devices were located within the radiation field of the detector, a worst rate of 10-20 SEU/h within all 216 RCU main FPGAs in total was expected [239]. Finally, a rate of 9.24 SEU/h occurred in all FPGAs during 7 TeV minimum bias $2.80 \cdot 10^9$ proton-proton collisions on 16/10/2010 [240] and 1552 SEUs in total between May and August 2011 [241]. This susceptibility lead to the fact that 9% of all data taking sessions had to be stopped due to TPC electronics errors [242].

In contrast to multiple SBU which accumulate over time, Multiple Bit Upsets (MBU) are defined as several bits flipping in parallel due to a single particle strike. It can also be understood as charge sharing between multiple sensitive areas of SRAM cells [243]. This effect is even boosted by the shrinking circuit manufacturing process technology, as a single ionizing particle's created electron-hole pair is progressively able to affect more than a single transistor in parallel as investigated in [244, 245, 246]. To give an example for Xilinx FPGAs made of SRAM cells, this meant an increase of MBUs at proton irradiation from Generation Virtex (250 nm) to Virtex-4 (90 nm, see table 2.1) of 0.04% to 3.05% according to [210]. Due to the increased LET, this effect is even worse for heavy ions: It increased from 21% in Virtex-II (150 nm XC2V1000) to 59% in Virtex-5 (65 nm XC5VLX50) according to [247]. In addition to these direct ionization effects, also nuclear reactions (see section 2.5.5.1) are considered to have a major impact on the MBU rate: Impacting nucleons may consecutively interact with multiple nearby lattice atoms and even lead to cascades as shown in figure 2.18 which create multiple closely related upsets [248]. This neutron impact on MBU has been confirmed in [249] by irradiating different microprocessor technology nodes.

In a natural radiation environment such as LEO, where nearly all manned spaceflights and space stations are located, the SBU/MBU ratio has been experimentally determined to 93.63% SBUs and 6.37% MBUs (up to 6 bit) for the radiation-tolerant

XQR Xilinx Virtex-4 FPGA [3, 4]. Therefore, the MBU-issue concerns all fields of radiation environments, not just the artificially created ones.

The increasing occurrence of MBUs is essentially complicating the whole process of conventional error mitigation by the use of design redundancy or error correcting algorithms (see section 2.7) and leads to logical domain crossing effects. The physical distance between separate memory cells thereby becomes a relevant factor which can only be influenced by the device manufacturers (see section 2.6).

2.5.4.3 SEFI - Single Event Functional Interrupt

By definition, a Single Event Functional Interrupt (SEFI) is "a soft error that causes the component to reset, lock-up, or otherwise malfunction in a detectable way, but does not require power cycling of the device (off and back on) to restore operability" [250]. More generally, it is defined as an "SEE that results in the interference of the normal operation of a complex digital circuit" [251]. This basically includes all supporting functionality within a device, such as reset circuits, clock generators, programming and configuration interfaces, or in case of FPGAs the configuration matrix itself [251]. Therefore, an SEFI is always related to device functionality and never considers bit upsets within plain user-data stored in an IC memory.

In parallel to the SEU cross-section (see section 2.5.4.2), some manufacturers indicate an SEFI cross-section to convey how many of the overall upsets can really confuse the basic functionality of a device. This SEFI cross-section is of course orders of magnitude lower than the SEU cross-section [64]. The total proton SEFI cross-section for the radiation-hardened Xilinx XQR5VX130 Virtex-5 for example is given with $\sigma_{\mathrm{SEFI}} = 7 \cdot 10^{-7}$ cm^2/device [252]. In addition, the SEFI ratio indicates the numerical relation of SEEs and SEFIs, given by the following formula according to [253]:

$$R_{\mathrm{SEFI}} = \frac{N_{\mathrm{SEE}}}{N_{\mathrm{SEFI}}}$$

The occurrence of an SEFI can have various and nearly unpredictable reasons. Just to give some examples: An SEU within the control registers of all different kinds of ICs can lead to undefined states and therefore lock-up or reset as soon as an internally running test cycle detects the error. This is, for example, the most frequent error within state controllers of Flash memories. Furthermore, an SEU within the program counter of a microprocessor can interrupt the correct program execution and may lead to infinite loops because of variables which have not been correctly set in advance. For complex programmable Xilinx SRAM FPGAs, there have been SEFI

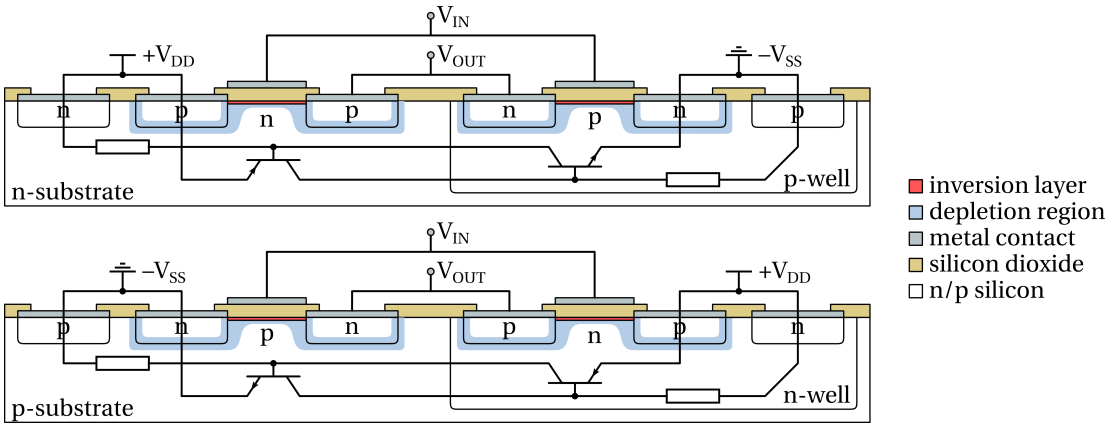


Figure 2.17: Simplified depiction of a sub-surface parasitic npn/pnp BJT forming a thyristor in C-MOSFET according to [205]. Once activated, the thyristor creates a cross-coupled positive feedback loop which mutually powers itself and increases the parasitic current between power (V_{DD}) and ground (V_{SS}) until device breakdown.

issues recorded, which caused a spontaneous Power-On Reset (POR). But the probability is relatively low and the SEFI was classified to occur only every 221 years for a radiation-hardened Xilinx Virtex-II XQR2V in GEO radiation field [251]. Less critical, but also a possible SEFI in FPGAs can be caused by an SET triggering the combinatorial reset wires of local design domains. Furthermore, an upset within the internal Frame Address Register (FAR) may completely crash the whole FPGA configuration if it occurs exactly in parallel to a programming cycle. Even the SelectMAP circuit responsible for accessing the FPGA configuration memory itself can be upset.

All of this indicates, that detection of such errors is a difficult task. But given that an SEFI is a temporal SEE, it can at least be recovered by a device reset or power cycle, maybe performed via an external watchdog. More information regarding SEFIs in Xilinx FPGAs can be found in section 2.5.7.

2.5.4.4 SEL - Single Event Latch-up

Beside of the p/n junctions within CMOS transistors that are required to provide correct electrical operation, there exist multiple others which are considered as parasitic. Therefore, every CMOS-pair of p-type and n-type SOB MOSFETs on the same substrate inherently provides two additional npn/pnp Bipolar Junction Transistors (BJT) by design that form a parasitic sub-surface silicon pnpn thyristor within the substrate [205] as depicted in figure 2.17. This thyristor, once activated, creates a cross-coupled positive feedback loop which mutually powers itself and increases

the parasitic current until device breakdown. Therefore, it has to stay switched off. During normal operation, this is guaranteed in MOSFETs by using a common contact for source and body which shorts the base-emitter junctions of the parasitic BJTs [254] and holds it in reverse biased blocking state. But due to the formed positive feedback circuit, the thyristor is susceptible to charge injection as carried out for example by large voltage spikes on the power supply wires or by heavily ionizing particles. Therefore, ions can act as a trigger within the substrate [254] by creating additional charge carriers which lead to a forward-biasing of the thyristor's base-emitter junction and switches it on. Consequently, the depletion region's width of the affected p/n regions gets reduced and hence a low impedance circuit short between power (V_{DD}) and ground (V_{SS}) is established. The subsequently flowing current within substrate- and well-silicon permanently switches on the C-MOSFET in a so called Single Event Latch-up (SEL) [205]. In this latched low-resistance state, the constantly drawn high current leads to time-dependent thermal degradation of the device.

As long as the device V_{DD} is sufficiently powered above the thyristor's hold condition V_H, the SEL remains activated. Immediate response by cutting off the power supply or by reducing the current via additional circuits and therefore breaking the feedback loop within the thyristor may prevent further and permanent damage. SEL is limited to local regions, but if not handled, it propagates and the device may burn out as shown in section 2.5.4.5. Furthermore, SEL can cause wrong output currents in power MOSFETs, which can immediately affect externally connected components. Since SEL is highly temperature dependent, the activation threshold value decreases at high temperature as indicated in [255].

The SEL issue has been solved for SOI MOSFETs, as the n/p regions within such devices have no contact to the substrate anymore due to the introduction of the intermediate insulating layer. Xilinx FPGAs have also been announced to be immune against SEL caused by neutrons [256], but in several other COTS devices there is a lack of mitigation techniques. To give some examples: The heavy ion SEL cross-section of the AMD K-5 microprocessor is about 10^{-3} cm^2/device at an LET of 12 $MeV \cdot cm^2 \cdot mg^{-1}$ [219], which is emitted for example by an ^{27}Al atom traversing with 1.7 MeV through the semiconductor's material. Another more practical example is given in [255] by experimental in-flight SEL data of different SRAMs: The most sensitive one, a low power high speed IS62WV20488BLL SRAM array, exhibited a latch-up rate of 0.2 SELs/day/device. 81% of these errors occurred while passing the SAA as mentioned in sections 2.4.5 and 1.1.

More information regarding the SEL process in CMOS technology can be found in the review [257].

2.5.4.5 SEB - Single Event Burnout

SELs in C-MOSFET devices result in permanent high current flowing in the silicon body region of parasitic bipolar transistors while thermally degrading the device as explained in section 2.5.4.4. If this process takes place over a specific period in time, it results in a destructive and permanent Single Event Burnout (SEB), also referenced as SEBO or SBO. Therefore, it can be found especially in high power applications, for example switching mode power supplies, which make use of critical components such as IGBTs, power MOSFETs or high-voltage diodes. Most of the power MOSFETs use an additional thin and planar epitaxial layer in the silicon which acts as the BJT collector but unfortunately is also highly susceptible to SEB [258]. This is due to the generation of electron-hole pairs by the avalanche multiplication effect, investigated in [259]. Since SEB requires interaction of heavily ionizing hadrons, the probability of its occurrence is relatively low for conventional CMOS circuits.

2.5.4.6 SEGR - Single Event Gate-Rupture and SHE - Single Event Hard Error

Beside of the most well known SEEs caused by ionizing particles, there are also some effects within low power semiconductors which occur with less probability and therefore are more difficult to discover. One of them is referenced as Single Event Gate-Rupture (SEGR) or Single Event Hard Error (SHE) and had been initially identified within SRAM cells because of stuck bits [260] or jammed bits [261]. But SEGR also affects many other devices, such as DRAMs [262], non-volatile EEPROMs [263] as well as power MOSFETs [264]. The basic principles behind this effect are well known for many years now and had been investigated for example in [265].

According to [260], the occurrence of SEGR requires irradiation with heavy ions featuring an LET of at least 30 $\mathrm{MeV \cdot cm^2 \cdot mg^{-1}}$. Such a heavy ion traversing through a single MOSFET's gate dielectric material transfers enough energy to create an electron-hole plasma which forms a low resistance conductive channel straight across the gate's oxide insulator. A sufficiently high electric field within this affected transistor subsequently causes a high current flowing between gate and substrate while immediately draining the transistor's capacitor. The hence arising heat dis-

sipation can cause rupture and meltdown of the gate oxide [263] which shorts the circuit and leads to a permanent *ON* state [261].

SEGR is also highly dependent on the impact angle of the striking heavy ion: Angles far from nominal incidence decrease the occurrence of gate ruptures [264]. Also temperature seems to have a marginal impact, whereas high temperatures of about 90 °C have shown less gate ruptures [264].

Some of the SEGR errors were reported to be annealing in minutes while others were permanently stuck [260]. Even more, the TID has influence on the probability of SEGRs [266]: As shown in [267], the number of stuck bits increased with TID and most of the annealed bits reappeared at the same position during radiation events. Therefore, SEGR is often considered to be a micro-dose effect within the gate oxide. As SEGR requires interaction of heavily ionizing hadrons, the probability of its occurrence is relatively low for conventional CMOS circuits. Xilinx FPGAs have therefore been announced to be immune against SEGR caused by neutrons [256].

2.5.5 Cumulative Effects

In contrast to SEE, cumulative effects are gradual accumulative effects collected over the entire runtime of a semiconductor device. They become visible as soon as a device-specific threshold value has been exceeded and can be separated according to non-ionizing as well as ionizing impact. The non-ionizing effect is known as lattice displacement, simply caused by nuclear collisions of penetrating particles with lattice atoms within the semiconductor. The second cumulative effect is known as total dose and represents the ionization's deposited energy value within a material. Among others, it is directly related to the ionizing particle's LET (see section 2.4.3) and therefore its type, energy, and size as well as on the semiconductor's gate material, size, and doping.

Total dose as well as lattice displacement can be mitigated by the so called annealing procedure, which implies to leave the device untouched at room temperature [268] or heat it up to accelerate the process. The annealing rate itself decreases over time [269]. Both cumulative effects are explained in the following subsections.

2.5.5.1 Lattice Displacement Damage and NIEL

Particles or photons traversing solid material have a specific probability to collide with its atoms while getting slowed down or even stuck. This non-ionizing interaction can change the arrangement of lattice atoms, leading to interstitial atoms as well as vacant positions as depicted in figure 2.18. A penetrating ion is initially

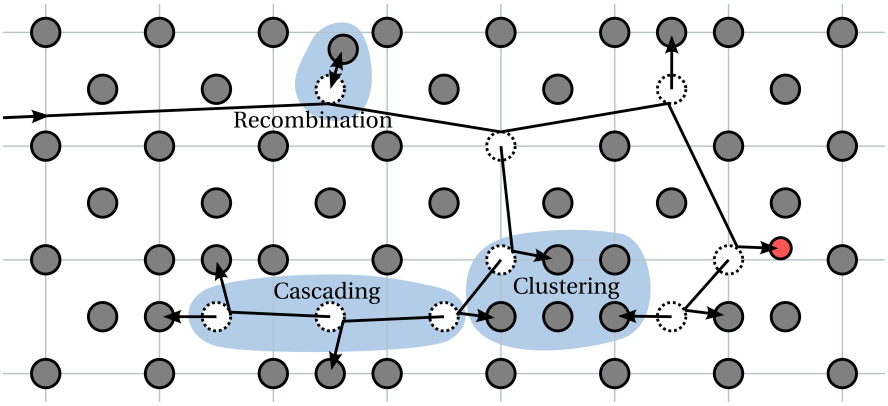


Figure 2.18: Simplified displacement damage in crystal lattices. A penetrating particle can pass its energy to a lattice atom by nuclear reactions while ejecting it from its position and creating Frenkel pairs. Subsequent processes can lead to recombination, cascading as well as clustering.

slowed down by electron interaction and therefore moves almost straight across the device. If it possesses enough energy, it is able to leave the device, otherwise, assuming sufficient slowdown, the probability of nuclear collisions increases. This may eject a lattice atom from its regular position (elastic scattering) if the required recoil threshold energy of 21 eV in silicon [187, 270] has been exceeded. These atoms in turn are able to interact with other lattice atoms while losing their energy and creating defect cascades. Every temporarily displaced atom leaves a vacancy when moving into an unused non-lattice interstitial position, creating a so called Frenkel pair [271], but it can also migrate and even recombine with another vacancy. The Frenkel pair recombination rate in silicon depends mostly on its distance in relation to the material's lattice constant. It is about 60% [272] of the total number of Frenkel pairs and locally increases to about 75% to 95% [273] in heavily disordered regions. Remaining interstitials and vacanvies can also interact with other impurity atoms or aggregate to long living immobility clusters [274] and therefore permanent displacement damage within the device. While isolated and solely misplaced atom defects are mostly created from low energy protons with less than 10 MeV in silicon, defect cascades are created from higher energy protons between 10 and 200 MeV as well as neutrons.

Lattice damages degrade the analog properties of the semiconductor junctions and therefore the device's electrical performance by introducing additional states between valence and conduction band in the electron band model. Among others, this enables electrons to easily move from the valence to the conduction band, in-

creasing the conduction band's current flow. Vice versa, they are able to capture electrons from the conduction band which ties holes from the valance band reducing the conduction band's current flow. Therefore, it also influences the semiconductor's doping characteristics. More details on this topic can be found in [272, 187].

Displacement damage and therefore nuclear interaction is a stochastic process, normally approximated and normalized to 1 MeV neutrons using by the Non-Ionizing Energy Loss (NIEL) equivalent, even for heavy ions [275]. It represents the non-ionizing energy loss per distance by a single penetrating particle in silicon and is given in $\mathrm{MeV} \cdot \mathrm{cm}^2 \cdot \mathrm{g}^{-1}$. According to [276], single isolated defects are dominating at low NIEL values below $5 \cdot 10^{-5}$ $\mathrm{MeV} \cdot \mathrm{cm}^2 \cdot \mathrm{g}^{-1}$, while clusters are dominating at high NIEL values above $2 \cdot 10^{-4}$ $\mathrm{MeV} \cdot \mathrm{cm}^2 \cdot \mathrm{g}^{-1}$.

Electronic devices such as bipolar transistors, which rely on current flow based on electrons and holes, suffer most from gain degradation and leakage current due to displacement damage. CMOS semiconductors are less sensitive, since the gate oxide volume (see figure 2.1) is fairly small and minor modifications within source and drain doping cannot force a p/n type inversion [277]. But also optoelectronic components may show displacement effects such as a reduced light-emitting diode (LED) illumination output or a reduced efficiency of Charge-Coupled Devices (CCD).

Displacement damage within certain limits can be easily mitigated by annealing the device and therefore provide enough energy for the Frenkel pairs to recombine.

2.5.5.2 TID - Total Ionizing Dose

The Total Ionizing Dose (TID) represents a cumulative effect referenced as deposited energy in form of ionization within a given material. This slowly degrades a semiconductor device and mainly results in modified threshold voltage as well as leakage current. As the reliability of a transistor is characterized by its ability to instantly switch from a low-conductance off-state to a high-conductance on-state as soon as the gate voltage threshold is exceeded, modification of this threshold may cause NMOS transistors to open sooner and PMOS transistors to open later. Moreover, leakage current within the isolating regions of a transistor makes it more difficult to distinguish between on and off states. These effects can overall lead to a CMOS transistor malfunction or even ultimate failure by switching NMOS transistors permanently on and PMOS transistors permanently off. The TID of semiconductors is mostly affected by heavy ions, low energy protons as well as higher energy electrons, whereas recent research also indicated a neutron impact [278, 279].

Ionizing radiation creates a dense plasma of electron-hole pairs within a semiconductor's oxide material (see section 2.4.2) in concentrations above the doping densities of most conventional devices [170]. This effect includes the gate silicon oxide, which isolates the metal contact from the bulk body substrate, as well as the bigger amount of amorphous field oxide, which is applied to passivate and protect the semiconductor's surface and isolate the MOS transistors from each other. Since the field oxide is thicker than the gate oxide (about 1 µm versus 5.5 nm in 240 nm technology), it is exposed to a much higher radiation dose [277]. Under normal conditions, the created electron-hole pairs try to recombine immediately, but powering a transistor generates an electric field which separates electrons and holes and therefore decreases the recombination rate. While the fast electrons are quickly moving towards the powered gate/gate-silicon-oxide interface of NMOS transistors to leave via the gate electrode, some of the slower holes remain trapped, forming a positive charge within the gate oxide. If this effect cumulates, it induces transistor threshold voltage shifts, noise [280], and immediate power leakage. Furthermore, the holes slowly migrate to the opposite interface (gate-silicon-oxide/bulk-silicon in NMOS; vice versa in PMOS) by electron hopping, while getting stuck by the so called deep hole trapping process [170]. This in turn induces transistor threshold voltage shifts, mobility degradation, and an additional, slowly increasing leakage current between source and drain. In addition, positive charge holes trapped within the field oxide, that normally do not contribute to device operation, can make it conductive, allowing current to pass formerly isolated device regions and increase leakage current between MOS transistors. But this field oxide leakage only occurs within NMOS transistors at the edges of two neighboring n-doped silicon regions [277], since the positive field oxide charge cannot interact with p-doped silicon. Leakage current increases the device's total power consumption and therefore its heat dissipation.

As the TID of a semiconductor mostly depends on size and thickness of the gate oxide, recent architecture scaling contributed in a positive manner to the TID limits of semiconductors [281]. From reference [282] follows that the threshold voltage shift is proportional to the gate oxide thickness squared, the total charge is proportional to the thickness and the capacitance is inversely proportional to gate thickness. Therefore, smaller transistors have shown better threshold voltage TID behavior than bigger ones and a gate size of only few nm may completely neutralize the threshold voltage shift due to the instant annealing process of all trapped charges near both gate oxide interfaces [283]. Therefore, TID becomes acceptable for gate oxide thicknesses below 5 nm as it can be found in 130 nm CMOS feature size and

below [245]. But the mentioned problems concerning unregulated leakage current flowing across the device are remaining the most dominating and limiting factor of modern CMOS devices. Even n-channel transistors based on SOI and SOS (see section 2.1.2) suffer from back-channel leakage as well as field oxide edge leakage [284, 285]. To mitigate these effects, several techniques have been developed. In principle, leakage current can be mitigated by using a hardened field oxide or at least p-doped guard rings between neighbored NMOS transistors. Furthermore, to prevent leakage current between source and drain it is possible to apply an annular gate electrode which separates both contacts from the gate oxide. But devices indicating TID effects due to ionizing radiation impact can also be cured within certain limits by annealing. Within this period, the separated electron-hole pairs try to recombine, regaining a stable state and therefore normal operation of the circuit. Especially oxide-trapped charge as well as noise decrease logarithmic with time [280, 170] due to electron tunneling from silicon into the oxide hole traps.

TID is given in Gray, whereas $\mathrm{Gy} = \mathrm{J} \cdot \mathrm{kg}^{-1} = 100$ rad. According to [286], it can be calculated in $\mathrm{rad} \cdot \mathrm{s}^{-1}$ by using the equation:

$$\mathrm{TID} = 100 \cdot \mathrm{LET} \cdot \Phi \cdot \mathrm{e}^{-}$$

whereas 100 indicates the Gy to rad conversation factor, LET references the linear energy transfer in $\mathrm{eV} \cdot \mathrm{cm}^2 \cdot \mathrm{kg}^{-1}$ as explained in section 2.4.3, Φ is the normalized particle flux in $\mathrm{s}^{-1} \cdot \mathrm{cm}^{-2}$ and e^{-} is the elementary charge of $1.602 \cdot 10^{-19}$C. A measured particle fluence therefore has to be normalized with regard to the recorded time period as well as the covered sensitive area. For example, the overall irradiated TID value of a beam test using ^{96}Ru ions of 1.69 GeV/u (accordingly LET = $3.3 \cdot 10^{12} \mathrm{eV} \cdot \mathrm{cm}^2 \cdot \mathrm{kg}^{-1}$) while irradiating with a total fluence Φ of $1.4 \cdot 10^{10} \mathrm{ions} \cdot \mathrm{cm}^{-2}$ can be calculated using the equation above to 740 krad.

TID threshold voltage and leakage current increment can be observed as long as a device is exposed to sufficient ionizing radiation. But devices acquiring a specific TID over a long time period have shown to behave different in comparison to devices which have been irradiated with the same TID in a few seconds [156]. Furthermore, it has been shown that TID also influences the SEU cross-section [287]. More details concerning TID in regard to FPGAs can be found in section 2.5.6. A study that investigates TID in commercially available CMOS processes, including an Intel 45 nm Wolfdale processor with tolerance up to 1 Mrad, can be found in [288].

Simulation of radiation-induced TID degradation effects within the CMOS sensitive oxide regions can be performed by using PSP [289], the advanced surface-potential-based MOSFET model for circuit simulation, and some extensions such as

FPGA series	Feature size	Max. TID	Reference
Virtex	220 nm	100 krad	[290]
Virtex-II	150 nm	200 krad	[290]
Virtex-II Pro	130 nm	250 krad	[290]
Virtex-4	90 nm	300 krad	[290]
Virtex-5	65 nm	~370 krad	estimation from figure 2.19
Virtex-6	40 nm	~410 krad	estimation from figure 2.19
Series 7	28 nm	>446 krad	[291]

Table 2.5: Maximum TID of Xilinx FPGAs according to the given references. Irradiation tests of Virtex to Virtex-4 generation devices have been performed according to MIL-STD-883 testing method 1019.5 at full 1019 dose rate [292]. Virtex-5 as well as Virtex-6 TID has been estimated from the trend line added to figure 2.19. The TID of a Series 7 Kintex-7 FPGA is taken from [291], obtained via irradiation with 180 MeV protons.

a physical model based on time-dependent kinetic calculations for charge generation, transport and trapping in silicon dioxide during exposure to ionizing radiation [281].

2.5.6 TID Impact on Xilinx COTS SRAM FPGAs

FPGA vendor Xilinx decided for a triple-oxide process, a composition of three different gate oxide thicknesses, to manufacture its in-house FPGAs as explained in section 2.1.3. In consequence, such a device's overall TID also involves three different maxima, but mostly reflects the worst oxide that fails first. Reference [290] contains a detailed current consumption analysis within all six transistor types of a 65 nm device, featuring thin, mid, and thick oxide in NMOS as well as PMOS while irradiating with a ^{60}Co source. It indicates, that the thin oxide transistor shows only very little radiation impact, while the thick oxide transistor exhibits the biggest variance in current consumption. Both CMOS transistors are affected, therefore solely field oxide impact can be ruled out.

An overview of the maximum TID for Xilinx FPGAs can be found in table 2.5 according to values indicated in [290, 291] as well as assumptions taken from the trend line added to figure 2.19. They reflect an increasing TID for shrinking transistor gate size. For 90 nm and 65 nm transistors, the authors furthermore forecast a maximum silicon TID of 1 Mrad with proper design margins [290], which include for exam-

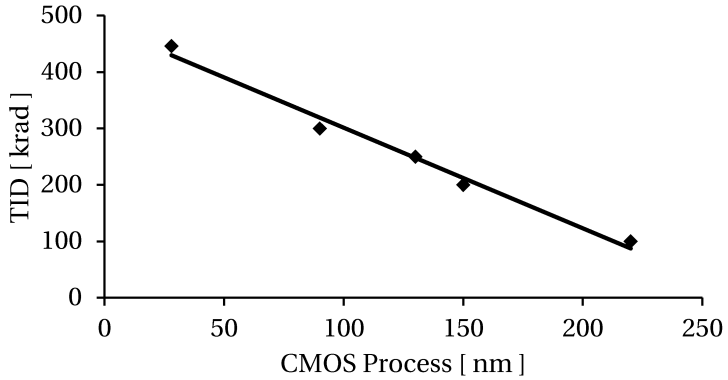


Figure 2.19: Maximum TID of Xilinx FPGAs according to values given in table 2.5. A trend line has been added which enables predictions for other feature sizes.

ple altered timing delays [170] due to the radiation induced shift in the transistor's ability to instantly switch from a low-conductance off-state to a high-conductance on-state as explained in section 2.5.5.2. A successful mitigation of such TID effects within FPGAs can only be performed by changing the material as well as design specifications of such devices. Therefore, Xilinx offers in-house radiation-hardened military- and space-grade qualified Virtex devices in parallel to the regular COTS devices, as depicted in section 2.6.3.

Beside of Xilinx, every company which aims to provide packaged semiconductors or even devices that include them to the global market has to take care of not exceeding a home country's local export restrictions. Radiation-hardened components or parts can be easily declared militarily critical and therefore export restricted as soon as they meet different requirements investigated in section 2.5.10. These circumstances may lead to in-house decisions in artificially increasing the radiation susceptibility of materials or at least to avoid extensive testing of the manufactured COTS components to be able to sell them unclassified in varying production batches and therefore open the global market.

2.5.7 SEE Impact and Mitigation on Xilinx COTS SRAM FPGAs

FPGAs, developed and offered by chip vendor Xilinx, are based on SRAM cell technology (see section 2.1.4) to provide a timing-efficient and speed-optimized basis for user-configurable logic designs, completed by a set of embedded ASICs to improve the response behavior of recurring basic calculations. Therefore, each device family

consists of a set of different components which can be interconnected by a global routing matrix. These components include fully user-configurable units, such as flip-flops, LUTs, or on-chip memory as well as different types of fixed and partially configurable units, such as clock managers, signal processors or even full PowerPC [130] and ARM microprocessors [132]. Therefore, an SEE within one of these components can result in different system behavior, ranging from unrecognized temporary glitches over storage bit errors up to global reset and clocking malfunction which may lock the whole system in an SEFI condition.

When taking into account, that it takes at least 10 SEUs in average to create an SEFI within a Xilinx FPGA [64, 221], then the SEFI cross-section of such a device lies at least an order of magnitude below the SEU cross-section. When furthermore taking into account that only about 10% of all configuration cells are used by a conventional FPGA design, this SEFI cross-section can easily differ at least two orders of magnitude from the plain device configuration bit cross-section [64] of a device. This has been more or less experimentally confirmed by the CERN ALICE TPC RCU, where 9% of all data taking sessions had to be stopped due to final electronics failures dominantly caused by SEUs within the Virtex-II Pro configuration memory [242]. [210] furthermore shows that the proton SEU cross-section is five to six and the MBU cross-section is two to three orders of magnitude larger than the SEFI cross-section. Giving some examples for the saturated SEFI cross-sections of Xilinx FPGAs at higher proton energies, the COTS Virtex-II XC2V1000 features $\sigma_{\mathrm{SEFI}} = 1 \cdot 10^{-6}\mathrm{cm}^2 \cdot \mathrm{device}^{-1}$ in average [293] while the radiation-hardened space-grade Virtex-5 XQR5VX130 device offers $\sigma_{\mathrm{SEFI}} = 7 \cdot 10^{-7}\mathrm{cm}^2 \cdot \mathrm{device}^{-1}$ [252]. Please note that the cross-sections are given for the full device and not per configuration bit. A detailed overview about different SEFI cross-sections (POR, SelectMAP, JCFG) in Xilinx FPGAs can be found in [293]. In addition, a comparison of FIT and MTBU rates of recent FPGAs can be found in appendix A.

As mentioned multiple times, Xilinx FPGAs are manufactured with three different gate dioxide thicknesses in a triple-oxide process as explained in section 2.1.3. In consequence, the individual layers used for on-chip memory, configuration cells, as well as input/output functionality result in different cross-sections and therefore upset rates for SBUs as well as MBUs. References [3, 4] have shown such experimentally gathered results for the radiation-tolerant Xilinx XQR Virtex-4 LX/SX devices in LEO: These results are depicted in table 2.6 and clearly indicate that the on-chip memory interconnects, made of the thinnest gate oxide, occupy only a small percentage of the whole device, but contribute with 37% to the overall upset ratio.

SRAM memory type	percentage of fabric	percentage of SEUs
CLB	91%	58% ± 2%
BRAM Interconnect	5%	37% ± 1%
DSP	0.7%	2.38% ± 0.3%
I/O Buffers	3%	2.52% ± 0.3%

Table 2.6: SEU distribution in radiation-tolerant XQR Xilinx Virtex-4 LX/SX devices during space flight in LEO between October 2011 and December 2012, according to [4].

The following subsections shall give a basic overview about which SEE are to be expected within the various components of Xilinx SRAM-based FPGAs and how they can be mitigated.

2.5.7.1 Signal Routing Impact

Routing resources within an FPGA are a composition of basically three components: First of all the static network of metal traces spread across the whole device area which is manufactured within the different CMOS layers of a silicon device. In case of longer wires, switchable buffers, controlled by SRAM configuration bits are added to improve the overall signal strength. The second component comprises multiple control bits within FPGA slices. They include the MUX configuration required for correct signal selection as well as several buffers used to provide configuration signals for embedded components such as LUTs. All of these configuration bits are made of SRAM cells and consequently they can be altered by radiation which may result in inverted, completely lost or incorrectly forwarded signals [294] as well as general component misbehavior. The third and proportionately largest component is the PSM with all its PIPs [139], whose configuration cells are fabricated as single SRAM cells to remain user-programmable, as explained in sections 2.3.1 and 2.1.4. In consequence, all PIPs are susceptible to ionizing radiation and can randomly be switched on or off. This results, according to [140], in the following SEUs as soon as a PIP is altered by an SET:

- Route breaking (PIP open): An active signal which is routed through a PIP gets mistakenly disconnected which results in information loss
- Route bridging (PIP short): A disabled PIP is altered to mistakenly connect two actively used signals which results in a concurrent data situation

- Route opening (PIP load): A disabled PIP is altered to mistakenly connect an active signal to an unused wire which results in unpredictable logic behavior if this signal drives another logic element and timing changes due to the additional load on the active wire

Furthermore, as soon as a PIP mistakenly shorts a power-driven wire to the device's ground level, an increase in power consumption can be observed [239] which additionally heats up the FPGA and therefore increases aging effects.

All of the previously mentioned, dynamically adjustable routing bits are included in the regular configuration bitstream of a Xilinx FPGA and can furthermore be read back from the device. This enables the possibility to compare their current state with the initial, golden configuration to easily identify routing errors, even at runtime. Moreover, recent devices, such as the Xilinx Series 7, can internally calculate an ECC checksum for single device configuration frames to simplify this process of error mitigation as shown in sections 2.7.7 and 5.2. A successful error mitigation approach for routing cells within the FPGA therefore is the refresh of configuration cells via the device's provided configuration interface. A possible dead-time between occurrence and correction of errors then has to be bypassed by adding circuit redundancy. Further details on configuration refresh and redundant firmware design can be found in section 2.7.

2.5.7.2 Look-up Table and Flip-Flop Impact

In addition to the conventional routing network, which contributes to the largest extent on the overall number of configuration bits, also LUT function, flip-flop initialization and flip-flop reset values need to be configured on device power-up and are therefore stored in SRAM cells. The frequency to upset one of these bits is smaller in comparison to the PSM, but given that every 6-input LUT can store $2^6 = 64$ bit for the calculation of a logic function, the total amount of plain LUT configuration cells is not negligible. The Xilinx Virtex-7 XC7V2000T for example has a configuration bitstream length of 447.34 Mbit [295] and contains 305,400 slices [146] which results in 78.18 Mbit of LUT as well as 2.44 Mbit of flip-flop configuration data (18%) that can suffer from radiation. An SEU within this combinatorial logic can therefore immediately change the output of an implemented logic function. But not all entries in an LUT contribute to the final function if the inputs are partially occupied and not every altered combinatorial output inevitably influences the final design behavior, since this normally depends on more than one signal. This of course is different when the LUT is used as distributed memory or shift register where an SEU imme-

diately changes the data word content. Furthermore, the LUT/memory/shifter and flip-flop/latch operation modes as well as several flip-flop timing interpretation parameters are defined by configuration bits stored in adjacent SRAM cells. A direct upset in these cells may result in data loss or timing errors as the DCM-generated clock phase is going to be misinterpreted.

Error mitigation for static LUT content can therefore be provided by refreshing the configuration cells of the FPGA. Dynamic LUT memory, shift register or flip-flop data can only be secured by adding redundant circuits to the whole design. Further details on configuration refresh and redundant firmware design can be found in section 2.7.

2.5.7.3 DCM and Clock Routing Impact

Clocking within recent Xilinx FPGAs is basically organized in Clock Management Tiles (CMT) (up to 24 in Series-7 devices [296]) which offer a DCM/MMCM with PLL and clock buffers as explained in section 2.3.1 to perform frequency synthesis, phase-shifting, de-skew, jitter filtering, and duty cycle programming for high input frequency ranges [296]. These clocking resources are not hardened against radiation, not even within Xilinx' Radiation-Hardened by Design (RHBD) FPGAs [252]. Hence, the clock buffer may get corrupted by SEUs, and PLLs may get desynchronized while suffering from SETs, resulting in clock corruption across CMTs. In consequence, the vulnerability of PLL circuits has been investigated in [297] with the result that first of all the sensitivity against SETs increases with the operating frequency and second that no pulses were lost up to a tested limit of 200 MHz. Only duty cycle errors were observed. This limiting factor can been neutralized in a clock distribution network by mitigating the radiation-caused jitter and adding redundancy as shown for an RHBD device in [298]. A sample circuit has been successfully built in 90 nm feature size and verified up to an LET of 100 $\mathrm{MeV \cdot cm^2 \cdot mg^{-1}}$. A deeper investigation of PLL and DCM under beam conditions can be found in [252], exemplary for the Xilinx RHBD Virtex-5Q. No permanent errors represented in the form of clock glitches, frequency changes or final breakdown that would have required a full device reconfiguration were observed up to an LET of 60 $\mathrm{MeV \cdot cm^2 \cdot mg^{-1}}$.

Successful error mitigation of clocking resources is a difficult task. It requires the definition of independent CMT regions on a single device which are fed by independent DCM/MMCM primitives that share an external oscillation crystal input. The regions can afterwards be populated with independently operating circuits whose results finally have to undergo a majority decision. Using a single shared clock net-

work for the whole design or even for redundant parts of the design does not result in a significant decrease in error susceptibility [299].

2.5.7.4 I/O Buffer Impact

External signals entering an FPGA as well as internal signals leaving it can be buffered to improve data integrity and match signals exactly to the internal clock phase. This is realized in Xilinx FPGAs by the use of user-configurable IOBs nearby the corresponding bonding wire pin. Some of these pins are capable to drive high speed serial communication while others are for general purpose IO only. While the data buffers of COTS FPGAs itself are made of regular flip-flops that can be upset by radiation impact, the IOB parameters such as signaling direction or I/O standard [152] are preserved in SRAM cells. An SEU within one of these parameters may therefore alter the basic behavior of the specific pin, resulting for example in spontaneous inversion of the signaling direction. This effect has been extensively investigated in [65] by taking the example of a Xilinx Virtex XCV1000 with the outcome that only 1 of 324 IOB configuration bits and 2 two-bit combinations were able to change the IOB behavior.

Error mitigation for the IOB configuration cells is handled via refresh of the corresponding SRAM cells within the device. But the I/O pins in general heve to be present redundant, to provide independent signals for later processing in different legs of the design. The Xilinx-recommended approach hereby is to hard-wire redundant pins outside of the FPGA to get split input signals as well as an analog majority decision for output signals by simply overwriting a wrong signal as stated in [157] or [154]. Exact requirements regarding trace length, drive strength or termination resistors can be found in [157].

2.5.7.5 BRAM On-Chip Memory Impact

To provide a flexible single/dual port fast access on-chip memory without the need for constant data refresh as known from DDR memory, Xilinx BRAM is made of SRAM cells featuring the smallest triple oxide thickness. In consequence, its SEU cross-section is slightly increased (up to five times [300]) in comparison to conventional SRAM configuration cells as used for the PSM [214, 59] or at least equal for features sizes of few nm (see section 2.5.4). Taking into account that the Virtex-7 XC7VX330T features $750 \cdot 36 \cdot 1024 = 27.6$ Mbit of BRAM [301] upon a total configuration bitstream size of 111.24 Mbit [295], at least 25% of all bits can be upset by BRAM SEUs. But a BRAM bit flip does not compulsorily change the behavior of a

circuit design, as it can only be sampled erroneous when read out. Therefore, bit errors in data memory are not considered that critical in comparison to the previously mentioned circuit components, precisely because flipped bits can be detected by a hardware-embedded SECDED circuit in case the BRAM primitive is configured to use this ECC feature [302, 301]. The hence additionally required data bits which preserve the Hamming(72,64) code have been added to the regular memory in the form of 4 extra bits for each 32 bit data word, forming 36 bit words. Since 4 bit are not sufficient for such an SECDED algorithm, two regular BRAM primitives have to be combined to a single ECC BRAM entity, which offers 64 bit words for user data as well as 8 bits for security. The ECC runs in background, fully transparent for the user and automatically returns corrected values as well as a status register to indicate the validation result. Unfortunately, erroneous data remains in the memory cells of COTS Xilinx FPGAs even after detection and in case errors accumulate in the same memory word they cannot be recovered anymore. Therefore it is necessary to check the ECC status register and initialize a write back cycle in case an SEU has corrupted BRAM data (see section 2.7.8 for details). This behavior is different in radiation-hardened FPGA devices [303] as described in section 2.6.3.

The actual cross-sections for Xilinx BRAM cells of various feature sizes can be found in [59]. In this vendor report, the BRAM vulnerability for series-7 devices is given with $\sigma = 6.32 \cdot 10^{-15}$ cm^2·bit^{-1} ± 18% and an FIT (failures per 10^9 hours) of 75 Mbit^{-1}. This means that for a device such as the mentioned XC7VX330T, the whole 27.6 Mbit of BRAM will take $1/(75\ \text{Mbit}^{-1} \cdot 27.6\ \text{Mbit} \cdot 10^{-9} \cdot 24 \cdot 365) = 55$ years to upset at least once from neutrons when operated in Earth's atmosphere at sea level. To verify the vendor-given FIT-rate the known formula from section 2.5.4.2 as well as the >20 MeV neutron flux at sea level of about $\Phi = 4 \cdot 10^{-3}$ s^{-1}·cm^{-2} [304, 191] can be used. In this case, it is calculated at a slightly increased value of $\text{FIT/Mbit} = \Phi \cdot \sigma \cdot 3600 \cdot 10^9 \cdot 10^6 = 91$ Mbit^{-1}, which may result from different values for Φ. Of course the FIT rate in actively irradiated environments such as particle accelerators is much higher and cannot be compared to these values. A general comparison of the FIT and MTBU rates for BRAM in recent FPGAs can be found in appendix A.

As with any other primitive, the various static configuration attributes of BRAM, such as EN_ECC_READ or EN_ECC_WRITE, as well as the static BRAM interconnect configuration are preserved in SRAM cells, unhardened, and therefore susceptible to radiation effects. This number even increases when the dual port BRAMs are configured to act as FIFOs. An upset within one of these cells may completely switch of

required functionality such as the ECC circuit without any notification to the user, therefore, a continuous refresh of this static configuration is inevitable (see section 2.7.7 for details).

2.5.7.6 DSP - Digital Signal Processor Impact

Multiple DSP slices within a Xilinx FPGA provide arithmetic functions as explained in section 2.3.6. While the total number of DSPs when introduced with the Virtex-II devives was limited to 168 [138], a recent Virtex-7 can hold up to 3600 [60] of such ASICs. In addition, the configuration options were extended over time, starting with basic read/multiply/accumulate operations and later-on adding Single-Instruction-Multiple-Data (SIMD) arithmetic units that operate in parallel. While the basic logic circuits are realized as hardware ASICs, the internal cache required for the high frequency operation as well as the DSP primitive configuration attributes are manufactured as unhardened flip-flops or SRAM cells and can hence be upset by radiation. This has been proven in [299] while irradiating with heavy ions.

A successful error mitigation approach therefore has to include the configuration refresh to recover correct behavior of the DSP primitive (see section 2.7.7) as well as a dynamic data preservation strategy which ensures at least detection of erroneously generated calculation results by the redundant use of circuits as explained in section 2.7. More information regarding recent FPGA DSP slices can be found in the technical documentation [153].

Recent simulations, which investigated functionally identical designs, implemented in DSPs on the one side as well as conventional FPGA logic circuits on the other side, have shown, that the overall error rates do not essentially differ between both solutions [305]. The only advantage of the DSP-implementation was found to be the significant reduction in configuration circuit usage.

2.5.7.7 Housekeeping-Primitives Impact

Beside of all functional primitives within an FPGA, some housekeeping circuits such as JTAG, SelectMAP, SPI, or the Internal Configuration Access Port (ICAP) are present only to provide the most basic programming, read-back as well as operational features of the device. These interfaces have global access to the FPGA resources, including permission to initiate a Power-On Reset (POR) as investigated in [214], which completely clears the whole device configuration in preparation for new firmware. Therefore, a critical radiation impact within one of their internal control registers or state machines may lock the whole device in an SEFI as observed

for the SelectMAP interface in [306]. This leads to the accumulation of upsets in the device until it finally gets unusable and has to be reset. Other critical upset issues include modifications of the ICAP port-select register, modifications in the 'GLUTMASK' register which excludes distributed LUT-memory areas from reconfiguration, or modifications of the 'GTS_CFG' register which causes all IOBs to drive high-impedance tri-state [295].

Since there is no possibility to mitigate radiation errors on these basic interfaces within a single FPGA, the only option is to use a combination of internal and external watchdog to monitor the behavior of such a device. This can be done for example by observing the behavior of the FPGA's 'DONE' pin, which indicates an internal POR that needs to be handled by initiating a full reconfiguration cycle. In the last instance, every kind of uncorrectable misbehavior or device hangup can be resolved by pulsing the 'PROGRAM_B' pin, which finally resets the device.

2.5.8 Radiation Impact on ASIC Semiconductors

In contrast to FPGAs, ASICs do not offer a flexible configuration matrix. The internal routing and function generators are fixed at the moment of production. But similar to FPGAs, they are manufactured in a CMOS process which undergoes TID effects. They also contain flip-flops as well as on-chip memory arrays made of SRAM or DRAM which store dynamic data but suffer from SEE. Irradiation of DRAM cells can furthermore decrease the retention time, defining the required memory cell refresh interval, which in consequence results in data loss as investigated in [307]. These radiation effects also apply to hard-wired micro-computers such as the PowerPC or ARM cores when embedded within FPGAs as seen from [308]. As explained in section 2.1.4, even the COTS Intel desktop and server CPU's die size is composed of up to 50% SRAM cells [75] to efficiently cache instructions and data, but also significantly increases the vulnerability to soft errors. Another prominent SRAM ASIC in the field of CBM (see section 1.3) is the SPADIC v1.0 TRD FEE readout chip, manufactured in UMC 180 nm feature size. It embeds 44 blocks of Faraday SRAM memory [309] which provide a shift register matrix in the digital part of the chip [310], but also occupy a significant amount of the total chip die as clearly seen from the layout.

Since there is no possibility to add radiation-tolerant design or fault-tolerant circuits to an ASIC a posteriori, the additional logic to fulfill such requirements has to be added already at the design stage. The mentioned SPADIC chip for example uses at least guard rings to protect the analog part of the chip. Other mitigation strategies for CMOS ASICs can be found in section 2.6.2.

2.5.9 Radiation Impact on Hybrid Semiconductors

The term 'hybrid' in the sense of semiconductor architectures and technologies defines a product that combines the characteristics of multiple device categories to form a completely new type. One of such devices is available from eASIC Corporation. This 'nextreme' chip combines the cost-effectiveness of ASICs in form of mass-production of a generalized device, with the programming capabilities of FPGAs in form of programmable LUTs and routing. This approach is comparable to the first PROMs (see section 2.2.4) and their high voltage vaporization of metal contacts which enabled a definition of custom routing. To provide such features, the eASIC chips offer multiple metal layers with fixed and more or less universally spread routing structures that are interconnected by standard vias. Some of these vias as well as a set of LUTs are one-time programmable when the ASIC gets initialized from a given design. A particular version of this chip furthermore offers reconfigurable LUTs made of embedded SRAM cells which have to be configured at power-up from an externally connected SPI memory device [311].

Such devices might offer a shorter time to market or cheaper manufacturing costs for smaller production series, but unless they are not qualified for the usage in radiation environments, the susceptibility has to be considered as a complex mixture of all involved technologies. Hybrid FRAM storage cells for example (see section 2.1.5) may be radiation-tolerant by design, but as long as they are embedded and make use of standard CMOS technology, for example to sense signal levels, they have to be considered as radiation susceptible. More information regarding FRAM as well as irradiation results can be found in sections 2.1.5 and 5.6.2.

2.5.10 ITAR - International Traffic in Arms Regulations

As the headquarters of all major FPGA vendors are located in the USA (Xilinx, Altera/Intel, Atmel, Lattice, Actel/Microsemi, QuickLogic, Aeroflex, SiliconBlue), part and component import becomes a major challenge as soon as the specification unveils radiation tolerance or military usability in its feature list. The justification for this issue lied within the U.S. Directorate of Defense Trade Controls' (DDTC) International Traffic in Arms Regulations (ITAR) §121.1, especially category XV.d [312], which forced nearly all of the space-radiation-hardened microelectronic circuits to get on the government-controlled United States Munitions List (USML), whether it could be used for a weapon or not. Therefore, a simple integrated circuit that offered radiation tolerance, including some FPGAs, was classified to be of defense grade as soon as it met the following criteria (according to [312]):

- at least 500 krad TID in silicon (see section 2.5.5.2)
- at least 500 Mrad/s dose rate upset threshold in silicon
- at least $1 \cdot 10^{14}$ n·cm^{-2} neutron fluence (1 MeV equivalent) (see section 2.5.5.1)
- an SEU rate of $1 \cdot 10^{-10}$ errors per bit per day or less (see section 2.5.4.2) for CREME96 geosynchronous orbit at Solar Minimum Environment
- SEL free (see section 2.5.4.4) and dose rate latch-up threshold of 500 Mrad in silicon

In June 2014, within the process of the export control reform, the previously relevant microelectronics radiation paragraphs §121.1 category XV.d and others were altered or removed from ITAR §121 [313], revised and integrated into the standard Export Control Classification Numbers (ECCN) within the Export Administration Regulations' (EAR) Commerce Control List (CCL) §774 [314]. These new ECCNs now comprehensively classify all items into ten broad categories, for example 3 for Electronics and 9 for Aerospace/Propulsion, as well as five product groups, for example A for end user items and D for software (see §772 [315]). A three-digit trailing numeric appendix finally defines each item. Therefore, a spacecraft may belong to 9-A-515 whereas the software for its development and operation belongs to 9-D-515. Each of these ECCNs now provides its individual export controls according to a list of exceptions and extensions regarding national security, weapon technology, and others. The new classification draws a clear line between special interest equipment and the more moderate dual-use components while specifying the maximum allowed shipping value (LVS) in parallel. This finally opened the market for a lot of commercially available rad hard integrated circuits as well as satellite systems which were previously listed on the ITAR §121 USML [312] and are now available under the EAR §774 CCL [314] Wassenaar Arrangement Munitions List (WAML) – still export controlled, but only restricted to a few well specified countries. Together with the ECCN classification, the radiation properties of such items have also been altered. According to the following excerpt taken from [314], export restrictions for several countries may apply for the items below which offer radiation hardness by design. In case any of the mentioned criteria is met, the EAR CCL has to be considered to investigate possible consequences before planning a project that involves such items.

- 2B007.c: Robots or end-effectors designed or rated as radiation-hardened
  - at least 500 krad TID in silicon without operational degradation
- 3A001.a: Electronic components including FPGAs (3A001.a.7) designed or rated as radiation-hardened to meet any of:
  - at least 500 krad TID in silicon

  - at least 500 Mrad/s dose rate upset threshold in silicon
  - at least $5 \cdot 10^{13}$ n·cm^{-2} neutron fluence in silicon (1 MeV equivalent)
- 4A001.a.2: Electronic computers and related equipment designed radiation-hardened (except civil aircraft) to meet any of:
  - at least 500 krad TID in silicon
  - at least 500 Mrad/s dose rate upset threshold in silicon
  - an SEU rate of $1 \cdot 10^{-8}$ errors per bit per day or less
- 4A101.b: Analog and digital computers designed as ruggedized or radiation-hardened for missiles
  - at least 500 krad TID in silicon
- 6A102: Radiation-hardened detectors designed for protecting against nuclear effects and usable for missiles
  - at least 500 krad TID in silicon
- 6A203.d: High-speed cameras, imaging devices, lenses designed or rated as radiation-hardened
  - at least 50 Mrad TID in silicon without operational degradation
- 9A515.d: Spacecraft microelectronic circuits designed or rated for defense articles or WAML items and rated to meet:
  - at least 500 krad TID in silicon
  - at least 500 Mrad/s dose rate upset threshold in silicon
  - at least $1 \cdot 10^{14}$ n·cm^{-2} neutron fluence in silicon (1 MeV equivalent)
  - an uncorrected SEU sensitivity of $1 \cdot 10^{-10}$ errors per bit per day or less for CREME-MC geosynchronous orbit at Solar Minimum Environment for heavy ion flux
  - an uncorrected SEU sensitivity of $1 \cdot 10^{-3}$ errors per part or less at a proton fluence (>50 MeV) of $1 \cdot 10^{7}$ p^{+}·cm^{-2}
- 9A515.e: Spacecraft microelectronic circuits designed for defense articles or USML Category XV items and rated to meet:
  - a TID between 100 krad and 500 krad in silicon
  - an SEE (SEL, SEB or SEGR) immunity to an LET of at least 80 MeV $\cdot$ cm^2 $\cdot$ mg^{-1}

Famous examples in the field of FPGAs that were subject to ITAR are the Actel/Microsemi Antifuse RTAX as well as the Xilinx radiation-hardened, space-grade Virtex QR series [317, 318, 303] (see section 2.6.3). To clarify recent changes for these products, table 2.7 summarizes the EAR CCL ECCNs of selected FPGA series. Items that moved from ITAR to EAR can be identified by the new category number 9A515. As

FPGA Product Series	EAR CCL ECCN
Microsemi RTSX	3A001.a.2.c
Microsemi RTAX	new 9A515.e
Microsemi ProASIC3-RT	3A001.a.2.c
Microsemi RTG4	new 9A515.e
Virtex XQR	new 9A515.E.2 [316]
Virtex-II XQR	new 9A515.E.2 [316]
Virtex-4	3A991.D [316]
Virtex-4QV	new 9A515.E.2 [316]
Virtex-5	3A991.D [316]
Virtex-5QV	new 9A515.E.2 [316]
Virtex-6	3A991.D [316]
Virtex-6Q	3A001.A.2.C [316]

Table 2.7: EAR CCL ECCNs of selected FPGA series. Items that moved from ITAR to EAR can be identified by the new category number 9A515.

seen from these classification, software can also be restricted by the government, as recently done for the Mentor Graphics® Precision® Hi-Rel tool (ITAR §121.1 category XV.f [319]) which automated the creation and synthesis of fault-tolerant HDL circuits [320] but had been discontinued later-on [321] (see section 2.9.1).

Working with ITAR products in non-US countries implies multiple difficulties, as ITAR includes technology transfer. Therefore, products which have been developed by the use of restricted information are automatically classified in this way. Since ITAR enforces very strict handling of technology and component use between all involved employees, even restricted to approved nationalities only, it is nearly impossible to develop or release electronic products or software based on these prerequisites. Even software licenses at CERN had to be canceled due to recent ITAR declaration.

In parallel to the USA, the European Commission adopted nearly the identical regulation for its own area of influence according to the Wassenaar Arrangement from 1996. It is known as 'Commission Delegated Regulation (EU) No 1382/2014' and defines export, transfer and transit of dual-use items to other countries [322].

2.6 Hardware Radiation Hardening

If semiconductors are intended to be specified for operation in radiation environments as depicted in section 2.4, special care has to be taken already at design phase to harden the internal circuits against the radiation effects explained in section 2.5. To reach this aim, various techniques are currently available:

One option lies in the selection of less critical doping, material and insulation, since substrate thickness and hence the maximum funneling length plays a major role in the generation of SET effects [323], especially for SOI MOSFETs (see section 2.1.2). Unfortunately, this mitigation technique has to be reconsidered for every new step in transistor feature size down-scaling and therefore capacitance change as it directly impacts the charge collection mechanism [231]. This also applies to the increasing sensitivity to single event cross-talk effects arising from the simultaneously shrinking etched metal wires [324].

Another technique comprises design rules for the mitigation of MBU events in logic memory arrays. Resilience can be improved by physically interleaving the placement positions of individual bits that belong to a single storage word [325]. This technique is of essential importance to improve the efficiency of logical SECDED algorithms.

Several other options to physically harden a devices against harmful radiation include the addition of circuit redundancy, guard rings, ELTs or other techniques as described within the following sections. If non of such options is available, additional shielding matter can be applied, though it demands synchronization with the characteristics of the operational radiation field and therefore it requires knowledge of the final application scenario.

In some literature, there is a distinction between the terms radiation-tolerant and radiation-hard. While radiation-tolerant usually refers to devices that can withstand at least 100 krad in silicon, radiation-hard guarantees a device qualification to handle at least 300 krad (Si). In addition, a semiconductor device is considered to be RHBD as soon as it can handle these qualification protocol dependent TID levels as well as withstand a particle single event energy of 120 MeV [326]. Sometimes, it furthermore includes the necessity for manual integration of circuit redundancy as in the Actel ProASIC3-RT [327]. In case of Xilinx COTS SRAM FPGAs, at least the TID requirement is already met for their latest devices from table 2.5. As total dose tolerance depends on the final usage scenario which can be completely different between space and particle accelerators, the following sections will use both terms synonymously.

In contrast to firmware fault tolerance (section 2.7), hardware radiation hardening includes all actions taken to prevent radiation from harmfully penetrating physical matter while the device firmware afterwards has to deal with the occurring faults in forms of detection and correction. Both processes are aggregating and as soon as radiation hardness exceeds its capabilities, fault tolerances takes over. Both definitions are often merged, as there exist devices that offer built-in fault-tolerant circuits by design that are fully transparent for the user. The following sections try to present a very basic overview about common practices in radiation-hard circuit design for ASICs and FPGAs to impart knowledge about its complexity and the differences to firmware fault tolerance.

2.6.1 Shielding Effects

Mitigation of radiation effects in biological or technological systems, including semiconductors, can be easily achieved by the application of shielding in a more or less moderate extent [274]. This statement may be true for the most medical and low energy terrestrial applications, but it should not be considered to be the only protection, as shielding can also induce new and opposing effects as soon as the radiation environment slightly changes as shown in the following paragraphs.

The general effect of shielding is to completely stop or at least weaken the incoming radiation until it cannot harm the vulnerable system anymore. Depending on the type of radiation, as explained in section 2.4, this aim can be reached by applying a simple sheet of paper in case of α or ultraviolet radiation, few aluminum foil in case of β particles, or dense matter with high atomic number, for example lead, in case of γ or X-ray radiation [328]. The denser and thicker the material, the more effective its shielding due to the deceleration-effects explained in section 2.4.1. But this effect rapidly reaches its maximum extent of usability in case of high energy particles such as neutrons, protons or heavy ions found in CME, GCR or particle accelerator beams. A charged proton with an energy of at least 2 GeV for example may easily penetrate lead in a depth of 1.3 m [162]. In addition, lead lacks of flexibility as well as chemical stability. In consequence, new composite materials are necessary when flexible shielding is required. [329] presents an isophthalate resin polymer with lead oxides to fill this requirement gap while [330] recommends to use a graded-Z shield made of sandwiched materials with high atomic numbers that provides an ideal trade-off between shielding and low mass, especially important for space microelectronics.

Nowadays COTS semiconductor devices implicitly use shielding to protect embedded silicon from harmful radiation as it occurs in Earth's terrestrial environment (see section 2.4.6) for which they have been specified. This shielding is provided by the thin, internal passivation layers as well as the thicker, external metal lid package. Both are designed to withstand a well defined spectrum of radiation only. Therefore, as soon as the irradiation energy rises, a significant increase in SEE can be observed before the energy exceeds a threshold value where particles just quickly pass by without much time for ionization. This effect has been shown in [288] and can be explained with the higher LET of low energy particles as well as the resulting Bragg Peak in case of energy depletion (see section 2.4.3). In consequence, it is of significant importance to adapt the shielding of a semiconductor to fit the radiation field in which it should be finally operated.

Shielding is also extremely important for space flight missions, since local Solar Particle Events (SPE) (see section 2.4.4) are able to generate radiation doses which can significantly harm the used electronic equipment. To give an example, one of the largest recorded SPE in human history, the Carrington event of 1859 [331] which caused visible auroras in Florida, was estimated to have a TID of 11 krad behind a shield of 1 mm of aluminum [332]. But this value can be decreased to 0.1 krad behind 20 mm of aluminum [332]. An even worse situation is given for the planned ESA satellite missions to the Jovian moon Europa, because of its local radiation belt which has been calculated to generate a TID of about 10 Mrad per year behind 4 mm aluminium shield [333]. More information regarding shielding requirements in space missions can be found in [334].

Finally, shielding might improve radiation mitigation, but it should not be overestimated, especially in applications with high energy particles, such as particle accelerators and space missions.

2.6.2 Radiation-Tolerant ASIC CMOS Design

All space grade CMOS ASICs available from BAE Systems, Aeroflex, Atmel, Honeywell, Intersil, Maxwell Technologies, STMicroelectronics, Texas Instruments, and many more have one thing in common: They have to offer physical radiation tolerance based on a combination of well investigated materials, special design techniques, as well as shielding. In contrast to the dynamically configurable switching matrix within FPGAs (see section 2.3.1), the routing between logic gates in ASICs is fixed by its function and therefore cannot be altered by radiation. This significantly reduces the sensitive area of an ASIC, but does not provide any solution

for transistors and hence all composited logic gates and storage elements such as flip-flops, registers, or memory arrays. To provide better protection for these elements, the major goal is to improve the transistor's critical charge Q_{crit} by increasing its node capacitance, which basically results in larger transistor design and higher power consumption. In addition, several advanced silicon manufacturing techniques are available, such as epitaxial SOB substrates [335] or SOI (see section 2.1.2) with SIMOX, ZMR or BESOI [336]. To furthermore protect such cells from being damaged by SELs, several combinable *Circuit Level* layout design techniques are available [337], for example:

- **Enclosed Layout Transistors (ELT)**: Reduction of leakage current between source and drain of transistors by completely surrounding source and drain with each other with the polysilicon gate in between them.
- **Shallow Trench Isolation (STI) / Deep Trench Isolation (DTI) Guard Rings**: Reduction of leakage due to charge hole trapping in the STI/DTI oxide between neighbored MOSFETs by application of heavily p-doped rings.
- **Layout Gapping**: Separation of a transistor's sensitive nodes by increasing their physical distance to each other. This requires additional silicon space and increases the total chip dimensions. To solve this problem, sensitive nodes of neighbored transistors can be interleaved with each other. The additionally required interconnection routing can be shifted to the overlying metal layers and silicon size remains unchanged or even shrinks.

Based on these methods, the resulting, hardened transistors are combined to basic function gates or cells. These cells, for example SRAM (see section 2.1.4), can furthermore be extended with fault-tolerant design techniques implemented directly within hardware. They exist with differing complexity as well as reliability benefit to SEE as well as cumulative effects. Most of them are based on redundancy by using four or more of the available standard storage nodes [338]. To give an impression about the complexity, a few techniques are selected:

- **Built-In Soft Error Resilience (BISER) cell**: A robust scan cell that extends the basic scan flip-flop with an output circuit for test and SEU resilience [339, 340]. This output circuit is be made of two redundant DMR latches with a Muller C-gate which offers hysteresis and hence returns the previous value in case of latch divergence. This successfully prevents SEU migration at minimum cost. A proposal for an improved version, CSER, can be found in [341].
- **Heavy Ion Tolerant (HIT) cell**: A standard HIT memory cell is composed of twelve transistors [342]. Each six of them form a structure that is cross-

coupled to the other one, similar to the SRAM cell (see section 2.1.4). Additional feedback signal paths provide SEU hardness but cannot mitigate SET on the input signal.

- **Single Event Resistant Topology (SERT) cell**: The SERT latch is composed of twelve transistors and offers two independent inputs as well as outputs [343, 344]. Both outputs will not be updated unless the input values match. Therefore, it is capable of mitigating an SEU within the cell as well as an SET on one of the input signals. Two SERT latches can be combined to an SERT register.
- **Dual Interlocked Storage Cell (DICE)**: The DICE cell (latch or register) is composed of four symmetric, interlocked CMOS inverters with twelve transistors in total [345]. They provide a dual-node storage redundancy cell (not to be confused with DMR) with a doubled input signal that is connected to both of them. This successfully mitigates SEUs as investigated in [346] and leads to excellent proton and heavy ion induced upset tolerance. The internal storage value of a DICE cell can only be corrupted by exceeding Q_{crit} of two redundant inverters simultaneously, for example by a particle strike from a few specific irradiation angles. Since SETs on the input signal can still upset a standard DICE cell, several extensions based on temporal delay filtering were proposed, for example Delay-Filter (DF-DICE) [347, 348], True Single-Phase Clock (TSPC-DICE) [349], Adaptive-Coupling Flip-Flop (DICE-ACFF) [350], Temporal DICE Master Slave Flip-Flop (Temporal DICE MSFF) [351] as well as Temporal DICE Flip-Flop (TDFF) [352] which furthermore offers an adjustable propagation delay to handle various SET pulse widths [353].

Finally, radiation-tolerant ASIC design does not only affect the qualification for radiation environments, it also increases the general device reliability by mitigating the numerous analog effects which arise from deviations in the initial wafer fabrication and subsequent aging, such as Electromigration, Stress Migration or Electrostatic discharge. Some of the proposed techniques, such as the DICE cell, can be directly applied to SRAM technology for FPGAs, while others strictly conflict with the basic principles of reconfiguration. But all of them have one major thing in common: They add a significant area overhead to increase radiation susceptibility and hence significantly increase the fabrication costs. More information regarding radiation- and fault-tolerant ASIC design can be found in [354, 355, 263].

2.6.3 Radiation-Tolerant FPGAs

Beside of the COTS consumer FPGAs (see section 2.3), there exists a subset of devices which are specifically designed for applications in radiation environments. Based on radiation-hardened ASICs, they mainly use selected shielding material, advanced silicon MOS design as well as internal redundancy to provide configurable memory cells that can tolerate penetrating ionizing and non-ionizing particles. As these mitigation techniques are integrated within the physical chip itself, a potentially sufficient radiation-hard qualification level can already be guaranteed without application of any fault-tolerant user-design. Appropriate hardening techniques, as described in section 2.6.2, have to be chosen depending on the underlying FPGA architecture, but nearly all of the available methods add area overhead, degrade speed and increase manufacturing and qualification costs significantly. Therefore, first research dealing with the insertion of redundant hardware circuits in Antifuse and SRAM FPGAs mainly focused on production yield improvements [356] and totally underestimated the repair capabilities after irradiation damages. The proposed techniques included the addition of spare-parts, extra wiring and special selector units to tolerate hardware damage.

To give a basic overview about the most common types as well as available representatives of radiation-tolerant FPGAs, antifuse, flash memory as well as SRAM FPGA architectures have been selected. For more information, please refer to [357].

2.6.3.1 Radiation-Tolerant Antifuse FPGAs

One-time programmable, non-volatile, antifuse FPGAs provide a cheap alternative to reconfigurable radiation-hardened FPGAs and have therefore been announced to be the best option at least for satellite applications [366] although flash based models caught up [327] during the last years (see section 2.6.3.2). The configuration of such devices is initialized/melted by applying a higher voltage (about 10 V to 20 V) to special high resistance, antifuse switches based on Oxide-Nitride-Oxide (ONO) material layers and therefore cannot be altered by SEE at runtime anymore. But the susceptibility of all remaining internal transistor-based elements within COTS antifuse FPGAs can be compared to conventional ASICs, therefore, the necessity of specially designed, radiation-hardened devices is obvious. Examples are given with the Actel/Microsemi 250 nm RTSX and 150 nm RTAX antifuse FPGA families. While both devices offer radiation hardened registers that hold an SEU LET_{th} of $37 - 40\ MeV \cdot cm^2 \cdot mg^{-1}$ as well as an SEL LET_{th} of $104 - 117\ MeV \cdot cm^2 \cdot mg^{-1}$, the RTSX is specified for TID_{max} = 100 krad whereas the RTAX is qualified for TID_{max} =

	Microsemi RTSX	Microsemi RTAX	Microsemi ProASIC3-RT
RHBD Device	yes	yes	no
ITAR prior the ECR	no	yes	no
EAR CCL ECCN	3A001.a.2.c	9A515.e	3A001.a.2.c
Feature Size [nm]	250	150	130
Technology	antifuse	antifuse	flash
Register SEU LET_{th} $[MeV \cdot cm^2 \cdot mg^{-1}]$	40 [358]	37 [359]	68 [360]
Register SEU rate in GEO $[Errors \cdot bit^{-1} \cdot day^{-1}]$	$1 \cdot 10^{-10}$ [358]	$1 \cdot 10^{-10}$ [359]	$\overline{TMR}$: $3.0 \cdot 10^{-6}$ [327] TMR: $1.6 \cdot 10^{-9}$ [327]
SRAM SEU rate in GEO $[Errors \cdot bit^{-1} \cdot day^{-1}]$	n.a.	$1 \cdot 10^{-10}$ [359]	$4.5 \cdot 10^{-5}$ [327]
Device TID_{max} [krad]	100 [358]	300 [359]	20 [327]
Configuration SEL LET_{th} $[MeV \cdot cm^2 \cdot mg^{-1}]$	104 [358]	104 [359]	55 [327]

	Microsemi RTG4	Xilinx Virtex-5QV	Atmel ATF280
RHBD Device	yes	yes	yes
ITAR prior the ECR	yes	yes	no
EAR CCL ECCN	9A515.e	9A515.e.2	3A001.a.7.b
Feature Size [nm]	65	65	180
Technology	flash	SRAM	SRAM
Register SEU LET_{th} $[MeV \cdot cm^2 \cdot mg^{-1}]$	37 [361]	n.a.	30 [362]
Register SEU rate in GEO $[Errors \cdot bit^{-1} \cdot day^{-1}]$	$1 \cdot 10^{-10}$ [361]	$3.3 \cdot 10^{-10}$ [363]	n.a.
SRAM SEU rate in GEO $[Errors \cdot bit^{-1} \cdot day^{-1}]$	$1 \cdot 10^{-10}$ [361]	$3.8 \cdot 10^{-10}$ [303] (38 Mbit)	$3.6 \cdot 10^{-13}$ [364] (1.7 Mbit)
Device TID_{max} [krad]	100 [361]	1000 [303]	300 [365]
Configuration SEL LET_{th} $[MeV \cdot cm^2 \cdot mg^{-1}]$	110 [361]	100 [303]	80 [365]

Table 2.8: Comparison of radiation-tolerant FPGAs. A register is considered as a 1 bit flip-flop in device feature size. For details regarding the particular radiation assurances, please refer to the indicated references. Costs of acquisition at least for the space-grade RHBD devices should not be underestimated, but at least for the 100,000 USD Xilinx Virtex-5QV there is a quantity discount on request.

300 krad in silicon [358, 359]. In consequence, the RTSX is freely available while the RTAX was subject to ITAR (see section 2.5.10). Nevertheless, cell hardening for both device's flip-flops is based on TMR of the D-type flip-flops in combination with asynchronous feedback logic. Larger internal memory array protection is provided via additional SECDED logic cores that can be integrated within the user design.

For a detailed comparison of radiation-tolerant antifuse FPGAs with other architecture FPGAs, please also refer to table 2.8.

2.6.3.2 Radiation-Tolerant Flash FPGAs

Considering radiation susceptibility, antifuse and flash FPGAs have many things in common: Their configuration cannot be altered by SEE and the operational cells have to be specifically protected or operationally shielded. In detail, the configuration of flash FPGAs is stored in floating gate transistors which require write operations at higher voltage - much higher than a striking ionizing particle is able to generate within today's silicon bulk material [367]. This advantage is immediately changed to the opposite by the additionally required charge pumps, that provide the erasing and programming voltages, but which can be damaged by SEGR during the configuration process [368] due to the insulator layer's charge trapping explained in section 2.5.4.6. In addition, the TID of flash based memory is relatively low. Considering Actel/Microsemi ProASIC3 devices, "significant degradation of propagation times" starts at about TID_{max} = 20 krad [327] while the configuration control logic degrades already at 15 krad [369]. These issues have been mitigated to about TID_{max} = 100 krad in recent RHBD devices such as the Microsemi RTG4 [361], but charge pumps are still untested and MIL-STD-883 qualification is pending. In comparison to antifuse and SRAM FPGAs, flash memory configuration cells can be user-configured several times while retaining their states during power cycles. But in contrast to dynamically configurable SRAM cells, this process is limited to static offline configuration, including a preceding cell erase as well as a subsequent POR. As configuration and user data in the Actel/Microsemi FPGAs can be provided side by side within the flash memory, it is important to know its specific flash SEL immunity. This has been investigated in multiple publications and revealed an LET_{th} of at least 55 $MeV \cdot cm^2 \cdot mg^{-1}$ for the ProASIC3 [370, 371] and RT ProASIC3 [327] devices. In case of the recently released RTG4, the SEL immunity LET_{th} of 110 $MeV \cdot cm^2 \cdot mg^{-1}$ is given by the chip vendor [361].

As previously mentioned, embedded SRAM memory cells, PLLs as well as all internal D-type flip-flops of flash-based FPGAs, that are used for the internal state

machines, buffers and caches, are susceptible to SEE [372, 370] and therefore have to be specifically protected, similar to antifuse and ASIC devices. But in case of the 130 nm Actel/Microsemi ProASIC3 FPGA family, such RHBD features are not available in hardware, neither for the COTS, nor for the radiation-tolerant space grade version. Therefore, each user has to add fault tolerance to the HDL system design on its own. If applied correctly, this can reduce the typical 130 nm CMOS register's SEU rate from $3.0 \cdot 10^{-6}$ SEU$\cdot$bit$^{-1}\cdot$day^{-1} without TMR [327] or $3.0 \cdot 10^{-7}$ SEU$\cdot$bit$^{-1}\cdot$day^{-1} [370] respectively to about $1.6 \cdot 10^{-9}$ SEU $\cdot$ bit$^{-1} \cdot$ day^{-1} by the simple use of TMR in a solar quiet GEO space environment [327]. For comparison, the previously mentioned antifuse RTAX-S FPGA returned $9.0 \cdot 10^{-9}$ SEU $\cdot$ bit$^{-1} \cdot$ day^{-1} in such an environment [327]. The latest 65 nm RHBD Microsemi RTG4 FPGA now offers improved redundancy with asynchronous feedback circuits in hardware which shifts the SEU rate to 10^{-10} SEU $\cdot$ bit$^{-1} \cdot$ day^{-1} in solar quiet GEO. To reach this aim, most of the internal 4-input LUTs are connected to special TMR-protected registers and SRAM consists of ECC bits accompanied by the corresponding SECDEC circuits. Even the PLL is now RHBD.

For a detailed comparison of radiation-tolerant flash FPGAs with other architecture FPGAs, please also refer to table 2.8.

2.6.3.3 Radiation-Tolerant SRAM FPGAs

COTS SRAM FPGAs are manufactured in a standard CMOS process to keep up with the decreasing feature size and maximize design density. They mostly utilize an efficient 6-transistor cell design (see section 2.1.4) to store the whole device configuration including routing switches, combinatorial logic, and internal setup parameters, as well as high speed on-chip memory blocks, while keeping everything reconfigurable, even at runtime (see section 2.7.7). But these SRAM cells are primarily vulnerable to SEE caused by ionizing and non-ionizing radiation as explained in section 2.5.3. Therefore, the tolerance techniques within such devices have to put special emphasis on the mitigation of these SEEs beside of the coexisting cumulative effects. Advanced SOB material hardening methods of SRAM FPGAs include gate oxide optimization (see section 2.5.5.2) to increase TID as well as ELTs, guard rings and wafer-manufacturing with an epitaxial layer that is known to reduce SEL susceptibility [335]. These technologies can be found in commercial radiation-hardened SRAM FPGAs [373, 374, 375, 376] such as the 180 nm non-ITAR Atmel ATF280 with TID_{max} = 300 krad(Si) and SEL LET_{th} = 80 MeV$\cdot$cm$^2\cdot$mg^{-1} [365], the 90 nm non-ITAR Xilinx Virtex-4QV with TID_{max} = 300 krad(Si) and SEL LET_{th} = 100 MeV $\cdot$ cm$^2 \cdot$ mg^{-1}

[318, 377] as well as the 65 nm ITAR Xilinx Virtex-5QV with TID_{max} = 1 Mrad(Si) and SEL LET_{th} = 100 MeV$\cdot$cm$^2\cdot$mg^{-1} [303]. Fabrication with these requirements in low quantities is expensive but successfully reduces TID and SEL – unfortunately it cannot prevent SEE in total. In consequence, a successful SEU and SET mitigation strategy has to include SRAM cell redundancy and EDAC-secured memory. Both can be realized by different approaches:

- **HDL triplication of logic circuits implemented in SRAM**: This is a clear firmware approach and therefore addressed in section 2.7.2. A prominent FPGA example is the Xilinx Virtex-4QV, which offers the mentioned TID and SEL hardware radiation tolerance techniques, but otherwise "there is no change in the structure, logic or circuit design, compared to standard devices" [375]. Therefore it has to be operated with an HDL design that has been prepared with TMR and memory scrubbing, obviously provided by the additional Xilinx TMRTool (see section 2.9.2) that provides "an elegant and cost-effective method to ensure reliable operation for almost all critical space applications" [376]. The success of this device family and its predecessors, the 150 nm Virtex-II QPro with TID_{max} = 200 krad(Si) and SEL LET_{th} = 160 MeV$\cdot$cm$^2\cdot$mg^{-1} [317] lead Xilinx to have many RHBD and COTS SRAM FPGAs in space, including Mars rovers [378], the International Space Station [379] as well as various satellites [380].
- **Internal Hardware Triplication of SRAM cells**: This method is known from antifuse FPGAs and hides the complexity of redundant system design from the end-user by internally managing triplicated SRAM cells, including asynchronous feedback paths, and transparently offering hardened standard cells to the synthesis tool. Redundancy data distribution in this case is completely handled within the device circuits. Since conventional 6-transistor cells already occupy a lot of silicon, a simple triplication for majority voting can hardly be afforded as it would significantly increase device size, signal delay and cooling issues. Therefore, this approach will hardly be seen in real hardware.
- **Improved SRAM cell design**: Saving valuable silicon surface while increasing radiation tolerance by the use of redundancy is most efficient by modifying the standard 6-transistor SRAM cell itself in parallel to the known TID and SEL hardening techniques. This method has been approved by chip vendors Atmel and Xilinx, which both are using an extended 12-transistor DICE cell design in their latest RHBD SRAM FPGAs: 180 nm Atmel ATF280 [381, 362] and 65 nm

Xilinx Virtex-5QV [381]. The advantage of the DICE cell regarding SEU mitigation is its intrinsic dual-node redundancy which can only be corrupted by a particle strike from a few specific irradiation angles as described in section 2.6.2. In addition to the hereby secured configuration and user memory SRAM cells, Atmel has designed all latches and D-type flip-flops "with the same SEU hardened cell" [373, 364]. Internal controllers are secured by TMR in hardware and clock as well as reset trees are protected by differential N/P resistive isolated paths [373, 364]. This increased the device's silicon size only by about 30% [362] but eliminated the need for "complex additional radiation effects mitigation techniques" [373, 364] such as SEU/SET TMR design. Unfortunately, the Atmel ATF280's logic capacity lies significantly behind the Virtex-5QV [382] which has been developed in the Single-event effects Immune Reconfigurable FPGA (SIRF) project in cooperation with the U.S. Air Force [383]. Within this RHBD Xilinx FPGA, both redundant nodes of the DICE cells, used for configuration, registers, IOBs and SERDES, are furthermore intentionally "spaced a good distance apart" [384] to reduce the probability of simultaneously exceeding Q_{crit}. But this can happen only at a very few incident angles where a penetrating particle with sufficient energy crosses exactly two related nodes which makes the device "nearly impervious to upset by proton interaction" [303]. To furthermore prevent SETs in the DICE cells, additional 800 ps transient filters have been applied to the clock, clock enable, set/reset and data inputs of registers [303]. Similar to the Atmel ATF280, internal controllers are secured by TMR and EDAC to prevent device SEFIs. BRAM is protected by SECDED ECC bits on word basis, equivalent to the COTS Virtex-5, but accompanied by an autonomously running EDAC circuit with writeback, which eliminates the requirement for corresponding user support logic [303]. Finally, the Xilinx Virtex-5QV offers many advantages but DSPs, MGT/GTXs, PCIe block and ethernet MAC are still unprotected against SEE [252] and since it is based on SRAM, it still requires external configuration and scrubbing management as instructed in [385].

For a detailed comparison of radiation-tolerant SRAM FPGAs with other architecture FPGAs, please also refer to table 2.8.

2.6.4 Radiation-Tolerant Micro-Controllers and Micro-Processors

Beside of the most famous radiation-hardened micro processors built for space applications, such as the 15 MHz Mongoose-V (Pluto probe New Horizons), the 16 MHz Aeroflex UT1750AR (Churyumov–Gerasimenko probe Rosetta) or the 200 MHz IBM RAD750 (Mars rover Curiosity [386]), there exists a large variety of such components also applicable to special case radiation scenarios such as particle accelerators. This ranges from the first commercial 3.58 MHz 8 bit RCA CDP1802 ASIC CPU [387] (Voyager, Viking and Galileo probes and still manufactured by Intersil) up to complex devices such as the fault-tolerant, dual-core Aeroflex Gaisler LEON3FT SPARC CPU. This LEON3FT for example has been designed in HDL and therefore can only provide fault tolerance which is not sufficient for use in ionizing radiation environments. To gain the additional radiation-hard and space-grade statuses, it can be integrated in different radiation-hardened architectures, such as the Microsemi ProASIC3-RT flash FPGA or the Microsemi RTAX antifuse FPGA [388]. It has also been manufactured as 250 nm stand-alone radiation-hard 66 MHz ASIC CPU, which finally offers all the ASIC hardening benefits (see section 2.6.2), combined with ECC protected flip-flops and memory cells to withstand a TID of 300 krad and SEL $\mathrm{LET_{th}} = 108\ \mathrm{MeV \cdot cm^2 \cdot mg^{-1}}$ [389].

This underlines that there are a couple of well defined, tested and verified ASIC microprocessors available on the market, which do not evolve and therefore are still manufactured and sold today, 40 years later. This is in clear contrast to the most dynamic FPGA designs, which can be easily migrated to a more recent device generation. The reason might be that radiation-tolerant 300 krad TID FPGAs with SEU and SET mitigation have not been available in the past or that power consumption and pricing exceeded all limits. But things might change as soon as microprocessor feature size down-scaling continues as today and therefore ASIC transistors become more radiation susceptible. Several COTS microprocessors with different feature sizes have been investigated under neutron irradiation [249] with the result that the SEU rate of embedded SRAM rises when moving below 40 nm. Even worse is the MBU rate as explained in section 2.5.4.2. The same effect is expected for flip-flops below 28 nm.

Therefore, even COTS microprocessor manufacturers will have to deal with terrestrial radiation effects in the near future (see section 2.4.6) to ensure correctness of all calculated results. An interesting microarchitecture-based approach towards this direction has been shown in [390] with the **DIVA architecture**, which adds a new

functional checker phase before a processor's commit phase. This DIVA checker takes the input operands as well as the calculated complex result of each operation and compares is against a functional description of the program. Afterwards, only correct results will be committed, otherwise the processor pipeline will be flushed and the calculation is restarted from the erroneous instruction. The advantage of this method is that the "overall design cost can be dramatically reduced because designers need only verify the correctness of the checker unit." [390].

Another approach towards a resilient microprocessor hardware has been taken by the **Intel Palisades** RISC processor architecture. It is able to dynamically detect transient timing errors in the processor's signal paths [391], caused by voltage anomalies that otherwise lead to miscalculation or signal loss. To reach this aim, additional Error Detection Sequentials (EDS) meter signal runtimes *between* and Tunable Replica Circuits (TRC) meter signal runtimes *in parallel* to the pipeline stages to register EDS data comparison delay and TRC pipeline delay timing errors. As soon as an illegal deviation has been detected, caused by temperature issues, device degradation or aging effects, the processing speed is adapted and the affected instruction is repeated until all parameters match [392, 393]. This 'self-tuning processor' or 'Intel Turbo-Boost' feature allows operation frequencies beyond of a device's specification, since the gained speedup outweighs the frequency of additionally occurring errors and therefore the necessary recalculation cycles. The EDS function of signal comparison between rising and falling clock edge may furthermore be used for short pulse SET detection when operating in radiation environments.

One practical example of a COTS automotive device that offers radiation tolerance shall be given by the **Texas Instruments TMS570** [394, 395]. This 32 bit ARM Cortex-R4F based microcontroller offers DMR in hardware via a dual-channel processor pipeline in lockstep architecture. Every instruction is calculated two times in parallel and the results are compared against each other in the CCM-R4 comparison unit. In addition, both CPUs offer physical protection via guard rings (see section 2.6.2) as well as a diverse placement in silicon to reduce angle dependent effects. The global clock has been split into two phase locked clock trees to independently serve both processors at the same frequency. Timer modules and on-chip communication buses offer redundancy. While the full microcontroller configuration or at least a boot loader is stored in flash memory, the dynamic program data is kept in SRAM. Both memories offer SECDED ECC. Unfortunately, data in the SRAM is not continuously checked and hence accumulates SEU. To prevent such behavior, development of a memory scrubber as described in section 4.2.3 is necessary.

Extensive hardware monitors are continuously monitoring the synchronicity of all components and offer multiple fail safe modes complemented by diverse diagnostic and reporting functions. Neutron and α irradiation tests for moderate terrestrial automotive environments have been performed in [396], indicating that the embedded protection mechanisms are working properly while all remaining failures were caused by unhardened peripheral components or propagation from non-redundant parts of the microcontroller.

Current research indicates multiple approaches to radiation-hardened microprocessors and microcontrollers on *Circuit Level* as well as *Register Transfer Level* with a subsequent circuit molding. Especially the last approach offers new possibilities for customized fault-tolerant system firmware design to benefit from the advantages of radiation tolerance when used in radiation environments. Latest development therefore points to the 'GR740: ESA Next Generation Microprocessor' [397], a quad-core LEON4FT microprocessor that will become available in VHDL, eASIC and radiation-tolerant ASIC variants.

2.7 Firmware Fault Tolerance

Firmware fault tolerance includes all actions that are taken to detect and correct errors within *user-addressable* logical units of a semiconductor, while the techniques of hardware radiation hardening (section 2.6) merely aim at prohibiting the basic generation of such errors to maintain correct device operation. Although both processes are complementary and mostly used in a merged context, firmware fault tolerance is a totally detached field of research which can only act on user-accessible and therefore user-addressable logical elements, such as device primitives or higher abstraction logic blocks. In any case, the most basic aim of firmware fault tolerance is to ensure that additionally introduced complex circuit redundancy reduces a design's overall error cross-section and not reverse. This behavior can always be validated either by participation in practical beam tests or by using SEE injection/simulation tools (see section 2.10).

FPGA firmware in particular combines the sequential as well as combinatorial logic elements which are available in modern devices. The sequential logic, such as flip-flops, samples input signals at designated clock edges, while the combinatorial logic, such as LUTs, generates function results according to the given input signals within a designated time scale. As soon as one of these elements is upset, the whole logic design may produce wrong output data or result in unwanted system behavior. Therefore, several fault tolerance methods, specialized on this sequential

and combinatorial behavior, have been proposed to mitigate such effects within a running system. The general approach of these methods for radiation susceptible FPGAs based on antifuse, flash, or SRAM technology always follows one of two basic practices: An extended system design with application of fault tolerance on *Register-Transfer Level* prior to synthesis and mapping (pre-synthesis), or a conventional system design with insertion of fault tolerance subsequent to synthesis and mapping by using *Logic Level* netlist primitives (post-synthesis). In both situations, redundancy in its various manifestations has shown to be the only solution in reaching all of these aims. Therefore, "fault tolerance is an exercise in exploiting and managing redundancy" [398]. It can be flexibly applied to firmware designs either by using a spatial, temporal or informational approach with different granularity, starting from a most basic flip-flop right up to a full device. Assuming that only one of the redundant instances or iterations is upset by external impact, a subsequent voter or error checking circuit is able to repair or at least identify the error and hence prevent unrecognized propagation throughout the adjacent circuits.

This firmware fault tolerance design allows to create reliable circuits that, in combination with universally configurable COTS or RHBD devices (see section 2.6), provide the robustness and flexibility as it is necessary to successfully operate FPGAs in specific radiation environments without the need for expensive space-grade products.

But fault tolerance is no panacea. Beside of the conventional firmware redundancy approaches, which aim in restoring every single faulty bit within an FPGA, there may also be special applications that include design parts which do not require mitigation. This includes for example circuits which are used during one-time test routines in controlled laboratory environments or large memory arrays that contain volatile data which mitigates autonomously by averaging multiple values as it can be found in feed forward equalizers. At least the last scenario can results in "71% saving of circuit complexity in comparison to XTMR" [399] (see section 2.9.2). Therefore, a full device redundancy may not always be an appropriate solution.

The following subsections now try to give a basic overview about common practices in the fields of fault-tolerant spatial, temporal or informational redundancy as it can be applied on *Register-Transfer Level*. Lower level techniques focusing on configuration shifting [400], routing switch box rewiring [401], or LUT optimization by defining the configuration state of *don't care* bits based on their statistic impact in logic gate trees [402] will not be addressed but can be found clearly summarized in [403]. For additional information about fault-tolerant systems, please refer to [398].

2.7.1 System and Device Redundancy

If logic complexity, timing constraints or spatial design restrictions do not allow an introduction of circuit redundancy within a single chip, a viable option to add fault tolerance to such a design lies in the replication of the whole firmware into identical hardware resources. In this setup, multiple redundant devices or even whole system boards are synchronously processing the same data and a subsequent physical hardware voter chip forms a majority decision about their calculated results to identify faulty components and mitigate possible errors. Such setups can be realized with two instances for error detection plus reset, three instances for result voting, four instances for redundant error detection with partial reset as well as more instances to offer a number of spare devices. Xilinx FPGA strategies for usage of redundant devices can be found in [251].

The device redundancy technique is easy to implement, even without deeper knowledge of fault tolerance, as it utilizes the nearly unmodified, plain design without additional internal voting logic. Therefore, behavioral simulation or verification does not differ from the regular process and affords no extra tools or licenses. Device redundancy with three or more instances is even more able to mitigate SEFIs that cause a whole chip to stop functioning. It furthermore drastically simplifies eventual firmware updates by enabling the possibility of distributing identical configuration files to all deployed devices. In comparison to the fault-tolerant extension of an existing design within a single chip, the firmware of redundant devices can be kept small, which results in multiple instances of smaller devices with less power consumption. It also eliminates the necessity of I/O pin multiplication (see section 2.5.7.4) and hence the frequent problem of pin shortage. The only necessary design feature that has to be implemented in all redundant firmware instances is a mechanism for re-synchronization after device reset. As soon as the external voter detects a recurring defective calculation result at one of its inputs, it has to initiate a reset for the specific device to repair the ongoing fault. This reset requires a process of re-synchronization with all other instances, either by replicating the current context of another device, by restoring all devices to a defined state or by intelligent usage of some sort of data sync entry point. In case of Xilinx FPGAs, an additional configuration controller (see section 2.7.7) may furthermore reduce the number of necessary reset cycles.

But device redundancy also entails a major drawback: Its effectiveness gets worse with increasing radiation intensity or increasing temporal length of the calculation chain, since errors within each instance are only checked by the final voter circuit.

As soon as they start to accumulate within multiple instances in parallel, a correct result may not be successfully voted anymore. In addition, the external voter, reset as well as the reconfiguration units constitute single points of failure and therefore need to be radiation hardened. The already mentioned increase in cost and power consumption due to the additional hardware may be partially relativized by the use of smaller devices, but the area overhead still persists and may increase the final PCB size.

A famous example for system/device redundancy is "Cube, a massively-parallel FPGA-based platform" [404], that offers 512 low-cost XC3S4000 Spartan-3 FPGAs [405], arranged in 8 stacked boards of 64 devices. Each board in turn is organized in 8 rows of 8 FPGAs. Fault tolerance redundancy is reached by a row-skipping scheme that "allows pairs of rows to be bypassed" [404] in case of an error in one or more of the FPGAs.

Another example for the use of triple device redundancy is the Maxwell SCS750 Super Computer for Space. It incorporates three IBM PowerPC 750FX SOI processors that operate in parallel. The chip is stated to "consume less power than a single rad-hard processor" [406] due to reduced core operating voltage and new features size. But this statement should not be attributed to all setups in general.

Some application scenarios such as particle accelerators simply cannot offer the necessary budget, power or area requirements for utilizing device redundancy. Sometimes, the error accumulation rate even demands a faster EDAC mechanism, which can only be realized by device-internal fault tolerance with feedback logic as explained in the next section 2.7.2.

2.7.2 Spatial Redundancy Techniques

Similar to system and device redundancy (see section 2.7.1), spatial redundancy describes the technique of circuit replication for parallel and synchronous data calculation at system level [407, 408, 409] to reduce the number of single points of failure within an existing firmware design. But in contrast, it is not limited to system level and can use fault tolerance methods on register transfer level or below to introduce fine grained redundant regions of combinatorial circuit blocks or modules across a single device, combined with local feedback circuits and voting elements. This of course occupies a significant number of additional device resources which are not available for implementation of custom logic functions anymore and therefore constitute the biggest penalty of spatial redundancy.

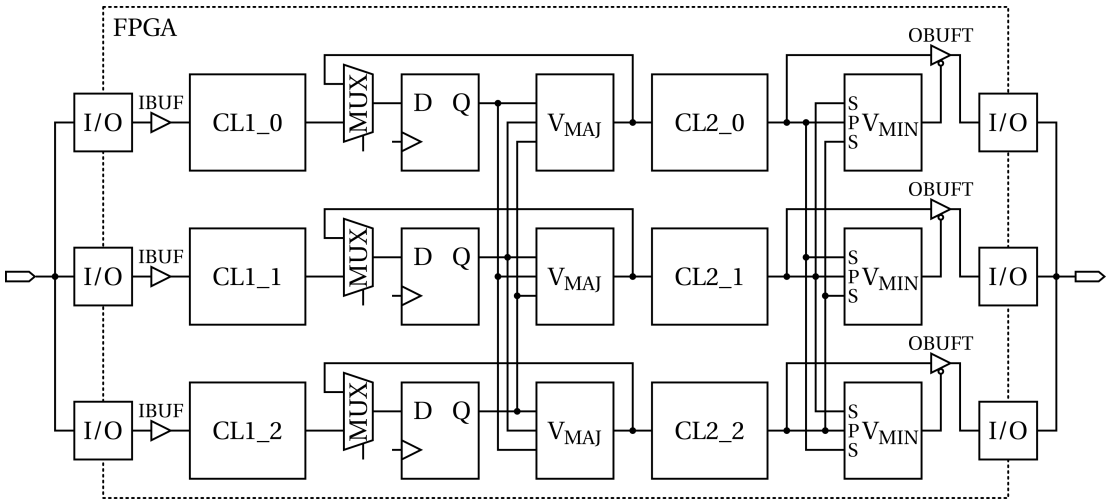


Figure 2.20: Depiction of a fault-tolerant XTMR design, designated for a Xilinx SRAM FPGA by summarizing the information given in [154]. Input and output pins are hard wired outside of the FPGA as explained in section 2.5.7.4. The IBUF input buffers are necessary to provide stable signal levels to the triplicated combinatorial logic paths CL1 and CL2. A subsequent register stores the calculation result for the majority voter decision and a surrounding feedback loop immediately corrects erroneous information between clock cycles. The OBUFT output buffers are tri-state logic elements, which are designated to forward the input signal under the prerequisite that most of the connected minority voter's input signals match the primary input 'P'. If this is not the case, the tri-state buffer enters high impedance 'Z' state and therefore does not contribute to the final output signal state. The logic tables of both voter types can be found in [154].

Spatial redundancy can be implemented with N-Modular redundancy of different granularity. A Dual Modular Redundancy (DMR) technique is able to detect SBUs by Duplication With Comparison (DWC). It can be easily realized within FPGAs in nearly any block/module size by the introduction of LUT *XOR* comparison functions between each two replicated logic instances. Normally, a detected DMR error results in data refusal or device reset, as it cannot be decided which of both instances is the faulty one, if not combined with other fault tolerance techniques such as Concurrent Error Detection (CED) as shown in [410]. DMR requires less device resources than Triple Modular Redundancy (TMR) but, in contrast, a TMR approach can detect single errors, form a majority voter decision and continue without interruption while returning a feedback signal to correct the erroneous signal. Four or more module redundancy, such as replicated DMR, provides additional coverage of multiple er-

rors in parallel or reserves spare instances that can take over in case of an instance breakdown.

Using TMR demands the implementation of additional voter logic. A TMR voter represents a simple logic function with three inputs that can be easily mapped into a single 3-input device LUT. In detail, there are two types of voters, majority and minority voters. While the majority voter simply comes to a decision by returning the most prevalent input signal, the minority voter compares all signals to a primary input and returns '0' (agree) only in case of a majority match and otherwise '1' (disagree). The logic tables for both voters can be found in [154]. The minority voter can be used for example to drive Xilinx tri-state buffers (BUFT) as shown in the output region of figure 2.20. The outputs of triplicated combinatorial logic blocks with their subsequent register for result buffering can be connected to either a single voter or three independent voters in parallel. Since utilization of a single voter would introduce another single point of failure, the common strategy is to use three voters, whereas each of them is fed by all inputs of the TMR registers to make individual decisions as shown in figure 2.20. The output of each majority voter is immediately returned by a feedback signal path to the input of its corresponding register to fix possible upsets between multiple clock phases that otherwise would accumulate over time. As all of these voters operate on bit level comparison, an MBU within two of three redundant legs in parallel and therefore an error in two of three majority voter inputs can not be detected. It furthermore mistakenly overwrites the correct result. This issue is addressed in [411] by the introduction of word-voters.

An efficient modular redundancy implementation furthermore requires supply with reliable input and output signals. To prevent irreversible data modification within the FPGA's I/O buffers, which may mistakenly feed all redundant inputs with wrong data or modify a voter's output signal before leaving the device, the guidelines described in section 2.5.7.4 have to be followed strictly. If the recommended hardwiring of I/O pins is physically impossible due to a predefined hardware platform or general pin shortage, signals need to be replicated directly after the I/O buffer's output to retain very short data paths.

A global TMR (GTMR) approach including replication of all combinatorial, sequential as well as I/O logic resources, completed by voters and feedback paths within Xilinx FPGAs, is called XTMR. XTMR offers sufficient protection for dynamic register data and is able to discard erroneous values from redundant combinatorial logic paths, but at very high cost: The overall resource usage of an XTMR approach as depicted in figure 2.20, applied to the whole design logic, grows the original re-

source usage up to six times [412]. Also power consumption, signal skew and domain crossings complications [413] grow with device size and increased logic utilization. XTMR can successfully handle SEU and SET, but does not tolerate MBU within redundant legs of the design. Also routing shorts between redundant instances that combine usually distinct regions and form bridging faults are considered to be critical [414, 415], since they are able to corrupt the voting mechanism. A strictly distinct TMR region placement and routing may help to mitigate such issues. It can be provided by tools such as the Reliability oriented place & Route Algorithm (RoRA) as explained in section 2.9.4. Furthermore, spatial redundancy in general cannot prevent error accumulation within the SRAM FPGA's basic device configuration itself. This issue is successfully handled by the configuration scrubbing technique explained in section 2.7.7.

Assistance in manually creating custom TMR designs is provided by a VHDL library with basic RTL functions [416]. Xilinx also provides a tool for automated insertion of XTMR into existing designs as explained in section 2.9.2, but the tool operates on logic level netlists only.

Beside of the XTMR technique depicted in figure 2.20, several other TMR approaches are available. Predominantly, they try to reduce the extensive resource consumption as well as the complexity of XTMR without jeopardizing design susceptibility. Reference [417] for example proposes to selectively identify the most sensitive parts of a system that have to be protected with TMR instead of following the blind GTMR/XTMR full device replication approach. This allows the user to "tradeoff circuit area with the hardness level" [417]. According to [413], there are four basic TMR techniques:

- **Block TMR (BTMR)** is considering the triplication of comprehensive combinatorial logic blocks with embedded flip-flops. A subsequent voter matrix forms a majority decision but without any feedback paths. This results in error accumulation and is referred to as "not an effective technique" [413].
- **Local TMR (LTMR)** is characterized by triplicated flip-flops across the design, which share data, enable and clock inputs from non-triplicated combinatorial logic outputs. Additional voters and feedback paths offer SBU mitigation and error correction. As clock and reset signals are still unprotected, LTMR is still vulnerable to SEFIs and therefore requires FPGA devices with hardened clocks [413].
- **Distributed TMR (DTMR)** extends LTMR by triplicated combinatorial logic to separate data paths. Clock and reset signals as well as I/O pins are still shared

between all components, therefore a glitch in one of these components can still upset the whole device. The overall resource and power consumption is lower then for the full GTMR approach [413] and due to the global use of a single, shared clock signal, the whole design might also become "slightly slower than GTMR" [413].

- **Global TMR (GTMR)** triplicates the entire design, including combinatorial logic, flip-flops, voters as well as I/O pins in independent domains with separate clock and reset signals as depicted in figure 2.20. By this method, only hidden internal FPGA logic is able to cause SEFIs. In general, GTMR "proves to be a great mitigation strategy" [413], but given that the available mitigation tools (see section 2.9) are not capable of providing RTL code for synthesis or simulation, its complexity is very high and therefore "can not be an embedded strategy" [413].

According to these characteristics, LTMR is most convenient for Antifuse as well as Flash FPGAs, DTMR can be used for Flash FPGAs and GTMR should be used for SRAM FPGAs only [413]. The practical evaluation of these TMR techniques in reference [418, 419], using an Actel ProASIC3E FPGA, returned an SEE LET_{th} of 2.8 MeV$\cdot$ cm$^2\cdot$mg^{-1} without TMR, 8.6 MeV$\cdot$cm$^2\cdot$mg^{-1} for LTMR, and $12-20$ MeV$\cdot$cm$^2\cdot$mg^{-1} for the DTMR approach. More evaluations and comparisons of TMR can be found in [420, 421], while a comparison of DMR techniques with information redundancy based systems (see section 2.7.4) is presented in [422]. Especially the latter-named publication comes to the conclusion that systems with spatially replicated functions "have a significant advantage over other CED schemes in providing protection against multiple failures" [422].

A practical example which makes intensive use of spatial redundancy is the many-core computer architecture with fault detection and recovery published in [423]. It uses 16 MicroBlaze soft processors within a single Virtex-6 device. While three running instances are coupled with TMR, 13 others are spare parts that can immediately replace faulty instances. This approach has already been successfully tested at high altitude and is furthermore planned to be examined in LEO space flight. But especially micro processors are perfect candidates for another fault tolerance technique by restarting erroneous calculations until a correct result has been obtained. This time dependent sampling approach is explained in the next section 2.7.3.

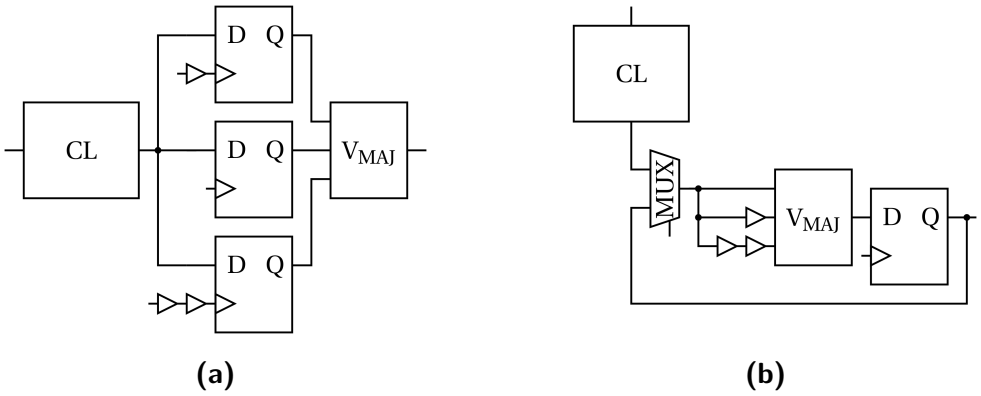


Figure 2.21: Depiction of temporal redundancy design techniques, according to [424, 425]. Both figures use a triplicated temporal sampling approach, but differ from each other by reason of clock handling as well as resource consumption. Subfigure (a) uses three phase shifted clocks to store the combinatorial logic (CL) output in separate flip-flops, while subfigure (b) delays the data signal itself without separate clocks. A feedback path furthermore improves signal retention. In both scenarios, an appropriate SET within the CL can be successfully mitigated by majority voting.

2.7.3 Temporal Redundancy

If full spatial XTMR is not available due to limitations in device resources or design complexity, a repeated sampling of a single, non-redundant logic block's output at different points in time may be a suitable option. This technique of temporal redundancy is a popular method to mitigate intermediately glitching SETs and the resulting SET-to-SEU capturing mechanism at the sampling clock edges of signal paths as explained in section 2.5.4.1. Based on its principle, the calculation delay between two such cycles has to be wider than the transient's maximum pulse width to prevent an SET from impacting more than one sampling cycle and preserve correct voter functionality. In recent designs, this can quickly become a limiting factor, as increasing operation frequencies are known to increase the SET-to-SEU capturing as shown in [426]. Therefore, additional timing delay needs to be scheduled for the entire design. Furthermore, temporal redundancy demands a stable device configuration to successfully process and vote the calculation result at least three times without interruption due to accumulating SBUs.

The logical implementation of temporal redundancy is shown in figure 2.21. The replicated output of a combinatorial logic block can be sampled either with phase shifted clock signals from separate device clock trees into buffering flip-flops (subfigure 2.21a) or the data signal can be temporally delayed itself (subfigure 2.21b). In

both cases, a subsequent voter circuit forms a majority decision about the correct result and therefore successfully mitigates a temporary SET glitch. The first solution can furthermore vote out an SEU in one of the flip-flops. Both approaches cannot decide about SBUs within the combinatorial logic block itself. Even more, the necessary stable configuration behavior cannot be guaranteed for SRAM FPGAs. Therefore, temporal redundancy for SRAM FPGAs is mostly combined with spatial redundancy as explained in section 2.7.5 to provide better mitigation results for SEUs and SETs.

As a well defined signal delay component in Xilinx SRAM FPGAs is difficult to implement without introduction of additional registers, the second approach is most advantageous for ASIC or antifuse FPGA devices, such as the Actel/Microsemi RTAX FPGA as done in [427]. A similar approach is followed in the Intel Palisades architecture [392, 393], where an erroneous signal gets recalculated after flushing the ASIC microprocessor's pipeline as explained in section 2.6.4. Further comparison with spatial redundancy techniques can be found in [420].

2.7.4 Information Redundancy

Beside of spatial and temporal redundancy, the introduction of fault-tolerant, self-checking logic circuits as shown in [428] can furthermore improve the overall system reliability. These methods typically exhibit lower resource cost in comparison to spatial redundancy, but at the price of additional computation, time which may critically slow down highly efficient algorithms that have to operate at the edge.

Information redundancy uses special data coding styles to generate additional check bits for the verification of user data or calculation results in designated firmware designs. This requires Concurrent Error Detection (CED) or Error Detection And Correction (EDAC) circuits in the user design to be able to verify or repair erroneous information, arising due to transfer faults or radiation impact on storage elements. For the latest Xilinx Virtex FPGAs, these circuits are embedded in hardware by default, at least ECC for the embedded BRAM (see section 2.3.4). Other systems may require the manual addition of such logic circuits.

While CED for spatial redundancy is based on the DMR approach with result comparison as explained in section 2.7.2, CED schemes for information redundancy target on the validation of internally calculated data, based on previously determined check bits. These check bits are generated from all input signals, based on a user-implemented function. Generation as well as validation are done fully transparent and in parallel to a running system. A minimalist CED unit may for example an-

alyze both input operands of a complex adder circuit to generate an all-even/odd signal and return a fault as soon as the subsequently returned result is odd. Such prediction rules can have any complexity up to static tables of valid results, which, in some situations, requires less device resources for implementation in comparison to a corresponding spatial redundancy approach. Beside of these implementation cost, CED is only capable of detecting errors in a running design, but not to correct them. Further issues are possible false positives/negatives due to massively simplified prediction functions as well as the complex design validation. An extensive analysis of such CED methods can be found in [429].

For EDAC algorithms, it has to be distinguished between error detection and error correction types. Simple parity check bits added to the storage registers or memory are based on either a single or multiple redundant bits which represent an even or odd number of ones within a group of data bits. This function can easily be implemented in FPGAs by the use of *XOR* gates. Parity can detect SBUs, while MBU detection only improves with multiple parity bits in parallel [430, 422]. Error correction is not possible. Nearly the same applies to Cyclic Redundancy Check (CRC): The fixed number of redundant bits, gathered from the CRC algorithm by binary division, is appended at the end of all data words stored in registers or memory. SBUs or even MBUs within these data words including the CRC bits can now be successfully detected but still not corrected. CRC data protection is very popular in firmware designs, as its algorithm is based on Egyptian/Russian multiplication which can be easily implemented by using multiplication with 2 (left shift), division of 2 (right shift) and a final addition.

But information redundancy can perform better. As soon as error correction with minimal storage overhead within data words becomes necessary, the usage of an EDAC SECDED ECC algorithm, such as Hamming Code with parity, is inevitable. Hamming code is based on a unique combination of XOR operations that, applied to a given data word, generates a set of parity bits. These parity bits are added at fixed positions within the data word itself and therefore extend it by a specific amount of digits, depending on initial length. For Xilinx FPGAs, the SECDED protection of a 32 bit data word with Hamming(39,32) contributes with 7 check bits [431] while a 64 bit data word with Hamming(72,64) contributes with 8 check bits [431] to the final word. As soon as a single error in data or parity has been detected, the erroneous position can be exactly determined and the bit upset should quickly be corrected to prevent accumulation of uncorrectable MBUs. Since MBUs caused by an SEU are locally related, a physical interleaving of all data bits that belong to a single Ham-

ming word can furthermore reduce the probability of uncorrectable errors. This basic interleaving technique is theoretically investigated in [432] and practically realized within Intel's Chipset Memory Controller Hubs by sharing words between all connected memory chips [433]. If none of such hardware controllers is available, interleaving requires design of a custom firmware. The same applies to a redundant Hamming encoder/decoder circuit. If not supported by hardware as mentioned in section 2.3.4, the additional logic block significantly increases the user design that, if not correctly protected against SEE, may negatively impact the whole radiation susceptibility. The block diagrams for a manual design of Hamming encoder/decoder parity bits can be found in [431].

A combined approach of CED and EDAC can be found in Residue Arithmetic Code (RAC), but only for arithmetic error detection and correction. RAC is based on comparison of the regular arithmetic calculation result in its residue-coded form with the result from an arithmetic operation on previously residue-coded operands. Depending on the protection strategy, RAC is available in several enhancement levels. While a single-modulus approach can detect SBU but is unable to correct them, a dual-moduli can correct SBU and detect double bit errors, and a triple-moduli approach corrects double bit errors and so on. At least the dual-moduli RAC has been investigated in [434], resulting in 2.6 times less area consumption than classic TMR and therefore "with the ability to detect and correct with much lower overhead than redundancy based RHBD techniques and less delay than temporal filtering techniques" [434].

Finally, it should be pointed out, that fault tolerance based on information redundancy can only address bit errors within the user data and not the within the device configuration or circuit layout itself. Therefore, it performs best in hard-wired devices or dense user memory arrays. Furthermore, additionally introduced CED or EDAC circuits in the user design will increase the overall design susceptibility and therefore have to be protected by spatial redundancy when used in SRAM FPGAs. This approach results in a combination of multiple fault tolerance techniques in the firmware design of today's FPGAs and will be explained in the next section 2.7.5.

2.7.5 Combined Redundancy

A combination of spatial redundancy (see section 2.7.2) and temporal redundancy (see section 2.7.3) is regularly used to efficiently mitigate SEU and SET in parallel, while reducing area overhead and therefore saving valuable design resources in comparison to a full XTMR approach. The common implementation comprises a

temporal sampling design as shown in figure 2.21a, but supplemented with spatially replicated combinatorial logic modules as well as voters.

A more efficiently combined redundancy approach is presented in [435, 436, 437]. It combines temporal data sampling (see section 2.7.3) with a DMR weighted voting scheme. Therefore, all shifted data signals from the temporally redundant design nodes get assigned predefined voting weights within the voter circuit. These weights are calculated inversely proportional to the probability of the node being erroneously upset. "The design technique not only eliminates all the single event upsets and single event transients but eliminates all double event upset as well" [435].

A different technique for arithmetic units only is explained in [438, 439]. It utilizes conventional DMR with CED (see section 2.7.2) for permanent error detection between two multiplier nodes. The operands of both multiplier units are furthermore protected by RAC (see section 2.7.4). By this method, an erroneous unit can be identified even in a DMR approach. Resource consumption as well as the number of I/O pins could be reduced in comparison to a full XTMR approach, while the error injection still recovered 100% of all injected faults.

Beside of all mentioned techniques for combined redundancy, there are even more custom solutions available, which have been developed and optimized for special application scenarios, such as extensive data transport [27] or embedded processor safety with lockstep, checkpoint and rollback features [286, 440, 441]. Especially in the field of embedded processor design, TMR or DMR with CED does not necessarily have to be the sole mitigation approach, since the linear calculation characteristic of a CPU is also perfectly suited for temporal redundancy methods such as a repeatedly execution of instructions in a time-multiplexed calculation scheme. But this does not mean that a fusion with spatial redundancy can be neglected completely. The most basic operational units, such as the internal program counter or state machines, still have to be protected in a way that they cannot be altered unexpectedly, as depicted in the next section 2.7.6. Otherwise, an upset in these components would quickly result in an SEFI with subsequent system reset. For more information on radiation-tolerant micro-controllers and micro-processors please refer to section 2.6.4.

2.7.6 State Machine Encoding

Finite State Machines (FSM), describing signals and their transitions in dependence of given input data, have direct impact on the sequential design behavior of

State	One-Hot	Two-Hot	Gray	Johnson	Hamming (d=3)
A	00000001	00011	000	0000	000000
B	00000010	00101	001	0001	010101
C	00000100	01001	011	0011	100110
D	00001000	10001	010	0111	110011
E	00010000	00110	110	1111	111000
F	00100000	01010	111	1110	101101
G	01000000	10010	101	1100	011110
H	10000000	01100	100	1000	001011

Table 2.9: Overview of common FSM encoding schemes. While a SBU in one-hot and two-hot state encoding passes the FSM directly into an undefined state, Gray or Johnson encoding upsets can spontaneously result in illegal transitions into other well defined states. Hamming encoding with a distance of d=3 therefore provides better protection against bit upsets by correcting SBUs in a corresponding circuit without FSM interference. If no such Hamming decoder circuit is available, a double bit upset still results in an undefined state that can be detected by the VHDL `others` clause.

a system firmware and therefore constitute a critical part within an FPGA device. FSMs are basically mapped into device flip-flops which dynamically store the current state, in BRAMs or LUTs to offer a static set of state transition rules as well as into routing configuration cells and device gate primitives to provide static connectivity. While LUT content as well as routing and gate interconnects are considered to have a static nature and therefore can be continuously refreshed in the background from a golden copy of the initial device configuration, the remaining registers are critical elements and have to be handled in a fault-tolerant firmware design. A plain spatial redundancy approach of the whole FSM module is therefore difficult to handle, since bit upsets within the registers will not be repaired automatically and continue to hold the FSM in an erroneous or undefined state. Synchronization between all replicated instances can solve this problem, but requires additional voter logic as known from extensive XTMR (see section 2.7.2). The asynchronous nature of SEUs furthermore complicates a mitigation approach especially for Moore FSMs where outputs depend only on the current state value and therefore can be altered immediately.

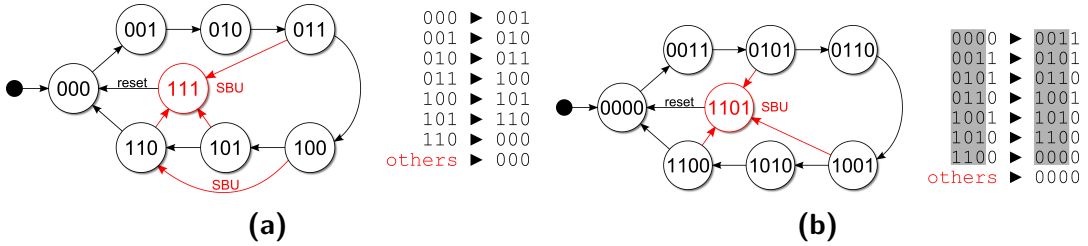


Figure 2.22: Depiction of SBU impact on state transition diagrams of (a) Gray or (b) Parity encoded FSMs. In case of (a) Gray encoding, the altered state register either performs an illegal transition to a well defined state or to an undefined state, which finally should result in a system reset if implemented correctly. In case of (b) Parity encoding, at least the illegal state transitions are ruled out.

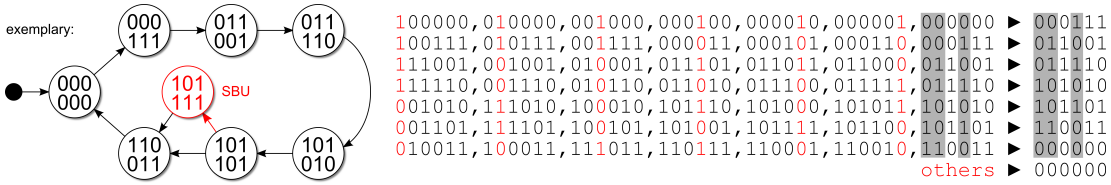


Figure 2.23: Depiction of SBU impact on the state transition diagram of a Hamming distance d=3 encoded FSM. The EDAC feature enables reliable detection of the upset condition and therefore allows continuation of operation without undefined states or transitions while eliminating the necessity of an FSM reset. This EDAC scheme works even without a Hamming decoder when auxilary states are added to the logic design [442].

Most of the regular FSMs use one-hot, two-hot, Gray or Johnson schemes to represent different states. These codes are at maximum capable of detecting SBU by entering undefined states such as seen from one-hot encoding where every state is represented by a single bit. But the more likely incident is that an SBU causes the FSM to enter a state that is well defined but illegally reached by an invalid state transition. This happens for example with Gray encoding where every state is represented by an increasing number as seen from table 2.9. Beside of unwanted system behavior, illegal states can furthermore result in deadlocks or system reset as depicted in figures 2.22. But a definition of clean reset states on logic level is heavily dependent on the synthesis tool's optimization process which may quickly remove logically unreachable states. Xilinx therefore offers options for safe encoding and implementation which "generates additional logic that forces the FSM to a valid state" [443]. But this feature does not add error correction and therefore state recovery. An adequate solution to protect FSM registers and to guarantee a deterministic behavior in the presence of SBUs is therefore to use a fault-tolerant state encoding scheme that is

able to autonomously perform EDAC while running. This practice increases the total number of occupied flip-flops, but if the frequency of error correction is higher than the frequency of upset injection, this minor drawback can be compensated.

EDAC state encoding is provided for example by Hamming code with a distance of d=3. With this method, every state differs from all others in at least 3 bits by introducing extensive parity which itself is protected by the encoding scheme. It enables double bit error detection and single bit error correction and therefore eliminates the possibility of illegal state transition due to single and double bit upsets. As soon as a Hamming decoder is available, it can furthermore trace illegally entered and undefined state conditions back to its origin, eliminating the necessity of a device reset. But as already mentioned, this requires a correct implementation of all possible states by the synthesis tool. It furthermore requires an additional circuit that continuously checks the current state value at runtime to prevent error accumulation. If no such Hamming decoder is available, all possible states can also be physically integrated into the design, accompanied by transitions which directly guide back to the originally upset state or forward to the originally planned next state as depicted in figure 2.23. This method is discussed in [442] and results are shown in section 5.4.2. It adds a significant amount of LUT content, but due to the static nature, it can easily be refreshed in the background from a golden copy as explained in the next section 2.7.7.

A detailed comparison of several state encoding schemes as well as their resource consumption and performance can be found in [444]. Another publication that focuses on a Hamming encoded FSM implementation while not utilizing LUT, but occupying embedded BRAM can be found in [445]. Using such an approach may furthermore benefit from several hardware-ECC features as provided by some BRAMs.

2.7.7 Dynamic Partial Reconfiguration and Scrubbing

Non of the presented device-, spatial-, temporal- or information redundancy and even non of the combined approaches is capable of handling the predominant accumulation of SBU within the most basic configuration cells of SRAM FPGAs. While a SBU within a data FIFO is usually overwritten quickly, even the best TMR approach becomes useless as soon as the routing information in two redundant legs has been damaged and generates wrong output. Therefore, the primary mitigation method in SRAM FPGAs, as manufactured by Xilinx, Altera/Intel or Lattice, deals with the essential conservation of a device's configuration matrix – to a more or less successful degree. This process is referenced as Dynamic Partial Reconfiguration (DPR).

Configuration Mode	Max Clock Rate	Data Width	Maximum Bandwidth
ICAP	100 MHz	32 bit	3.2 Gbps
SelectMAP	100 MHz	32 bit	3.2 Gbps
Serial Mode	100 MHz	1 bit	100 Mbps
JTAG	66 MHz	1 bit	66 Mbps

Table 2.10: Maximum bandwidth of all configuration ports in Xilinx Virtex FPGAs. All data is taken from table 6-1 in [141].

DPR in general deals with the partial modification of a device firmware by loading a specifically prepared configuration file. The most efficient devices can even perform this process dynamically in the background, simultaneously to device operation and without interruption, as seen from Xilinx Virtex, Kintex, Artix, Zync and UltraScale FPGAs [141, 446]. This enables usage for a variety of applications dealing with device over-provisioning and dynamic feature exchange [447, 448, 449, 450]. A special class of DPR that continuously writes initially defined bit values into a device's configuration memory is called configuration scrubbing. Since most of the configuration in conventional designs is static by nature, for example routing (see section 2.3), these parts can be easily refreshed in background without changing the behavior of a design. While useless for normal operational conditions, the benefit from such a method pays off as soon as the SRAM configuration cells are illegally upset, especially by radiation, and repaired on the fly. Scrubbing does not protect from configuration memory upsets in general, it only repairs them as quickly as possible to prevent error accumulation. Therefore, the process has to run continuously in the background at maximum speed or at least in consecutively fixed intervals which are shorter than the statistical SBU rate. In case of Xilinx FPGAs, where configuration refresh is available since the very first Virtex devices [375, 451], scrubbing can be performed via the following device configuration interfaces: JTAG, SelectMAP, Serial [451] or ICAP [452]. Due to their individual features, they operate at different bandwidths as indicated in table 2.10. But scrubbing speed is an essential criterion when selecting an appropriate interface – the faster the scrubbing process operates on the device, the less errors are able to accumulate within the configuration memory. While JTAG offers multiple device chaining at lower serial bandwidth, SelectMAP and ICAP, which internally access the SelectMAP interface, are operated with parallel interfaces at higher bandwidth. The ICAP scrubbing primitive is furthermore initialized device-internally only [453] and therefore addresses security constraints

arising with bitfile encryption. A comparison of all configuration interfaces can be found in [454].

To successfully initiate configuration scrubbing on a supported device, an internal or external controller is required which has access to the specifically prepared device configuration file. This controller as well as all connected memory chips need to be radiation- and fault-tolerant when operated in radiation environments to securely operate the scrubbing cycles and reload configuration data. A publicly available, FPGA-internal Xilinx Soft Error Mitigation (SEM) controller circuit that detects bit upsets with a typical latency of 25 ms [455] is currently available for all Series-6 FPGAs [456, 457] as well as for all Series-7 and Zynq FPGAs [455]. It optimally utilizes specifically built-in hardware silicon primitives and therefore minimizes additional logic resource consumption and maximizes correction speed. Former device generations did not offer such hardware components and therefore had to implement a fully firmware-based, fault-tolerant logic core that was either custom designed or rated space grade and therefore not publicly available. The scrubbing process itself can be operated according to one of the following methods that are currently available for Xilinx FPGAs:

- **Blind Scrubbing**: The scrubbing controller for blind scrubbing is used to continuously write a full partial bitfile from an external source to the connected device, whether there have been bit upsets or not. Therefore, it does not matter if a SBU or MBU needs to be corrected, all static data is overwritten with the original content. This is by far the simplest controller implementation, but given that the configuration interface itself can be upset by radiation, a damaged write pointer may have tremendous consequences by damaging the whole device configuration and making it completely unusable. Such errors have been observed in [306]. In addition, the chip-internal CRC has to be disabled, as it is violated by the partial bitfile. Practical implementation details for the blind scrubbing controller can be found in section 4.1.
- **Selective Frame Scrubbing**: The scrubbing controller for selective frame scrubbing continuously reads back the actual device configuration frame by frame (see section 2.3.1) and compares each one's static content against its initial golden version, either by following a full bit-by-bit approach or by validating the actual frame's CRC value against its preconfigured one, initially copied from bitfile to the FPGA. In case of bit-wise comparison, it does not matter if a SBU or MBU needs to be corrected, while in CRC mode, although extreme unlikely, an MBU may corrupt the arithmetic CRC calculation and therefore

remains undetected. The golden data, required for error identification, has to be taken from an external memory or, in case of CRC matching, it is stored within the device memory itself. In case of successful upset detection, only the affected frame is reconfigured from the golden bitstream data. This speeds up the error detection process and reduces the probability that a damaged configuration interface immediately misconfigures the whole device. A final readback cycle for the scrubbed frame furthermore ensures correct processing.

- **Self-Scrubbing**: The process of single frame EDAC known from selective frame scrubbing is sped up for younger FPGA devices by introducing the FRAME_ECC hardware primitive which offers analytical access to device-internally stored, Hamming encoded ECC for single frames [456]. While Xilinx Spartan FPGAs never provided such ECC features in hardware [456, 458], SECDED for the Virtex-4, Virtex-5 and Virtex-6 chips can be realized by utilizing the corresponding FRAME_ECC_VIRTEX4 [459], FRAME_ECC_VIRTEX5 [460] and FRAME_ECC_VIRTEX6 [461] primitive. Although the error detection mechanism in the mentioned devices is running autonomously as soon as readback is triggered for a selected configuration frame, error correction still requires custom user logic to be integrated in conjunction with the scrubbing controller. Therefore, the FPGA self-scrubbing concept works only in a combination of FRAME_ECC primitives, the ICAP configuration interface, as well as the controller circuit itself, which embeds all required error correction circuits. This controller can be either the elder Xilinx SEU Monitor (Virtex-4 and Virtex-5), the current Xilinx SEM logic core (Virtex-6 and later) [457, 456, 455] or a custom design as presented in [462, 450]. While Xilinx Virtex devices prior to Series-7 required major parts of the SECDEC functionality to be embedded with device resources in firmware, which itself had to be protected by fault tolerance, Series-7 FPGAs introduced a new FRAME_ECCE2 primitive, that is now able to completely detect double bit upsets and correct SBUs in hardware [295] within a well defined time frame of 610 to 915 µs [455], depending on the manufactured silicon architecture. MBU detection of more than 2 errors within a single frame, that may not be detected by the Hamming ECC, is realized in the SEM controller by finally checking a device-spanning CRC. To furthermore reduce this MBU susceptibility in single frames, physically adjacent device bits have been logically assigned to different frames as explained in sections 2.7.4 and 2.6. Since erroneous conditions arising from MBU cannot be repaired internally, a watchdog within the scrubbing controller is continu-

ously looking at the connected primitive's status signals. It initiates a frame scrubbing cycle as soon as a double bit upset within a single frame (ECCERROR) has been successfully detected. It also triggers a full device reconfiguration if more than three upsets occurred (CRCERROR). But this requires the controller to have access to the specifically prepared partial configuration file from a connected memory source. A recently released self-scrubbing design concept for the CBM TOF detector readout electronics (see section 1.3.2) that supports FPGA frame data requests via an externally connected, radiation-hardened GBT-SCA [463] communication ASIC has been presented in [464]. Unlike the blind and selective frame scrubbing approaches, self-scrubbing can also be operated completely without external impact if only SBUs within the FPGA's configuration cells are expected for the proposed application scenario.

Configuration scrubbing has grown to a standard for static firmware preservation that cannot be neglected when operating SRAM FPGAs in radiation areas. Many publications therefore offer various solutions to combine classic redundancy approaches with DPR. Reference [465] for example offers local place and route for redundant TMR regions and limits the configuration scrubbing to device frames which are covered by the erroneous regions only. [466] implements DMR modules that are individually scrubbed in case of a configuration upset. [450] offers the same functionality for reconfigurable TMR and voter modules. Finally, in reference [286], the basic functionality of a whole DMR RISC CPU is synchronized with background blind scrubbing cycles to ensure reliable calculation results even in case of a configuration bit upset.

But reconfiguration by scrubbing is also subject to several constraints and limitations, as not all configuration cells within an FPGA have to be used statically by default. LUTs in Xilinx FPGAs for example can be configured to serve as synchronous, distributed memory (SLICEM) or shift registers (SRL16) without occupying flip-flop resources. This easily extends the device's limited dynamic storage capabilities, but actually precludes a static handling for these cells. To bypass this issue, LUT memory usage can be completely disabled during synthesis or so called global LUT mask (GLUTMASK) tags can be defined [300] for such cells, to automatically exclude them from being scrubbed and therefore from being replaced with the initial golden data. Considering the occurrence of SEU within the SRAM FPGA's configuration cells, the use of GLUTMASK reduces the amount of LUT data that can be repaired by scrubbing and therefore forces the use of other mitigation techniques on register-transfer

level as explained in section 2.7.2. Therefore, if possible, the use of distributed memory should be avoided for designs in radiation susceptible applications.

Another limitation is imposed by the chip-internal BRAM (see section 2.3.4). All Xilinx FPGAs are embedding large memory arrays to provide storage RAM or ROM features that can be initialized with user data. This initialization data is included at the end of the FPGA's configuration bitfile and can therefore easily be used for configuration scrubbing. Practically, this can and should only be applied to ROM blocks and needs to be excluded for RAM data by removing the respective commands from the configuration file. But given that radiation does not omit the thin oxide BRAM cells and since their susceptibility is even higher in comparison to regular configuration cells (see section 2.5.7.5), the mitigation of upsets in RAM cells is an urgent task that has been addressed in the next section 2.7.8.

2.7.8 On-Chip Memory Scrubbing

On-Chip memory in Xilinx FPGAs can be realized by using different approaches. Storing data in registers is an easy task that allows high performance access, since every bit can be directly accessed for further processing, but the size of registers is highly limited by the number of available flip-flops in the selected device. This kind of memory in firmware designs has to be kept valid over runtime by using the redundancy techniques explained in section 2.7, such as feedback paths and voters. A second approach is the use of specifically prepared slices with memory LUTs (SLICEM) within the device that provide read and write operations for the internally stored bit information. Similar to registers, this distributed memory can be protected in the firmware design by using redundancy. Furthermore these LUTs have to be excluded from the static configuration scrubbing techniques explained in section 2.7.7 unless they provide only ROM data.

Beside of these two logic resource approaches, the designated way of storing larger quantities of user data in FPGAs is to use the embedded BRAM primitive explained in section 2.3.4. Depending on the FPGA model, these hardware blocks may provide internal ECC techniques for error mitigation. In case of Xilinx Virtex FPGAs and starting with Virtex-4 [150], the BRAM primitives offer SECDED functionality for read and write operations, fully transparent to the user via embedded hardware circuits as explained in section 2.5.7.5. But this additional feature cannot prevent the accumulation of errors within the 64 bit user data words or within the additionally stored ECC parity bits in general. To reduce the probability of situations that cannot be corrected by the BRAM-internal SECDED algorithm anymore, a continuous

memory refresh for every single memory word is mandatory. This does not come for free. It has to be manually implemented and connected to every BRAM primitive and therefore occupies valuable logic resources. But erroneous data that has been identified and corrected during a readback cycle is not automatically written back to the memory [300]. The error resides, until it has been corrected by manually initiating a word write operation at the respective memory address. This process of continuously reading through the whole memory and correcting SBU, similar to a voltage level capacitor refresh in DRAM, is referenced as memory self scrubbing. Using hardware error correction provided by the memory primitives eliminates the requirement for additional modular redundancy and therefore efficiently reduces the number of occupied device primitives that are normally lost to redundancy techniques.

Memory self scrubbing becomes slightly more complicated when ECC in hardware is not supported, as known from the cheaper Spartan series, although additional memory bits for parity storage are physically available within the device memory. In consequence, the selected algorithm for such a device is arbitrary but has to be manually implemented by the use of valuable logic resources while simultaneously considering fault-tolerant design fundamentals. The alternative to this approach is to use the memory primitives with a well established modular redundancy technique that offers error mitigation by mirroring the entire memory content. In this case, additionally introduced voters ensure that each data, synchronously read from the independent memory blocks, undergoes a majority decision to mitigate SBU and even MBU within a single memory block. The penalty of increased resource consumption, arising from the voter usage, is compensated by savings from the missing SECDED encoder/decoder, but the necessity of continuous memory self scrubbing still persists to prevent error accumulation. A practical implementation of such a TMR memory scrubber for Xilinx FPGAs that operates on a configuration with multiple BRAM primitives can be found in [467]. It can be automatically generated and applied to a selected user design by using the Xilinx SEU mitigation TMRTool described in section 2.9.2.

2.8 Software Fault Tolerance

Although error mitigation by following the hardware and firmware approaches explained in sections 2.6 and 2.7 offers extensive capabilities for the protection of circuits against SEEs, these methods are restricted to circuit designers or specific hardware devices. In consequence, they cannot be applied later on to fixed, hard-

wired ASICs. This limitation also applies to FPGA-embedded complex primitives, as known from the PowerPC and ARM versions of Xilinx's Virtex and Zynq device series. The sole exception comes along as soon as the ASIC in question accepts to run a customizable set of commands that allows to compare or recalculate data as commonly supported by microprocessors. This software-based approach introduces error mitigation to higher system layers and is commonly referenced as software fault tolerance or Software Implemented Hardware Fault Tolerance (SIHFT). The basic errors that occur during software execution in such a processor's data and control flow are analyzed in [468]. Basically, they can be summarized to:

- Computational errors which cause erroneous calculations, such as `add ecx,03` instead of `add ecx,01`
- Control flow errors which modify the instruction sequence, for example `cmp ecx, edx` and `je _exit` instead of `jne _exit`
- Memory errors caused by incorrect values or addressing, such as `mov ecx,3` instead of `mov edx,3`

The mitigation of these soft errors can be handled at different levels of granularity. The most coarse-grained approach to software fault tolerance is handled at application level and features the parallel execution of two redundant program instances on a multithreading processor [469, 470, 471, 472] or in two virtual machines [473]. An associated hypervisor automatically takes care of consistency checks and output comparison. It furthermore monitors all instances and checks for errors. Since this approach is fully compatible to all operating systems and compilers, it provides the maximum flexibility at the price of the highest memory consumption and calculation overhead.

A more fine-grained and commonly used approach of SIHFT operates on single processor instructions. It uses either plain N-version redundancy [474] to multiply variable assignments that are voted with additional comparison functions when read, or adds data checksums [475] to limit the redundancy overhead. As these approaches result in extensive software program modifications, conceivably performed automatically as shown in [476], it might cause difficulties with existing compilers and their corresponding optimization strategies, similar to the behavior of synthesis tools as mentioned earlier. To simplify this process, a SIHFT implementation called SoftWare Implemented Fault Tolerance (SWIFT) has been released [477]. SWIFT offers improved control-flow checking that reduces the number of validation instructions, necessary to check variable synchronicity. It also reduces cache and memory consumption by substituting the variable duplication approach with

ECC support and adds methods for interrupt and exception handling. But although SWIFT is a robust fault tolerance technique, it can miss errors as explained in [478] and therefore has to be combined with other mitigation approaches, such as CRiticAlity based Fault Tolerance (CRAFT) [479, 480] or Profile-Guided Fault Tolerance (PROFiT) [478].

Since the manual implementation of SIHFT is not practical, several techniques have been integrated in special custom compilers to provide transparent usage:

- **Error Detection by Duplicated Instructions (EDDI)** [481] is based on the automatic duplication of program instructions at the software compilation stage to enable the detection of errors at system runtime. It tries to minimize the inevitable performance overhead by the optimization of instruction parallelism. This fault tolerance approach is similar to DWC for hardware and firmware error mitigation but without the necessity of circuit redundancy. EDDI duplicates register and memory data structures to mirror a program and eliminate any interference between both instances. The additionally required comparison instructions that decide about the correctness of a calculation result have been added prior to storing a value in memory, since "there is no need to compare intermediate computation results that propagate and corrupt final results" [481]. It has furthermore been proposed that EDDI provides "over 98% fault-coverage without any extra hardware for error detection" [481]. But finally, EDDI can not detect control flow / branching errors.
- **Control Flow Checking using Software Signature (CFCSS)** [482] offers software error detection in a program's control flow by monitoring the correct traversal of functions at custom granularity. It is based on the prior calculation of each function's unique signature as well as the generation of a control flow graph that reflects the correct traversal tree. These signatures as well as instructions for error detection are afterwards added to the program code at the compilation stage. The software program itself is now able to perform error detection by comparing the currently called function's signature with the stored set of signatures that are locally valid for correct program flow. This CFCSS approach has been announced to miss only "3.1% of branching faults [that] produced undetected incorrect outputs" [482]. Since the use of software signatures to mitigate control flow errors underlies some limitations when used completely without hardware mitigation techniques as shown in [483], hybrid solutions that utilize a combination of software and hardware / firmware fault tolerance for microprocessors have been devel-

oped. Therefore, the improved control flow error mitigation method becomes available as soon as an external controller can be connected to the CPU, for example when using FPGA-embedded processor cores. This controller then acts as a watchdog and continuously checks the full program control flow, not only the local one, based on the previously generated signatures and control flow graph. In detail, the CPU announces the signature of every called function, either by shared memory or message passing, to the watchdog, which immediately checks validity based on the full progression graph. This approach is depicted in [484, 485] and has been continued in the Hybrid Error-detection Technique through Assertions (HETA) [486, 487] to offer "full fault detection against control flow errors with performance and area overhead up to 11%" [486]. Other hybrid solutions that provide reliability against transient errors in microprocessors are given for example in [488] or [489].

- **Error Detection using Diverse Data and Duplicated Instructions (ED^{4}I)** [490] is based on the parallel execution of two slightly modified program instances with the same functionality and a subsequent result comparison. Therefore, all integer, floating point and exponential numbers in the duplicated program instance are transformed with a given algorithm before any calculation takes place. The returned results are afterwards transformed backwards by using an inverse operation for comparison with the unmodified program instance. This approach is similar to RAC (see section 2.7.4) and therefore works only with arithmetic calculations and an optimized transformation algorithm.

Whether the proposed fault tolerance technique is completely software based or a hybrid solution, the common disadvantage in all approaches results from the added instruction redundancy: It increases program and data memory and degrades performance [482, 491]. A practical test of SIHFT considering the use in space radiation environments has furthermore proven that a COTS processor exhibited more failures, although most of them were corrected, and crashed more often than a radiation hardened processor, which was "the more reliable of the two" [492]. For additional information about software fault tolerance, please refer to the reviews in [493, 494, 398].

2.9 Automated Fault Tolerance Tools

Fault-tolerant system design does not necessarily have to be understood in every particular detail. A large variety of currently available as well as formerly discontinued error mitigation tools offer general assistance when applying the well estab-

lished TMR technique to an existing user design while operating more or less in a black box. All of them aim on the reduction of development time as well as necessary expertise but all of them have one major thing in common: They operate on netlist descriptions only (see section 2.2.1). The following sections provide a basic overview of these tools and tool sets, available from educational institutions to commercial enterprises.

A first algorithm towards this direction is given in [495] with Selective Triple Modular Redundancy (STMR). It analyzes the netlist description of a current design, identifies critical elements based on a set of definitions and introduces TMR and voter circuits based on the previous analysis results. While trying to reduce the logic resource consumption in comparison to XTMR, it is announced to "provide immunity against SEUs comparable to that with full module TMR, with less area overhead" [495].

2.9.1 Mentor Graphics Precision Rad-Tolerant Tool

In 2010, Mentor Graphics® released their Precision® Hi-Rel (formerly Precision® Rad-Tolerant) synthesis product for vendor independent high-reliability applications [320]. Developed under the guidance of NASA, the tool focused on the mitigation of soft errors by automatically introducing TMR during the RTL logic synthesis process for better optimization results, mostly to complement the reliability of specifically designed radiation-tolerant FPGAs (see section 2.6.3), but also capable of improving COTS devices. The Precision® Hi-Rel tool was capable of mitigating SEUs and SETs by handling TMR in combinatorial logic, flip-flops, I/Os, buffers, RAMs, DSPs as well as their combinations in conventional registers or shift registers within various devices of different vendors. Logic equivalence checkers and fault injection routines could be used to verify correct behavior of the generated netlist description before starting with place and route. The additional FSM processor was capable of securing FSMs by introducing parity bits to reach a state encoding Hamming distance of d=3 and therefore be able to correct invalid states at runtime without interruption. This automated workflow, designated for antifuse, flash memory, and SRAM FPGAs, could be used to avoid a time consuming manual design of TMR with all its advantages and disadvantages (see section 2.7). In 2012, the Mentor Graphics® Precision® Hi-Rel software had been export restricted by the U.S. Department of State [319] and later-on discontinued [321]. Therefore, it is currently not available for custom designs anymore.

2.9.2 Xilinx TMRTool

Specifically offered for FPGAs made by chip vendor Xilinx, the TMRTool software extends the logic circuits of an existing user design in EDIF or NGC netlist description with XTMR fault tolerance functionality (see section 2.7.2). In combination with the well known configuration scrubbing process explained in section 2.7.7, XTMR provides better protection in contrast to classical TMR and promises "full SEU and SET immunity for any Virtex FPGA design" [496]. Similar to the Mentor Graphics® Precision® Hi-Rel tool, it follows an automated approach that has been integrated between synthesis and mapping (see section 2.2.2) of the conventional tool chain while staying fully transparent for the user. To make use of it, a given non-TMR RTL description has to be synthesized into a device-specific NGC or a translated NGO format, especially when using EDIF, to be qualified as input for the Xilinx TMRTool software [157]. The subsequently generated, fault-tolerant output file is given in EDIF again and can therefore be transferred to other devices or re-integrated in the conventional tool chain for further processing and final device configuration bitfile creation.

The automatic enhancements of the Xilinx TMRTool basically involve triplication of I/O pins and buffers, clocks, combinatorial logic, voters, and also include the insertion and synchronization of feedback paths [496, 157]. The software also triplicates conventionally used BRAM primitives or primitive arrays, integrates BRAM voters and automatically adds memory scrubber macros [467]. In contrast to the Mentor Graphics® Precision® Hi-Rel software, FSMs are implemented with TMR and the synchronicity between each redundant leg is guaranteed by introducing locally voted feedback paths [496, 497]. Although the option for half-latch removal in Virtex and Virtex-II devices is optional by default, it should be considered mandatory when using such a device in radiation susceptible applications as explained in section 2.3.7. The same applies to LUT shift register (SRL16) extraction [496], since shift register LUTs are not capable of being refreshed by static configuration scrubbing. Finally, a fully XTMR-protect design can consume up to six times more resources than the original one [412], therefore a general usage may not be economical in all cases.

Beside of the many disadvantages that are arising from redundancy approaches, such as an increased resource consumption, wattage and PCB complexity as well as a decreased maximum clock speed, the Xilinx TMRTool offers significant improvements regarding the mitigation of SEE in SRAM FPGAs while in parallel keeping design productivity at a very high level by saving valuable development time. Unfortu-

nately, the latest version of the TMRTool is currently stuck at Xilinx ISE® v13.2 and cannot be used with the most recent software, especially not with the latest Vivado® Design Suite. Furthermore, the device family support has been officially restricted to the radiation-tolerant space-grade FPGA series Virtex-4QV and Virtex-5QV [497].

2.9.3 BYU-LANL TMR Tool

Brigham Young University (BYU) and Los Alamos National Lab (LANL) offer an academic open source GNU GPL Java tool that is able to identify and protect SEU-sensitive circuit structures within the EDIF netlist description of a given user design. The sensitivity is hereby defined via the possibility of causing a persistent soft error. Persistent in this context means that the error cannot be corrected by static configuration scrubbing (see section 2.7.7) and therefore causes permanent miscalculation of results as known from circuits with loopback signal paths. Critically identified structures are subsequently partially protected with DTMR to improve resource and power consumption in comparison to a full GTMR approach [498] (see section 2.7.2).

The automated fault tolerance software is called BYU-LANL TMR (BLTmr) tool. It operates on an algorithm that separates the designated circuit design into three major sections: Circuit structures that contain feedback signal paths as well as structures that provide input to and output from these identified feedback circuits. The mitigation scheme afterwards operates on these structures and initially triplicates all logic blocks that contain feedback signal paths, adds triplicated voters, and realigns the feedback signals to be fed from the corresponding voter's majority decision. In a second step, all circuits providing input to these recently generated redundant legs are triplicated themselves. Finally, all circuits that are involved in the further processing of output signals coming from the feedback structure itself are also triplicated.

The major advantage of this automated approach, beside of the significant time saving in comparison to a manual design, lies in the trade-off between an affordable resource consumption and the mitigation necessity. The available device resources are optimally utilized by protecting only the most sensitive circuit structures that contribute with the highest probability to a permanent design failure. This allows bigger designs to be used in smaller and hence cheaper FPGAs. In consequence, some parts of the design may remain unmitigated and therefore can cause miscalculation until a scrubbing cycle repaired the corresponding upsets and the erroneous data was shifted out of the device.

2.9.4 Politecnico di Torino RoRA Tool

As explained in section 2.7.2, spatial redundancy, including XTMR, fundamentally relies on the disjoint implementation of all redundant entities to be able to correctly mitigate SEU. On the other hand, synthesis, place, and route algorithms try to minimize device usage while unifying functionally redundant primitives to save valuable device resources and to increase the overall system performance. These two approaches form a contradiction that has to be resolved to prevent single points of failure. Sharing LUT configuration bits between redundant voters for example results in uncorrectable situations as soon as one of these bits has been upset.

The Reliability-oriented place & Route Algorithm (RoRA) from Politecnico di Torino focuses exactly on this problem. It is a custom TMR place and route software for SRAM-Based FPGAs that improves the routing distance between all redundant domains [499, 294, 500]. The algorithm itself takes a mapped design as input, identifies all logic functions, triplicates them and introduces a set of new voter circuits. Afterwards, the custom placement tool physically allocates primitives for all logic functions and voters in non-overlapping partitions and the subsequent routing process finally tries to find the shortest connection paths between each two connected primitives while avoiding domain cross-overs to bypass additional single points of failure. Details on the exact algorithm as well as on the graph representation can be found in [500].

RoRA has proven SEU mitigation reliability during error injection, while revealing 62% less errors in comparison to conventional TMR [294]. Unfortunately, the physically widespread distribution of logic elements results in significant device performance degradation of about 40% in comparison to the non-TMR designs and about 25% in comparison to conventional TMR [499]. Furthermore, the PIP routing resource consumption increased by about 25% in comparison to conventional TMR [500], which will quickly fill a device, especially the cheaper devices with less routing resources such as the Xilinx Spartan series.

2.10 SEE Simulation Tools

SEE Simulation is an essential task while preparing an FPGA firmware design for use in radiation environments. First of all, it allows the determination of a given design's initial SEE susceptibility to enable comparison against the final, fault-tolerant designed version. This ensures that the selected fault tolerance strategy does not increase SEE susceptibility when implemented fragmentary while adding single

points of failure. Simulation is furthermore able to estimate the total number of susceptible bits within a specific device configuration, as most of the available bits remain unused or do not impact the functionality of a running design when upset.

SEE simulation, monitoring and analysis in general can easily be performed by the use of an HDL simulator such as Mentor Graphics® ModelSim® that operates on the models of a firmware design but which is required anyway to debug logic and timing behavior. ModelSim® therefore offers a 'force' command that applies a well defined SEU for a specified time to a given signal of a designated model. Since all signals have to be simulated independently, this injection method requires a significant amount of time before the SEU propagates through the whole design, but offers the highest flexibility when it comes to the simulation of precisely designed combinations of device primitives. An automated tool that follows the ModelSim® approach is the Nebrija SEUs Simulation Tool (SST) from Universidad Nebrija [501]. It is based on a set of scripts that allows to effectively simulate SEUs within a user design and check the result in a surrounding test bench. A graphical user interface is available to guide through all simulation parameters.

As soon as a first hardware prototype becomes available and due to efficiency and randomness concerns, SEE simulation for Xilinx FPGAs is mostly done by injecting errors either within the static device configuration or in the dynamically processed data stream or clock tree. The static error injection itself can furthermore take place offline, before powering the device, by flipping bits in the initial configuration file or it can be done online, while operating a device, by modifying bits in the specifically prepared scrubbing configuration file (see section 2.7.7). Both require a deactivation or adaption of the bitfile-internally stored CRC value. While the first method allows an observation of the reset behavior in case of a permanently damaged configuration bit, the second method investigates the general miscalculations of an erroneously operating design or misconfiguration due to a broken scrubbing controller and therefore allows better coverage of the general usage scenario in real-time.

The following enumeration tries to give a basic overview of currently available, automated SEE simulation tools for custom firmware designs to ease selection:

- The latest **Xilinx Soft Error Mitigation Controller LogiCORE** [455] is not only capable of performing FPGA configuration self scrubbing as explained in section 2.7.7, it also officially supports configuration error injection to evaluate the reliability of an application. A specifically designed error injection interface is used to easily send basic control commands as well as designated bit addresses to the controller. All commands can be sent FPGA-internally or

from an externally connected injection controller. The monitor interface can afterwards be used to check the injection or its correction state.

- The **STatic AnalyzeR (STAR)** [502] software from Politecnico di Torino is a verification tool that operates on a given fault-tolerant FPGA design to identify sensitive device primitives that are shared between multiple legs of spatial redundancy and therefore affect multiple voter domains when upset. It returns a list of critical configuration bits that can be used as input data for several SEE simulation tools that operate on configuration bit manipulation. The tool has been created by reverse engineering the configuration bitfiles of different FPGAs and therefore supports only few devices, predominantly Xilinx.

- The **FLIPPER** [503] error injection platform for SRAM based FPGAs from IASF Milano focuses on the emulation of SEUs within configuration memory and flip-flop primitives of Xilinx SRAM FPGAs. It follows a base- and daughterboard approach, whereas the test FPGA is located at the daughter board and therefore can easily be exchanged. The SEU and MBU injection itself is handled via partial reconfiguration frame scrubbing with manipulated bitstreams. In consequence, errors between several injections are accumulating as long as no reset has been triggered. A subsequent set of test vectors is sent to the FPGA to generate the synthetic output signals which are logged for later analysis. Unfortunately, the development has stopped with Xilinx Virtex-II FPGA, although the basic concept allows quick replacement of the FPGA.

- The **Fault Tolerance UNiversity of Sevilla HArdware DEbugging System (FT-UNSHADES)** [504] is a specifically designed, Xilinx-based, error simulation platform that allows SEU injection for flip-flop and memory primitives in real-time. This functionality is provided by combining partial reconfiguration with previously calculated test vectors that cover all critical bits of the designated user design. To be able to operate in real-time, the test design is duplicated and implemented twice within a large SRAM FPGA. One of the simultaneously working instances remains unchanged, while the other one is modified with the injected error. All input vectors feeding both instances are stored in the on-board memory chips. The output signals are immediately compared by a subsequent voter and differences are logged for later SEE analysis. Although the Virtex-II FPGA on the FT-UNSHADES board is obsolete, the basic concept remains beneficial and has furthermore been updated to simulate SET and MBU [505].

- The **SET Analysis (SETA)** tool [506] from Politecnico di Torino is able to analyze the SET sensitivity of a test design for flash-based FPGAs. Since propagation of an SET within the logic gates affects its voltage amplitude the longer it traverses, a detailed map of the internal signal paths is required to successfully estimate the final SET pulse shape and PIPB effect (see section 2.5.4.1). Therefore, the SETA tool offers a library for at least the Actel/Microsemi ProASIC3 FPGA, that has been built based on a self-developed analytical model. Even if the SET pulse shapes do not match exactly the real conditions, they can be easily adjusted in the tool.

- The **BYU SEU Simulator** [507] from Brigham Young University is based on the SLAAC1-V system board, that offers three conventional Xilinx Virtex FPGAs. Two of them are operating synchronously with the same initial test design. The third FPGA has a supporting role. It provides all input vectors for the test design and compares the generated output signals. As soon as a difference has been detected, the error is logged for later analysis. A difference can occur as soon as one of the test design FPGAs is modified by error injection via a specifically prepared partial reconfiguration bitstream, while the other FPGA remains unchanged. The final list of sensitive locations within the FPGA design can afterwards be viewed in a graphical representation of the FPGA.

- The **USU SEU Simulator** was a former simulation system available from Utah State University. Test design as well as simulation controller circuit were independently partitioned into a single FPGA by the use of the Xilinx PlanAhead software. An On-Chip Peripheral Bus (OPB) connected both entities for error detection and output redirection to a host computer. Since the partition frame boundary data and simulation software were directly integrated in the full configuration bitstream, the simulation controller had to operate ICAP as well as timer functionality and therefore required a significant amount of chip resources. The simulation covered all configuration and flip-flop data within the test partition but not beyond and without considering embedded BRAM content.

All of the above mentioned tools are focusing on the simulation of SEE in given firmware circuits. They cannot cover errors within embedded CPU hard cores as available for example in the Xilinx Virtex-4 FX FPGA series. When it comes to the simulation of software errors, a software simulator as presented in [508] has to be chosen. The simulator integrates specific commands among the native program

code that are able to selectively modify the processor's vulnerable registers and data storage at a given time and therefore act as artificial error injector. A software testbench may furthermore automate the process of randomized error injection and result validation. The herein given code example from [508] was based on an 8051 CPU simulation and caused 76% upsets that did not impact the system behavior since they occurred in registers that were not in use, while 24% of the upsets were observable. The authors state that 36% of the observable errors caused a permanent system hang, 60% issued a temporary out of time command execution while only 4% were erroneously altered command executions [508]. But this does not generally mean that firmware errors should be neglected completely when using FPGA-embedded processors, as there are plenty of circuits required to successfully connect and operate such a device. This specific susceptibility has been addressed for example in [509, 510] while running a Linux operating system on a Virtex-4 PowerPC. Both processor and firmware vulnerabilities therefore have to be summed up to be aware of the final device susceptibility.

In any case, when simulating software for the execution on radiation susceptible processors, special care has to be taken regarding the CPU-internal cache memories that cannot be accessed from outside. They are built of conventional CMOS storage arrays and therefore show radiation effects as well. Data, which resides within such memory for a longer period of time may get damaged and has to be considered with an increased upset probability during simulation.

3 Approaching Fault Tolerance for FPGAs

As soon as the advantages of FPGAs become an essential design requirement, several aspects have to be considered before being able to use semiconductors in critical radiation environments and form a final decision about their device specifics such as technology or capacity. Based on all given information regarding radiation effects and fault tolerance for semiconductors summarized in chapter 2, a basic concept for a successful usage of FPGAs in radiation environments, spanning all layers of modern computer architecture, has to be defined. This concept, an essential part in the construction phase of particle accelerator readout electronics for example, should combine the best of all available techniques while incorporating considerations about the available financial as well as personnel resources to stay within the budget, speed up custom firmware development and keep the design complexity at a reasonable level. The following chapter now tries to summarize a basic approach to this concept of using FPGAs in critical radiation environments as well as some ideas for improvement. Looking only at the physical composition of FPGAs (see section 2.3) suggests, that this requires different strategies: The combinatorial configuration matrix for example has a static context, while the processed data changes dynamically at runtime.

3.1 Device and Board Selection

When starting a new project with parallel information processing based on FPGAs, that further on will be used in ionizing radiation environments, the most basic PCB design layer first of all requires the selection of one or more adequate FPGAs. Due to the frequently occurring radiation effects in semiconductors as indicated in section 2.5, it is obvious that action has to be taken with regard to radiation tolerance, especially TID (see section 2.5.5.2) and SEE cross-section (see section 2.5.2). Since every device technology, manufacturing process and even every silicon waver batch of the same device exhibits varying radiation sensitivity, the optimal choice would be to use a specifically designed, radiation-tolerant FPGA with controlled and guaranteed silicon characteristics as explained in section 2.6.3. Since these devices are the most expensive ones available on the market, this choice usually becomes un-

available as soon as a higher number of devices is finally planned. Therefore, the use of COTS FPGAs is mostly inevitable. In this case it is recommended, to purchase a single device in advance and run a custom radiation test to determine all critical silicon characteristics independently. As soon as all results indicate usability, more devices of exactly the same batch can be purchased. Since this individual testing procedure is very time-consuming, a third but most inaccurate option is to rely on external test results as provided in publications or component databases such as available from the CERN Radiation Working Group (RadWG). Unfortunately, these external sources usually do not reference a specific silicon waver batch. In addition to the central processing FPGAs, the same testing procedure also applies to all other components placed on the PCB that get in contact with critical radiation but which exhibit individual TID and cross-section specifications.

As soon as suitable COTS FPGAs have been selected, several PCB design guides for fault-tolerant system design come into operation. They include the Xilinx recommendation to hard-wire all redundant input and output pins and therefore replicate the internal IOBs as explained in section 2.5.7.4. This in turn significantly reduces the number of totally available signal pins and may require a larger, more expensive device package to be selected in advance. The necessity of firmware fault tolerance (see section 2.7), that fundamentally relies on the disjoint implementation of I/O pins, increases the resource consumption in addition. SRAM FPGAs furthermore require configuration provisioning and preservation which requires hardware components and interfaces to be placed on the PCB which themselves have to be tolerant against the expected radiation levels. All this increases initial acquisition costs but also stresses power consumption and therefore continuous operational costs. Moving PCB and FPGA out of the radiation field might mitigate the necessity of this overhead for some fields of application, but especially in case of particle accelerators, generated secondary particles may also reach and harm distant electronic parts.

A conscientious weighting of these first aspects about **radiation tolerance** and the associated selection of an adequate hardware platform builds the essential base for any ongoing development of firmware and software. It furthermore decides about the necessity of additional **fault tolerance** techniques as well as their extent and therefore defines the more or less comprehensive investment, necessary for development resources given in the following sections.

3.2 Static Configuration Scrubbing - Benefits and Limits

The flexibility of fast and efficient SRAM FPGAs is given by the opportunity to program and re-program a wide range of configurable routing and logic elements within the device for custom use (see section 2.3). This configuration, once programmed, is kept within the logic cells as long as the device is powered or the content has intentionally been changed. Unfortunately, it can also be altered accidentally by ionizing radiation as depicted in section 2.5.7 and therefore has to be refreshed before too many upsets start to emerge in parallel. This requires a fast and efficient configuration controller, which can be realized FPGA-internally without additional costs or externally by planning dedicated hardware. The internal Xilinx SEM controller is promoted to scrub in 25 ms [455] while the speed of an external version mostly depends on the custom controller's efficiency, the utilized scrubbing technique as well as the configuration interface itself as seen from table 2.10. All types of scrubbing techniques as well as additional analysis can be found in section 2.7.7. The advantage when opting for an external unit is that additional watchdog capability comes for free, just by wiring a few dedicated signals. This scheme was the chosen one on all SysCore developent platforms [511, 512, 513] (see section 5.1.3). The disadvantage, beside of the construction, development and operating costs, lies in the radiation susceptibility of the introduced controller itself. Therefore, it has to be clarified in advance, that an external scrubbing controller part does not increase the overall radiation susceptibility and that it is specified for the expected total dose that accumulates over the entire operating period.

The process of continuous configuration scrubbing at runtime via DPR easily prevents the accumulation of errors in device routing and logic, but is limited to the static parts of the device. This means that configurable components such as BRAM, SLICEM, SLR16 or simple flip-flop primitives, which contain dynamic data, need to be bypassed and have to be mitigated by using a separate fault tolerance technique later on. Since all of the configuration data for static and dynamic components are coupled within a single configuration bitfile for initial programming, the bypassing can be easily handled by providing an additional mask file that contains information about the according bits and that is processed by the scrubbing controller.

Although only about 40% of the routing bits in a conventional design are utilized [251], configuration scrubbing, when implemented correctly, significantly improves the FPGA behavior in ionizing radiation environments. It is the most ba-

sic method to preserve static context within SRAM FPGAs without the necessity of power-cycling the device after each upset. Although a full reset can also be considered as a simplistic form of scrubbing which might be suitable for low radiation applications with lesser errors, re-initiating a device requires valuable time and furthermore eliminates all data which has been calculated so far. Finally, configuration scrubbing has to be considered as a prerequisite for the operation of COTS SRAM FPGAs in radiation environments and as a necessity for all supplementing fault-tolerant design as depicted in the next section 3.3.

3.3 Redundant Firmware Design

Preparing a COTS FPGA design for use in ionizing radiation environments should not solely rely on static configuration scrubbing, which is able to repair upsets within the device's configuration, but cannot cover erroneous calculation results in-between scrubbing cycles or changes within dynamic data itself. Especially temporary configuration upsets in FSM controller logic may critically change a full system's behavior that can only be repaired by a full system reset. Therefore, the second requirement for COTS FPGAs in ionizing radiation environments is firmware fault tolerance. This kind of safeguarding in all its variations (see section 2.7) enables "fault avoidance, fault masking, detection of erroneous or compromised system operation, containment of error propagation, and recovery to normal system operations" [514]. It is all along based on redundancy and therefore requires additional device resources. The most secure version is XTMR (see section 2.7.2) offering circuit and pin redundancy as well as redundant, local feedback paths. It is able to vote out dynamic storage upsets and restore transient faults that have passed into storage cells.

As mentioned earlier, XTMR results in a very high resource consumption – up to six times of the original design size [412]. If this maximum protection level is urgently required, additional costs for bigger FPGA devices shall not be strictly refused, special radiation hardened devices are far more expensive. Even if redundancy is merely optional, since erroneous calculations can be repeated or hardware can be reset without information loss, filling up the remaining device resources with fault tolerance may reduce the overall number of necessary interventions and therefore increase data throughput.

Manually applying fault tolerance to an existing design on RTL is a highly time consuming and therefore expensive as well as complex and therefore error-prone process, that, if implemented fragmentary, may result unnoticeably in weakness of the whole design. It furthermore results in bloated hardware descriptions that are

difficult to understand and nearly impossible to maintain or extend. In addition, the synthesis tools, whose tasks are lying in the reduction of design resource consumption as well as timing optimization, afterwards try to remove all forms of introduced logic signal path redundancy if not explicitly excluded on an individual signal base. For a first impression, please refer to the guidelines given in section 4.5. To the knowledge of the author, there is currently no software tool available on the market that provides technical assistance in adding and managing fault-tolerant circuits, including the required tooling statements, on RTL.

Furthermore, when it comes to place and route of the firmware design, particular attention has to be paid on the parallel usage of components between independent redundancy domains. A significant TMR placement and routing distance between these domains can prevent inter-domain routing shorts with bridging faults, that critically corrupt multiple instances of the redundancy voting mechanism as depicted in [414, 415]. Since manual definition of such regions is even more time consuming and error-prone, an automated tool such as the Reliability oriented place & Route Algorithm (RoRA) promises assistance (see section 2.9.4).

Finally, fault-tolerant firmware design can significantly increase the operating reliability of COTS FPGAs in ionizing radiation environments, but only in combination with static configuration scrubbing that refreshes all redundant logic parts and prevents error accumulation. A manual design approach on RTL is possible, with all assets and drawbacks, but requires high financial efforts. Furthermore, it is impossible to give a quality assurance in advance without running post synthesis SEE simulations (see section 2.10) or performing adequate irradiation tests. Using quality-proven, fail-safe design packages or all-in-one fault-tolerant Intellectual Property (IP) cores may reduce this risk and can increase development speed.

3.4 Automatic Firmware Redundancy

To eliminate some major drawbacks introduced with manual, fault-tolerant firmware development, such as the increased design complexity and error-proneness, several software tools are available that automatically apply redundancy to existing circuit designs. Educational versions free of charge as well as commercial products offer different approaches to TMR or context sensitive routing as see from section 2.9. They furthermore dispense with the necessity of designer expertise regarding all firmware redundancy techniques explained in section 2.7 to provide quick out-of-the-box results. Unfortunately, all of them are operating on netlist description level (see section 2.2.1), which makes it impossible to exert influence

on the mitigation process or its extent. Especially the commercial Xilinx and Mentor Graphics® tools are well integrated into the existing design tool workflows. But given that the market demand for such products is not that high and export restrictions may furthermore limit usage in some countries (see section 2.5.10), most of these tools are deprecated or finally discontinued. The Xilinx TMRTool for example is currently stuck at version 13.2 of their ISE® and furthermore cannot be used with the latest Vivado® Design Suite. Even the device family support has been officially restricted to the radiation-tolerant space-grade Virtex-4QV and Virtex-5QV FPGA series [497].

Finally, using an automatic software tool to apply fault tolerance may solve some problems, but it also introduces new ones, such as the limited scope of influence as well as the missing transparency and maintainability. Minor updates within the original HDL design in particular require a complete regeneration of the fault tolerance design including the execution of all subsequent simulation, verification or testing tasks. Due to the automated approach, there is also no possibility to limit the resource consumption at designated instances which may finally result in a redundant design that is larger than the current chip or even larger than the biggest available FPGA.

Most of the other drawbacks of fault-tolerant firmware design remain unchanged, even with the automated mitigation approach: The complexity of the PCB layout remains raised, the increased redundant design size still requires acquisition of a much bigger FPGA, the redundant calculation logic will still stress power consumption and increase the continuous operational costs, the more complex routing still decreases system performance, and the probability of an SEFI still persists, even with XTMR [251]. Considering all of these aspects, the use of such fully automated tools for error mitigation is not recommended in general. A semiautomatic approach that allows selective mitigation schemes might be a more suitable solution.

3.5 Semi-Automatic Fault Tolerance with Steering

As most of today's projects are strictly limited in budget or operate close to the edge of technical feasibility, extensive fault tolerance introduced by automated software might not be available in every case. But on the other hand, manual, fault-tolerant firmware design quickly becomes highly complex and nearly unmanageable for larger designs. Therefore, the perfect compromise between automatic and manual error mitigation approaches at least on the firmware level of a full system design would be a software tool that increases firmware fault tolerance usability by

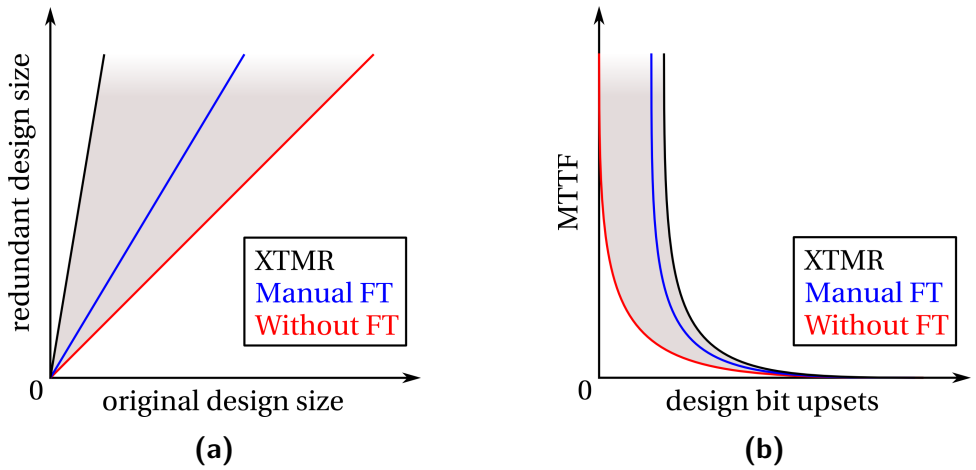


Figure 3.1: Simplified comparison between conventional Xilinx FPGA firmware designs without fault tolerance techniques (red line) and particularly prepared Xilinx Triple-Modular-Redundancy (black line) designs for use in radiation environments. The possibilities of a manual firmware optimization (grey area) are situated in-between and depend totally on the kind of design as well as the specific implementation techniques added by the developer. Its expected benefit (blue line) is shown for example. The comparison includes firmware resource consumption (left diagram) as well as Mean Time To Failure (MTTF) (right diagram).

introducing assisted redundancy with user-scalable granularity based on regions or signals that have manually been identified by the designer due to individual importance criteria. The resource consumption of such a selective redundancy approach in relation to its original design size should therefore stay below that of conventional XTMR as depicted in figure 3.1a to enable usage of smaller, cheaper FPGAs or bigger firmware designs. In consequence, the effectiveness of error mitigation might be reduced and does not cover the MTTF of XTMR anymore, but it should clearly be better than a firmware design which does not use fault tolerance at all as illustrated in figure 3.1b. This behavior of course depends to a high degree on the successful identification of critical design paths that have a major impact on the whole design functionality, such as controller FSMs. As long as the software operates on RTL, it also needs to add compatible synthesis-software ignore-statements that prevent an optimization of all redundant instances which otherwise may defeat all error mitigation efforts. In addition to the introduction of redundancy, the software might also check and warn for configuration scrubbing incompatibilities by identifying significant device primitives or IP cores that indicate distributed memory.

This whole functionality may provide a device family independent flexibility which has never been reached before via existing tools or scripts. It would enable the firmware designer, who knows best about his critical signal paths, to explicitly distinguish between regions that need to be protected and regions that can handle temporary glitches. XTMR firmware designs that would have never fit into a given FPGA may be prepared with fault tolerance and FPGAs that have available resources may use them to implement additional protection. Reducing the total number of occupied device resources also improves signal timing and therefore device performance. Some practical examples which utilize the developer's knowledge in a selected redundancy approach, but which have been built completely manually without any software assistance can be found in [286], [27] and [399], in case of the last one with "71% saving of circuit complexity in comparison to XTMR" [399].

As long as the proposed software tool operates on RTL to perform a source-to-source transformation of VHDL or Verilog files, it can be compared with a code washing machine as known from software projects to generate compiler-optimized versions of C++ sourcecode as announced in [515].

3.6 Dynamic Data Retention in Memory Arrays

Setting up full microprocessor system designs based on FPGAs inevitably requires large memory arrays with high speed interfaces. As data retention via configuration scrubbing only covers static firmware components and as fault-tolerant design only bridges temporary configuration upsets and preserves basic storage cells via locally voted feedback paths, both techniques are unsuitable to cover errors in any of such memories. Moreover, a redundant use becomes mostly impossible due to limited storage availability or would result in waste of valuable resources as soon as the internally or externally connected storage component provides custom, hardware-embedded error mitigation features. Fault-tolerant dynamic memory use without feedback signal paths furthermore cannot prevent error accumulation in the storage cells, therefore, the technique of memory scrubbing as explained in section 2.7.8 cannot be ignored.

A custom dynamic memory scrubber that covers all connected storage devices therefore has to be designed for every fault-tolerant system. It needs to operate continuously in the background to be able to successfully prevent any accumulation of errors. Since it will be integrated into the conventional FPGA firmware, it has to be designed by using fault tolerance. The exemplified, hand-triplicated self-scrubbing macro given in [467] fulfills these requirements for FPGA-internal BRAM,

<table>
<tr><td colspan="4">Data Criticality</td><td>Low</td><td colspan="2">Medium</td><td>High</td></tr>
<tr><td colspan="4">Error Persistance</td><td>No</td><td colspan="3">Yes</td></tr>
<tr><td rowspan="4">SEU Rate</td><td rowspan="4">Low
High</td><td rowspan="4">Operating Window</td><td>Minutes</td><td colspan="2">No Mitigation</td><td colspan="2">XTMR</td></tr>
<tr><td>Days</td><td rowspan="3">Scrubbing</td><td rowspan="3">Scrubbing
XTMR</td><td rowspan="3">Redundant
Devices</td></tr>
<tr><td>Months</td></tr>
<tr><td>Continuous</td></tr>
</table>

Table 3.1: Xilinx SEU Mitigation Strategy Selection Guide taken from [251]. Depending on criticality of data, error persistence, SEU rate, and operation window, various fault tolerance approaches are recommended.

but is available in the Xilinx TMRTool only and therefore has to be redesigned for unsupported FPGA families (see section 2.9.2).

3.7 System-wide Fault Tolerance

Fault tolerance, "the capability of a system to recover from a fault or error without exhibiting failure" [514], is a highly complex and interconnected process when applied to all layers of modern computer architecture, beginning with basic hardware components up to complex software solutions. But only the combination of these individual techniques enables the maximum device reliability that can be applied to data processing systems based on COTS SRAM FPGAs when used in critical radiation environments. Dynamic memory scrubbing would be less effective without device hardware support, firmware fault tolerance would be less effective without static configuration scrubbing or board pinout support and finally, software fault tolerance for microprocessors (see section 2.8) would be less effective and more error-prone without all of them. Every new architectural layer benefits from the summarized mitigation approaches of all underlying layers and an error that has already been corrected within the device memory cannot cause calculation errors in software anymore. But as long as the time between error event occurrence, fault detection and fault correction is non-zero, even software fault tolerance has a justified existence to be able to distinguish between correct and incorrect data while preventing situations where "one and one" suddenly equals three. The extent of a

suitable mitigation approach on firmware level can potentially be reduced due to prior analysis of the processed data criticality as well as the upset rate of the underlying hardware platform – a general recommendation of different strategies can be found in table 3.1, taken from [251]. But even system-wide fault tolerance cannot solve the impossible. Therefore, few application scenarios which are characterized by long term operation with highly critical data might still require a redundant use of full devices.

4 System Implementation Details

The following chapter provides the interested reader with implementation details about the error mitigation techniques used for the fault-tolerant system design. It covers aspects of static configuration scrubbing, dynamic memory scrubbing as well as fault-tolerant firmware design for multiple revisions of the designated SysCore board platform and its different Xilinx FPGA families as shown in section 5.1.3. In addition, it contains implementation details about tools and fault tolerance software that has been developed in order to improve the overall process of firmware redundancy design.

4.1 Static Configuration Scrubbing

To comply with most of the fault tolerance design guidelines from chapter 2, the basic decision about a suitable hardware platform led to the selection of a SysCore board that had been available in its first and second version at that time. This development and irradiation test board, described in detail in section 5.1.3, fulfills most of the mentioned prerequisites for optimal error mitigation when no radiation-hardened device is available and COTS FPGAs have to be preferred. The only drawback is its FPGA size. Since the offered Xilinx FX20 FPGA is comparatively small and almost the smallest one in its device family [130], all redundancy approaches have to be limited to smaller, mostly experimental firmware designs.

Most of the relevant static configuration scrubbing basics about the Virtex-4 FPGA's internal structure was taken from the specific Xilinx documentation [516, 295]. This includes the fixed-length frame addressing scheme within the configuration bitstream as well as the composition of the configuration words with their various configuration registers. The process of communicating with the FPGA on low level is fully vendor-specific and can be compared to the operation of a complex machinery where only a specific handling sequence leads to the designated result. Reference [516] also recommends usage of an external configuration controller that holds the bitstream configuration and performs active readback with CRC check via the configuration interface. But in order to keep the controller's firmware implementation effort at a reasonable level that can be handled even by the simplest external hardware chips while operating at high speed, blind scrubbing via the parallel SelectMAP configuration interface was selected. Deactivation of the CRC fea-

ture furthermore enabled custom modifications within the bitstream command sequence, necessary for example during BRAM, LUT-RAM, SRL16, or DCM/MGT Dynamic Reconfigure Port (DRP) memory [295] removal.

This scrubbing practice has shown very good results for the Virtex-4 based SysCore v2 boards and therefore has been adapted for the updated SysCore v3 boards [511, 512, 513], which make use of a cheaper Spartan-6 COTS FPGA while keeping all former scrubbing functionality as described in section 5.1.3. The experimental results of this blind scrubbing technique can be found in section 5.2.

4.1.1 SysCore and Xilinx SelectMAP Interface

SelectMAP is a parallel configuration interface for Xilinx FPGAs. It supports master or slave operation with different word widths of 8 and 32 bit [459]. As seen from table 2.10, it provides 100 MHz operation at 32 bit data word width, which finally results in 3.2 Gbps maximum bandwidth. The interface width of SelectMAP is optimized for the 32 bit wide configuration words of Xilinx FPGAs and therefore can operate gapless without padding. A Virtex-4 FX20 firmware design without BRAM, containing about 5.5 million configuration bits (see appendix A), can therefore be transferred in about 1.7 ms. Practically, this number is limited by the maximum transfer speed of the used controller as well as the access speed of the external memory that stores the configuration data. Therefore, the general recommendation for the SelectMAP interface is a readback-based operation that returns the whole FPGA configuration, calculates the CRC on the fly and finally compares it to a previously stored value in memory. The subsequent scrubbing cycle is only performed in case of a CRC mismatch. Although this technique can detect SBUs much faster, it may fail on MBUs and requires more time for a full scrubbing cycle due to the readback overhead. Therefore, it can be improved by only reading back single fixed-length FPGA frames and checking the embedded ECC bits. In case of upset, only the specific area can be frame-scrubbed. As this procedure may also fail in case of MBUs, the combination with CRC promises maximum protection.

It is important to note, that special care has to be taken regarding the configuration register CTL. Its third bit has to be defined with logic one to guarantee persistence of the SelectMAP interface after initial configuration, otherwise it will be disabled and scrubbing will not be possible. Furthermore, bits 4 and 5 have to be set to zero to enable basic configuration read and write. Finally, bit number 8, GLUTMASK (see section 2.7.7), has to be fixed to logic zero to disable LUT-RAM and SRL16 reconfiguration and protect the dynamic data contained herein.

The SelectMAP interface of the Virtex-4 FPGA is operated in static slave configuration mode `M(2:0)=110` with an 8 bit wide bidirectional data bus `D(7:0)` according to [459]. Therefore, the maximum data transfer speed is reduced accordingly by a factor of 4^{-1} to 800 Mbps. The SelectMAP direction is selected by switching the RDWR_B port signal between logic zero for configuration and logic one for readback. To operate synchronously with the scrubbing controller, a dedicated configuration clock input signal has to be provided on the CCLK port, which is sampled at rising edge. The CCLK clock port can also be driven in output mode as soon as RDWR_B switches to the configuration readback mode. To delay its sampling in readback mode, an additional DOUT_BUSY port is available, but has to be sampled in the controller itself. During configuration or readback, the chip select CS_B port has to be kept low. As soon as the initial configuration has finished, the DONE port is driven with logic one to indicate completion and INIT_B indicates if a CRC error occurred. As configuration readback via the scrubbing controller is currently not supported, RDWR_B remains static at logic zero and DOUT_BUSY is not available.

The scrubbing controller on the SysCore board is physically provided by a dedicated Actel/Microsemi ProASIC3 A3P125 FPGA. This FPGA is based on flash memory technology and therefore its configuration is immune against SEUs. While being too slow to perform high performance tasks designated for the Xilinx Virtex FPGA, it can perfectly act as concurrent controller on the board. Its primary objectives are the initial Virtex configuration as well as its continuous blind configuration scrubbing from the connected memory chips. The ProASIC3 FPGA furthermore acts as a dedicated FPGA watchdog that asynchronously resets the full FPGA via the PROGRAM_B port in case of any unexpected behavior. Therefore, it is observing a transmitted, alternating signal that is calculated at various positions in the Virtex firmware design to indicate correct operation. In addition, it is monitoring the susceptible SelectMAP interface itself [517], which contains the configuration registers COR and STAT that can be upset (see section 2.5.7.7) and cause an SEFI as observed in [306]. Such errors can only be fixed by a full device reconfiguration.

All configuration data for the Xilinx Virtex FPGA is stored in two externally connected Macronix MX29LV640 flash memory chips which are immune against SEUs and offer 5 ns SET glitch protection on the control signals. Due to constructional reasons, they share the same 22 bit address space and therefore have to be selected via both separate chip enable signals. These chips have been replaced on the SysCore v3 board with more radiation characterized and more recent versions of Micron MT29F4G16 Single-Level Cell (SLC) ECC NAND flash memory with durable charge

pumps and support for a burst read mode to improve the overall scrubbing speed. Especially the SLC memory provides "greater radiation robustness, reliability, and endurance" [518] in comparison to the Multi-Level Cell (MLC) chips. While the first chip stores the initial, full configuration bitfile, the second one holds the prepared blind scrubbing bitfile. Since there is no external interface at the ProASIC3 FPGA, programming of the memory chips has to be done via a dedicated Virtex FPGA design, that forwards all programming data from a management computer to the ProASIC3 memory controller. Reading and writing from and to the flash cells is handled by correctly setting the 22 bit address register as well as the chip enable, output enable and write enable control signals. Since direct overwriting of flash memory cells is not possible, they have to be erased in advance by programming logic ones. Subsequently, the write cycle is able to change particular cells to logic zero to define the final programming. This additional erase cycle, which is required to get stable signals, increases the programming time of the flash memory cells significantly. Due to the hardware-limited flash memory access time with single byte operation at a speed of 90 ns per byte, reading the whole 5.5 million CLB (see A) configuration bits of a Virtex-4 FX20 FPGA from the flash memory theoretically takes about 61 ms. Due to protocol and controller overhead this process finally takes about a second and therefore a full scrubbing cycle cannot perform faster.

4.1.2 Xilinx SelectMAP Configuration Protocol

The communication between the configuration scrubbing controller running in the Actel/Microsemi ProASIC3 FPGA and the SelectMAP configuration interface within the Xilinx Virtex-4 FX20 FPGA has to follow a specific protocol to ensure correct operation. A detailed description of all available commands as well as options and restrictions can be found in [459]. Therefore, this sections only gives a basic overview about the selected strategy. As depicted in figure 4.1, the signal communication sequence follows a basic scheme: First of all, the device is reset via PROGRAM_B. While waiting for this internal, asynchronous process to finish, configuration write mode is selected with RDWD_B. As soon as INIT_B signals general readiness, CS_B enables data sampling from the 8 bit DATA port on each triggered CCLK rising edge. The behavior of CCLK is manually controlled since the scrubbing controller has to wait for the memory controller to finish reading from the external flash memory chips. As soon as the last configuration word has been sent, the FPGA performs final setup routines that require CCLK cycles and communicates correct configuration by not touching the INIT_B CRC error indicator and finally rising the

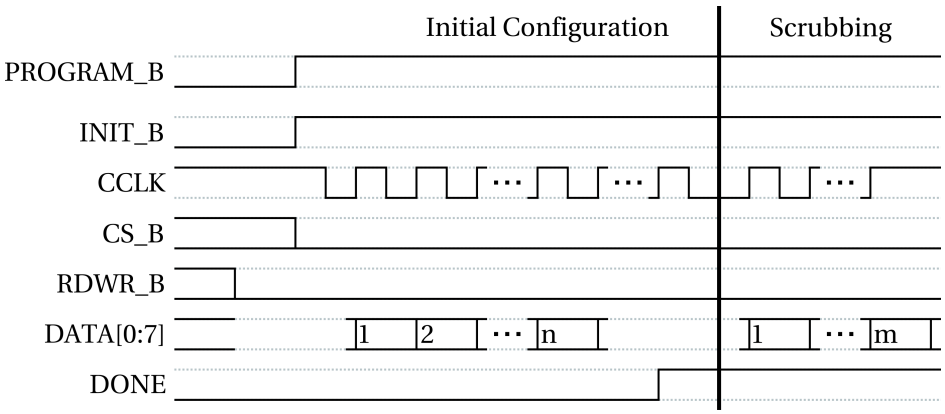


Figure 4.1: Signal diagram of FPGA initialization as well as subsequent static configuration scrubbing via the SelectMAP interface. The 8 bit configuration input DATA[0:7] is sampled non-continuous via a user controlled CCLK scrubbing clock that is triggered as soon as new data from the configuration flash memory is available. The diagram is partially derived from [459] (figures 2-16 and 2-18).

DONE signal. All subsequent configuration scrubbing cycles follow exactly the same pattern except of the initial device reset via PROGRAM_B/INIT_B as well as the final DONE assignments. Since the FPGA configuration can only be performed for full frames of $41 \cdot 32$ bit words, 164 scrubbing cycles are necessary before a frame register of the Virtex-4 FPGA is filled and its data is applied to the active device configuration. The total number of CLB configuration frames for the Virtex-4 FX20 FPGA used on the SysCore 2.0 board is 4200. To enable correct identification of beginning and end of a configuration sequence, the Xilinx-internal configuration controller furthermore requires specific synchronization and desynchronization words to be sent as well as information about the total number of 32 bit configuration words. For more information please refer to the next section 4.1.3.

4.1.3 Partial Bitfiles for Configuration Scrubbing

The whole Xilinx FPGA configuration data, composed of CLBs, BRAM, BRAM interconnect, and few other interface definitions are stored in a single programming bitfile. In consequence, the uncompressed size of this file is directly proportional to the number of configuration units. On the one hand, this approach requires a flexible way of data storage that satisfies all device models, on the other hand it needs to comply with the basic principle of serially transmitting data from the configuration interface to a designated internal device frame. Therefore, Xilinx has chosen to

store all configuration bits sequentially within the file, only providing global identifiers that specify frame address position and data length. The file is afterwards surrounded with general header information and a few controller commands following a simple protocol as explained for Virtex-4 FPGAs in [459, 516, 517]. This composition of commands and data can now be used to directly modify selected storage cells within the FPGA and to finally prepare a special bitfile that only partially reconfigures a device.

Preparation in this manner means to skip the leading device header until `DUMMYWORD` and `SYNCWORD` are detected. Every data input sent to the device before or between both commands is ignored by the configuration interface. In the subsequent command sequence, programming of the configuration option register is performed. This register contains for example the device's internal CRC bypass setting as well as clocking information and therefore has to remain in the file. Afterwards, the device ID register is filled with device family and size information. The subsequent command register `SWITCH` is necessary to adjust the CCLK input clock frequency. All following control register write commands as well as their masking definitions can be removed from the partial bitfile as they have already been set on initial configuration. This is easily performed by replacing them with non-operational NOP commands. NOPs are used regularly during configuration transfer to introduce timing control and guarantee correct internal command execution. The next FAR write command assigns the configuration's start frame with section, column, row and minor address. Manipulation of this FAR enables selective frame scrubbing capabilities. If just blind scrubbing is performed, all values can remain zero to indicate the device's first index. To gain knowledge about device frames and their exact storage position within the bitfile, an interpretation library as used in [519] can be consulted. Finally, the frame data register needs to be filled with the number of words that will be sent to the FPGA for configuration. In case of a full device blind scrubbing, containing only CLB frames, this number of CLB words can be determined with the Xilinx data2mem tool by running the following command: `data2mem -bt filename.bit -d`. The CLB data itself directly follows this command and therefore has to fit in length with the specified number of words.

In case of dynamic BRAM content is embedded within the original full configuration, it has to be removed when generating the partial bitfile to prevent interruption of the running design. The amount of BRAM data varies between different devices, follows the CLB configuration in the bitfile, and is terminated by a write command to the CRC register. The CRC command also needs to be removed to prevent inter-

ruption of the configuration process. While the full configuration bitfile contains device reset and high impedance commands to initialize the FPGA, this is not allowed in partial configuration mode, and therefore all following GRESTORE, DGHIGH/LFRM, AGHIGH, and START commands have to be removed. A terminating DESYNC word indicates the end of the configuration file towards the configuration controller. All following data words are ignored until the configuration process is restarted.

4.1.4 Automatic Partial Bitfile Generator

To simplify the process of partial bitfile generation as explained in section 4.1.3, a command line tool has been developed that automatically considers all mentioned guidelines while offering a verbose mode to indicate command explanations as well as all bitfile modifications in detail to the user. It takes a regular Xilinx-generated bitfile for input, automatically detects the FPGA-type from the general header, selects CLB and BRAM configuration sizes, and prepares the file for blind configuration scrubbing. In contrast to the Xilinx-provided tools, it operates without an NCD (see section 2.2.2) and generates a binary configuration file that can be used with any programming interface, including JTAG. This allows cheap and comfortable test operation setups based on conventional FPGA USB device-programmers in combination with COTS development boards that do not offer configuration scrubbing hardware support as provided by the SysCore architecture. The tool has furthermore been designed to support upcoming FPGA generations without the necessity of extensive code modification.

An exemplified output of this tool can be found in appendix B for the modification of a Xilinx Virtex-4 4VFX20FF672 FPGA. As the tool operates on a 32 bit word basis, all bitfile positions are given in 4-byte steps. Modified command and data words are indicated for convenience by a small asterisk and commands are extensively itemized with single bit interpretation. The subsequent CLB configuration data is not indicated, as it is left untouched. The same applies to dynamic BRAM data, which is cut from the output file to prevent overwriting of essential runtime data (see section 2.7.8). Finally, after issuing the DESYNC command, 16 TYPE-1 NOP words are added to flush the command pipeline. For a detailed overview of CLB and BRAM sizes for various Xilinx FPGAs that have been used within this tool, please refer to appendix A.

4.1.5 Prerequisites for Design Scrubbing

Summarizing static configuration scrubbing, multiple requirements in regard to hardware platform, FPGA, and configuration bitfiles have been shown, which go hand in hand to provide the most basic fault tolerance functionality. In addition to this, some major guidelines related to the system design itself have to be observed. First of all, the use of SLICEM or SLR16 primitives that store dynamic runtime data in distributed memory or shift registers has to be omitted. Explicit options for such automatic primitive extraction can furthermore impact the synthesis process and therefore have to be disabled. This includes shift register extraction, avoidable via the `-shreg_extract NO` switch as well as distributed memory extraction avoidable by using the `-ram_style Block` statement. All data contained in such primitives is continuously reset by the utilized blind scrubbing process or at least ignored when correctly masked but therefore susceptible to error accumulation. The only viable option to use such distributed memory resources without additional efforts is in read-only mode, protected by static scrubbing. If usage cannot be omitted, additional fault tolerance design is necessary (see section 2.7) to provide an adequate error mitigation. The second, urgent configuration step is to instruct the FPGA in assigning a termination signal as soon as a scrubbing cycle has successfully finished. This ready-signal is required by the scrubbing controller to initiate a new cycle as well as it acts as watchdog signal to indicate proper function of the internal configuration interface. Ideally, the SelectMAP interface's DONE port can handle this task. It needs to be enabled via the `-g DriveDone:yes` bitgen option, that avoids a permanent connection to an internal, static pull-up resistor. The third requirement is to keep the SelectMAP configuration interface itself functional and the configuration signals assigned exclusively without being used for other data transfers. To guarantee this condition, the `-g Security:None` and `-g Persist:Yes` bitgen options need to be used that enable reconfiguration via SelectMAP. If this condition is neglected, the configuration interface ports used by the scrubbing controller will be closed for several reasons, mostly design readback protection and pin availability. This does not apply when using the JTAG interface for configuration scrubbing. Finally, the last basic requirement for a successful scrubbing design considers the use of embedded memory primitives. As already mentioned, dynamic BRAM content cannot be protected by static configuration scrubbing and therefore has to undergo a different error mitigation strategy which makes use of hardware-specific features provided by the FPGA primitive, such as ECC, if available. This method is explained in detail within the next section 4.2.

4.2 Dynamic run-time Memory Scrubbing

As mentioned in the previous sections 2.7.7 and 2.7.8, BRAM and distributed memory usage in radiation environments requires special care. Static configuration scrubbing cannot protect these primitives, therefore it is mandatory to use a separate, fault-tolerant memory scrubbing unit that continuously refreshes all words of the whole memory array and corrects bit upsets. This specific controller circuit needs to be adapted to match the expected radiation environment's upset rate to be able to successfully retain all data without error accumulation between two complete checking cycles, while the implemented EDAC algorithm needs to include considerations about the device's CMOS feature size and therefore the probability of MBUs in closely related memory cells. In the worst case, an SECDED protected memory entity has to be fully doubled to be able to take correct decisions about its content in case of MBU when hardware storage cell interleaving of the physical memory is unavailable (see section 2.6).

As the availability of hardware support to implement EDAC features varies between different Xilinx FPGA series, two separate memory scrubbers for the fault-tolerant system have been utilized: The first one makes use of the hardware provided ECC circuits by Xilinx Virtex FPGA BRAMs (see section 4.2.1) and the second one from [520] according to [467, 154] provides custom EDAC for the cheaper Spartan FPGA series which does not offer SECDED ECC in hardware (see section 4.2.2). If required, this last circuit entity can be adapted to protect SLICEM primitives that have been chained to provide distributed memory. A similar 'CoreEDAC' for Actel/Microsemi FPGA's block memory can be found in [521] and as even commercial ASIC microcontrollers contain embedded memory, the same scrubbing functionality has furthermore been realized by [522] in a software-based approach that is able to improve even fixed-circuit devices and therefore reduce the probability of uncorrectable errors, data loss as well as reset cycles (see section 4.2.3). The experimental results for all implemented memory scrubbers can be found in section 5.4.1.

Another essential requirement for the memory scrubbing circuit is its ability to operate without significantly slowing down a running system. Ideally, it is operating unattended in the background via a dedicated memory port or it uses idle cycles to perform the error correction. In every case, the memory scrubber has to be protected by radiation-tolerant hardware or fault-tolerant circuit design, similar to the rest of the FPGA firmware when operated in radiation environments.

4.2.1 Xilinx Virtex FPGA

As Xilinx Virtex FPGA devices, starting from Virtex-4, offer a surplus of 4 extra bits for every single 32 bit embedded BRAM word which can be easily combined to a full 64 bit word-with, Hamming(72,64) protected memory with hardware encoding and decoding as explained in sections 2.3.4 and 2.5.7.5, they are well equipped for use in radiation environments. Unfortunately, the memory content is error-checked only on a readback request [467, 154] and all memory correction steps have to be performed manually to prevent error accumulation. Therefore, a simple version of the ECC-BRAM memory scrubber continuously reads back every single word, checks the hardware-provided error status register and, if necessary, writes back SBU-corrected data. In case of data integrity cannot be guaranteed anymore due to a detected MBU within a single storage word, an error indicator is permanently set to inform the overall system about the mismatch. The decision about the initiation of a subsequent system reset can then be made based on the criticality of the contained data. To bypass this issue, an improved version of the memory scrubber duplicates the whole ECC-BRAM block and continuously double-checks every single word while providing exactly the same interface as the simple version. This enables the correction of MBU in a single ECC-BRAM as soon as the corresponding status register indicates the fault. The first memory scrubber does not loose a significant amount of memory resources, only the surplus bits are required to store the ECC, but it cannot handle MBU in single words. This is fixed with the improved version, but at the cost of a 50% BRAM capacity overhead. The trade-off between data protection and resource consumption has to be chosen based on the upset rate of the designated radiation environment (see section 2.5).

Unfortunately, both memory scrubbers cannot make use of the synchronous, dual-port BRAM interface [147] to independently access the second PORT_B connection, as this is not available for the ECC-BRAM. To bypass this issue, it encapsulates the memory and implements a new access layer with custom interface that operates only when the memory is idle and not accessed by conventional operation to minimize any interference with the regular design.

4.2.2 Xilinx Spartan FPGA

In contrast to the Xilinx Virtex FPGAs, Spartan devices do not offer hardware circuits for BRAM primitive SECDED ECC encoding and decoding, although the required surplus storage bits are available and user-accessible. Therefore, this error-correction functionality has to be embedded manually in the conventional firmware

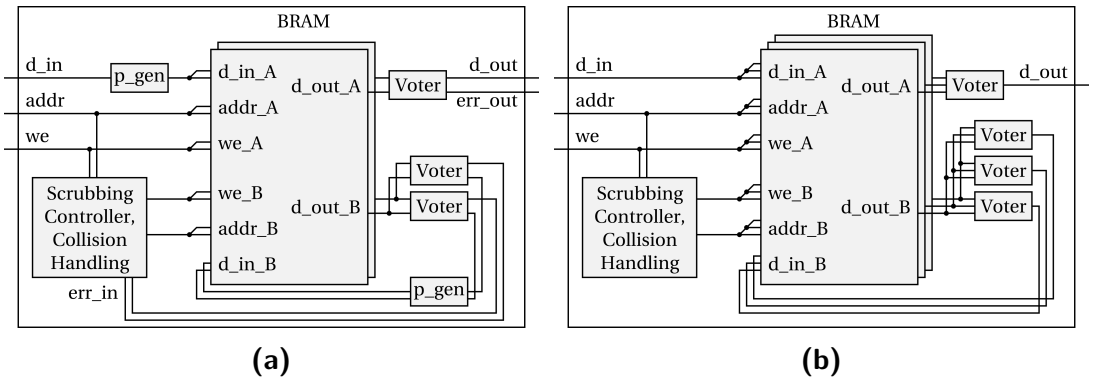


Figure 4.2: Simplified depiction of dynamic memory scrubbers for Xilinx FPGAs in DMR (left) and TMR (right) configuration. Both designs can be used in devices without ECC-BRAM support, such as the Xilinx Spartan series. While the DMR voter is based on a special combination of data and parity checker, the TMR voting follows all prerequisites given in 2.7.2 to eliminate the single point of failure in case of a corrupted voter within a critical feedback path. The encapsulated entity provides a custom interface to guarantee data input mirroring, data output voting as well as internal read/write conflict handling.

design while occupying valuable device resources. It furthermore has to be designed fault-tolerant, which stresses logic resource consumption in addition. In case of a full 64 bit Hamming encoder and decoder, this requirement should not be underestimated. To bypass this issue, BRAM can easily be used in a TMR configuration by mirroring every word write command and voting every word read request. Due to the availability of the synchronous, dual-port BRAM interface [147], the continuous memory scrubbing that prevents error accumulation within the memory itself can operate at maximum speed, fully transparent for the rest of the firmware on the second PORT_B connection. As all memory access requests between PORT_A and PORT_B have to be organized, conflict detection and handing needed to be added. All conventional memory operations are handled via PORT_A as usual. An encapsulating entity with custom interface only ensures that every write access is correctly distributed to all connected memory blocks as well as every read request is correctly voted before being issued. Unfortunately, the BRAM overhead consumption for a TMR approach is high and 66% of the overall memory capacity is lost. As a compromise between complex Hamming EDAC circuits and TMR replication, a DMR coupling of each two BRAMs with individual XOR parity has been implemented. Therefore, every 8 interleaved bits of a single 32 bit word are protected by a separate XOR parity bit. All parity bits are efficiently stored in the 4 bit surplus area of the BRAM by utilizing the provided 'Byte Write Enable' feature with a total byte

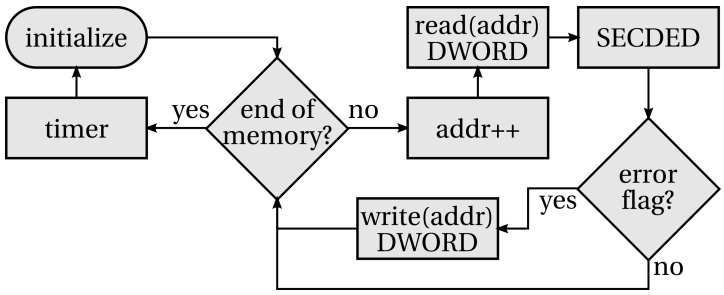


Figure 4.3: Flowchart of the TMS570 memory scrubber, depicting the continuous loop of 64 bit double word read, error check and flag analysis as well as the subsequent conditional write of data and ECC bits in the SRAM-based memory. A timer controls the repetition speed. Picture derived from [522].

size of 9 bits [302]. The parity generation is performed independently for both duplicated BRAMs during every write command so that data and parity in each two BRAMs are mirrored. A read request checks the parity of both BRAMs and returns the 8 bit data of the first correct one. An additional background memory scrubber continuously checks all storage bits and in case of an error, the erroneous data is replaced with the correct word from the second, mirrored BRAM. For this solution, the BRAM overhead capacity loss is only 50% in exchange for few device logic resources to implement the parity check, mostly LUTs. Both implementations of this memory scrubber, DMR and TMR, have been realized in [520] for $512 \cdot 36$ bit BRAM primitives in 'Read-First' configuration. The TMR implementation uses triplicated voters and triplicated flip-flop counters for the scrubbing feedback path to eliminate this single point of failure as explained in section 2.7.2. It furthermore provides all necessary statements to prevent undesired design optimization as mentioned in section 3.3. The DMR solution implements a special DMR voter that combines data comparison with parity check. Both solutions are depicted in figure 4.2. A comparison of their individual resource consumption and effectiveness can be found in section 5.4.1.

4.2.3 Texas Instruments TMS570 Microcontroller

Utilizing an FPGA to implement a conventional or fault-tolerant CPU might not be the best option anymore as soon as commercial ASICs with equivalent features but less power consumption and size hit the market at a lower price. A valuable competitor in the field of fault-tolerant micro controllers is the Texas Instruments

TMS570 [395, 394] that is approaching from the automotive industry which also pays special attention to data safety. It offers a 32 bit redundant dual-channel processor pipeline in lockstep architecture with static configuration matrix and dynamic on-chip memory as described detailed in section 2.6.4. While configuration scrubbing is unnecessary for such a fixed routing device, its embedded, SRAM-based dynamic storage memory still remains susceptible to ionization impact and therefore accumulates errors when exposed to critical radiation. Fortunately, this memory provides CPU-coupled Hamming(72,64) SECDED ECC hardware support that can be used to detect and correct errors. It is held in 8 additional bits provided for each two 32 bit words, similar to Xilinx BRAM (see section 2.5.7.5). Unfortunately, corrupted data within this memory does not repair automatically. To bypass this issue, a continuous dynamic memory refresh has to be carried out by periodically running an efficient assembler routine in software. This concept of background memory scrubbing running on the microcontroller has been implemented in [522]. Its basic data flow can be found in figure 4.3. The assembler routine consecutively reads every 64 bit double word from the SRAM-based memory, waits for the CPU-coupled ECC check to complete, analyzes the correction result register flag and in case of a successful repair subsequently writes back data and ECC bits from the CPU register to the corresponding memory address. This operation is repeated after a customizable time interval between 5 ms and one hour, adjustable even at runtime. If more than one error has been detected that cannot be repaired, the program enters a routine that handles the microprocessor reset. Due to the intelligent error flag interpretation, the overall number of memory write cycles is lower in comparison to a simple blind scrubbing approach, which refreshes every single memory cell, and therefore does not significantly slow down the overall memory speed, especially in ionizing radiation applications which are characterized by a low number of single bit errors. The scrubbing speed is furthermore enhanced by the internal memory layout of the TMS570 microcontroller, as a single 64 bit double word is always read via two coupled 32 bit memory block interfaces offering separate pipelines. This theoretically doubles the scrubbing speed in comparison to two regular 32 bit sequential memory accesses. To practically prove the correct behavior of the implemented assembler routine, a proton particle beam test has been performed. It is shown in section 5.4.1.

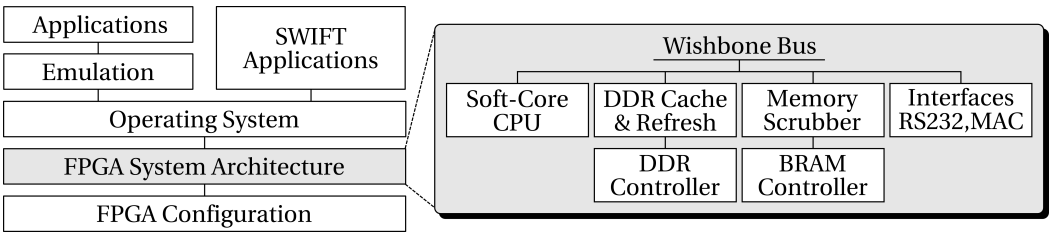


Figure 4.4: Schematic concept of a fault-tolerant microcontroller system that spans all layers of modern FPGA design. The basic configuration layer is protected by static scrubbing, the system architecture offers a set of intelligently designed fault-tolerant standard components to provide the optimal basis for a regular operating system that can furthermore be protected by SWIFT-compiled applications or regular applications that make use of an interpreter/emulator to introduce fault tolerance.

4.3 Fault-Tolerant System Design

When error mitigation on the most basic user-accessible FPGA configuration layer had been cleared by the successful setup of static device configuration scrubbing, the fundamental decision about a fault-tolerant FPGA system architecture design approach had to be taken. As designers know best about their critical control and data paths, manual design-hardening by analysis of all logic components' functionality, explained in section 2.7, promises an optimal error mitigation trade-off in comparison to automatic TMR approaches, enumerated in section 2.9. Based on this fact and, in addition, based on the necessity of modern embedded computer systems to offer a central microprocessor to efficiently operate standard interfaces such as Ethernet, the decision was made to manually design a set of components on the FPGA system architecture layer that are able to successfully operate a microcontroller in radiation environments. This schematic concept is shown in figure 4.4. Within this context, the following components have been designed: An MIPS R2000/R3000 compatible microprocessor in section 4.3.2 by [286], a dynamic memory scrubber as already introduced in section 4.2 by [520], as well as several interfaces in sections 4.3.3 by [523] and 4.3.4 by [524]. All of them are mutually connected by the central Wishbone bus protocol and designed by steadily focusing on the aim of avoiding XTMR where not urgently required. Coupling these components on the FPGA system architecture layer can massively foster a successful operation of SRAM-based FPGAs in radiation susceptible environments and furthermore forms a perfect basis for the overlying operating system and application layers, finally resulting in a modern, fault-tolerant, embedded FPGA microprocessor system.

4.3.1 Logical Design Decisions

Manually designing a fault-tolerant, MIPS-based computer system without the use of extensive XTMR required some preparatory thoughts, first of all about the basic implementation of the central bus system and the connected processor with sequential instruction handling. The limitation to sequential in-order execution of instructions simplified these thoughts.

To limit the amount of valuable logic resources occupied by the CPU for example, it can be sufficient to simply know about a processing error instead of immediately being able to correct it. The erroneous instruction can be repeated with ease when no memory or register content has been damaged as soon as the affected SRAM configuration cells that caused the error have been repaired by a subjacent scrubbing cycle. This concept implied the implementation of a DMR approach with doubled instead of tripled pipeline stages. But as processing errors can occur in all pipeline stages, starting from instruction fetch, instruction decode, operand fetch, instruction execute, up to memory load/store, a permanent background comparison of pipeline registers becomes necessary to prevent the CPU from data miscalculation, illegal address jumps, undefined program progression or even hang-up in an undefined state. As soon as this comparator detects a discrepancy, the whole processor pipeline is flushed and the program counter restarts from the address where the first error occurred and all data was still valid. This results in a loss of processing time for the last five instructions, but ensures that no erroneous data has been used for calculation, and even more important, that no faulty data has been written back to the memory block. Special attention has furthermore to be taken with the program counter as this central element of the microprocessor is shared between all pipeline instances and an upset immediately causes both instances to fail simultaneously without a chance of error detection. The challenge of implementing this concept of a fault-tolerant microprocessor has been taken by [286].

The selection of a suitable standard bus protocol in a fault-tolerant computer system mostly depends on its susceptibility to bit upsets, as it cannot be modified for custom needs without violating compatibility. A complicated bus with features like latency compensation and buffering is vulnerable by design as it requires a lot of additional logic resources, while a simple bus that just interconnects all components by simple routing and communicates via cable select might be the better solution. Additional properties that have to be considered are word-width, device maximum, size of address space or expandability. In the present case, the selected open Wishbone bus [525] for example follows a simple single master / multiple slaves wiring

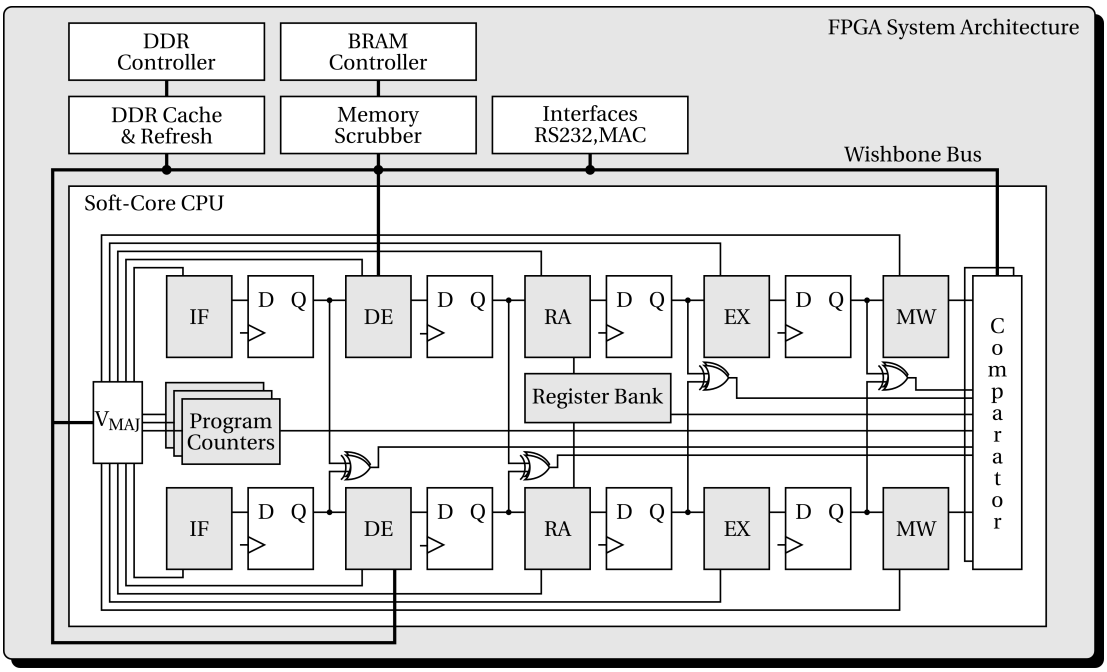


Figure 4.5: Depiction of the fault-tolerant MIPS R2000/R3000 compatible CPU with doubled pipeline stages according to [286]. Comparators between all operational stages ensure an immediate reaction on spontaneously occurring upsets. All program counters are tripled and voted to eliminate this single point of failure. A final comparator guarantees that no erroneous data is written to memory or external interface. (Remark: Component depiction not equivalent to area consumption.)

concept without overhead and it basically offers tag fields for both data and address lines that can be used to transfer additional user data without violating the official bus specification. These tag fields have therefore been used to transmit additionally generated Hamming ECC bits to enable error checking on the system bus.

One major limitation when providing fault-tolerant firmware for conventionally designed hardware is, that none of the external interfaces connected to the FPGA, such as Ethernet or RS232, is designed to use redundancy. This means, that all I/O pins are present only once instead of using a replicated approach as explained in section 2.5.7.4. In consequence, possible mitigation techniques for these interfaces are limited in their efficiency and the redundancy needs to be applied directly after the IOBs as in [520] and [524].

4.3.2 Central Processing Unit

The CPU of a computer system in a radiation susceptible environment has to ensure, that all instructions of a running program are executed in the designated, deterministic order and that no erroneously calculated data is committed to the storage memory or changes the behavior of the whole system. To ensure this behavior, a reliable error detection mechanism is required. Different approaches based on signal analysis [390], functional validation [392, 393] or others are available. If any unrecoverable discrepancy within the processor's calculation chain has been detected, the erroneous instruction and all of its following instructions have to be discarded and correctly recalculated since they may be based on wrong input information. This aim can be reached by simply flushing the whole pipeline and restarting it from the defective instruction, similar to a regular branch prediction failure. This method has been successfully used in many microprocessor architectures, such as DIVA [390] or Palisades [392, 393], and has also been used in the integrated work by [286].

Many modern FPGAs offer embedded, hard-wired processors beside of the regular functional gates on a single silicon die. This offers efficient accessibility and high speed data exchange. As by design, non of these embedded processors offers sufficient fault tolerance to sustain ionizing radiation effects, investigated by [308], and cannot be protected by redundancy in the same way as the remaining logic device, the decision was made to create a customized soft-core CPU.

This 32 bit MIPS R2000/R3000 compatible processor, depicted in figure 4.5 and developed by [286] for the current work, utilizes a DMR approach applied to the processor pipeline, accompanied by comparator units between each pipeline stage to immediately detect spontaneous calculation discrepancies. As soon as a difference between both concurrently calculating pipelines has been reported, the error signal is forwarded to the write-back stage and triggers the described repair mechanism. In the meanwhile, the SRAM configuration memory scrubber, running continuously in the background, repairs all routing information that may have caused the miscalculation and a subsequent recalculation may restore the system to normal operation. Just in case the same error occurs multiple times and cannot be repaired, a counter entity assumes an SEFI and triggers the whole microprocessor to reset.

The following sections now merely outline an overview of the fault-tolerant microprocessor's basic concepts which are necessary to support the approach of this thesis. The processor core itself has been implemented and tested by [286].

4.3.2.1 Processor Pipeline

As constituted in section 4.3.1, the decision was made to implement an MIPS R2000/R3000 compatible microprocessor with doubled pipeline stages, by following the spatial and temporal redundancy guidelines from sections 2.7.2 and 2.7.3. This includes the following replicated set of independently operating pipeline stages:

- Instruction Fetch (IF): In cooperation with the instruction execute stage, the program counter address is either incremented or, in case of a jump command, set to the previously calculated value. This address is afterwards fetched via the central Wishbone bus from the program memory block and the operation code is stored in both pipeline registers.
- Instruction Decode (DE): The previously fetched instruction is decoded and all corresponding select signals for the subsequent pipeline stages, for example the operand registers or the combination logic, are set.
- Operand Fetch (RA): All operand registers are fetched from the register bank as selected and provided to the next execution stage. These 32 registers with 32 bit word width occupy about 1000 flip-flops within the device.
- Instruction Execute (EX): This stage finally applies the instruction's logic operation to the fetched registers. In addition, possible result jump and branch conditions from the following instruction are analyzed and forwarded to the IF stage. While addition, subtraction and division are handled in device logic, multiplication and division are utilizing the embedded hardware DSP primitives (see section 2.5.7.6). As there have been enough DSP blocks on the FPGA, both pipeline instances offer individual cores.
- Memory Load/Store (MW): Storing the calculated result in the processor's destination register or sending it to a designated peripheral on the connected Wishbone bus finalizes the pipeline chain.

Both pipeline chains are fed with the same calculation data and share program counter as well as register bank, but internally drive their own control signals for independent progression with minimized timing delay. The error detection and therefore the comparison process between all relevant signals of both instances is handled separately for each of the five pipeline stages, but the error handling is initiated by a final comparator only with at most one clock cycle delay after detection. This enables immediate reaction on upsets and improved timing behavior as explained in the next section 4.3.2.2.

4.3.2.2 Error Detection and Handling

Error detection in a DMR approach is achieved easiest by comparison of both instances - the same applies to the microprocessor's mirrored pipeline stages. To efficiently indicate an error and handle the corresponding incorrect instruction, the microprocessor supports instruction flagging by additional error bits. The error detection itself is handled in a two-staged process. First of all, the synchronized outputs of each two pipeline stages with validly flagged instructions are compared against each other. This involves multiple signals and is depicted in figure 4.5 by the use of simplified XOR gates. It allows error detection at an early stage without the timing performance loss that would arise if all pipeline signals were compared simultaneously at a fixed time. But this introduces another vulnerability, as the resulting signal itself needs to be buffered for the final comparator entity and therefore can fail due to an SBU. To bypass this issue, each comparison has been implemented with DMR and the redundant results are directly fed into the global DMR comparator. In this second step, all of the individual comparison signals are brought together to be continuously analyzed. In conjunction with the combinatorial output-comparison of the last MW stages, any error detected in any of the five pipeline stages is handled with at most one clock cycle delay. This ensures that no incorrect data is written back to the register bank or to an external interface. Error detection within the comparator initiates error handling. It enforces the current MW-instruction to be marked invalid and flushed back to all other pipeline stages. With this program counter forwarded to the IF stage, the calculation can start again from the point of the previously faulty instruction with unmodified memory conditions, while in the meantime, a scrubbing cycle repairs all underlying SEU in the device's SRAM cells. All other pipeline stages with instructions marked invalid are idling and no comparison takes place. In the event that the error cannot be resolved, the last pipeline stage's program counter remains unchanged over multiple clock cycles, which can easily be detected by connecting an external watchdog that initiates a microprocessor reset.

4.3.2.3 Program Counter and Register Bank

As mentioned in section 4.3.1, the most critical part of the fault-tolerant microprocessor is the shared program counter, as it constitutes the information basis of the whole fault tolerance concept of error-detection with instruction reset and re-execution. If just a single bit is upset and pointing to a wrong program address, a logical instruction disorder cannot be detected. In consequence, the program counter of each pipeline stage had to be implemented with voted TMR to eliminate this is-

sue. As the logic resource consumption is minimal in comparison to the register bank, the triplicated implementation did not significantly increase the design size and therefore its cross-section. Unfortunately, the same strategy could not be followed for the processor's register bank due to its considerably high flip-flop utilization that would have entailed additional comparators as well as circuit overhead to mitigate error accumulation (see section 2.7.8). Even duplication of these flip-flops would not have increased fault tolerance, as it cannot be decided which of two registers contains valid data without implementing at least an additional parity checking circuit which in turn is accompanied by timing delay. The decision was therefore to use a single shared register bank and keep the FPGA-internal signal wires to this memory short.

4.3.2.4 System Bus

Wishbone has been selected as central system bus to organize the signal transport between all functional components within the FPGA. It utilizes a simple single master / multiple slave handshaking concept, whereas the microprocessor acts as master device to orchestrate all slave peripherals such as BRAM, DDR, RS232 or Ethernet. Beside of the 32 bit data, 30 bit address and few control bits, the Wishbone specification [525] contains additional tag fields, assigned to both data and address lines. In the current implementation, they have been defined to 7 bits each to take up a set of Hamming ECC. This enables error checking on the system bus and effectively prevents SBU from being simultaneously sampled in both pipelines stages. As a full Hamming encoder/decoder circuit would significantly increase the resource consumption and therefore the cross-section of a design (see section 2.7.4), its functionality has been limited to double bit error detection for both pipeline instances. The missing error correction feature can be compensated without circuit overhead by exploiting the bus protocol itself: Erroneous transmissions from the microprocessor to any peripheral can just be canceled with an error signal. This signaling immediately causes error handling within the processor in the same way a pipeline mismatch would do, finally resulting in recalculation and subsequent retransmission.

4.3.3 Serial Communication Interface

One of the basic communication interfaces between a conventional personal computer and a hardware development platform for decades now is the serial interface. Nevertheless, it becomes a sensible part of the whole system if critical watch-

dog or status information is transferred. Therefore, it has to follow the same fault tolerance design guides such as all other components in a running system. Multiple implementations of fault-tolerant serial interfaces have been developed for this thesis. All of them use spatial redundancy as explained in section 2.7.2, due to its simplicity and reliability.

One set of such interfaces has been developed as standalone versions implementing DMR and TMR [523]. Beside the regular communication ports such as data in, data out, write enable, acknowledge and others, an additional error port has been added. It indicates the error status of the serial component itself and can be collected individually across all components of the fault-tolerant system within the FPGA. One drawback to sustain compatibility across multiple development boards was the solely usage of the communications I/O pins, since the author has not seen a single board which follows the Xilinx recommendation to triplicate I/O ports and hard-wire them outside of the FPGA (see section 2.5.7.4). The same constraint was used for the internal communication ports to sustain compatibility across multiple FPGA designs. The DMR implementation internally compares data and reset signals via XOR and rises the error signal as soon as there is a logical difference when sampled at rising clock edge. The defective data can afterwards be internally requested again. In addition, the RS232 protocol's parity bit is sent after data transmission. Vice versa, defective data words are re-requested whenever this received parity bit is wrong. This mechanism is currently realized by sending the predefined word `00010010` but has to be supported by the communication tool running on the personal computer. If the data output vector gets corrupted while currently sending a word, the fault tolerance mechanism simply stops before sending the RS232 protocol stop bit. This is a simple process without overhead. It is also indicated by the module's error port. The TMR implementation utilizes three independent voters to feed all data input ports of the triplicated combinatorial logic instances. Their outputs are afterwards voted again, before they are assigned to the physical TxD output pin. This step can be skipped when using three independent output pins according to [157]. To get a basic impression of the physical design size requirements of the non-fault-tolerant, DMR as well as TMR modules within an FPGA, table 4.1 offers the number of occupied design primitives within a Xilinx Virtex-4 FPGA. The DMR solution itself includes a minor overhead due to the additionally utilized parity bit features and is therefore slightly bigger than a clean DMR implementation.

Another improved serial interface which utilizes the Wishbone bus and therefore perfectly integrates into the global fault-tolerant system was developed for the RISC

design primitive	non-fault-tolerant	DMR	TMR
LUTs	162 (100%)	421 (260%)	493 (304%)
registers	113 (100%)	252 (223%)	368 (326%)

Table 4.1: FPGA module size requirements of non-fault-tolerant, DMR and TMR serial interfaces according to [523]. The DMR solution includes additional parity features. Comparison based on Xilinx Virtex-4 implementations.

CPU described in section 4.3.2. It is also based on spatial redundancy and duplicates the I/O pins for the serial communication directly at the FPGA buffers to minimize this single point of failure as described in 2.5.7.4 and 2.7.2. The redundant signals are afterwards directly fed into the serial module to keep routing short by using design constraints which force location placement nearby the physical IOBs. In addition, the output pin is only written if both DMR instances contain identical values. Incoming data from the Wishbine bus' connected CPU, which needs to be sent via the serial TX pin, is stored in TMR registers, to be able to perform a retransmission if necessary. It is voted into the doubled serial module. In case of a transmission error, no stop-bit is sent, both modules are reset, the stored value is requested again from the TMR registers and transmission restarts. While this process is running, no incoming data is acknowledged to the Wishbone bus. Outgoing data which arrived from the serial RX pin is forwarded immediately to the CPU, since correctness cannot be checked. Signal simulation of the transmission reset by error injection is shown exemplary in figure 4.6.

4.3.4 Ethernet Controller

Another communication interface that is frequently used in conventional microcontrollers is Ethernet. Ethernet for FPGAs is usually a combined approach between Physical Layer (PHY) hardware, Media Access Control (MAC) firmware and further, protocol-related, software. While the on-board PHY-chip is limited to the basic functionality of OSI layer 1 and cannot be modified, the Ethernet MAC (EMAC) implements all features of OSI data link layer 2, including flow control, buffering, as well as CRC error checking, and is therefore an ideal choice for the implementation of fault tolerance for the Ethernet interface in an SRAM-FPGA-based system. To improve usability within the FPGA, OSI network layer 3 and transport layer 4 have been integrated for UDP packets and ICMP-based ping requests. Following the guidelines

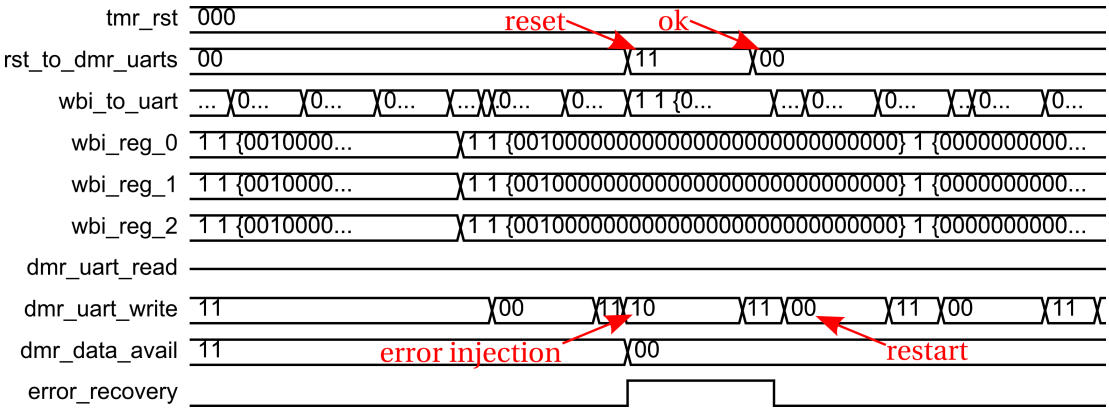


Figure 4.6: Signal Simulation of the transmission reset by error injection into the serial module's DMR instances. After the interface was reset, the transmission restarts with correct data from the background TMR registers.

of fault-tolerant system design by using spatial and temporal redundancy (see section 2.7), a DMR solution has been implemented by [524] that was initially based on [526].

Beside of the Wishbone bus integration, the non-redundant base-design has been modified to support BRAM primitives instead of FIFO elements for the storage of all ARP, IP frame, UDP packet and ICMP reply buffers. The advantage of BRAM in contrast to an FIFO is its ability to directly address data which becomes necessary as soon as a retransmission request is triggered by an SBU. The BRAM has been implemented for stand-alone use by applying information redundancy with DMR and parity (see section 2.7.4) as well as a custom control logic for the parity handling. But it can also be coupled with a central dynamic memory scrubber which orchestrates all BRAM data transfer and prevents error accumulation (see section 2.7.8). Other improvements include the duplication of the critical package handling core as well as the introduction of corresponding comparators that are monitoring data and control signals. All comparison results are subsequently forwarded to a central control unit, that can initiate packet re-transmission after the protocol-dependent *LineWait* cycles. This control unit also monitors the error recovery and communicates a possible accumulation of internal module errors via an external *frameError* flag. The spatial logic consumption of this DMR approach is about twice the size of the optimized module without fault tolerance, which is en enormous saving in comparison to XTMR. The functionality of this design has been proven via simula-

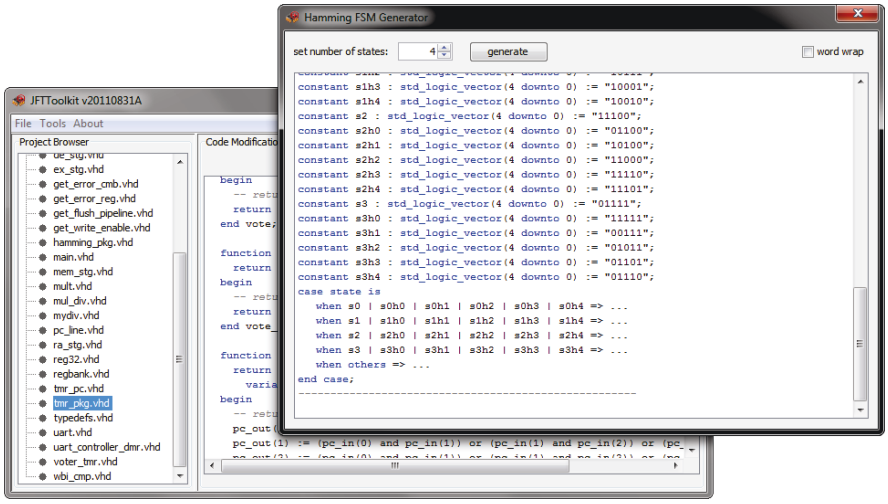


Figure 4.7: Screenshot of the JFTToolkit GUI. The Hamming FSM Generator module (see subsection 4.4.4) is shown in the foreground and provides the manual definition of a four state machine with a complete set of states and transitions that can be reached with a distance of d=1.

tion only, as there was not enough time to implement all protocol features for a fully functional Ethernet module.

4.4 JFTToolkit - The Java Fault Tolerance Toolkit

The manual design of fault-tolerant circuits on RTL is a very time consuming and complex process, associated with additional simulation and documentation effort as explained in section 3.3. In contrast, the use of automated tools, if available, requires netlist level application and simulation as seen from section 2.9. But in most commercial development scenarios, manual fault-tolerant system design as shown in the previous section 4.3 can be a show-stopper due to the significantly increased time to market (see section 2.2.3). Making use of already available, standardized, fault-tolerant components may defuse this situation, but this practice can not be applied in general, especially not to custom logic parts. Therefore, the approach of simplifying the manual design process by providing a tool that operates on RTL resulted in the development of the Java Fault Tolerance Toolkit (JFTToolkit) shown in figure 4.7 and explained in the following subsections.

The basic aim of the JFTToolkit is to analyze existing HDL, create a parsable object tree, identify basic HDL elements and advise the user in VHDL error mitigation options such as DMR, TMR and FSM improvement. It does not operate completely

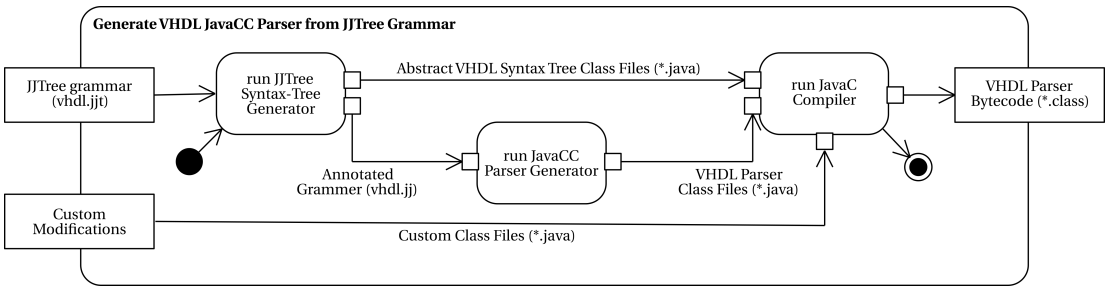


Figure 4.8: Activity diagram (UML 2.0) of the JavaCC™ VHDL parser generation. The JJTree grammer input file (vhdl.jjt) is used to generate an Abstract Syntax-Tree (AST) that can be used to traverse all node objects. Further processing of the annotated grammar file (vhdl.jj) with the JavaCC™ Parser-Generator creates a syntax parser which is able to check conventional VHDL files for errors.

without user-interaction, as it cannot decide whether a data path or design process is critical and therefore requires special handling. To remain independent from a specific description language, the underlying model of JFTToolkit has been designed to support multiple HDLs, but as the current implementation focuses on VHDL, all descriptions, examples as well as results are given for this single HDL only. The Verilog counterpart, as soon as implemented, can operate in a similar way and therefore should not differ in handling. Unfortunately, the JFTToolkit software did not reach a production state that allows stand-alone usage, and therefore requires additional development.

The following sections now give a basic overview about the currently implemented state, including the generation of a Java VHDL parser in subsection 4.4.1, an illustration of the proposed working algorithm in subsection 4.4.2, a declaration of the most important program interconnects via an UML 2.0 package and class diagram in subsection 4.4.3 as well as an integrated submodule for Hamming FSM generation in subsection 4.4.4. All hereby utilized methods are finally summarized in the general guidelines for fault-tolerant VHDL design of section 4.5.

4.4.1 VHDL, Chomsky Hierarchy and the Parser Package

VHDL is known to be a context-sensitive type-1 grammar in the Chomsky hierarchy, therefore it cannot be parsed without collecting contextual information from all included libraries and package files in advance. Parameterized function calls such as `sigA := sigB(10);` for example are used equivalent to local array indexes as well as equivalent to array indexes from parameter-less local or imported package

function calls themselves. Parsing this syntax requires contextual knowledge about all function signatures and return types in the local scope. This requires at least the analysis of all imported project files – recursively. The JFTToolkit operates on single files only, therefore this information is not collected and the functional range is reduced to scope and context that is available by the subset of the context-free type-2 grammar. As type-2 grammars can furthermore be represented by an Extended Backus–Naur Form (EBNF), building a parse tree for this subset becomes possible.

The VHDL grammer used for the JFTToolkit is contained in the VHDL-93/VHDL-AMS (IEEE 1076.1) parser package of [527]. It provides all required production rules and terminal symbols to build an Abstract Syntax-Tree (AST) for a given VHDL file. Embedding the grammar into a larger application required only minor adjustments regarding automated code generation as well as package namespace. To finally build the AST in Java, the grammar file has been prepared with the JavaCC™/JJTree v5.0 [528] application. This process is depicted in figure 4.8. Therefore, the grammar file (.jjt) is handed over to the JJTree tree bilder, a preprocessor to the final JavaCC™. JJTree generates the whole set of Java classes and interfaces used in the AST and inserts additional AST building commands as well as object references in the grammar file, resulting in a new annotated grammar file (.jj) that is used as input for the final JavaCC™ parser generator. JavaCC™ then generates the operational tokenizer/lexer and parser class files which operate on the JJTree classes and interfaces to assemble the bottom-up parse-tree from a given VHDL file. It therefore starts with the given input symbol stream and tries to find a set of production rules by stepping backwards until it reached the start symbol. Each nonterminal is hereby represented by the corresponding node class.

The whole process of JJTree and JavaCC™ construction has been automated in a custom make script to simplify the processing of upcoming releases. It generates the complete content of the `JFTToolkit.HDL.VHDL` parser package, including all `JFTToolkit.HDL.VHDL.Syntax` classes and interfaces which are referenced by the AST. In parallel, the development of a custom Java Model for VHDL has been started.

As mentioned initially, JFTToolkit models support for multiple HDLs. Therefore, each HDL in the `JFTToolkit.HDL` package is inherited from a `LanguageHandler` class and implements a `LanguageHandlerInterface` interface that forces all instantiated objects of the corresponding code class to offer common methods such as the return and assignment of a set of all fault tolerance options as well as code dumps with syntax highlighting. Finally, all HDL classes throw unified `LanguageHandlerException` exceptions to simplify the error handling. All files in a directory

selected by the user are handled in a central store, the `LanguageHandlerProject`. When initialized with a project directory on program start or via menu selection, this class is responsible for loading all selected files, identifying the file type by sending a request to all language handlers and organizing all requests from and to each file handler. By this organizational structure, switching between different project files without losing modified content becomes possible.

4.4.2 Basic Working Algorithm and VHDL Example

For the purpose of imparting general knowledge about the basic working principle of the proposed algorithm that applies fault tolerance to a conventional VHDL description, an easy to understand example shall be given. It incorporates the identification of fault tolerance options by the corresponding `VHDLHandler` class as well as their specific implementation concerning a user-selection of DMR on the one hand and TMR on the other hand. The example itself realizes a very basic VHDL design that triggers an external LED with a clock counter. All modifications are shown for simplicity side by side in tables 4.2 (Architecture Declarative Part) and 4.3 (Architecture Statement Part). They have been synthesized with Xilinx ISE® Project Navigator 14.7 (nt64) and the resulting resource consumption is given in table 4.4. Finally, all hereby applied modifications as well as additional, unused improvements are summarized as guidelines in section 4.5. For simplification purpose, the given example uses the term 'signal' equivalent for all type declarations. The algorithm includes the following sequential steps:

- Instantiate a `CLOCK-IGNORE` list that contains all identified clock signals.
- Instantiate a `SIGNAL-MAPPING` list that contains all identified data signals and their user-selected redundancy option (None/DMR/TMR).
- Identify all clock signals and add them to the `CLOCK-IGNORE` list. Clocks are routed in separate timing nets and handled as explained in section 2.5.7.3. This identification includes:
  - Input and output signals assigned to instantiated Xilinx DCM LogiCORE component ports such as CLK0, CLK90, CLK180, CLK270, CLKDV, CLK2X, and others. In the given example the signals `clk250_p`, `clk250_n` and `clk100` are identified.
  - Entity Declaration Header's Ports Clause's Signal Declarations where the identifier starts or ends with `clk`.
- Handle signal declarations in the architecture declarative part. Therefore, present a replication option (none/DMR/TMR) for each local signal declara-

```
entity nonFT is port (
 clk250_p: in std_logic;
 clk250_n: in std_logic;
 LED_RED: out std_logic := '1'
 );
end nonFT;
architecture Bhv of nonFT is
 component clkgen250to100 port (
  CLKIN_N_IN: IN std_logic;
  CLKIN_P_IN: IN std_logic;
  CLKDV_OUT: OUT std_logic;
  CLKIN_IBUFGDS_OUT: OUT
  std_logic;
  CLK0_OUT: OUT std_logic;
  LOCKED_OUT: OUT std_logic );
 end component;
 component FD is port (
  D: in std_logic;
  C: in std_logic;
  Q: out std_logic );
 end component FD;
 attribute IOB: string;
 attribute IOB of LED_RED:
 signal is "true";
 signal clk100: std_logic;
 signal trigger: std_logic;


 signal ctr: std_logic_vector(25
  downto 0) := (others => '0');


 signal ff_D_in: std_logic :=
 '1';


 signal ff_Q_out: std_logic;
```

```
entity DMR is port (
 clk250_p: in std_logic;
 clk250_n: in std_logic;
 LED_RED: out std_logic := '1';
 error: out std_logic := '0');
end DMR;
architecture Bhv of DMR is
 component clkgen250to100 port (
  CLKIN_N_IN: IN std_logic;
  CLKIN_P_IN: IN std_logic;
  CLKDV_OUT: OUT std_logic;
  CLKIN_IBUFGDS_OUT: OUT
  std_logic;
  CLK0_OUT: OUT std_logic;
  LOCKED_OUT: OUT std_logic );
 end component;
 component FD is port (
  D: in std_logic;
  C: in std_logic;
  Q: out std_logic );
 end component FD;
 attribute IOB: string;
 attribute IOB of LED_RED:
 signal is "true";
 signal clk100: std_logic;
 signal trigger: std_logic;
 signal trigger_dmr[0|1]:
 std_logic;

 signal ctr: std_logic_vector(25
  downto 0) := (others => '0');
 signal ctr_dmr[0|1]:
 std_logic_vector(25 downto 0)
 := (others => '0');


 signal ff_D_in: std_logic :=
 '1';
 signal ff_D_in_dmr[0|1]:
 std_logic := '1';


 signal ff_Q_out: std_logic;
 signal ff_Q_out_dmr[0|1]:
 std_logic;


 attribute keep: string;
 attribute keep of trigger_dmr
 [0|1]: signal is "true";
 attribute keep of ctr_dmr[0|1]:
  signal is "true";
 attribute keep of ff_D_in_dmr
 [0|1]: signal is "true";
 attribute keep of ff_Q_out_dmr
 [0|1]: signal is "true";
 attribute
 equivalent_register_removal:
 string;
 attribute
 equivalent_register_removal of
 trigger_dmr[0|1]: signal is "no
 ";
 attribute
 equivalent_register_removal of
 ctr_dmr[0|1]: signal is "no";
 attribute
 equivalent_register_removal of
 ff_D_in_dmr[0|1]: signal is "no
 ";
 attribute
 equivalent_register_removal of
 ff_Q_out_dmr[0|1]: signal is "
 no";
```

```
entity TMR is port (
 clk250_p: in std_logic;
 clk250_n: in std_logic;
 LED_RED: out std_logic := '1'
 );
end TMR;
architecture Bhv of TMR is
 component clkgen250to100 port (
  CLKIN_N_IN: IN std_logic;
  CLKIN_P_IN: IN std_logic;
  CLKDV_OUT: OUT std_logic;
  CLKIN_IBUFGDS_OUT: OUT
  std_logic;
  CLK0_OUT: OUT std_logic;
  LOCKED_OUT: OUT std_logic );
 end component;
 component FD is port (
  D: in std_logic;
  C: in std_logic;
  Q: out std_logic );
 end component FD;
 attribute IOB: string;
 attribute IOB of LED_RED:
 signal is "true";
 signal clk100: std_logic;
 signal trigger: std_logic;
 signal trigger_tmr[0|1|2]:
 std_logic;
 signal trigger_vote: std_logic;
 signal ctr: std_logic_vector(25
  downto 0) := (others => '0');
 signal ctr_tmr[0|1|2]:
 std_logic_vector(25 downto 0)
 := (others => '0');
 signal ctr_vote:
 std_logic_vector(25 downto 0)
 := (others => '0');
 signal ff_D_in: std_logic :=
 '1';
 signal ff_D_in_tmr[0|1|2]:
 std_logic := '1';
 signal ff_D_in_vote: std_logic
 := '1';
 signal ff_Q_out: std_logic;
 signal ff_Q_out_tmr[0|1|2]:
 std_logic;
 signal ff_Q_out_vote: std_logic
 ;
 attribute keep: string;
 attribute keep of trigger_tmr
 [0|1|2]: signal is "true";
 attribute keep of ctr_tmr
 [0|1|2]: signal is "true";
 attribute keep of ff_D_in_tmr
 [0|1|2]: signal is "true";
 attribute keep of ff_Q_out_tmr
 [0|1|2]: signal is "true";
 attribute
 equivalent_register_removal:
 string;
 attribute
 equivalent_register_removal of
 trigger_tmr[0|1|2]: signal is "
 no";
 attribute
 equivalent_register_removal of
 ctr_tmr[0|1|2]: signal is "no";
 attribute
 equivalent_register_removal of
 ff_D_in_tmr[0|1|2]: signal is "
 no";
 attribute
 equivalent_register_removal of
 ff_Q_out_tmr[0|1|2]: signal is
 "no";
```

Table 4.2: VHDL code comparison of a non-fault-tolerant (left), DMR (middle) and TMR (right) design (Architecture Declarative Part).

```
begin --entity nonFT
 clkgen250to100 port map (
  CLKIN_N_IN => clk250_n,
  CLKIN_P_IN => clk250_p,
  CLKDV_OUT => clk100,
  CLKIN_IBUFGDS_OUT => open,
  CLK0_OUT => open,
  LOCKED_OUT => open );
 Inst_D_FlipFlop: FD port map (
  D => ff_D_in,
  C => trigger,
  Q => ff_Q_out );
 LED_RED <= ff_Q_out;
 trigger <= ctr(25);

 counter: process (clk100) begin
  if rising_edge(clk100) then
  ctr <= ctr + 1; end if;
 end process;

 flicker: process (trigger)
 begin
  if rising_edge(trigger) then
  ff_D_in <= not ff_D_in; end if
  ;
 end process;

end Bhv;
```

```
begin --entity DMR
 clkgen250to100 port map (
  CLKIN_N_IN => clk250_n,
  CLKIN_P_IN => clk250_p,
  CLKDV_OUT => clk100,
  CLKIN_IBUFGDS_OUT => open,
  CLK0_OUT => open,
  LOCKED_OUT => open );
 Inst_D_FlipFlop: FD port map (
  D => ff_D_in_dmr0,
  C => trigger_dmr0,
  Q => ff_Q_out );
 LED_RED <= ff_Q_out_dmr0;
 trigger <= ctr_dmr0(25);
 trigger_dmr[0|1] <= trigger;
 ff_Q_out_dmr[0|1] <= ff_Q_out;

 error <= '0' when (trigger_dmr0
 =trigger_dmr1) and (ctr_dmr0=
 ctr_dmr1) and (ff_D_in_dmr0=
 ff_D_in_dmr1) and (
 ff_Q_out_dmr0=ff_Q_out_dmr1)
 else '1';

 counter_dmr0: process (clk100)
 begin
  if rising_edge(clk100) then
  ctr_dmr0 <= ctr_dmr0 + 1; end
  if;
 end process;
 counter_dmr1: process (clk100)
 begin
  if rising_edge(clk100) then
  ctr_dmr1 <= ctr_dmr1 + 1; end
  if;
 end process;

 flicker_dmr0: process (
 trigger_dmr0) begin
  if rising_edge(trigger_dmr0)
  then ff_D_in_dmr0 <= not
  ff_D_in_dmr0; end if;
 end process;
 flicker_dmr1: process (
 trigger_dmr1) begin
  if rising_edge(trigger_dmr1)
  then ff_D_in_dmr1 <= not
  ff_D_in_dmr1; end if;
 end process;

end Bhv;
```

```
begin --entity TMR
 clkgen250to100 port map (
  CLKIN_N_IN => clk250_n,
  CLKIN_P_IN => clk250_p,
  CLKDV_OUT => clk100,
  CLKIN_IBUFGDS_OUT => open,
  CLK0_OUT => open,
  LOCKED_OUT => open );
 Inst_D_FlipFlop: FD port map (
  D => ff_D_in_vote,
  C => trigger_vote,
  Q => ff_Q_out );
 LED_RED <= ff_Q_out_vote;
 trigger <= ctr_vote(25);
 trigger_tmr[0|1|2] <= trigger;
 ff_Q_out_tmr[0|1|2]<= ff_Q_out;
 trigger_vote <= (trigger_tmr0
 and trigger_tmr1) or (
 trigger_tmr0 and trigger_tmr2)
 or (trigger_tmr1 and
 trigger_tmr2);
 ctr_vote <= (ctr_tmr0 and
 ctr_tmr1) or (ctr_tmr0 and
 ctr_tmr2) or (ctr_tmr1 and
 ctr_tmr2);
 ff_D_in_vote <= (ff_D_in_tmr0
 and ff_D_in_tmr1) or (
 ff_D_in_tmr0 and ff_D_in_tmr2)
 or (ff_D_in_tmr1 and
 ff_D_in_tmr2);
 ff_Q_out_vote <= (ff_Q_out_tmr0
  and ff_Q_out_tmr1) or (
 ff_Q_out_tmr0 and ff_Q_out_tmr2
 ) or (ff_Q_out_tmr1 and
 ff_Q_out_tmr2);
 counter_tmr0: process (clk100)
 begin
  if rising_edge(clk100) then
  ctr_tmr0 <= ctr_vote + 1; end
  if;
 end process;
 counter_tmr1: process (clk100)
 begin
  if rising_edge(clk100) then
  ctr_tmr1 <= ctr_vote + 1; end
  if;
 end process;
 counter_tmr2: process (clk100)
 begin
  if rising_edge(clk100) then
  ctr_tmr2 <= ctr_vote + 1; end
  if;
 end process;
 flicker_tmr0: process (
 trigger_tmr0) begin
  if rising_edge(trigger_tmr0)
  then ff_D_in_tmr0 <= not
  ff_D_in_vote; end if;
 end process;
 flicker_tmr1: process (
 trigger_tmr1) begin
  if rising_edge(trigger_tmr1)
  then ff_D_in_tmr1 <= not
  ff_D_in_vote; end if;
 end process;
 flicker_tmr2: process (
 trigger_tmr2) begin
  if rising_edge(trigger_tmr2)
  then ff_D_in_tmr2 <= not
  ff_D_in_vote; end if;
 end process;
end Bhv;
```

Table 4.3: VHDL code comparison of a non-fault-tolerant (left), DMR (middle) and TMR (right) design (Architecture Statement Part).

Logic Resource	Original Design	DMR Design	TMR Design
Slice Flip Flops	34 (100%)	61 (180%)	88 (259%)
4-Input LUTs	12 (100%)	27 (225%)	40 (333%)
Occupied Slices	30 (100%)	51 (170%)	59 (197%)

Table 4.4: Comparison of the resource consumption of a simple VHDL design, implemented in three versions (see section 4.4.2): completely without fault tolerance, with DMR and with TMR. All percentage values are given in relation to the original design without fault tolerance. The TMR implementation is not equivalent to XTMR (see section 2.7.2), as it does not separate the triplicated instances and clock nets, and therefore requires less resources.

tion (except signals already on CLOCK-IGNORE) to the user. If a DMR/TMR option has been selected, add the corresponding signal and option to the `SIGNAL-MAPPING` list. Furthermore, for each TMR selection:

- Add redundancy and voter signals to the architecture declarative part:

```
signal trigger_tmr[0|1|2]   : std_logic;
signal trigger_vote         : std_logic;
signal ctr_tmr[0|1|2]       : std_logic_vector(25 downto
   0) := (others => '0');
signal ctr_vote             : std_logic_vector(25 downto
   0) := (others => '0');
signal ff_D_in_tmr[0|1|2]   : std_logic := '1';
signal ff_D_in_vote         : std_logic := '1';
signal ff_Q_out_tmr[0|1|2]  : std_logic;
signal ff_Q_out_vote        : std_logic;
```

- Add a signal splitting statement to the architecture statement part's concurrent statements if it is not a target signal in a sequential statement (process):

```
trigger_tmr[0|1|2] <= trigger;
ff_Q_out_tmr[0|1|2] <= ff_Q_out;
```

- Add a voter signal to architecture statement part's concurrent statements

```
trigger_vote <= (trigger_tmr0 and trigger_tmr1) or (
   trigger_tmr0 and trigger_tmr2) or (trigger_tmr1 and
   trigger_tmr2);
ctr_vote <= (ctr_tmr0 and ctr_tmr1) or (ctr_tmr0 and
   ctr_tmr2) or (ctr_tmr1 and ctr_tmr2);
```

```
ff_D_in_vote <= (ff_D_in_tmr0 and ff_D_in_tmr1) or (
   ff_D_in_tmr0 and ff_D_in_tmr2) or (ff_D_in_tmr1 and
   ff_D_in_tmr2);
ff_Q_out_vote <= (ff_Q_out_tmr0 and ff_Q_out_tmr1) or (
   ff_Q_out_tmr0 and ff_Q_out_tmr2) or (ff_Q_out_tmr1
   and ff_Q_out_tmr2);
```

and for each DMR selection:

- Add redundancy signals to the architecture declarative part:

```
signal trigger_dmr[0|1]   : std_logic;
signal ctr_dmr[0|1]       : std_logic_vector(25 downto 0)
    := (others => '0');
signal ff_D_in_dmr[0|1]   : std_logic := '1';
signal ff_Q_out_dmr[0|1]  : std_logic;
```

- Add a signal splitting statement to the architecture statement part's concurrent statements if it is not a target signal in a sequential statement (process):

```
trigger_dmr[0|1] <= trigger;
ff_Q_out_dmr[0|1] <= ff_Q_out;
```

- Add a signal checker to the architecture statement part's concurrent statements. This error checker collects all redundant signals and returns the comparison result as outgoing external.

```
error <= '0' when (trigger_dmr0=trigger_dmr1) and (
   ctr_dmr0=ctr_dmr1) and (ff_D_in_dmr0=ff_D_in_dmr1)
   and (ff_Q_out_dmr0=ff_Q_out_dmr1) else '1';
```

- Add Xilinx-specific `equivalent_register_removal` and `keep` attributes for each redundant signal in the `SIGNAL-MAPPING` list to prevent synthesis signal optimization. For a TMR selection in the given example, this results in:

```
attribute keep: string;
attribute keep of trigger_tmr[0|1|2], ctr_tmr[0|1|2],
   ff_D_in_tmr[0|1|2], ff_Q_out_tmr[0|1|2]: signal is "true
   ";
attribute equivalent_register_removal : string;
attribute equivalent_register_removal of trigger_tmr[0|1|2],
   ctr_tmr[0|1|2], ff_D_in_tmr[0|1|2], ff_Q_out_tmr[0|1|2]:
   signal is "no";
```

while for a DMR selection, the added statements are:

```
attribute keep: string;
```

```
attribute keep of trigger_dmr[0|1], ctr_dmr[0|1], ff_D_in_dmr
   [0|1], ff_Q_out_dmr[0|1]: signal is "true";
attribute equivalent_register_removal : string;
attribute equivalent_register_removal of trigger_dmr[0|1],
   ctr_dmr[0|1], ff_D_in_dmr[0|1], ff_Q_out_dmr[0|1]: signal
    is "no";
```

- Handle concurrent statements in the architecture statement part (except of all previously added signal redundancy statements). Target signals are hereby ignored as they are already covered by the concurrent signal redundancy. All other signals from the SIGNAL-MAPPING list, acting as sources, have to be replaced with the corresponding voter (TMR) or one of the duplicated signals (DMR). In the given example, this corresponds to:

```
LED_RED <= ff_Q_out_vote;   |  LED_RED <= ff_Q_out_dmr0;
trigger <= ctr_vote(25);    |  trigger <= ctr_dmr0(25);
```

- Handle sequential statements (processes) in the architecture statement part. Therefore, present a replication option (none/DMR/TMR) for each process to the user:
  - If the None-Option has been selected, adjust all signals in the process that are listed in SIGNAL-MAPPING with a redundancy option. In case of a TMR target signal, triplicate the signal assignment while using the corresponding redundant signal names, such as `sig_tmr0 <- xyz`, `sig_tmr1 <- xyz`, and `sig_tmr2 <- xyz`. TMR source signals are replaced with the previously introduced voter signal, such as `sig <- xyz_vote` In case of a DMR target signal, duplicate the signal assignment while using the corresponding redundant signal names, such as `sig_dmr0 <- xyz` and `sig_dmr1 <- xyz`. DMR source signals are replaced with a redundant signal name, such as `sig <- xyz_dmr0`.
  - If the DMR-Option has been selected, duplicate the process and label the new instances for clarity with `proc_dmr[0|1]` (replace the original process). Then replace all source/target signals that can be found in the SIGNAL-MAPPING list with their corresponding redundancy instance. In the DMR example, this results in:

    ```
    ctr_dmr[0|1] <= ctr_dmr[0|1] + 1;
    ff_D_in_dmr[0|1] <= not ff_D_in_dmr[0|1];
    ```

  - If the TMR-Option has been selected, triplicate the process and label the new instances for clarity with `proc_tmr[0|1|2]` (replace the original

process). Then replace all target signals that can be found in the SIGNAL-MAPPING list with their corresponding redundancy instance. Replace all source signals with their previously introduced voter signals to improve the fault tolerance, except if they are used in the sensitivity list or in combination with edge detection. In the given TMR example, this results in:

```
ctr_tmr[0|1|2] <= ctr_vote + 1;
ff_D_in_tmr[0|1|2] <= not ff_D_in_vote;
process (trigger_tmr[0|1|2])
rising_edge(trigger_tmr[0|1|2])
```

- Handle component instantiation by modifying all assigned signals that are included in the SIGNAL-MAPPING list. For component ports declared as outputs, this is handled via the previously introduced signal error checker and replication statements, but for all component ports with input declaration, the signal names have to be adjusted. The given example therefore matches the signal names to the user-selected redundancy options with TMR voters or an instance of a DMR signal:

```
D => ff_D_in_vote,   |  D => ff_D_in_dmr0,
C => trigger_vote,   |  C => trigger_dmr0,
```

- As soon as one of the SIGNAL-MAPPING signals uses DMR, add a new error output signal declaration in the entity declaration header's port clause list. In the DMR example this is reflected by the lines:

```
signal error : std_logic := '0';
port ( error : out std_logic := '0' );
```

- If an FSM process has been detected by identifying the corresponding state type declarations or signal names that match state, count the number of states and replace this process with FSM generator output (see subsection 4.4.4), modify the FSM signals and add the FSM constants.

As most of the above mentioned options are suitable to be selected either for DMR or TMR error mitigation methods, depending on the specific requirements and therefore user-selection, the JFTToolkit software has to detect and inform about all illegally chosen combinations. Such conflicts can arise for example when not selecting signals for replication but using them in redundant processes and therefore would cause illegal multiple source errors in VHDL.

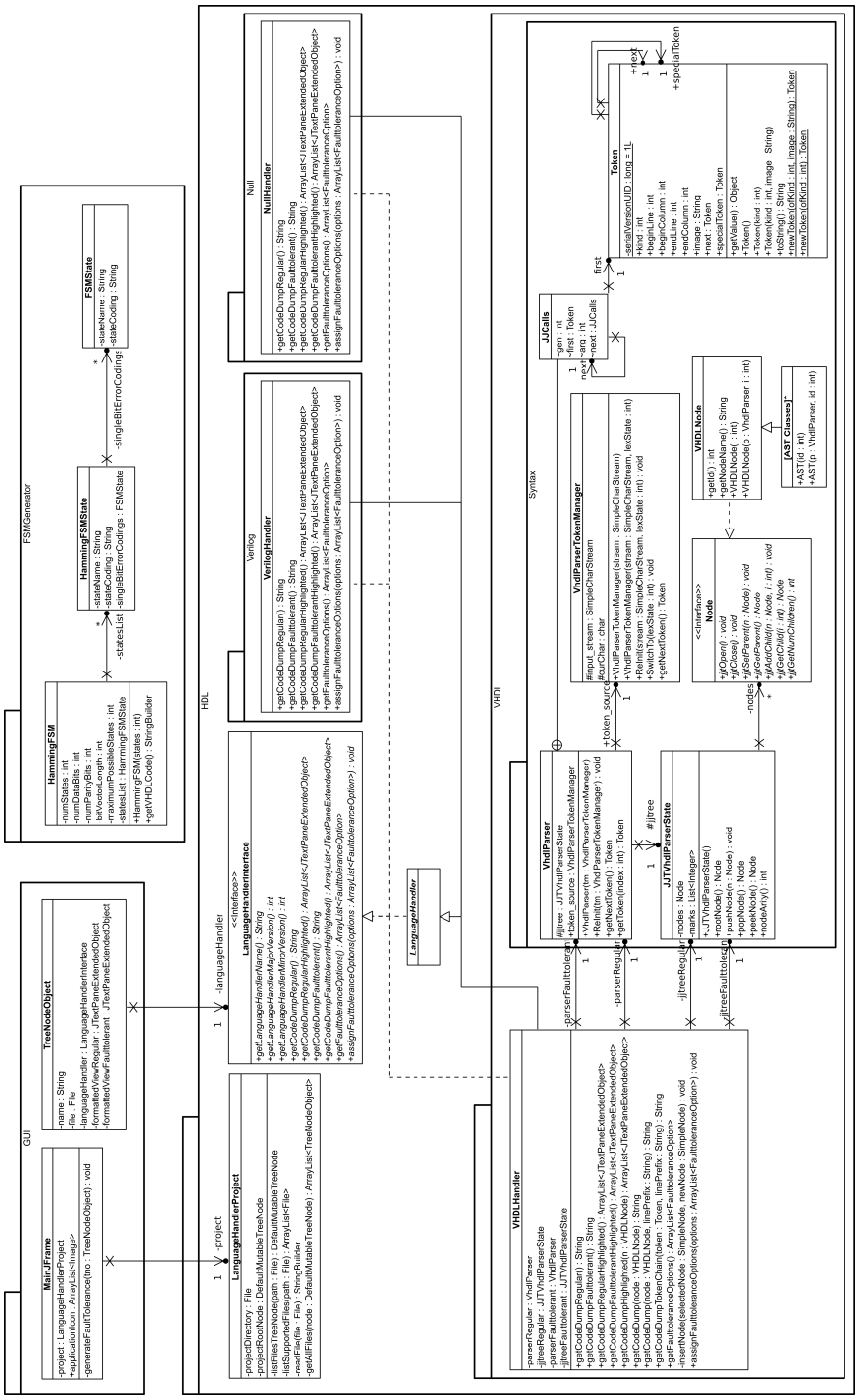


Figure 4.9: UML package diagram of the JFTToolkit software, containing the most important classes and interfaces as well as their relationship to each other. Attributes and methods have been reduced to improve readability. All AST classes have been combined into a single one.

4.4.3 UML Package and Class Diagram

The basic operation of the JFTToolkit is depicted in the UML 2.0 package diagram in figure 4.9. To improve the readability while keeping a summarizing character of the packages and classes, only the most important attributes and methods are shown. Running the Java application jar file's `MainJFrame` class instantiates all necessary objects to display the GUI and requests the user to specify a project directory from where all contained files are read into a `LanguageHandlerProject`. Based on these files, the project object afterwards generates a list of `TreeNodeObject` entries for the GUI, whereas each one references a custom `LanguageHandler` that provides language-related options. The correct `LanguageHandler` for a specific file is determined automatically while handing over its content to the constructor method of all available handlers. Those which did not succeed in parsing the file throw a specific `LanguageHandlerException` to indicate the failure. Finally, if not a single handler returned a successful parse result, the `NullHandler` takes care of the file, which may generate a simple text viewer in the final version. By implementing the `LanguageHandlerInterface`, every `LanguageHandler` is forced to offer individual but unified methods for:

- Code analysis
- Identification of signals and processes which are suitable for the application of fault tolerance techniques
- Application of fault tolerance techniques to a user-selected subset of the HDL code
- Generation of before-and-after output in a conventional as well as code-highlighted mode

These unified interfaces ensure language-independent processing in separate HDL handlers, such as the `VerilogHandler` or `VHDLHandler`, even in a single project directory. Each handler can furthermore be easily extended with custom functionality without disturbing the rest of the system. This may include methods for code addition, code modification or code removal as well as several other commands related to the corresponding language.

As the selection of a suitable parser varies with the HDL, it needs to be provided in conjunction with the respective language package. In case of VHDL, its basic functionality as well as the process of parser generation are described in section 4.4.1. Initializing the `VHDLHandler` class with VHDL content afterwards launches the `VhdlParser` from the `JFTToolkit.HDL.VHDL.Syntax` package, which, in the first instance, launches the `VhdlParserTokenManager` tokenizer/lexer to split the

whole input stream into separate Token objects, specified in the implemented VhdlParserConstants interface. This includes for example VHDL basic identifiers, bit values, string literals, basic operations, line breaks, comments, the semicolon, as well as language constants such as begin, generic, or downto. The type of token is stored in the kind attribute and its value in image while the position in the whole VHDL input stream is referenced by accompanied beginLine, endLine, beginColumn, and endColumn attributes. Finally, all Token objects are concatenated in a linear object chain via the next class attribute. The only exception is made for comments, which are stored in the specialToken attribute of the subsequent token and need to be referenced from there.

On top of this token-chain, the VhdlParser makes use of its implemented grammar production rules to combine all tokens via a bottom-up approach into a valid parse tree of non-terminal AST Classes. These AST Classes are inherited from a custom VHDLNode class and implement the Node interface to enable usage from the VhdlParser. Additional firstToken and lastToken attributes ensure access to the corresponding Token objects of this specific Node sub-tree. As long as a non-terminal produces a terminal symbol, firstToken and lastToken equal each other. If a production creates other non-terminals, the parent VHDLNode is assigned an array of children objects. As soon as the bottom-up production reaches the start symbol, indicating valid VHDL input, the root note of the parse tree becomes available via the VHDLHandler.jjtreeRegular AST object. It represents the initial content of a selected VHDL file and can be used to modify the code by introducing fault tolerance statements, which results in the final VHDLHandler.jjTreeFaulttolerant parse tree being presented to the user at the end of the entire code improvement process.

4.4.4 The Hamming FSM Generator

An integrated submodule of the JFTToolkit is provided in the Hamming FSMGenerator package. According to a given number of required FSM states, it generates a *complete* set of predefined signals and transitions that form a Hamming-secured Moore machine without the necessity of a Hamming decoder itself. The term *complete* in this specific context means that every basic state, manually defined with Hamming d=3 to all basic neighbor states, is accompanied by a set of auxiliary d=1 Hamming states, reachable only by SEEs. In addition, a *complete* set of transitions is given, which reference the basic states from the auxiliary states. More details as well

as advantages and disadvantages of this technique are given in section 2.7.6, while experimental test results can be found in section 5.4.2.

The definition of all legally and illegally reachable states as well as the corresponding state transitions in VHDL is handled by the `HammingFSM` class constructor. Afterwards, the created class instance itself contains individual `HammingFSMState` objects for manipulation as well as major attributes related to the Hamming encoding. In addition, the generated output can be used as template for custom firmware implementations in any other VHDL project. A simple example of the output generated for an FSM that encodes two Hamming d=3 states in a single data bit plus two parity bits is given below. To keep the manual state encodings, it is mandatory to use the synthesis option `-fsm_extract NO`.

```
constant s0   : std_logic_vector(2 downto 0) := "000";
constant s0h0 : std_logic_vector(2 downto 0) := "100";
constant s0h1 : std_logic_vector(2 downto 0) := "010";
constant s0h2 : std_logic_vector(2 downto 0) := "001";
constant s1   : std_logic_vector(2 downto 0) := "111";
constant s1h0 : std_logic_vector(2 downto 0) := "011";
constant s1h1 : std_logic_vector(2 downto 0) := "101";
constant s1h2 : std_logic_vector(2 downto 0) := "110";
signal current_state : std_logic_vector(2 downto 0) <= s0;
signal next_state : std_logic_vector(2 downto 0);
case current_state is
  when s0 | s0h0 | s0h1 | s0h2 => next_state <= s1;
  when s1 | s1h0 | s1h1 | s1h2 => next_state <= s0;
  when others => next_state <= s0;
end case;
```

4.5 Guidelines for Fault-Tolerant VHDL Design

As seen from the previous chapters, improvement of a VHDL design with fault tolerance is a difficult task, as behavioral simulation and circuit performance in radiation environments can differ considerably from each other when not considering several design guidelines. The fundamental reason for this behavior is founded in the optimization strategies of nowadays synthesis and implementation tools that transform HDL into logic circuits. Following a minimalist approach is counterproductive for designs which are based on redundancy, whether they have been manually designed or by using automated tools. Therefore, the following basic guidelines are strongly advised when designing fault-tolerant circuits, hereafter given for the Xilinx ISE®:

- Use **Code Conventions**: Improved reading characteristics require the use of a global naming scheme for redundant circuits. In the following VHDL examples, a constant suffix use of `_tmrN` or `_dmrN` for all replicated instances has proven beneficial:

```
signal reset_tmr0 : std_logic;
signal reset_tmr1 : std_logic;
signal reset_tmr2 : std_logic;
```

- Prevent apparently unused signals from being removed by adding the predefined **keep** attribute in the VHDL architecture body's declarative part:

```
attribute keep: string;
attribute keep of reset_tmr0: signal is "true";
attribute keep of reset_tmr1: signal is "true";
attribute keep of reset_tmr2: signal is "true";
```

  Alternatively, this functionality can be added via the design's user constraints file (UCF):

```
NET "reset_tmr0" keep;
NET "reset_tmr1" keep;
NET "reset_tmr2" keep;
```

- Prevent the removal of apparently equivalent registers by adding the predefined **equivalent_register_removal** attribute in the VHDL architecture body's declarative part:

```
attribute equivalent_register_removal : string;
attribute equivalent_register_removal of reset_ff_tmr0 :
   signal is "no";
attribute equivalent_register_removal of reset_ff_tmr1 :
   signal is "no";
attribute equivalent_register_removal of reset_ff_tmr2 :
   signal is "no";
```

  In general, this option can also be disabled by using the synthesis option `-equivalent_register_removal NO`.

- Force the assignment of IOBs for constant timing delay in redundant signals when soldered together outside of the FPGA, as explained in section 2.5.7.4, by adding the predefined **IOB** attribute in the VHDL architecture body's declarative part:

```
attribute IOB : string;
attribute IOB of reset_in_tmr0 : signal is "true";
attribute IOB of reset_in_tmr1 : signal is "true";
attribute IOB of reset_in_tmr2 : signal is "true";
```

Use furthermore an independent voter to directly feed each of the output IOBs inside the FPGA, whether the corresponding pin has been triplicated or not:

```
reset_ack <= ((reset_tmr0 and reset_tmr1) or (reset_tmr0 and
    reset_tmr2) or (reset_tmr1 and reset_tmr2));
```

- Use **BRAM** with hardware-assisted ECC calculation if available (Xilinx Virtex, Artix, Kintex) or apply a custom SECDED algorithm in any other case. To prevent the accumulation of errors within the memory, implement a preceding memory scrubber which continuously refreshes all data words (see section 4.2).

- Handle **distributed memory** with special care. When using SLICEM in RAM mode, the corresponding LUTs will be included in the bitfile's mask file and therefore excluded from the static configuration scrubbing. In this case, error accumulation has to be mitigated by the use of a custom dynamic memory scrubbing approach. This is different in ROM mode, where the LUT content is static and can be continuously refreshed without hesitation. Including ROM LUTs in the golden configuration scrubbing file requires manual modification of the output bitfile's mask file. To prohibit automatic SLICEM allocation, the synthesis option: `-ram_style Block` is highly recommended, which occupies BRAM instead.

- Use pass through **shift registers**, realized in flip-flops or SLICEMs, only in TMR mode and ensure that the SBU rate multiplied by the data retention time of a single bit within the shift register is lower or equal to one to prevent error accumulation. In addition, prevent the automatic extraction of additional, unprotected shift registers by using the synthesis option: `-shreg_extract NO`

- To enable **configuration scrubbing** on the SysCore board, get sure to use the bitgen options `-g DriveDone:yes`, `-g Persist:Yes` and `-g Security:None`, which enable done pin signaling, SelectMAP configuration interface persistence as well as readback and reconfiguration in general.

- Prevent design **fanout optimization**, that adds unprotected flip-flops to the design, by applying the synthesis option `-register_duplication NO`.

- If **automatic FSM encoding without error recovery** is applied, the use of synthesis options `-fsm_extract YES`, `-fsm_encoding One-Hot` and `-safe_implementation Yes` is highly recommended, as it prevents the state machine at least from entering undefined states and implements additional logic circuits that force a deterministic transition into the user-defined `others` reset clause in case of an SBU. If this option cannot be used globally for all state machines in the design, an individual definition is possible by defining the predefined **fsm_encoding**, **safe_implementation** and `safe_recovery_state` attributes in the VHDL architecture body's declarative part:

```
attribute fsm_encoding : string;
attribute fsm_encoding of state : signal is "one-hot";
attribute safe_implementation : string;
attribute safe_implementation of state : signal is "yes";
attribute safe_recovery_state : string;
attribute safe_recovery_state of state: signal is "recovery";
```

- If **custom FSM encoding with SBU error recovery** becomes necessary (see section 2.7.6), the use of synthesis option `-fsm_extract NO` is mandatory. The definition of the FSM *User* encoding algorithm via `-fsm_encoding User` can furthermore be counterproductive, as it results in state extraction with One-Hot handling, especially for states defined via enumerated types:

```
type states is (s0, s1);
attribute enum_encoding: string;
attribute enum_encoding of states : type is "000 111";
```

  In addition, an appropriate Hamming basic state encoding with distance d=3, the introduction of auxiliary states with distance d=1 to cover SEE, as well as a complete set of transitions are recommended. To bypass the necessity of the Hamming encoder/decoder, all FSM states and SBU auxiliary states as well as transitions can be directly embedded within the HDL. All of the routing and LUT resources added in this context are static circuits and require configuration scrubbing to successfully operate in the background. The feature of automatically generating such a complete FSM is provided by the tool described in section 4.4.4. To give an example, the FSM description can look like:

```
constant s0   : std_logic_vector(4 downto 0) := "00000";
constant s0h0 : std_logic_vector(4 downto 0) := "10000";
constant s0h1 : std_logic_vector(4 downto 0) := "01000";
```

```
constant s0h2 : std_logic_vector(4 downto 0) := "00100";
constant s0h3 : std_logic_vector(4 downto 0) := "00010";
constant s0h4 : std_logic_vector(4 downto 0) := "00001";
constant s1   : std_logic_vector(4 downto 0) := "10011";
constant s1h0 : std_logic_vector(4 downto 0) := "00011";
constant s1h1 : std_logic_vector(4 downto 0) := "11011";
constant s1h2 : std_logic_vector(4 downto 0) := "10111";
constant s1h3 : std_logic_vector(4 downto 0) := "10001";
constant s1h4 : std_logic_vector(4 downto 0) := "10010";
constant s2   : std_logic_vector(4 downto 0) := "11100";
constant s2h0 : std_logic_vector(4 downto 0) := "01100";
constant s2h1 : std_logic_vector(4 downto 0) := "10100";
constant s2h2 : std_logic_vector(4 downto 0) := "11000";
constant s2h3 : std_logic_vector(4 downto 0) := "11110";
constant s2h4 : std_logic_vector(4 downto 0) := "11101";
signal s_curr : std_logic_vector(4 downto 0) <= s0;
signal s_next : std_logic_vector(4 downto 0);
case s_curr is
  when s0 | s0h0 | s0h1 | s0h2 | s0h3 | s0h4 => s_next <= s1;
  when s1 | s1h0 | s1h1 | s1h2 | s1h3 | s1h4 => s_next <= s2;
  when s2 | s2h0 | s2h1 | s2h2 | s2h3 | s2h4 => s_next <= s0;
  when others => s_next <= s0;
end case;
```

As errors in the FSM state registers can also accumulate, it is important to refresh these bits continuously, either by ensuring that its operation frequency is higher than the upset rate or by implementing a feedback path.

- Place each redundant leg of the design in one or more non-overlapping FPGA I/O banks and use a replicated **DCM** for each individual leg. This enables resetting of the DCM right after SET desynchronization by scrubbing all static and dynamic data within the corresponding bank with GLUTMASK disabled. A validation circuit can hereby compare the DCM output with expected values [252].

- Do not apply additional **register replication** if these registers are already part of a redundant leg in the design. Define clock-synchronized voters instead, which involve all redundant instances and form a majority-decision to feed the corresponding registers.

- Implement internal or external **watchdog timers** with reset capabilities wherever it is possible, especially if there is no possibility to perform a manual reset

for the whole device. This retains system operation even after a locking condition.

Without entitlement to completeness, these guidelines should be considered as general recommendations when designing fault-tolerant circuits in VHDL. Not every aspect is necessary across all application scenarios. Please consider the introductory sections 2.7 about firmware fault tolerance before making a decision about a general mitigation strategy. In some situations, less overhead but replicated low-power devices might be the golden path to reach radiation tolerance of SRAM-based FPGAs.

5 Experiments and Results

5.1 Radiation Experiments

SEE simulation tools, as depicted in section 2.10, provide immediate assistance when it comes to the estimation of radiation faults within FPGA SRAM storage cells of a designated feature size. Unfortunately, they are limited to known radiation effects as well as physical design primitives that are accessible and that can be altered via the device's programming interfaces offered by the chip vendor. Major disadvantages are also a lack of options to simulate the behavior of internal status registers, which are responsible for device operation such as programming and reset, as well as the difficulty of observing embedded ASICs as mentioned in section 2.5.8. In consequence, the only testing method which promises reliable results is an irradiation beam test. This test has to be run with a device whose type and feature size matches the one for later use, including the full package without device thinning. Only by this way, every known and more important every unknown radiation effect can carefully be investigated. But using a DUT without device thinning requires selection of a radiation source that is capable of providing a sufficiently high particle energy while being able to traverse the chip package to penetrate the underlying semiconductor material. While crossing this additional matter, secondary particles are released from the shielding material as explained in section 2.6.1. They add radiation impact which cannot be simulated by the use of simple upset emulation. Another reliability investigation, which can effectively be performed only via beam test, is natural aging and the according FIT. Irradiation causes accelerated aging effects within devices, which normally have to be gathered by simultaneously recording long-term FIT rates of an extensive array of devices within earth's regular radiation field at high altitude. This offers significant advantages, especially for expensive semiconductors such as the IBM POWER devices, which can be irradiated with 150 MeV protons [529]. In case of accelerated TID tests with heavy ions, it is even more important to carefully calculate the particle energy in relation to the irradiated material, since heavy ions with insufficient energy and therefore higher LET are causing major damage or tend to get stuck within the semiconductor's lattice structure as explained in section 2.5.4. Due to all these prerequisites, the range of suitable irradiation facilities is often limited – a basic overview to simplify a selection is given in [194].

The following sections try to introduce the basic test equipment, DUT configuration, flux monitoring as well as some basic radiation considerations, that were necessary to successfully accomplish SEE or TID irradiation tests with the designated FPGA semiconductors and other electronic devices.

5.1.1 Radiation Monitoring

Determining the cross-section of a DUT during particle beam test implies to sum up its error rate and, in parallel, measure the particle fluence over time. In addition, it is important to record the irradiation angle, as discussed in [243, 338, 64]. Fluence measurement for particle radiation can for example be done via conventional particle counters such as scintillators or ionization chambers. Scintillators are based on photo-diode measurements of radiation-induced light flashes within dark solids, mostly crystals or plastic. They can be efficiently combined within hodoscopes, that allow complex reconstruction of particle tracks within all three dimensions. Since the basic principle of scintillation is dependent from the half-life period of light and therefore its decay time, a saturation level is reached when too many impulses are generated in too short time intervals, which results in missed events. Therefore, scintillation devices are typically used within a range of 10^4 to 10^7 particles per second. Higher particle rates within a 10^6 to 10^{10} per second flux frame are typically measured with an ionization chamber based on a large high voltage capacitor within a gaseous system and a corresponding electrometer to count emitted electrons from the irradiated gas. Another radiation measurement method which outperforms scintillators especially in applications with high background noise or closely coupled signals is the use of Germanium semiconductor diodes [188].

But radiation monitoring for semiconductor devices at sufficient particle flux can also be performed by irradiating the memory cells itself [530], which also applies to FPGA SRAM cells. Therefore, an easy way to monitor the FPGA particle flux is to put a second sensitive device in parallel to the DUT within the particle beam. This monitoring FPGA has to have a well known cross-section to allow a subsequent backwards calculation using an adapted version of the formula given in section 2.5.2:

$$\Phi = \frac{N_R}{\sigma \cdot N_S}$$

with Φ the beam particles per unit time per unit area (flux), N_R the number of reactions per unit time, σ the known cross-section in unit area per scattering centers and N_S the number of scattering centers. Assuming a measured SEU-rate of

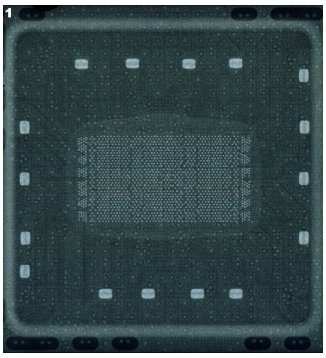
0.33 s^{-1} within a Virtex-4 FPGA that features a cross-section of $1.55 \cdot 10^{-14}$ $cm^2 \cdot bit^{-1}$ for 7200256 configuration bits, the flux calculates to:

$$\Phi = \frac{0.33\ s^{-1}}{1.55 \cdot 10^{-14}\ cm^2 \cdot bit^{-1} \cdot 7200256\ bit} = 2.96 \cdot 10^6\ s^{-1} \cdot cm^{-2}$$

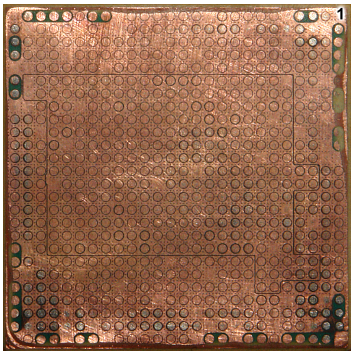
The SEU-rate N_R of the FPGA itself within a given time frame is gathered by reading back the device configuration bitstream at two different points in time via an externally connected JTAG or SelectMAP interface controller and perform a bit-wise comparison of both recorded files. Since there was no tool available which directly offered comparison and counting features on single bit level, a basic 'diffbit' tool had been developed and is available for download at the author's website. It offers the following features: full file comparison, single/multiple byte position check and byte range check. All of these options can be used with an additional counter option and a mode switch to enable or disable the display of identical data. The tool can furthermore be automated to continuously monitor incoming data, which is particularly useful for in-beam DUT alignment with the use of a remote controllable X/Y positioning system such as [531]. This read-back and comparison method has been used for several experiments within the field of this thesis. The concept of using SRAM FPGAs as real-time alignment and flux monitors for FPGA irradiation tests has furthermore been published in paper [532]. It is based on the well known ESA SEU Monitor [533, 534], which has been evaluated at GSI [535], but is not manufactured anymore.

5.1.2 FPGA X-ray Analysis

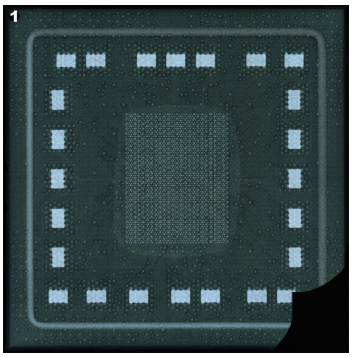
Using FPGAs in radiation experiments, especially for radiation detection and measurements made it necessary to know more about their internal stack-up as well as silicon dimensions. But gathering such internal characteristics directly from the chip manufacturer is unlikely crowned with success. A second approach is to remove the device lid and thin the top copper heat spreader, thermal interface material, as well as the bulk silicon by milling or etching as described in [214, 293]. But this process irreversibly damages the whole device and maybe internal structures. Furthermore, Xilinx COTS FPGAs are manufactured in a so called 'flip-chip' technique, which implies that the bulk silicon is placed atop of the device, hiding the real semiconductor's size as well as bonded wires or internal solder bumps [536]. Therefore, a non-intrusive way of getting insight into such devices has been chosen by taking X-ray images. The results of these FPGA X-ray imaging can be found in figure 5.1. Before irradiation, both chips have been removed from damaged boards



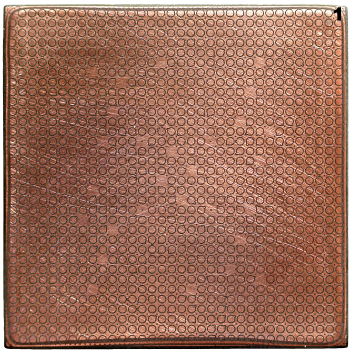
(a) XC4VFX20FF672 top view



(b) XC4VFX20FF672 bottom view



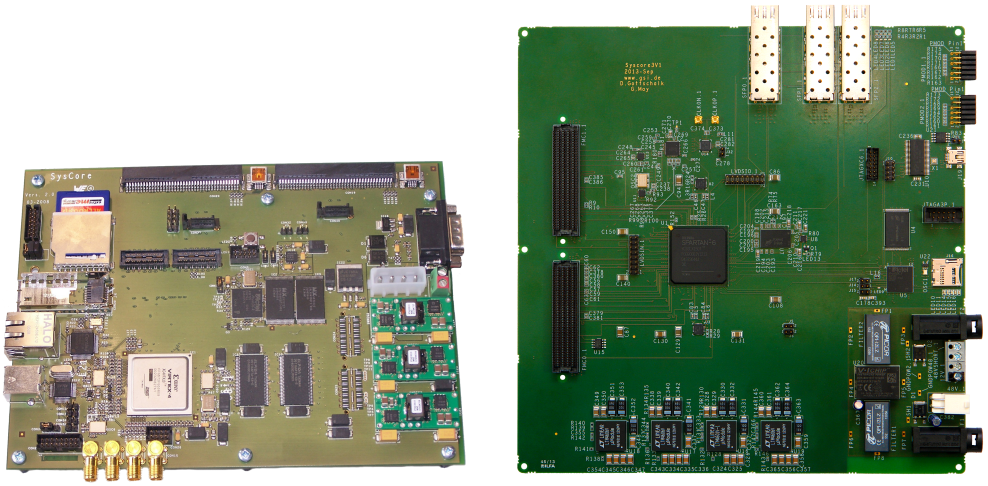
(c) XC4VLX40FF1148 top view



(d) XC4VLX40FF1148 bottom view

Figure 5.1: X-ray as well as copper contact plate views of two different Xilinx Virtex-4 FPGAs: The XC4VFX20FF672 (a, b) as well as the XC4VLX40FF1148 (c, d). Both X-ray images (a, c) have been taken with 70 kV irradiation for 0.8 seconds. For the copper contact plate images taken with a conventional camera, all solder balls have been removed by the use of abrasive paper. Remarks: The chip size exceeded the imaging capabilities in sub-figure (c), therefore it is a combination of 3 separate images, missing just a small edge. Furthermore, the horizontal borders of figure (a) are cut due to the same reason.

by the use of a conventional heat gun. Afterwards, all remaining soldering balls needed to be carefully abraded by the use of a fine-grained abrasive paper to eliminate metal noise in the X-ray imaging process. Finally, a 70 kV X-ray irradiation has been applied for different time periods, whereas an exposure time of 0.8 seconds has shown best results for this kind of devices. In contrast to the LX40 chip, the FX20 offers additional high speed serial Multi-Gigabit Transceivers (MGT). This special feature is fixed to designated pins within the device and can be clearly spotted due to the separately isolated receiver/transceiver pairs indicated in figure 5.1b as well as in its X-ray image 5.1a. Finally, the dimensions of both devices have been calculated to $0.664\ \text{cm} \cdot 1.310\ \text{cm} = 0.87\ \text{cm}^2$ for the XC4VFX20FF672 as well as $1.06\ \text{cm} \cdot 1.32\ \text{cm} = 1.40\ \text{cm}^2$ for the bigger XC4VLX40FF1148.



(a) (b)

Figure 5.2: SysCore v2.0 and v3.1 FPGA DUT platforms. The earlier version 2.0 (left) board features a central Xilinx Virtex-4 XC4VFX20 FPGA while the recent version 3.1 (right) is based on a central Xilinx Spartan-6 XC6SLX150T FPGA [512, 513]. Both boards are supporting Xilinx FPGA SelectMAP configuration scrubbing as well as GPIO watchdogs via an adjacent Actel ProASIC3 FPGA. External functionality can be added by universal interfaces or mezzanine cards.

5.1.3 SysCore DUT Platform

The FPGA test platforms that have been used for fault tolerance experiments are SysCore v1.0 and SysCore v2.0. Both boards are nearly identical, therefore the following sections will refer to the second version only, shown in figure 5.2a. This SysCore v2.0 board was the second instance that followed the universal SysCore architecture – a concept that is based on a static processing core part and a flexible interfacing part. The static part contains a central COTS high performance data processing FPGA and its corresponding components required for basic operation while the flexible part offers a rich set of universal interfaces which allow addition of various external functionality. The SysCore v2.0 board therefore was designed to be reusable for multiple application scenarios as they can be found in today's particle accelerator detector's FEE readout systems. One of such applications required the use in radiation environments, therefore the base system was temporarily extended with components that were necessary to perform efficient FPGA configuration scrubbing via the SelectMAP interface. But this functionality was planned to be finally moved to a flexible part to become a detector-specific removable feature. In consequence, the static part of the SysCore v2.0 development board now offers a Xilinx

Virtex-4 XC4VFX20 SRAM-based FPGA which covers the main data processing tasks but is accompanied by an Actel ProASIC3 A3P125 flash-based FPGA responsible for configuration scrubbing as well as watchdog tasks. The configuration data of the Actel FPGA can be stored within the device itself, while the golden and scrubbing configuration files of the Xilinx FPGA are stored in two additionally placed 64 Mbit Macronics MX29LV640 flash-based memory chips. Beside of additional copper and optic interfaces, the dynamic part of the SysCore v2.0 board is covered mostly by ERNI and Samtec QSE connectors.

Since SysCore v2.0 was such a success but always low in quantity, it has been updated with recent low-cost FPGAs as well as interfaces [512, 513]. The resulting platform is shown in figure 5.2b. The previously required configuration scrubbing approach had been kept due to the board's usage scenario as development platform. SysCore v3.1 now features a central Xilinx Spartan-6 XC6SLX150T SRAM-based FPGA, accompanied by a slightly bigger Actel ProASIC3 A3P600 flash-based FPGA for scrubbing. The additional flash memory required to store the SRAM configuration files, a 4 Gbit Micron ECC SLC NAND MT29F4G chip, has now been selected to be less susceptible to radiation due to the single bit per cell technology, internal ECC as well as durable charge pumps [537]. The dynamic interfacing part is covered by two Samtec Vita 57.1 connectors [538] which both nearly offer a high pin count (HPC) layout and therefore enable the CBM detector groups to efficiently extend the SysCore v3.1 board with designated external functionality. Linear Technology LTM4601 POL (Point-of-Load) DC/DC converters round up the whole system to be more radiation- and fault-tolerant as the previous platform. Summarizing the version upgrade, there is nothing to be said against an advancement of all proposed SysCore v2.0 fault-tolerant FPGA techniques to the latest SysCore v3.1 platform for further use in future applications.

5.1.4 Experimental Configuration

The experimental setup during DUT irradiation in beam test experiments is depicted in figure 5.4. As indicated, the DUT was placed together with various other physical experiments in a closely related setup chain. To get reliable values for the actual particle fluxes, the dosimetry was placed closely related before the investigated FPGA. It was chosen depending on the used irradiation particle as well as the expected flux rates between a scintillator, ionization chamber or a second FPGA board with subsequent backwards calculation as explained in section 5.1.1. To ensure that the FPGA had been correctly placed within the center of the particle beam,

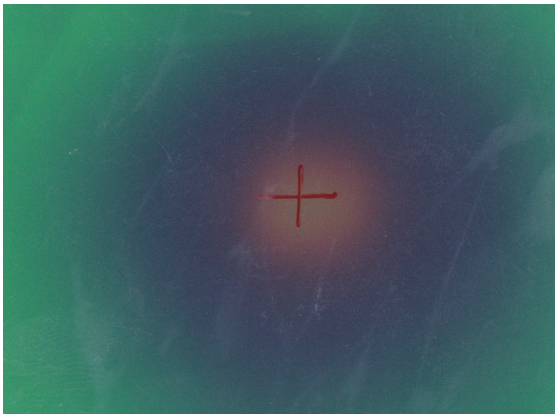


Figure 5.3: Sample picture for a radiation sensitive radiochromic film (GAFCHROMIC EBT by ISP) used for alignment purposes within the beam line. The picture has been taken during proton irradiation on August 5th, 2014 at COSY in Jülich, Germany.

a radiochromic film (GAFCHROMIC EBT by ISP) was attached above it. This foil is darkened by passing ionizing particles as shown in figure 5.3 and therefore it allows immediate detection of alignment issues. An additionally placed Axis M1014 Network Camera beside the setup furthermore allowed a visual live inspection during irradiation. A recorded animation of this darkening process is available at the author's website.

Readout of all experimental data from the investigated FPGA was performed via a conventional PC located within the cave about 1 m directly below the setup. Although this computer was not directly placed within the beam line, it showed abnormal behavior during some proton particle irradiation tests with intensive flux rates of about 10^9 particles per spill. This PC was also used to control the SelectMAP configuration scrubbing process handled by the on-board Actel FPGA. It could furthermore bypass this process and switch to manual configuration and read-back via the JTAG interface and a connected Xilinx HW-USB-II programming device.

None of the SysCore boards used for irradiation tests has ever been permanently damaged due to SEE or TID effects. It therefore promises to be a reliable and robust platform for such experimental setups, provided that the annealing periods between each two tests are sufficiently long.

5.1.5 TID Tool

Semiconductor TID calculation in advance of irradiation tests is particularly important to prevent such devices from permanent damage as explained in sections 2.5.5.2 and 2.5.6. To simplify this process, the development of a calculation tool has been supervised within a project internship. It easily visualizes eligible particles for irradiation of a specific FPGA within the given time frame. The output of this tool

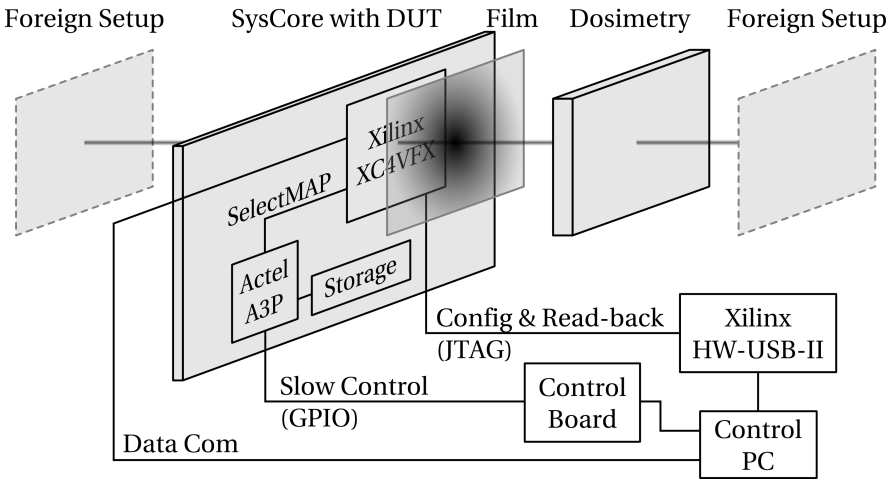


Figure 5.4: Irradiation beam test setup and readout chain used in all FPGA experiments. The central DUT was located in a chain with several other experiments. Individual dosimetry as well as radiochromatic film were attached directly in front of the FPGA. Experimental data readout was performed via a conventional PC in the cave. Scrubbing and read-back could be switched between the on-board Actel FPGA and its attached memory and the JTAG interface with an attached PC. The Actel slow control features were also provided from this PC.

for a 300 s penetration of SiO_2 material with 50 MeV per atomic mass unit for instance is shown in figure 5.5. The user only needs to specify irradiation time as well as expected particle flux. The maximum TID value is selected from a drop down list which contains several preconfigured FPGA devices, but it can also be entered manually. The subsequent calculation displays a color-coded periodic table, whereas greenish cells are promising a risk-free usage and orange cells are getting close to the physical TID limit. Particles in red cells should definitely be avoided to prevent physical device degradation.

The tool furthermore supports calculation of a particle's penetration depth within the selected target material as well as the time to reach a designated TID value. Finally, even SEU rates per second can be returned. All of these features sum up to a helpful tool which simplifies the particle selection in advance of a beam test.

5.2 Dynamic Partial Reconfiguration Experiments

The best way to practically test the efficiency of an implemented scrubbing method for SRAM FPGAs is to build a custom input data processing firmware design that occupies most of the designated device's static resources and returns out-

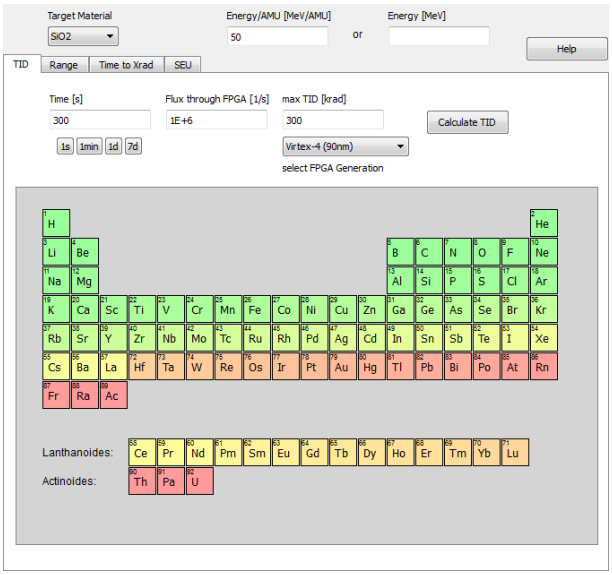


Figure 5.5: Screenshot of the designed TID, SEU and range calculation tool to easily visualize eligible particles for beam test experiments. The depicted scale has been generated for a 300 s penetration of SiO_2 material with 50 MeV per atomic mass unit. Greenish cells are promising a risk-free usage while orange cells are getting close to exceed the physical TID limit. Particles in red cells should definitely be avoided to prevent physical device degradation.

put values that have been routed through the entire chip while undergoing error injection by irradiation or simulation. These values subsequently have to be checked for validity to determine erroneous calculations. This method even allows to draw conclusions about components whose actual state cannot be determined by a configuration readback interface. Since it is impossible to occupy 100% of all routing resources by this method and nearly impossible to prevent the usage of flip-flops in a clocked design, the determined error rates can only give a basic impression about the effectiveness of scrubbing and should not be used to calculate an SRAM device's configuration bit cross-section. This is different for other devices, such as Flash memory, where the routing is insensitive against spontaneous modifications and where the selected primitives contribute solely to the final test results. The FPGA-internal data input/output interface required for communication with the external test controller is usually provided by an optimized serial interface that is limited to the designated transfer protocol and speed and that has manually been improved with fault tolerance to limit the amount of incidental, unwanted impact.

To test the blind configuration scrubbing performance of the SysCore v1 and v2 development boards, two firmware designs have been prepared: A shift register as well as a multiplier design. The FPGA-internal communication interface used in both tests has been prepared and manually placed nearby the corresponding IOBs to provide fast and efficient data transfer to the connected PC without interruption. The partial configuration bitfiles required by the external scrubbing controller have been prepared according to section 4.1 and finally provided to the external blind

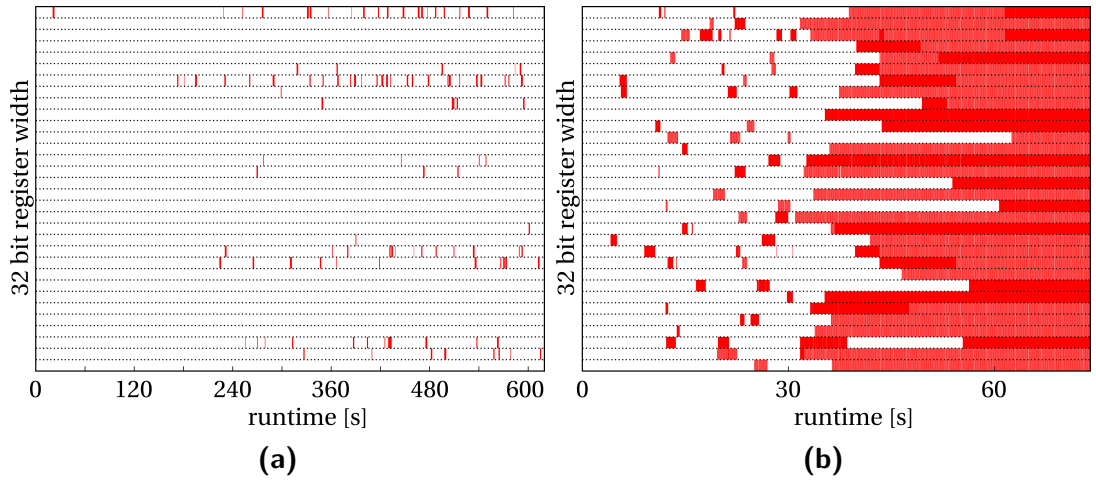


Figure 5.6: Static configuration upsets in a Virtex-4 FX20 FPGA, indicated by red bars within a 32 bit wide shift register. Binary test patterns have been routed through the whole device and externally recorded at the end of each bit chain. Blind scrubbing continuously repaired the configuration in background. The width of each bar therefore represents the downtime of the specific chain due to routing upsets. Single flip-flop damage may be indicated by the narrowest lines but cannot be distinguished from configuration upsets. The measured results depicted in subfigure (a) have been taken during 2.3 GeV proton particle irradiation with a flux of $3 \cdot 10^6$ $p^+ \cdot s^{-1} \cdot cm^{-2}$ and scrubbing turned on all the time. The experimental values shown in subfigure (b) have been taken during 1.69 GeV ruthenium heavy ion particle irradiation with a flux of $1-3 \cdot 10^4$ $ions \cdot s^{-1} \cdot cm^{-2}$ and scrubbing turned off after 30 seconds. Both irradiation tests have been performed at the GSI linar accelerator in Darmstadt (Germany) between 2008 and 2009.

scrubbing controller. The static configuration bits embedded within these files have been continuously copied at runtime to the Xilinx DUT FPGA.

The first shift-register test design has been selected to occupy a maximum and homogeneous routing area across the whole FPGA while utilizing nearly all flip-flops of all slices within all configuration frames of the device. The shift-register itself was 32 bit wide and offered an overall depth of 524 bits. This occupied 99% of all device flip-flops while leaving most of the LUTs untouched at only 1% occupation rate. At this point, it is important to note that LUT shift register (SRL16) extraction had been disabled to prevent the dynamic test data from being hold in static LUTs from which they would have been continuously overwritten by the configuration scrubbing process. Alternating input data with a designated binary test pattern, sent via the serial interface from the external control PC, have now been shifted between the generated flip-flops until they reached the end of the chain to be returned. As soon as the routing of this chain broke at random parts within the FPGA, new incoming data

could not be forwarded correctly and the output of the whole shift-register turned wrong as soon as the specific position was reached. It changed back to correct operation short time after a scrubbing cycle successfully repaired the broken chains. This behavior was recorded by the connected PC. In consequence, the time of persistence of such an error is a direct indicator for the performance of the scrubbing controller. If just a single bit or a set of bits was upset at a specific timestamp, then a flip-flop might have directly been damaged.

The second multiplier test design was selected to utilize all static components within an FPGA. Therefore, all LUTs and with it an accompanying, maximum amount of routing resources had to be occupied. This has been reached by synthesizing multiple hierarchical $32 \cdot 32$ bit integer multipliers taken from [539] into static LUTs while disabling the FPGA-embedded DSPs. As a result, 89% of all LUTs and only 3% flip-flops were occupied. Similar to the shift-register design, binary test patterns have been sent from the external control PC to the FPGA via the serial communication interface and the returned results have been immediately checked. An upset in the static LUT or routing elements has shown direct impact on the calculation results of a multiplier component as long as a scrubbing cycle was able to successfully repair the corresponding positions.

Both designs have been irradiated at the GSI linar accelerator in Darmstadt (Germany) with 2.3 GeV proton particles in August 2008 as well as with 1.69 GeV ruthenium particles in February 2009. The DUT was a SysCore v1 board with Virtex-4 FX20 FPGA in FF672 BGA package. It has been directly irradiated from the top cover at a 90 degree rectangular impact angle. Due to the high particle energies, no device thinning had to be performed. Low energy conditions normally require package removal and silicon substrate thinning of the upside down manufactured chips in flip-chip assembly. The package was also kept in order to reflect real-life conditions with additional material impact. Neither a heatsink, nor a cooling fan were present. The experimental results are depicted in figure 5.6. As clearly visible, scrubbing performed well in repairing the broken static device configuration bits. Figure 5.6b also shows the accumulation of errors when scrubbing is turned off. The horizontal width of an error bar directly reflects the total downtime of the specific shift register before a scrubbing cycle repaired all involved, erroneous routing cells. The effectiveness of any configuration scrubbing technique therefore depends mostly on the reconfiguration speed and in consequence on the selected interface and controller. A good error detection performance of 25 ms [455] can be reached for example with the ICAP interface as explained in section 2.7.7.

All of these results clearly indicate, that the blind scrubbing process implemented on the SysCore boards is working properly and can be well integrated in the combined approach of a fault-tolerant system based on COTS FPGAs.

5.3 Experimental Particle Flux Determination

As described in section 5.1.1, SRAM-based FPGAs can serve as radiation detectors beside of the well established dosimetry devices which are commonly used to determine the flux rate of a particle beam. This method becomes available as soon as the irradiation causes sufficient SEE in the device silicon and therefore SBU in its transistors without damaging the device due to accumulating TID (see sections 2.5.4 and 2.5.6). This kind of dosimetry can be considered as a very simple and rapidly realizable method, as is requires neither auxiliary equipment, such as expensive sensors or high voltage power supply, nor special preparation in terms of alignment or complex gauging.

This type of SRAM detector was realized in the scope of this thesis by using a conventional Xilinx Virtex-4 FX20 FPGA with known cross-section of $\sigma = 1.55 \cdot 10^{-14}$ cm$^2 \cdot$ bit^{-1} for protons exceeding 50 MeV and $N_S = 7200256$ scattering centers (5488 configuration frames, each containing $41 \cdot 32$ bit words). The FPGA, embedded on a SysCore v1 development board (see section 5.1.3), was therefore read back continuously via the JTAG protocol from a conventional Xilinx platform cable connected to a personal computer nearby the beam table. In all cases where a constant particle flux could be provided by the accelerator test facility, the DUT FPGA itself was used to monitor the particle flux before loading the actual firmware test design. Otherwise, a second SysCore board was mounted in front of the actual DUT to monitor varying flux rates. The periodically retrieved configuration bitstream was immediately compared against its previously gathered data to stay informed about every single bit flip and back flip as described in [532]. The number of reactions per unit time N_R was afterwards calculated by simply summing up these upsets at run time and normalizing them to s^{-1}. Finally, the flux Φ could then be calculated according to the formula given in section 5.1.1.

To provide experimental test results for this concept of flux calculation via SRAM readback, a long term 2.4 GeV proton beam test has been scheduled in December 2010 at the COSY accelerator in Jülich, Germany. During this test, a scintillator was operated and logged in parallel to the proposed SRAM FPGA while monitoring the upset readback results. The total continuous irradiation period of this test run was 14 hours and 53 minutes (53580 s). At the end of this period, the scintillator indi-

cated a proton flux of $\Phi = 3.02 \cdot 10^4\ \mathrm{s}^{-1} \cdot \mathrm{cm}^{-2}$ while the FPGA readback returned 292 SBUs, which equates to $N_R = 5.45 \cdot 10^{-3}\ \mathrm{s}^{-1}$ and finally:

$$\Phi = \frac{5.45 \cdot 10^{-3}\ \mathrm{s}^{-1}}{1.55 \cdot 10^{-14}\ \mathrm{cm}^2 \cdot \mathrm{bit}^{-1} \cdot 7200256\ \mathrm{bit}} = 4.88 \cdot 10^4\ \mathrm{s}^{-1} \cdot \mathrm{cm}^{-2}$$

Comparing both values of Φ indicates that the SRAM readback technique with backwards flux calculation can be considered equivalent to the conventional dosimetry practices and therefore provides reliable measurements but at much lower effort.

5.4 Fault-Tolerant Firmware

Nearly all of the implemented, fault-tolerant firmware designs have been verified for correctness and efficiency, either by simulation, practical irradiation test or both. In the first instance, this includes dynamic memory scrubbing, which ensures that upsets in a memory array are not able to accumulate over time by continuously reading all data words and their corresponding ECC and refreshing the content where necessary. It has been implemented and tested for the Xilinx Virtex and Spartan FPGAs as well as for the Texas Instruments TMS570 COTS microcontroller. These results are shown in the following subsection 5.4.1. A related publication is currently pending.

In addition to the conventional modular redundancy techniques that can be applied to the firmware of SRAM-based FPGAs to mitigate SBU and MBU effects, protection of FSMs is an essential part, as state machines are used to define the most basic deterministic behavior of a design. The short period between occurrence and correction of a configuration bit or register error can be sufficient to disturb their correct operation. The proposed FSM encoding with a complete set of user-defined basic and auxiliary Hamming states as well as the corresponding transitions has been generated with the JFTToolkit described in section 4.4 and tested during a particle beam test. The results are shown in subsection 5.4.2 and have been published in [442].

Other interesting test results of the fault-tolerant MIPS R3000-compatible microprocessor are given in subsection 5.4.3. They comprise simulation by error injection as well as practical irradiation results that have been taken during a beam test together with [286]. In principle, they can be seen in contrast to the Texas Instruments TMS570 hard-wired microprocessor when operated with a working instance of memory scrubbing. The results have been published in [540].

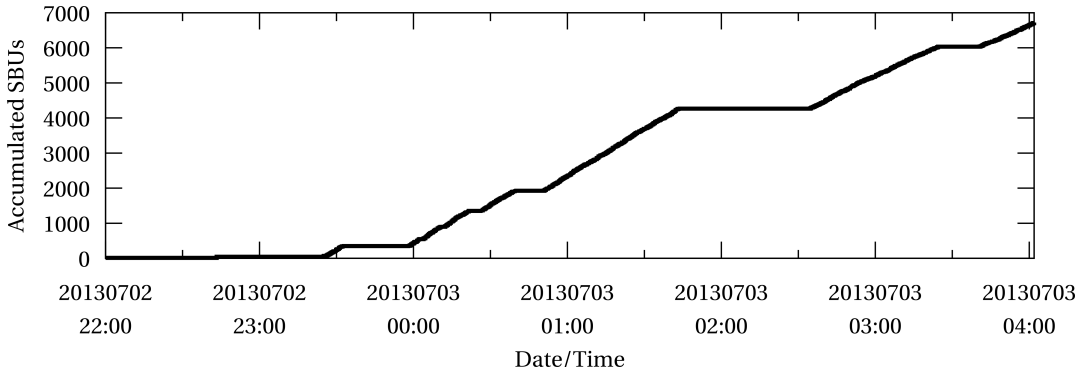


Figure 5.7: Proton irradiation results of the Texas Instruments TMS570 with enabled software-implemented dynamic memory scrubbing assembler routine. As depicted in the accumulated graph, SBUs occurred evenly during active irradiation phases while being absent in the official beam pauses. In total, 6722 upsets have been detected, corrected and successfully written back to the memory blocks. The 2 GeV proton particle beam had an intensity of $5 \cdot 10^6$ $p^+ \cdot s^{-1} \cdot cm^{-2}$ to ensure adequate ionization impact.

5.4.1 Dynamic Memory Scrubbing

Scrubbing of dynamic on-chip memory has been implemented for Xilinx FPGAs as well as for the Texas Instruments TMS570 as an example for COTS microcontrollers. For implementation details please refer to the subsections of 4.2.

The dynamic memory scrubber for Xilinx FPGAs is necessary as soon as the device is exposed to ionizing radiation which generates a sufficient amount of upsets that allows multiple SBU in the BRAM to accumulate. Even hardware SECDED support as offered in the Virtex FPGA series would then no longer be in the position to correct such erroneous memory; Spartan does not even offer EDAC. In consequence, this functionality has been added manually. To find a solution that keeps the logic consumption at a reasonably low level, the efficiency as well as resource consumption of a triplicated BRAM vs. duplicated BRAM with $8+1$ bit parity implementation has been investigated. In direct comparison, the DMR implementation of a $2048 \cdot 32$ bit BRAM in a Xilinx Spartan-6 FPGA requires 37 flip-flops and occupies 200 LUTs while utilizing 8 BRAM primitives. The corresponding TMR scrubber occupies the same number of 37 flip-flops, only 108 LUTs, but 12 BRAM primitives. At first glance, the DMR approach might be the better solution. Now assuming that an SBU has damaged a storage word of a single BRAM. An uncorrectable MBU situation for the TMR approach arises as soon as a second upset hits exactly the same physical address in one of the other two units before a scrubbing cycle was able to carry out repairs. Two bit positions out of $3 \cdot 32 = 96$ are therefore known to be critical. For the DMR

approach, with one parity bit protecting each 8 regular bits, an uncorrectable MBU situation arises as soon as either the same 9 bit parity protected block is hit twice at different positions or each of both mirrored parity blocks is hit once at an arbitrary position. This results in 17 out of $2 \cdot 36 = 72$ critical bit positions for the second upset. Assuming now p as the probability to hit a single bit in the BRAM. The probability to generate an MBU is then given with $p_{\mathrm{TMR}} = \left(3 \cdot 32 \cdot p\right) \cdot \left(2 \cdot p\right) = 192 \cdot p^2$ as well as $p_{\mathrm{DMR}} = \left(2 \cdot 36 \cdot p\right) \cdot \left(17 \cdot p\right) = 1224 \cdot p^2$, which indicates an increase of 6.4. This disadvantage outweighs the additional resource consumption of TMR. Triplicated BRAM is therefore more reliable in application scenarios where no SECDED is available and where SBU accumulate in the memory because their frequency of occurrence is faster than the scrubbing period. TMR is furthermore able to correct multiple bit upsets which, in the worst case, span a whole memory primitive.

The dynamic memory scrubber designed for the COTS Texas Instruments TMS570 microprocessor's on-chip SRAM-based storage has been validated in July 2013 during a particle beam test at the COSY accelerator in Jülich, Germany. The available proton ions had a momentum of 2 GeV and the total particle flux was about $5 \cdot 10^6$ $\mathrm{p}^+ \cdot \mathrm{s}^{-1} \cdot \mathrm{cm}^{-2}$ to ensure an adequate impact with measurable results. The microcontroller was placed on a COTS Hercules TMDX570LS31HDK development kit base board, together with all components for powering and operation. Readout was performed via a serial interface with a conventional computer, placed nearby the beam table. The DUT itself was mounted directly at beam level and the TMS570 was directly irradiated with full impact at an angle of 90 degrees. While performing the irradiation tests, the microprocessor's internal error reporting function has been activated to get external logging output. It has been observed that a massive number of upsets with corresponding logging effort massively degraded the overall device performance. Therefore, logging should be disabled for productive use.

The TMS570 with dual pipeline in lockstep mode has been operated for about 53 hours with nearly 14 hours of irradiation. No physical damage to the DUT could be observed. During this test cycle, a total of 11 device resets has been registered, including 5 resets caused by uncorrectable MBU in the memory. In addition, 13216 SBUs have been detected, corrected and successfully written back to the corresponding memory blocks. The longest run is depicted in figure 5.7 in which a total of 6722 SBUs and 7 resets have been logged. Without the technique of background memory scrubbing, multiple SBU would have easily accumulated in the SRAM-based on-chip memory, causing plenty of ECC failures and therefore device resets.

Design	#FF	#4LUT	Flux $[1/(s \cdot cm2)]$	Runtime $[min]$	FSM reset errors $[1/(h \cdot 10^5)]$	Misc errors $[1/(h \cdot 10^5)]$
(A)	59	1584	$8.61 \cdot 10^3$	302	23.1	0.0
			$4.82 \cdot 10^5$	354	2.8	0.0
(B)	169	7812	$1.46 \cdot 10^5$	147	3.0	0.0
(C)	169	13952	$1.72 \cdot 10^4$	59	0.0	0.0
			$2.77 \cdot 10^4$	95	0.0	0.0
			$2.37 \cdot 10^4$	11	0.0	0.0
			$2.15 \cdot 10^5$	61	0.0	2.2

Table 5.1: Hamming FSM beam test data taken from three different designs: (A) without fault tolerance or FSM auxiliary transitions, (B) with TMR base system but without FSM auxiliary transitions and (C) with TMR base system and with Hamming(6,3) FSM state encoding scheme (distance d=3) plus auxiliary transitions. All errors have been normalized to 1 hour, 100% chip resources and 10^5 proton particles. I/O pins as well as DCMs are single-use in all designs. Flux and runtime are shown for information purposes only. The results have been published and discussed in [442].

5.4.2 Hamming FSMs

As section 2.7.6 explained in detail, fault tolerance for FSMs is a critical task to handle while manually protecting a custom firmware design against ionizing radiation. Therefore, Hamming encoding with a distance of d=3 is known to provide optimal EDAC capabilities to prevent SBU, but requires a Hamming decoder for the error correction approach. To bypass this step, a complete set of states as well as transitions can be directly embedded in static device logic. To prove correct behavior of the proposed FSM modifications when introducing auxiliary states, a practical beam test at the cooler synchrostron (COSY) accelerator in Jülich, Germany has been performed in November 2011, while using 2.4 GeV proton particles for electronics irradiation. The results have been discussed and published in [442].

The synthetic Virtex-4 FX20 FPGA firmware designs investigated in this test are listed in table 5.1. All designs performed blind configuration scrubbing via the SelectMap interface in background as explained in section 2.7.7 to prevent an accumulation of static configuration errors. I/O pins as well as DCMs have been available only once. A single Hamming(15,11)-encoded FSM with a distance of d=3 and a total of 2048 states was running in all three test designs. The bit values of this

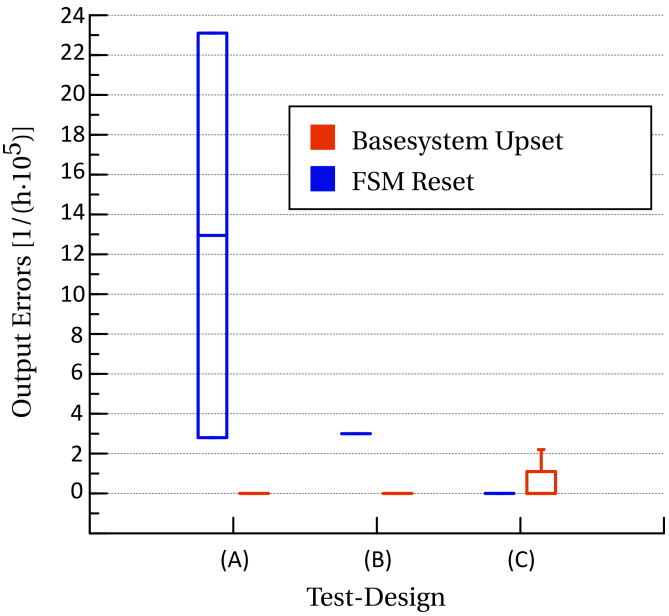


Figure 5.8: Hamming FSM beam test results boxplot (median and quartiles), normalized to 1/h, 100% chip resources and 10^5 proton particles. The test designs are using blind configuration scrubbing (see section 2.7.7) and (A) no fault tolerance or FSM auxiliary transitions, (B) a TMR base system but no FSM auxiliary transitions and (C) a TMR base system and a Hamming(6,3) FSM state encoding scheme (distance d=3) with auxiliary transitions. I/O pins as well as DCM are single-use in all designs. The results have been published and discussed in [442].

FSM's current state flip-flops have been sent to an externally connected computer for logging and analysis. Design (A) contained no TMR fault tolerance or FSM auxiliary transitions at all and therefore acts as positive control to show the device's general susceptibility and correct alignment in the beam line. Designs (B) and (C) have been improved by introducing a manually protected, fault-tolerant TMR base system (without FSM) to guarantee correct submission of all values. This increased flip-flop and LUT usage. The only difference between designs (B) and (C) is that design (C) additionally features $2^{15}-2^{11} = 30720$ auxiliary transitions with a distance of d=1 to automatically react on SBU. This increased the device LUT utilization, but the number of flip-flops remained unchanged. The FSM itself has been automatically generated by using the tool described in section 4.4.4.

Designs (B) and (C) can therefore be used to compare the efficiency of the auxiliary state and transition approach. These results are shown in table 5.1 and boxplot 5.8. To enable comparison, all captured errors have been normalized to 1 hour, 100% chip resources as well as 10^5 proton particles. The actual flux was determined by using the FPGA configuration readback procedure described in section 5.3 and the literature-given cross-section for this type of FPGA [59, 156, 541, 542]. The subsequent offline log file analysis lead to the identification of the following output error categories:

- **Short SBU in non-auxiliary FSM**: short output of wrong data when entering an undefined state; immediately repaired by configuration scrubbing and FSM reset; observed in designs (A) and (B)
- **Long SBU in non-auxiliary FSM**: long looping output of data when entering an undefined state; repaired by configuration scrubbing and FSM reset after a longer period; maybe caused by a local charge accumulation; observed in designs (A) and (B)
- **Device SEFI**: permanent output interface crash maybe caused by an SEE in the IOB or clock net; repaired by manual device reset; observed in design (C)

Finally, the results prove that protecting FSMs by a Hamming encoding scheme with a minimum distance of d=3 accompanied by auxiliary states and transitions that are reached by SBU with a distance of d=1 decreases the overall number of FSM failures to zero. All additionally required static device resources can successfully be preserved by configuration scrubbing. Their amount decreases significantly with the total number of FSM states. But most important: The consumption of dynamic flip-flops is kept low in comparison to an XTMR mitigation approach.

5.4.3 Fault-Tolerant CPU

The fault-tolerant MIPS R3000-compatible microprocessor explained in section 4.3.2 has been tested for compliance with the general requirements of radiation application scenarios by artificially injecting errors as well as practically performing a particle beam test as published in [540].

When it comes to logic resource consumption, the fault-tolerant microprocessor design with doubled pipeline, single register bank and two-step comparison involved additional 169% slices compared to the non-fault-tolerant version. In detail, this includes a surplus of 60% flip-flops, 234% LUTs as well as 100% DSPs without including any surrounding interfaces. The number of flip-flops increased due to the pipeline duplication as well as the added error detection techniques. The LUT occupancy increased mostly due to the additional Hamming and comparison logic which both rely on combinatorial LUTs. Finally, the number of DSPs used for multiplication is directly associated to the number of processor pipelines. The duplication therefore linearly increased their number. These numbers prove that the total logic resource consumption has been significantly reduced in comparison to extensive XTMR without violating error detection necessities while just limiting the error correction capabilities for application scenarios where this is sufficient. Additional

investigations considering a doubled pipeline implementation as well as a single comparison approach can be found in [286].

The initial upset simulation by error injection into an operating firmware design can be performed in multiple ways. A basic overview of such methods and tools is given in section 2.10. To simplify the process for the fault-tolerant CPU, a basic configuration file manipulation scheme has been selected. This process of randomly altering a single bit in the configuration file, transferring this file to the FPGA device and evaluating the serial output becomes available as soon as the programming CRC check has been disabled according to section 4.1.3. As programming of the whole bitfile for every single modification accumulates to a reasonable amount of time when considering seven million bits, a faster approach is available with DPR by writing only single frames to a running design (see section 2.7.7). This improved method reproduces real SBU impact even better and therefore increases the result significance. To unify the test results between processor designs with varying resource consumption, a designated test area has been defined within the FPGA and all designs have been placed and routed into this area. According to [286], the random configuration bitfile error injection lead to the following results: While for the non-fault-tolerant microprocessor 15.5% of all injected errors lead to a functional failure, the fault-tolerant microprocessor design with doubled pipeline, single register bank and two-step comparison exhibited 38.6% errors with only 1.8% uncorrectable functional errors and 36.8% successfully corrected upsets. This reduction of error susceptibility does not completely reach the effectiveness of XTMR, but requires significantly less logic resources. MBU and SET impact have not been investigated during the error injection tests.

Some of the successfully simulated designs were subsequently evaluated in a particle beam test with 1.69 GeV/u ^{96}Ru at GSI in Darmstadt (Germany) during March 2009. According to the heavy ion Weibull Fit for devices from the Xilinx Virtex-4 generation, manufactured in a 90 nm process, this corresponds to a cross-section of $\sigma = 2$ to $4 \cdot 10^{-9}$ cm$^2 \cdot$ bit^{-1} (see section 2.5.2). The particle accelerator's spill rate was $5 \cdot 10^6$ ions in 15 seconds extraction slots. The DUT which held all test designs running standard Fibonacci, multiplication and division operations was a SysCore v1 board (see section 5.1.3), read out via the on-board serial interface. It has been placed centered in the beam line with an irradiation angle of 90 degree according to the experimental setup explained in section 5.1.4. No device thinning has been applied. Due to the DUT distance of about 15 m to the last magnet and the corresponding increased beam opening angle, the particle spill fluence was reduced to 10^5 ions.

CBM Subsystem	Total Dose $[\text{year}^{-1}]$	NIEL $[\text{n}_{\text{eq}} \cdot \text{cm}^{-2} \cdot \text{year}^{-1}]$	High-Energy Hadron Flux $[\text{s}^{-1} \cdot \text{cm}^{-2}]$	SEUs in Xilinx Virtex-4 FPGA $[\text{s}^{-1}]$
STS	8 Mrad	$2 \cdot 10^{13}$	$1 \cdot 10^{6}$	$1.1 \cdot 10^{-1}$
TRD	10 krad	$5 \cdot 10^{11}$	$5 \cdot 10^{4}$	$5.6 \cdot 10^{-3}$
TOF	10 krad	$1 \cdot 10^{11}$	$1 \cdot 10^{4}$	$1.1 \cdot 10^{-3}$
PSD	1 Mrad	$1 \cdot 10^{14}$	$4 \cdot 10^{6}$	$4.5 \cdot 10^{-1}$
electronics cave	20 rad	$2 \cdot 10^{9}$	100	$1.1 \cdot 10^{-5}$

Table 5.2: FLUKA Monte-Carlo simulation results for the CBM detectors, assumung an Au particle beam at 35 AGeV on a 250 µm Au target with 1% interaction rate [543]. Total dose, NIEL as well as ion flux indicate the maximum expected values, taken from [543]. On this basis, the last column adds calculated SEU rates for a 90 nm CMOS Xilinx Virtex-4 FPGA when using $N_S = 7200256$ and $\sigma = 1.55 \cdot 10^{-14}$ $\text{cm}^2 \cdot \text{bit}^{-1}$.

In combination with the DUT's Xilinx FX20 FPGA which offers about 7.2 million configuration bits (see appendix A), the final ^{96}Ru particle fluence affecting the test designs was in the order of 10^3 to 10^4 upsets per spill. While the non-fault-tolerant processor design with single pipeline and without scrubbing operated correctly for 1.2 s on average in 2894 runs, the fault-tolerant version with doubled pipeline and blind configuration scrubbing via the on-board controller operated 15.7 s on average in 4930 runs without indicating a functional error. The relatively short time period of both designs is owed to the intensive ionizing particle beam. An actual firmware design that uses fault tolerance based on DMR can never be successfully operated in such an intense radiation field, but considering the measurements of both designs in comparison to each other confirms the previous simulation results.

While the simulation by error injection indicated an improvement-factor of 9 when switching from the non-fault-tolerant design to the fault-tolerant version, the beam test confirmed a factor of 13 that can even be improved when increasing the scrubbing frequency by the use of selective frame scrubbing.

5.5 Considerations for the CBM Experiment

Even the best rated, fault-tolerant hardware, firmware and software cannot be operated successfully if the application scenario's radiation impact exceeds critical limits. As depicted in sections 2.4 and 2.5, these limits have to be considered independently in terms of tolerated SEE and TID capabilities but can be calculated in

CBM Subsystem	Total Dose $[\text{CBM-life}^{-1}]$	SEUs in Xilinx Virtex-4 FPGA $[\text{CBM-life}^{-1}]$	Mean Time between SEUs
STS	48 Mrad	$3.5 \cdot 10^6$	9 s
TRD	60 krad	$1.7 \cdot 10^5$	3 min
TOF	60 krad	$3.5 \cdot 10^4$	15 min
PSD	6 Mrad	$1.4 \cdot 10^7$	2 s
electronics cave	120 rad	$3.5 \cdot 10^2$	25 h

Table 5.3: Accumulated simulation results from table 5.2 for the CBM detectors over a CBM-lifetime of 6 years.

advance for each component as soon as the expected irradiation intensity and total operation period are known. In case of the CBM experiment and its different detectors (see section 1.3.2), these values are predicted from extensive FLUKA Monte-Carlo simulations as shown in table 5.2. All values indicated in this table are based on an Au particle beam at 35 AGeV on a 250 µm Au target with 1% interaction rate as stated in [543]. The operational time for the CBM detector is herein given with 2 months per year ($5.184 \cdot 10^6$ s), which equates to 10^9 Au ions per second and $5 \cdot 10^{15}$ beam particles in total.

Assuming furthermore an expected CBM lifetime of 6 years ($3 \cdot 10^7$ s of operational time), the total dose rates accumulate as depicted in table 5.3. As the TID of a Xilinx Virtex-4 FPGA is known to be at least 300 krad (see section 2.5.6), the TRD and TOF detectors as well as the electronics cave can easily host such FPGAs, while PSD and STS exceed the silicon-given threshold value (without considering annealing and aging effects at room temperature).

A similar situation arises from the SEU rates, simplified in the same table 5.3: While PSD and STS would significantly suffer from the massive amount of failures, that may even cause uncorrectable MBUs (see section 2.5.4.2), the TRD and TOF detectors show a moderate number of upsets that can be easily handled by using the methods proposed in this thesis, including FPGA fault-tolerant firmware design in combination with device and memory scrubbing.

But the calculations also indicate that there is a slight radiation impact to the electronics cave which contains the 2nd tier of data combining and processing FPGAs and therefore urgently needs to be shielded from the rest of the detector to eliminate at least this point of failure.

5.6 COTS Component Evaluation

As today's modern circuit boards, especially those offering complex microcontrollers or FPGAs cannot be operated solely without a significant amount of closely bound auxiliary parts such as resistors, capacitors, and even assembled breakout circuit modules, especially POL voltage regulators, it is worth looking at different other components when operating powered silicon in radiation environments. In this context, the Radiation Working Group (RadWG) at CERN maintains its own database for components and their relevant radiation tests that have been carried out so far [544]. Beside of these tests, a couple of CBM-related components have been analyzed in this thesis and the most relevant ones are shown in this section.

First of all, a Lattice FPGA from the HADES detector has been investigated regarding SEE and TID performance in section 5.6.1. It is manufactured in the same 90 nm CMOS as the Xilinx Virtex-4 competitor and uses SRAM-based storage cells that suffer from ionizing radiation. An emerging alternative to this SRAM technology has been identified with the FRAM storage cell. It has been tested in the second section 5.6.2. Finally, the COTS switching mode power regulator used on both SysCore v1 and v2 boards has been actively irradiated as it was suspected of being responsible for several board issues. These results are shown in section 5.6.3.

5.6.1 Lattice ECP2M FPGA

In contrast to Xilinx, chip vendor Lattice offers SRAM FPGAs at a much lower price scale for a cost-optimized market. Beside of the obviously lower number of available SRAM cells and routing resources, one major limitation of these devices comprises the absence of a runtime error mitigation technique for internal static configuration cells, similar to scrubbing that is available for Xilinx FPGAs. As cost-optimization is a major objective even in the most complex applications such as particle accelerator detector electronics, a Lattice LFE2M20E-5FN256C SRAM FPGA in 90 nm CMOS with 20k LUTs and 1.2 Mbit of BRAM has been analyzed in terms of SEE and TID performance to get measurements for comparison with Xilinx devices. The reason for selecting this type of FPGA for testing purposes was that the LFE2M20E's basic slice layout, consisting of two 4-input LUTs and two D-type flop-flops, is directly comparable to the Xilinx Virtex-4 technology FPGA in 90 nm which has been used throughout the rest of this thesis.

The SEE test has been performed by irradiation with a 2.1 GeV proton particle beam possessing a flux of 10^7 $s^{-1} \cdot cm^{-2}$ at the COSY accelerator in Jülich, Germany during August 2012. The LFE2M20E DUT was directly irradiated with full impact

at an angle of 90 degrees and dosimetry was provided by an ionization chamber. Correct DUT-alignment in the beam line was furthermore ensured by applying radiochromic film (GAFCHROMIC EBT by ISP) as shown in figure 5.3. Due to the missing static configuration refresh feature at runtime, all routing-dominated firmware tests, implementing four 8 bit wide and 432 bit long shift-registers with 98% resource utilization, failed immediately with an MTBF of 0 s after the device irradiation started. This happened due to broken signal paths in the register chains. Therefore, a BRAM matrix test with 96% chip utilization, initialized with an alternating bit pattern of zeros and ones, was performed. This time, the device failed every 31 s on average (155 measurements), which results in an MTBF of 1.1 s, but the continuous data readback returned an error rate of 19 SBUs in 21 s on average (78 measurements), that calculates to a cross-section of $\sigma = 7.7 \cdot 10^{-14}$ cm$^2 \cdot$ bit^{-1} which is in the range of 90 nm SRAM FPGAs.

A subsequent long-term heavy ion irradiation test should be used to analyze the LFE2M20E's TID. The 90 nm CMOS device was expected to show physical damage between 100 krad and 300 krad as seen from figure 2.19. Therefore, two particle accelerator tests have been scheduled at the GSI Helmholtzzentrum für Schwerionenforschung GmbH in Darmstadt, Germany. The first test run in August 2012 used 1.4 GeV ^{84}Kr with an LET of 2.326 MeV$\cdot$cm$^2\cdot$mg^{-1} and was split into several irradiation/annealing phases. The total irradiation period summed up to 11.6 h, resulting in a TID of 84 krad with regard to beam distribution and flux, calculated according to the formula given in section 2.5.5.2. The subsequent, second TID test at the same accelerator facility in October 2012 used 1.7 GeV ^{58}Ni with an LET of 1.386 MeV$\cdot$cm$^2\cdot$mg^{-1} and was also split into several irradiation/annealing phases. It accumulated 288 krad in 6.5 h of irradiation. In both experiments a polycrystalline CVD diamond, attached directly to the back side of the FPGA, was used for dosimetric flux measurement and DUT alignment was checked with radio-chromatic foil (GAFCHROMIC EBT by ISP).

While neglecting the annealing phases of both experiments at room temperature, the total accumulated dose of 372 krad was not sufficient to finally destroy the Lattice LFE2M20E FPGA, but ruined the auxiliary voltage regulators on the test board that were not protected from the ionizing radiation. The FPGA itself was still operational when powered externally. Further results of both SEE and TID experiments have been published in [545].

5.6.2 FRAM Irradiation

While looking for promising alternatives to the radiation susceptible SRAM and flash memory used on the SysCore board, which are able to fully replace or at least improve the current components, FRAM has drawn the author's attention. The technological basics of this memory, including some manufacturer's neutron irradiation tests, can be found in section 2.1.5. To prove these given specifications, a custom irradiation test has been performed at the COSY accelerator in Jülich, Germany, 5th to 10th August 2014, using 2 GeV protons, a beam spot diameter of 1 cm at the last magnet and a measured total flux of $5 \cdot 10^9$ $p^+ \cdot spill^{-1}$, which was about $6 \cdot 10^8$ $p^+ \cdot s^{-1}$ in average for 7 to 10 seconds spill duration.

Two COTS FRAM chips placed on conventional development boards have been selected for irradiation. In detail, they provide the following features:

- Fujitsu MB85RS256B FRAM chip on mikroElektronika MIKROE-1486 breakout board
  - 256 kbit (32,768 words · 8 bits) plain memory array
  - memory controller with SPI supporting max. 25 MHz read and max. 33 MHz write speed
  - endurance of 10^{12} read/write operations
  - unknown nm CMOS process
- Texas Instruments MSP430FR5739 FRAM chip on MSP-EXP430FR5739 Experimenter Board
  - 16 MHz 16 bit RISC Microcontroller with FRAM storage
  - 15,744 Byte FRAM, 2 KiB SRAM, 40 IOs
  - 130 nm CMOS process

Since the FRAM's ferroelectric storage cells are reasonably promoted to be unsusceptible against magnetic fields as well as radiation, the surrounding CMOS transistors required for controlling are still vulnerable to radiation effects. To investigate this assumption, two boards of each type have been irradiated in parallel within the same beam line. The first ones were powered and readout continuously in beam while the second ones were hold completely passive and unpowered to be analyzed directly at the end of the beam test.

Both Fujitsu memory chips were initialized with 50% logical '1' and 50% logical '0' to get a prediction about the probability of 1/0 as well as 0/1 bit flips. Readout was performed by a custom-made 10 MHz SPI controller implemented on a Xilinx Spartan-3A FPGA development board. In addition, Voltage and Current of the DUT were logged every 10 ms using a Rohde & Schwarz HMC8043 power supply. As pub-

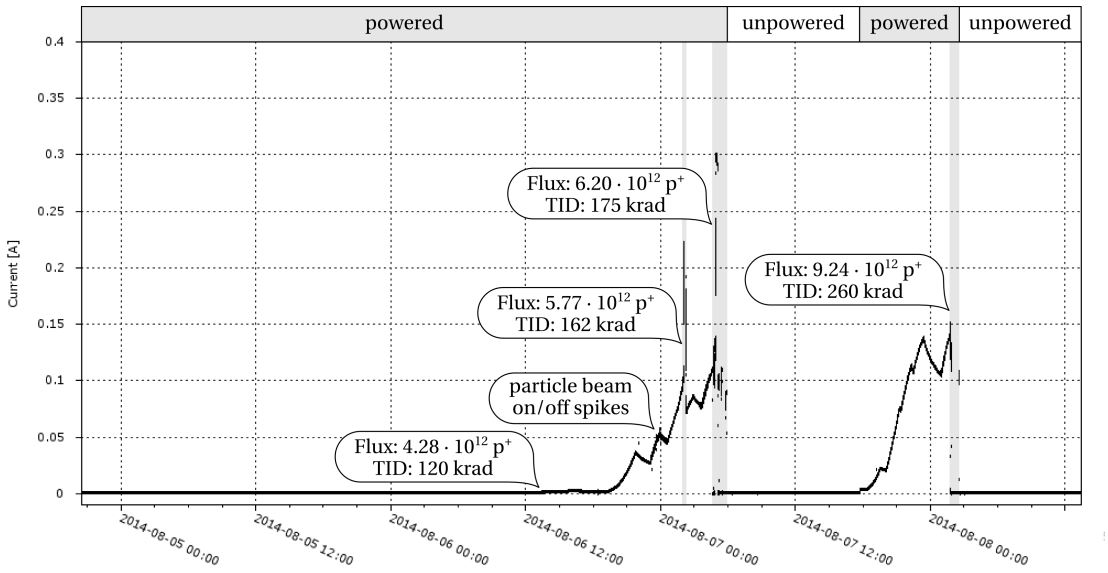


Figure 5.9: Current log of mikroElektronika MIKROE-1486 break-out board during 2 GeV proton irradiation, taken with Rohde & Schwarz HMC8043 in 10 ms intervals. The board's chip has been continuously read-back in beam. Read-back failed during the grayly depicted intervals; the corresponding total flux as well as TID values are indicated accordingly. Annealing at room temperature was performed during the unpowered periods indicated above, leading to device regeneration.

lished in [546], during the whole beam test, not a single upset in the FRAM memory could be detected, neither in active read-back mode, nor in passive mode at the end. Unfortunately, the constantly powered device broke down due to a leakage failure after about 160 krad as depicted in figure 5.9. Current drain raised gradually from an initial operation value of 0.0007 A up to a final value of 0.100 A. After an annealing period of multiple hours at room temperature without any voltage operation but partially irradiated, the device was fully functional again and the stored configuration was still unchanged. Only a short period after a few successful read-back cycles, the device's current drain raised again above 0.140 A. After a final annealing period, the device was fully functional again and the FRAM configuration was unchanged. Therefore, the total amount of irradiation caused this cumulative TID effect.

Both Texas Instruments FRAM chips were initialized with a counter test-pattern containing 50% logical '1' and 50% logical '0' to get a prediction about the probability of 1/0 as well as 0/1 bit flips. Readout was performed via USB with the manufacturer-provided MSP430 Flasher Software. It has been configured in a way that in case of a static bit error within the read-back configuration, that lasts mul-

Memory Address [0x]	Initial Data [0b]	Read-back Data [0b]
00003351	01010000	0**0**010000
00003352	01010001	0**0**0**0**000**0**
00003353	01010010	0**0**010010
00003354	01010011	0**0**0**0**001**0**
00003355	01010100	010**0**0**0**00
00003356	01010101	0**0**0**0**010**0**
00003357	01010110	0**0**0**0**0**0**10
00003358	01010111	0**0**0**0**0**0**0**0**
00003359	01011000	01011**1**0
0000335A	01011001	01011**1**1
0000335B	01011010	**1**1011**1**1
0000335C	01011011	**1**1011011
0000335D	01011100	**1**1011100
0000335E	01011101	**1**1011**1**1
0000335F	01011110	**1**1011110

Table 5.4: MSP-EXP430FR5739 logfile excerpt from 20140806-0305, indicating the observed read-back error event in device A. Fifteen 8 bit words within a single chain have been upset and mistakenly rewritten during the continuous dynamic read-back of the FRAM memory cell in beam. The wrong values statically remained until a manual reconfiguration.

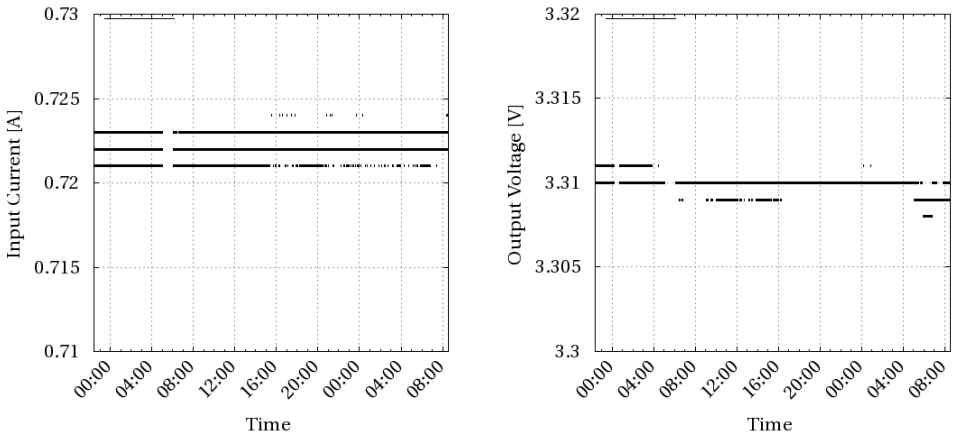
tiple read-back cycles, the initial configuration pattern is rewritten to the device. While the passively irradiated FRAM chip has not shown a single bit flip, the actively operated one observed a single but critical failure throughout the whole beam test. It lead to an upset of multiple stored bit values in a single continuous chain as depicted in table 5.4. While in the first instance, logical ones have been upset to zeros, this process reversed by changing logical ones to zeros. All values remained statically stored within the FRAM cell and have been read-back multiple times until the readout tool reset them to their initial configuration. Without knowledge about the internal circuit structures of the tested CMOS device, multiple reasons seem to be able to cause such an effect. First of all, sense AMPs in CMOS are usually realized as 4-transistor latch-type circuits. Furthermore, a 2T2C FRAM cell adds two additional transistors to the overall circuit, irrespective of word line and plate line which also require controlling. This basic circuit structure is exemplified in figure 2.6. But CMOS transistors are known to be susceptible to radiation and therefore cause SETs as explained in section 2.5.4. In consequence, an SET in the amplifier transistor's cross-coupled inverter whose pulse width overlaps with the sensing interval can lead to incorrect cell *read* results. Reading an FRAM cell is a destructive

process as explained in section 2.1.5, therefore, every cell *read* results in a subsequent cell *write*. This may have caused the SET's static *latching* into an SEU, even in the radiation unsusceptible FRAM cells. The characteristic structure of the observed upsets furthermore clearly allows to draw conclusions about the internal memory structure: The sense amplifiers are arranged in parallel to the memory rows which enables their reuse across the whole memory columns. This means if a single amplifier is upset, a single bit position within multiple words is affected for the duration of the transient pulse width. This happens even if it does not change the final result because initial and upset values are equal. A second point of failure can be the transistors which are both connecting the 2T2C FRAM cells to the bit lines. They are controlled by the word line and a transient on this wire is able to affect the power circuit between bit line and plate line to be correctly opened or closed. Therefore, the ferroelectric element may loose its stored value without performing the required rewrite cycle when being mistakenly read out.

Finally, the relatively large 130 nm CMOS manufacturing process has contributed positively to the device's radiation properties and therefore improved the results. Further down-scaling will increase FRAM susceptibility similar to conventional CMOS devices.

5.6.3 Power Regulators

All of the high performance FPGAs used within the CBM experiment require multiple DC voltages for operation. Therefore, every PCB contains a set of DC/DC converters, which take the general input voltage and transform it into all necessary destination voltages directly at the point of load (POL) to simplify power generation and distribution in the final experiment. Since some of these PCBs will be operated in temperature and radiation critical scenarios, a selection had to be made which of such components can be optimally operated. Linear regulators were found on some of the previously manufactured SysCore v2 boards, maybe due to the low pricing, but they emitted more heat than expected and therefore had to be additionally cooled with passive heat sinks. Less temperature critical and therefore more power efficient are switching regulators, since their voltage output is not enabled continuously but switched in a sufficient interval to provide a designated output voltage. This solved the temperature issue, but left the question about radiation tolerance. A use of radiation qualified components was also impossible, because it would have caused overly high costs for large quantities as required for the detector FEE. These components furthermore may probably be export restricted (see section 2.5.10).



(a) **(b)**

Figure 5.10: Beam test logs of a Texas Instruments PTH05010 [547] switching mode power regulator. Data has been taken during 2.4 GeV proton particle irradiation with an average flux of $2 \cdot 10^5$ $s^{-1} \cdot cm^{-2}$ and a total fluence of at least $2 \cdot 10^{10}$ protons. Apparently parallel lines only result from slight value variations but clearly indicate the precision of the used measurement instruments. No values beyond the depicted ranges were observed.

Therefore, a beam test was prepared in December 2010 at COSY in Jülich, Germany, to investigate the radiation impact on a COTS switching mode power regulator. The selected device, a Texas Instruments PTH05010 [547] 15 A 5.0 V input power module, was tied to 3.3 V for operation of a simple ceramic resistor. Output voltage as well as input current of the DC/DC have been logged during irradiation with 2.4 GeV proton particles at fluxes of at least 10^4 $\cdot s^{-1} \cdot cm^{-2}$ for 22 h and 10^6 $\cdot s^{-1} \cdot cm^{-2}$ for 5 h. This results in an average flux of $2 \cdot 10^5$ $s^{-1} \cdot cm^{-2}$ and a total fluence of at least $2 \cdot 10^{10}$ protons. The fluence is given "at least", because due to shift work, there was a time period of additional irradiation without any recorded particle flux. The beam test results can be found in figure 5.10. As depicted, no significant degradation could be observed, therefore switching POL regulators have been chosen for the SysCore v3 board, which uses Linear Technology LTM4601 and LTM4606EV regulators to generate all required PCB voltages of 1.2 V, 1.5 V, 2.5 V and 3.3 V.

6 Conclusions and Outlook

Summarizing the development process of this thesis, it can certainly be said that the extensive amount of background information required to fully understand all semiconductor and radiation topics as well as the variety of options and exceptions available for error mitigation have been totally underestimated. The various FPGA technologies, combined with the wide range of nowadays available and continuously shrinking transistor feature sizes and the resulting challenges make all this a highly complex and never ending topic.

When focusing on the hardware aspect only, the fundamental causes of temporary and permanent ionizing and non-ionizing radiation effects in semiconductors are mostly understood, including electron-hole pairs, nuclear scattering, parasitic sub-surface silicon thyristors, back-channel leakage current, thermally caused shorts or NBT stress. They can be influenced by modification of many different variables, such as silicon doping, transistor type, node capacitance, node size, operating voltage, isolation and even natural radioactive contamination in the package material. Therefore, device sensitivity assurance has to be given on a statistical basis, determined by independently testing a number of devices, including statements about waver manufacturing, vendor packaging and finally the chip-layout design itself. Consistent values are normally assumed for a single manufacturing batch only, which makes this an extremely expensive procedure.

The ongoing process of CMOS feature size scaling, following Moore's law, modifies most of these variables and therefore raises radiation sensitivity, mainly due to the decrease of operation voltage, reduction of sensitive node distance or thinning of the gate electrode in the upcoming FinFET. This sensitivity becomes visible even on ground level during normal operation in terms of SBUs, an increased number of MBUs as well as boosted aging degradation [548]. In consequence, all chip vendors have to address this issue sooner or later, as done by Intel [549]. A suitable error mitigation method for the most recent FinFET CMOS for example would be to operate them in Tri-Gate Multi-Mode which simply merges three independent gates to increase signal strength and resilience. Beside of radiation effects, additional hardware fault tolerance may also mitigate security issues such as chip attacks for RSA crypto compromising [550].

FPGA manufacturers have intensified the development of devices that can deal with radiation stress, but production quantities are still low, prices are high and

the technology node is mostly years behind recent generations. In parallel, new error-resilient storage technologies that may replace SRAM cells in the near future are evaluated – promising candidates are FRAM, MRAM, CRAM and RRAM. At least FRAM has been tested in an ionizing particle beam within the scope of this thesis. But today's high performance FPGAs still rely on radiation susceptible 6-transistor SRAM cells, therefore error mitigation remains an essential task. While spontaneously occurring SBUs are nowadays recovered predominantly in the firmware layer of COTS FPGAs, and rarely with embedded hardware support, the growing issue of MBUs has been addressed in very few devices and components only. The mitigation of such errors is mostly related to embedded, high density memory arrays, utilizing storage cell interleaving to finally break down an uncorrectable MBU into several correctable SBUs. Additional research focuses on the improvement of SRAM cell design itself, trying to eliminate the single point of failure within these cells where a sole transistor is able to flip the whole cell's state. As soon as this technology is adapted for conventional COTS devices, there is no need for any further SRAM fault tolerance on the firmware level. This tendency may have already been recognized by most of the TMR tool vendors and answered with the discontinuation of their specific products. But from the current point of view, it cannot be said when this will happen, therefore errors will continue to emerge in critical radiation environments and fault tolerance remains a necessary design task.

While data refresh of the static LUT and routing configuration in a Xilinx SRAM FPGA can be handled independently from the actual design by operating an internal or external scrubbing unit, dynamic data held in flip-flops, distributed memory and embedded block memory arrays has to be specifically protected against SBUs by considering different firmware design techniques with a combination of spatial, temporal and information redundancy, maybe completed by a partitioning approach [551]. These types of redundancy furthermore became inevitable for bypassing temporary miscalculations as appearing between two configuration scrubbing cycles. Otherwise, internal state machines or output data generators may be corrupted, which finally can result in the failure of a complex running system. But beside of all benefits when using fault tolerance on firmware level, major drawbacks are inevitable. In addition to the significantly increased resource requirements of redundant firmware, HDL designs are composed of multiple functional components which follow a strictly defined timing specification. This timing becomes obsolete after introduction of TMR voting circuits and signal feedback paths and therefore may cause the whole system synthesis to fail due to a single delay in a critical de-

sign path. Removal of these elements on the other hand results in weak overall error mitigation performance and therefore should be urgently avoided. In such a case, a DMR approach with global reset or error flagging might provide better results. But plain DMR should always be used with caution, as it increases the device cross-section without providing any data recovery.

In summary it can be said, that manual fault-tolerant firmware design on RTL is a complex and time consuming process which generates additional documentation and simulation effort and therefore increases development cost and time to market, which might be a show-stopper for the most of the companies. In contrast, the few available, *automated* tools which may speed up this process are operating on netlist level and *do not offer* fine-grained options about mitigation techniques or critical design paths and therefore complicate simulation and transparency. The use of third-party LogiCORE black box netlist modules may reduce this effort, but does not cover custom logic parts. To solve all of these issues, the development of a *semi-automatic* tool that introduces fault tolerance on RTL for multiple HDLs has been started. Beside of the user-interactive coordination of mitigation strategies for various vulnerable design components, it has to take care of introducing synthesis-tool specific instructions to prevent design minimization and component reuse to achieve timing closure for the whole system but that finally would counteract all previously added redundancy. At the current moment, the tool did not reach a production state and therefore requires additional development. An integrated Hamming FSM generator module provides full sets of state and transition definitions that cannot be upset due to a SBU. Its efficiency has been tested in an ionizing particle beam. In addition, a set of fault-tolerant system components and tools has been developed, which can be used for integration in custom designs, scrubbing bitfile handling as well as radiation assessment. Some of them have also been validated in a particle beam test.

As illustrated, explained and proven in the context of this thesis, the operation of currently available COTS SRAM FPGAs in ionizing radiation environments requires the team play of multiple error mitigation technologies that are available on different system layers. Only this approach successfully introduces *System-wide fault tolerance for FPGAs*. The proper selection of a suitable *radiation-tolerant* hardware base, such as the SysCore development board, that can deal with the expected amount of ionizing and non-ionizing radiation stress to provide stable powering and operation, almost decides on victory or defeat. All overlying layers then have to deal with the arising errors by implementing *fault tolerance*. Whether configu-

ration scrubbing, design redundancy, memory refresh or information redundancy – none would be as efficient without the others and every new layer builds on the efficiency of all the underlying ones. Software fault tolerance may complete this chain on the top, posing other issues, but has not been primarily addressed in this thesis and therefore shows only state of the art.

This study has shown many advantages of using COTS SRAM FPGAs in ionizing radiation environments, especially for the CBM experiment at the GSI/FAIR particle accelerator in Darmstadt, Germany. But it has also proven that the whole system is only as stable as its weakest part. It is fairly difficult, although not impossible, to meet all specific requirements of a fault-tolerant system. In particular the improvement of error mitigation characteristics via RTL design in modern HDLs has proven to be a challenging task, as the vendor's synthesis tools vehemently try to minimize all such efforts in adding redundant logic.

Concerning the continuous improvements in silicon manufacturing, technology substitution, makro cell design, firmware optimization as well as the general tendency towards modern HLS, nearly all subjects referenced in this thesis seem to remain an ongoing and highly interesting topic of research, as for sure "it would be illogical to assume that all conditions remain stable."

Appendices

Appendix A

Neutron MTBU for Xilinx FPGAs

This table reflects the Mean Time Between Upset (MTBU) for recent Xilinx FPGAs while separating CLB configuration memory and on-chip BRAM. All device information have been taken from the following datasheets: Virtex-4 [459, 130], Virtex-5 [460, 131], Spartan-6 [458, 552], Virtex-6 [461, 553]. CLB sizes have been taken from device-specific data2mem output (see section 4.1.3). Neutron cross-sections have been taken from table 2.1. FIT and MTBU rates have been calculated by using a neutron flux at sea level of $\Phi = 4 \cdot 10^{-3}$ $s^{-1}cm^{-2}$ according to [304, 191]. The total FPGA programming bitfile is composed of the CLB and BRAM configuration bits as well as additional command protocol overhead.

Device	Total Configuration Bits		Device FIT		MTBU [years]	
Name	CLB	BRAM	CLB	BRAM	CLB	BRAM
Virtex-4						
XC4VLX15	3705088	884736	827	349	138	327
XC4VLX25	6250368	1327104	1395	524	82	218
XC4VLX40	10181120	1769472	2272	698	50	164
XC4VLX60	14295552	2949120	3191	1164	36	98
XC4VLX80	19024000	3686400	4246	1455	27	78
XC4VLX100	25599744	4423680	5714	1745	20	65
XC4VLX160	34227456	5308416	7640	2094	15	55
XC4VLX200	44240640	6193152	9875	2444	12	47
XC4VSX25	6407808	2359296	1430	931	80	123
XC4VSX35	9611712	3538944	2145	1396	53	82
XC4VSX55	15964416	5898240	3563	2327	32	49
XC4VFX12	3705088	663552	827	262	138	436
XC4VFX20	5510400	1253376	1230	495	93	231
XC4VFX40	11351424	2654208	2534	1047	45	109

continued on next page

continued from previous page

Device	Total Configuration Bits		Device FIT		MTBU [years]	
Name	CLB	BRAM	CLB	BRAM	CLB	BRAM
XC4VFX60	15565568	4276224	3474	1687	33	68
XC4VFX100	24600000	6930432	5491	2734	21	42
XC4VFX140	35691648	10174464	7966	4014	14	28
Virtex-5						
XC5VLX30	7011328	1179648	676	673	169	170
XC5VLX50	10516992	1769472	1015	1009	113	113
XC5VLX85	17790720	3538944	1716	2018	67	57
XC5VLX110	23720960	4718592	2289	2691	50	42
XC5VLX155	32957440	7077888	3180	4036	36	28
XC5VLX220	45048832	7077888	4346	4036	26	28
XC5VLX330	67573248	10616832	6519	6054	18	19
XC5VLX20T	4723200	958464	456	547	251	209
XC5VLX30T	7336704	1327104	708	757	161	151
XC5VLX50T	11005056	2211840	1062	1261	108	91
XC5VLX85T	18278784	3981312	1764	2270	65	50
XC5VLX110T	24371712	5455872	2351	3111	49	37
XC5VLX155T	33608192	7815168	3243	4457	35	26
XC5VLX220T	45699584	7815168	4409	4457	26	26
XC5VLX330T	68549376	11943936	6614	6811	17	17
XC5VSX35T	9299456	3096576	897	1766	127	65
XC5VSX50T	13949184	4866048	1346	2775	85	41
XC5VSX95T	24938496	8994816	2406	5129	47	22
XC5VSX240T	57402624	19021824	5538	10847	21	11
XC5VTX150T	33167360	8404992	3200	4793	36	24
XC5VTX240T	51608832	11943936	4979	6811	23	17
XC5VFX30T	9467392	2506752	913	1429	125	80
XC5VFX70T	18934784	5455872	1827	3111	62	37
XC5VFX100T	27268608	8404992	2631	4793	43	24
XC5VFX130T	34085760	10985472	3289	6264	35	18
XC5VFX200T	48648960	16809984	4694	9586	24	12
Spartan-6						
XC6SLX4	4218240	221184	607	70	188	1629

continued on next page

continued from previous page

Device	Total Configuration Bits		Device FIT		MTBU [years]	
Name	CLB	BRAM	CLB	BRAM	CLB	BRAM
XC6SLX9	4218240	589824	607	187	188	611
XC6SLX16	6190080	589824	891	187	128	611
XC6SLX25	10535200	958464	1517	304	75	376
XC6SLX25T	10535200	958464	1517	304	75	376
XC6SLX45	18903040	2138112	2722	677	42	169
XC6SLX45T	18903040	2138112	2722	677	42	169
XC6SLX75	31998720	3170304	4608	1004	25	114
XC6SLX75T	31998720	3170304	4608	1004	25	114
XC6SLX100	42232320	4939776	6081	1565	19	73
XC6SLX100T	42232320	4939776	6081	1565	19	73
XC6SLX150	56659200	4939776	8159	1565	14	73
XC6SLX150T	56659200	4939776	8159	1565	14	73
Virtex-6						
XC6VHX250T	57915648	18579456	10508	3050	11	37
XC6VHX255T	57915648	19021824	10508	3123	11	37
XC6VHX380T	86873472	28311552	15762	4648	7	25
XC6VHX565T	121772160	33619968	22094	5519	5	21
XC6VLX75T	19237824	5750784	3491	944	33	121
XC6VLX130T	32063040	9732096	5818	1598	20	71
XC6VLX195T	46578240	12681216	8451	2082	14	55
XC6VLX240T	55893888	15335424	10141	2517	11	45
XC6VLX365T	78102144	15335424	14171	2517	8	45
XC6VLX550T	117153216	23298048	21256	3825	5	30
XC6VLX760	154897920	26542080	28105	4357	4	26
XC6VSX315T	74556288	25952256	13527	4260	8	27
XC6VSX475T	111834432	39223296	20291	6439	6	18

Appendix B

Parbitgen conversion log

Custom parbitgen tool conversation and modification log while generating a partial reconfiguration bitfile for a Xilinx 4VFX20FF672 device. Every 32 bit word from the input file's configuration header is indicated with its absolute position, value and semantics. Modified words are asterisked and inserted immediately at the position of occurrence.

Position	*DataWord	Semantics
00000075	FFFFFFFF	DUMMY WORD
00000079	AA995566	SYNC WORD
0000007D	20000000	TYPE-1 NOP
00000081	30008001	TYPE-1 Write 1 words to register CMD (Command Register)
00000085	00000007	Command: RCRC
00000089	20000000	TYPE-1 NOP
0000008D	20000000	TYPE-1 NOP
00000091	30012001	TYPE-1 Write 1 words to register COR (Configuration Option Register)
00000095	01043FE5	Data: 01043FE5
		CRC_BYPASS: 0 (CRC enabled)
		DONE_PIPE: 0 (No pipeline statue for DONEIN)
		DRIVE_DONE: 1 (DONE is actively driven High)
		SINGLE: 0 (Read-back is not single-shot)
		OSCFSEL: 000010 (Select CCLK frequency in Master configuration modes)
		SSCLKSRC: 00 (CCLK)
		DONE_CYCLE: 011 (Startup cycle 4)
		MATCH_CYCLE: 111 (No Wait)
		LOCK_CYCLE: 111 (No Wait)

continued on next page

continued from previous page

Position	*DataWord	Semantics
		GTS_CYCLE: 100 (Startup cycle 5)
		GWE_CYCLE: 101 (Startup cycle 6)
00000099	30018001	TYPE-1 Write 1 words to register IDCODE (Device ID Register)
0000009D	01E64093	Data: 01E64093
		Revision Code: 0x0
		Family Code: b0001111 (Virtex 4 FX)
		Device Size: 0x064 (sum of device rows and columns)
		Manufacturer ID: 0x049
		JTAG IDCODE: 0x1 (always 1 to conform to the JTAG IDCODE specification)
000000A1	30008001	TYPE-1 Write 1 words to register CMD (Command Register)
000000A5	00000009	Command: SWITCH
000000A9	20000000	TYPE-1 NOP
000000AD	3000C001	TYPE-1 Write 1 words to register MASK (Masking Register for CTL)
000000AD	*20000000	TYPE-1 NOP
000000B1	00000600	Data: 00000600
000000B1	*20000000	TYPE-1 NOP
000000B5	3000A001	TYPE-1 Write 1 words to register CTL (Control Register)
000000B5	*20000000	TYPE-1 NOP
000000B9	00000600	Data: 00000600
		ICAP_SEL: 0 (Top ICAP Port Enabled)
		SBITS: 00 (Read/Write OK)
		PERSIST: 0 (No)
		GLUTMASK: 0 (Readback all 0s from SRL16 and Distributed RAM. (Active Device Readback)
		GTS_USER_B: 0 (I/Os placed in high-Z state)
000000B9	*20000000	TYPE-1 NOP

continued on next page

continued from previous page

Position	*DataWord	Semantics
		[000000BD to 000012B1 TYPE-1 NOP]
000012B5	3000C001	TYPE-1 Write 1 words to register MASK (Masking Register for CTL)
000012B5	*20000000	TYPE-1 NOP
000012B9	00000600	Data: 00000600
000012B9	*20000000	TYPE-1 NOP
000012BD	3000A001	TYPE-1 Write 1 words to register CTL (Control Register)
000012BD	*20000000	TYPE-1 NOP
000012C1	00000000	Data: 00000000 ICAP_SEL: 0 (Top ICAP Port Enabled) SBITS: 00 (Read/Write OK) PERSIST: 0 (No) GLUTMASK: 0 (Readback all 0s from SRL16 and Distributed RAM. (Active Device Readback) GTS_USER_B: 0 (I/Os placed in high-Z state)
000012C1	*20000000	TYPE-1 NOP
000012C5	30008001	TYPE-1 Write 1 words to register CMD (Command Register)
000012C9	00000000	Command: NULL
000012CD	20000000	TYPE-1 NOP
000012D1	30002001	TYPE-1 Write 1 words to register FAR (Frame Address Register)
000012D5	00000000	Data: 00000000 Top/Bottom Bit: 0 Block Type: 000 (CLB/IO/DSP/CLK/MGT) Row Address: 0x00 Column Address: 0x00 Minor Address: 0x00
000012D9	30008001	TYPE-1 Write 1 words to register CMD (Command Register)
000012DD	00000001	Command: WCFG

continued on next page

continued from previous page

Position	*DataWord	Semantics
000012E1	20000000	TYPE-1 NOP
000012E5	30004000	TYPE-1 Write 0 words to register FDRI (Frame Data Register Input)
000012E9	50036EF0	TYPE-2 Write 225008 words
000012E9	*5002A0A8	TYPE-2 Write 172200 words
000DCEAD	30000001	TYPE-1 Write 1 words to register CRC (CRC Register)
000DCEAD	*20000000	TYPE-1 NOP
000DCEB1	2AEA9E25	Data: 2AEA9E25
000DCEB1	*20000000	TYPE-1 NOP
000DCEB5	30008001	TYPE-1 Write 1 words to register CMD (Command Register)
000DCEB9	0000000A	Command: GRESTORE
000DCEB9	*00000000	Command: NULL
000DCEBD	20000000	TYPE-1 NOP
000DCEC1	30008001	TYPE-1 Write 1 words to register CMD (Command Register)
000DCEC5	00000003	Command: DGHIGH/LFRM
000DCEC5	*00000000	Command: NULL
		[000DCEC9 to 000DD055 TYPE-1 NOP]
000DD059	30008001	TYPE-1 Write 1 words to register CMD (Command Register)
000DD05D	0000000A	Command: GRESTORE
000DD05D	*00000000	Command: NULL
000DD061	20000000	TYPE-1 NOP
000DD065	30008001	TYPE-1 Write 1 words to register CMD (Command Register)
000DD069	00000000	Command: NULL
000DD06D	20000000	TYPE-1 NOP
000DD071	30002001	TYPE-1 Write 1 words to register FAR (Frame Address Register)
000DD075	00008AC0	Data: 00008AC0

continued on next page

continued from previous page

Position	*DataWord	Semantics
		Top/Bottom Bit: 0
		Block Type: 000 (CLB/IO/DSP/CLK/MGT)
		Row Address: 0x02
		Column Address: 0x2B
		Minor Address: 0x00
000DD079	30008001	TYPE-1 Write 1 words to register CMD (Command Register)
000DD07D	00000005	Command: START
000DD07D	*00000000	Command: NULL
000DD081	20000000	TYPE-1 NOP
000DD085	3000C001	TYPE-1 Write 1 words to register MASK (Masking Register for CTL)
000DD085	*20000000	TYPE-1 NOP
000DD089	00000008	Data: 00000008
000DD089	*20000000	TYPE-1 NOP
000DD08D	3000A001	TYPE-1 Write 1 words to register CTL (Control Register)
000DD08D	*20000000	TYPE-1 NOP
000DD091	00000008	Data: 00000008
		ICAP_SEL: 0 (Top ICAP Port Enabled)
		SBITS: 00 (Read/Write OK)
		PERSIST: 1 (Yes)
		GLUTMASK: 0 (Readback all 0s from SRL16 and Distributed RAM. (Active Device Readback)
		GTS_USER_B: 0 (I/Os placed in high-Z state)
000DD091	*20000000	TYPE-1 NOP
000DD095	30000001	TYPE-1 Write 1 words to register CRC (CRC Register)
000DD095	*20000000	TYPE-1 NOP
000DD099	F6CC7F47	Data: F6CC7F47
000DD099	*20000000	TYPE-1 NOP
000DD09D	30008001	TYPE-1 Write 1 words to register CMD (Command Register)

continued on next page

continued from previous page

Position	*DataWord	Semantics
000DD0A1	0000000D	Command: DESYNC
		[000DD0A5 to 000DD0E1 TYPE-1 NOP]

Bibliography

[1] H. Garrett, I. Katz, I. Jun, W. Kim, A. Whittlesey, and R. Evans, "The Jovian Charging Environment and Its Effects - A Review," *Plasma Science, IEEE Transactions on*, vol. 40, no. 2, pp. 144–154, Feb 2012.

[2] E. Stassinopoulos and J. P. Raymond, "The Space Radiation Environment for Electronics," *Proceedings of the IEEE*, vol. 76, no. 11, pp. 1423–1442, Nov 1988.

[3] H. Quinn, P. Graham, K. Morgan, Z. Baker, M. Caffrey, D. Smith, M. Wirthlin, and R. Bell, "Flight Experience of the Xilinx Virtex-4," in *Radiation and Its Effects on Components and Systems (RADECS), 2012 13th European Conference on*, Sept 2012, pp. 1–4.

[4] ——, "Flight Experience of the Xilinx Virtex-4," *Nuclear Science, IEEE Transactions on*, vol. 60, no. 4, pp. 2682–2690, Aug 2013.

[5] C. Bergin, "Dragon enjoying ISS stay, despite minor issues – Falcon 9 investigation begins," NASASpaceflight, Oct. 2012.

[6] M. A. Shea and D. F. Smart, "Recent and Historical Solar Proton Events," *Radiocarbon*, vol. 34, no. 2, pp. 255–262, 1992.

[7] C. Dyer, "Space Weather: Past, Present and Future," Sept 2013, invited Talk at RADECS 2013 Conference.

[8] Bundesamt für Strahlenschutz, Öffentlichkeitsarbeit, "Strahlung, Strahlenschutz - Eine Information des Bundesamtes für Strahlenschutz, 4. Auflage," 2008.

[9] R. C. Baumann, "Determining the impact of alpha-particle-emitting contamination from the Fukushima Daiichi disaster on Japanese manufacturing sites," in *Radiation and Its Effects on Components and Systems (RADECS), 2011 12th European Conference on*, Sept 2011, pp. 784–787.

[10] D. M. Harland and R. Lorenz, *Space Systems Failures*. Springer, 2005.

[11] D. Gabel, "Bor-Neutroneneinfangtherapie von Tumoren," *Chemie in unserer Zeit*, vol. 31, no. 5, pp. 235–240, 1997.

[12] M. Durante, C. Trautmann, V. Ferlet-Cavrois, A. Menicucci, O. Angerer, and E. Daly, "Application of FAIR for Space Radiation Investigations," RADECS 2013 Conference Data Workshop DW-18, 2013.

[13] A. Bracco, P. Chomaz, J. J. Gaardhøje, P.-H. Heenen, G. Rosner, E. Widmann, and G.-E. Körner, Eds., *NuPECC Long Range Plan 2010: Perspectives of Nuclear Physics in Europe*. European Science Foundation, Nov. 2010.

[14] H. Stoecker and C. Sturm, "FAIR - The Facility for Antiproton and Ion Research," *Romanian Journal of Physics*, vol. 58, no. 9-10, pp. 1023–1030, 2013.

[15] M. Pfalz, "Antiprotonen unter Beschuss," *Physik Journal, Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim*, vol. 14, no. 7, pp. 6–7, Jul. 2015.

[16] ——, "FAIR bleibt fair," *Physik Journal, Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim*, vol. 14, no. 11, pp. 11–11, Nov. 2015.

[17] I. A. Shovkovy, "Two Lectures on Color Superconductivity," *Foundations of Physics*, vol. 35, no. 8, pp. 1309–1358, Aug. 2005.

[18] B. Friman, C. Höhne, J. Knoll, S. Leupold, J. Randrup, R. Rapp, and P. Senger, Eds., *The CBM Physics Book: Compressed Baryonic Matter in Laboratory Experiments*, ser. Lecture Notes in Physics. Springer-Verlag Berlin Heidelberg, 2011, vol. 814.

[19] The CBM Collaboration, "Technical Design Report for the CBM Superconducting Dipole Magnet," GSI Darmstadt, Tech. Rep., 2012.

[20] R. Turchetta, J. Berst, B. Casadei, G. Claus, C. Colledani, W. Dulinski, Y. Hu, D. Husson, J. L. Normand, J. Riester, G. Deptuch, U. Goerlach, S. Higueret, and M. Winter, "A monolithic active pixel sensor for charged particle tracking and imaging using standard VLSI CMOS technology," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 458, no. 3, pp. 677–689, 2001.

[21] The CBM Collaboration, "Technical Design Report for the CBM Silicon Tracking System (STS)," GSI Darmstadt, Tech. Rep., 2013.

[22] ——, "Technical Design Report for the CBM Ring Imaging Cherenkov (RICH) Detector," GSI Darmstadt, Tech. Rep., 2013.

[23] ——, "Technical Design Report for the CBM Muon Chambers (MuCh)," GSI Darmstadt, Tech. Rep., 2014.

[24] D. Emschermann, "TRD status update ," Universität Münster, Institut für Kernphysik, Tech. Rep., Apr. 2014, 23rd CBM Collaboration Meeting.

[25] The CBM Collaboration, "Technical Design Report for the CBM Time-of-Flight Detector (ToF)," GSI Darmstadt, Tech. Rep., 2014.

[26] C. Lippmann and W. Riegler, "Detector Physics of Resistive Plate Chambers," in *Nuclear Science Symposium Conference Record, 2002 IEEE*, vol. 1, Nov 2002, pp. 600–604 vol.1.

[27] S. Manz, "Radiation Mitigation for SRAM-Based FPGAs in the CBM Experiment," Ph.D. dissertation, Goethe University Frankfurt, 2015.

[28] The CBM Collaboration, "Technical Design Report for the CBM Projectile Spectator Detector (PSD)," GSI Darmstadt, Tech. Rep., 2014.

[29] "Pre-Construction Memorandum of Understanding for the Execution of the Initial Phase of the Compressed Baryonic Matter (CBM) Experiment at FAIR – Annex 9," Mar. 2012.

[30] The CBM Collaboration, "Compressed Baryonic Matter Experiment – Technical Status Report," GSI Darmstadt, Tech. Rep., Jan. 2005.

[31] W. F. J. Müller, "The CBM Experiment @ FAIR – New challenges for Front-End Electronics, Data Acquisition and Trigger Systems," *Journal of Physics: Conference Series*, vol. 50, no. 1, p. 371, 2006.

[32] F. Lemke, "Unified Synchronized Data Acquisition Networks," Ph.D. dissertation, Universität Mannheim, 2012.

[33] P. Schäfer, "BSc Thesis: Synchronization of Front-End Electronics in a Data Acquisition Interconnection Network," Master's thesis, University of Heidelberg, 2012.

[34] W. F. J. Müller, "HUB vs GBTx," CBM FEE/DAQ Workshop, Feb. 2014.

[35] J. de Cuveland, V. Lindenstruth, and the CBM Collaboration, "A First-level Event Selector for the CBM Experiment at FAIR," *Journal of Physics: Conference Series*, vol. 331, no. 2, 2011.

[36] A. Chaudhry, "Nanoscale Effects: Gate Oxide Leakage Currents," in *Fundamentals of Nanoscaled Field Effect Transistors*. Springer New York, 2013, pp. 25–36.

[37] M. Santarini, "Xilinx 16nm UltraScale+ Devices Yield 2-5X Performance/Watt Advantage," *Xilinx Xcell Journal*, no. 90, pp. 8–15, Feb. 2015.

[38] *White Paper 1201: The Breakthrough Advantage for FPGAs with Tri-Gate Technology*, Altera Corporation, Jun. 2013.

[39] *Arria 10 Device Overview*, Altera Corporation, Sep. 2014.

[40] *White Paper WP003 v1.0: Challenges of High-Speed Ethernet at 100G, 400G and Beyond - The Spacetime Perspective*, Tabula, Mar. 2013.

[41] *Achronix Speedster22i Product Brief v2.7*, Achronix, Dec. 2014.

[42] C.-H. Lin, R. Kambhampati, R. Miller, T. Hook, A. Bryant, W. Haensch, P. Oldiges, I. Lauer, T. Yamashita, V. Basker, T. Standaert, K. Rim, E. Leobandung, H. Bu, and M. Khare, "Channel doping impact on FinFETs for 22nm and beyond," in *VLSI Technology (VLSIT), 2012 Symposium on*, June 2012, pp. 15–16.

[43] M. Poljak, V. Jovanovic, and T. Suligoj, "SOI vs. bulk FinFET: Body doping and corner effects influence on device characteristics," in *Electrotechnical Conference, 2008. MELECON 2008. The 14th IEEE Mediterranean*, May 2008, pp. 425–430.

[44] D. James, "Intel Ivy Bridge unveiled - The first commercial tri-gate, high-k, metal-gate CPU," in *Custom Integrated Circuits Conference (CICC), 2012 IEEE*, Sept 2012, pp. 1–4.

[45] S. Tawfik, Z. Liu, and V. Kursun, "Independent-gate and tied-gate FinFET SRAM Circuits: Design guidelines for reduced area and enhanced stability," in *Microelectronics, 2007. ICM 2007. Internatonal Conference on*, Dec 2007, pp. 171–174.

[46] R. A. Wolkow, L. Livadaru, J. Pitters, M. Taucer, P. Piva, M. Salomons, M. Cloutier, and B. V. C. Martins, "Silicon Atomic Quantum Dots Enable Beyond-CMOS Electronics," in *Field-Coupled Nanocomputing*, ser. Lecture

Notes in Computer Science, N. G. Anderson and S. Bhanja, Eds. Springer Berlin Heidelberg, 2014, pp. 33–58.

[47] F. Wessely, "CMOS ohne Dotierstoffe: Neuartige siliziumbasierte Nanodraht-Feldeffekt-Bauelemente," Ph.D. dissertation, Technischen Universität Darmstadt, 2011.

[48] M. Casu and P. Flatresse, "History effect characterization in PD-SOI CMOS gates," in *SOI Conference, IEEE International 2002*, Oct 2002, pp. 62–63.

[49] L. Dreeskornfeld, J. Hartwich, E. Landgraf, H. Luyken, W. Rosner, T. Schulz, M. Stadele, D. Schmitt-Landsiedel, and L. Risch, "Comparison of Partially and Fully Depleted SOI Transistors Down to the Sub 50nm Gate Length Regime," The Electrochemical Society (ECS), 203rd Meeting, J3, Tech. Rep., Apr. 2003.

[50] R. Hulfachor, K. Kim, M. Littlejohn, and C. Osburn, "Comparative Analysis of Hot Electron Injection and Induced Device Degradation in Scaled 0.1 μm SOI n-MOSFET's Using Monte Carlo Simulation," *Electron Device Letters, IEEE*, vol. 17, no. 2, pp. 53–55, Feb 1996.

[51] T. Sakurai, A. Matsuzawa, and T. Douseki, *Fully-Depleted SOI CMOS Circuits and Technology for Ultralow-Power Applications*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[52] T. Liu, W. Chen, P. Gui, C.-A. Yang, J. Zhang, P. Zhu, A. Xiang, J. Ye, and R. Stroynowski, "Total ionization dose effects and single-event effects studies of a 0.25 μm Silicon-On-Sapphire CMOS technology," in *Radiation and Its Effects on Components and Systems, 2007. RADECS 2007. 9th European Conference on*, Sept 2007, pp. 1–5.

[53] J. Tomayko, *Computers in Spaceflight: The NASA Experience*. National Aeronautics and Space Administration, Scientific and Technical Information Division, 1988.

[54] E. Fluhr, J. Friedrich, D. Dreps, V. Zyuban, G. Still, C. Gonzalez, A. Hall, D. Hogenmiller, F. Malgioglio, R. Nett, J. Paredes, J. Pille, D. Plass, R. Puri, P. Restle, D. Shan, K. Stawiasz, Z. Deniz, D. Wendel, and M. Ziegler, "5.1 POWER8TM: A 12-core server-class processor in 22nm SOI with 7.6Tb/s off-chip bandwidth," in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*, Feb 2014, pp. 96–97.

[55] H. Lee, C.-H. Lee, D. Park, and Y.-K. Choi, "A study of negative-bias temperature instability of SOI and body-tied FinFETs," *Electron Device Letters, IEEE*, vol. 26, no. 5, pp. 326–328, May 2005.

[56] W. G. Bennett, "Single Event Upset Mechanisms in Emerging Memory Technologies," Ph.D. dissertation, Vanderbilt University, Department of Electrical Engineering, 2014.

[57] D. Tang, Y. Li, G. Zhang, C. He, and Y. Fan, "Single event upset sensitivity of 45 nm FDSOI and SOI FinFET SRAM," *Science China Technological Sciences*, vol. 56, no. 3, pp. 780–785, 2013.

[58] F. El Mamouni, "Single-event-transient effects in sub-70 nm bulk and SOI FinFETs," Ph.D. dissertation, Vanderbilt University, Department of Electrical Engineering, 2012.

[59] Y. C. Wang, *Xilinx User Guide UG116 v10.1: Device Reliability Report Second Quarter 2014*, Aug. 2014.

[60] *Xilinx Product Specification DS180 v1.16.1: 7 Series FPGAs Overview*, Xilinx, Inc., Dec. 2014.

[61] A. Telikepalli, *Xilinx White Paper WP223 v1.2: Power vs. Performance: The 90 nm Inflection Point*, May 2006.

[62] J. D. Cressler and H. A. Mantooth, Eds., *Extreme Environment Electronics*. CRC Press, 2013, ch. 73: Radiation-Hard Multichannel Digitizer ASIC for Operation in the Jovian Environment, pp. 849–862.

[63] M. Klein, *Xilinx White Paper WP298 v1.0: Power Consumption at 40 and 45 nm*, Apr. 2009.

[64] A. Lesea, S. Drimer, J. Fabula, C. Carmichael, and P. Alfke, "The Rosetta Experiment: Atmospheric Soft Error Rate Testing in Differing Technology FPGAs," *Device and Materials Reliability, IEEE Transactions on*, vol. 5, no. 3, pp. 317–328, Sept 2005.

[65] N. Rollins, M. Wirthlin, M. Caffrey, and P. Graham, "Reliability of Programmable Input/Output Pins in the Presence of Configuration Upsets," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, Sep. 2002.

[66] S. Platt, Z. Torok, C. Frost, and S. Ansell, "Charge-collection and single-event upset measurements at the ISIS neutron source," in *Radiation and Its Effects on Components and Systems, 2007. RADECS 2007. 9th European Conference on*, Sept 2007, pp. 1–6.

[67] E. Normand, K. Vranish, A. Sheets, M. Stitt, and R. Kim, "Quantifying the Double-Sided Neutron SEU Threat, From Low Energy (Thermal) and High Energy (> 10 MeV) Neutrons," *Nuclear Science, IEEE Transactions on*, vol. 53, no. 6, pp. 3587–3595, Dec 2006.

[68] M. Olmos, R. Gaillard, A. Van Overberghe, J. Beaucour, S. Wen, and S. Chung, "Investigation of Thermal Neutron Induced Soft Error Rates in Commercial SRAMs with 0.35 µm to 90 nm Technologies," in *Reliability Physics Symposium Proceedings, 2006. 44th Annual., IEEE International*, March 2006, pp. 212–216.

[69] R. P. Preston, *Design of High-Performance Microprocessor Circuits*, 1st ed. Wiley-IEEE Press, 2001, ch. Register Files and Caches, pp. 284–308.

[70] L. Walker, J. Manoliu, and R. Rung, "Four-Transistor Static CMOS Memory Cells," in *Electron Devices Meeting, 1977 International*, vol. 23, 1977, pp. 402–405.

[71] J. Yang and L. Chen, "A New Loadless 4-Transistor SRAM Cell with a 0.18 µm CMOS Technology," in *Electrical and Computer Engineering, 2007. CCECE 2007. Canadian Conference on*, April 2007, pp. 538–541.

[72] K. Ma, H. Liu, Y. Xiao, Y. Zheng, X. Li, S. Kumar Gupta, Y. Xie, and V. Narayanan, "Independently-Controlled-Gate FinFET 6T SRAM Cell Design for Leakage Current Reduction and Enhanced Read Access Speed," in *VLSI (ISVLSI), 2014 IEEE Computer Society Annual Symposium on*, July 2014, pp. 296–301.

[73] T. Song, W. Rim, J. Jung, G. Yang, J. Park, S. Park, Y. Kim, K.-H. Baek, S. Baek, S.-K. Oh, J. Jung, S. Kim, G. Kim, J. Kim, Y. Lee, S.-P. Sim, J. S. Yoon, K.-M. Choi, H. Won, and J. Park, "A 14 nm FinFET 128 Mb SRAM With V_{MIN} Enhancement Techniques for Low-Power Applications," *Solid-State Circuits, IEEE Journal of*, vol. 50, no. 1, pp. 158–169, Jan 2015.

[74] F. Hamzaoglu, U. Arslan, N. Bisnik, S. Ghosh, M. Lal, N. Lindert, M. Meterelliyoz, R. Osborne, J. Park, S. Tomishima, Y. Wang, and K. Zhang, "A 1Gb 2GHz embedded DRAM in 22nm tri-gate CMOS technology," in *Solid-State Circuits*

Conference Digest of Technical Papers (ISSCC), 2014 IEEE International, Feb 2014, pp. 230–231.

[75] "JEDEC SmartBrief 06-21-2013: Can SRAMs be scaled down to 20nm and further?" 2013.

[76] M.-T. Chang, P. Rosenfeld, S.-L. Lu, and B. Jacob, "Technology Comparison for Large Last-Level Caches (L3Cs): Low-Leakage SRAM, Low Write-Energy STT-RAM, and Refresh-Optimized eDRAM," in *High Performance Computer Architecture (HPCA2013), 2013 IEEE 19th International Symposium on*, Feb 2013, pp. 143–154.

[77] V. C. Kumar and B. Underdahl, *Texas Instruments FRAM MCUs for Dummies*. John Wiley & Sons, Inc., Hoboken, New Jersey, 2012.

[78] *AN-21377: FRAM MCU Key Strengths and Applications*, Fujitsu Semiconductor, Sep. 2010.

[79] H. Ishiwara, M. Okuyama, and Y. Arimoto, Eds., *Ferroelectric Random Access Memories: Fundamentals and Applications*. Springer-Verlag Berlin Heidelberg, 2004.

[80] A. B. Kaufman, "An Expandable Ferroelectric Random Access Memory," *Computers, IEEE Transactions on*, vol. C-22, no. 2, pp. 154–158, Feb 1973.

[81] R. Womack and D. Tolsch, "A 16kb Ferroelectric Nonvolatile Memory with a Bit Parallel Architecture," in *Solid-State Circuits Conference, 1989. Digest of Technical Papers. 36th ISSCC., 1989 IEEE International*, Feb 1989, pp. 242–243.

[82] K. Kim, "1T1C FRAM," in *VLSI Technology, Systems, and Applications, 2001. Proceedings of Technical Papers. 2001 International Symposium on*, 2001, pp. 81–84.

[83] S. Masui, W. Yokozeki, M. Oura, T. Ninomiya, K. Mukaida, Y. Takayama, and T. Teramoto, "Design and Applications of Ferroelectric Nonvolatile SRAM and Flip-Flop with Unlimited Read/Program Cycles and Stable Recall," in *Custom Integrated Circuits Conference, 2003. Proceedings of the IEEE 2003*, Sept 2003, pp. 403–406.

[84] *FRAM Guide Book (MN05-00009-6E)*, 6th ed., Fujitsu Semiconductor, Sep. 2010.

[85] D. Takashima and I. Kunishima, "High-Density Chain Ferroelectric Random Access Memory (Chain FRAM)," *Solid-State Circuits, IEEE Journal of*, vol. 33, no. 5, pp. 787–792, May 1998.

[86] S. Drapatz, "Parametric Reliability of 6T-SRAM Core Cell Arrays," Ph.D. dissertation, Technischen Universität München, Feb. 2012.

[87] I. Prejbeanu, W. Kula, K. Ounadjela, R. Sousa, O. Redon, B. Dieny, and J.-P. Nozieres, "Thermally assisted switching in exchange-biased storage layer magnetic tunnel junctions," *Magnetics, IEEE Transactions on*, vol. 40, no. 4, pp. 2625–2627, July 2004.

[88] D. Nguyen and F. Irom, "Radiation effects on MRAM," in *Radiation and Its Effects on Components and Systems, 2007. RADECS 2007. 9th European Conference on*, Sept 2007, pp. 1–4.

[89] J. Heidecker, G. Allen, and D. Sheldon, "Single Event Latchup (SEL) and Total Ionizing Dose (TID) of a 1 Mbit Magnetoresistive Random Access Memory (MRAM)," in *Radiation Effects Data Workshop (REDW), 2010 IEEE*, July 2010, pp. 4–4.

[90] C. Hafer, M. Von Thun, M. Mundie, D. Bass, and F. Sievert, "SEU, SET, and SEFI Test Results of a Hardened 16Mbit MRAM Device," in *Radiation Effects Data Workshop (REDW), 2012 IEEE*, July 2012, pp. 1–4.

[91] G. Tsiligiannis, L. Dilillo, A. Bosio, P. Girard, A. Todri, A. Virazel, S. McClure, A. Touboul, F. Wrobel, and F. Saigne, "Testing a Commercial MRAM Under Neutron and Alpha Radiation in Dynamic Mode," *Nuclear Science, IEEE Transactions on*, vol. 60, no. 4, pp. 2617–2622, Aug 2013.

[92] Y. Guillemenet, L. Torres, G. Sassatelli, N. Bruchon, and I. Hassoune, "A non-volatile run-time FPGA using thermally assisted switching MRAMS," in *Field Programmable Logic and Applications, 2008. FPL 2008. International Conference on*, Sept 2008, pp. 421–426.

[93] O. Goncalves, G. Prenat, and B. Dieny, "Radiation hardened LUT for MRAM-based FPGAs," in *Semiconductor Conference Dresden-Grenoble (ISCDG), 2012 International*, Sept 2012, pp. 33–36.

[94] ——, "Radiation Hardened MRAM-Based FPGA," *Magnetics, IEEE Transactions on*, vol. 49, no. 7, pp. 4355–4358, July 2013.

[95] E. S. Kuh and C.-P. Hsu, *The Best of ICCAD - 20 Years of Excellence in Computer-Aided Design*. Springer US, 2003, ch. Physical Design Overview.

[96] *Proceedings of the 1st IEEE European Design Automation Conference (EURO-DAC)*, Sep. 1992, Hamburg, Germany.

[97] *Proceedings of Design, Automation and Test in Europe Conference (DATE)*, Feb. 1998, Paris, France.

[98] VHSIC Program Office, "Very High Speed Integrated Circuits (VHSIC) Final Program Report 1980-1990," U.S. Department of Defense, Tech. Rep., Sep. 1990.

[99] *IEEE Standard 1076-1987: VHDL Language Reference Manual*, IEEE Standards Board, 1988.

[100] *IEEE Standard 1076-1993: VHDL Language Reference Manual*, IEEE Standards Board, 1994.

[101] *IEEE Standard 1076-2000: VHDL Language Reference Manual*, IEEE-SA Standards Board, 2000.

[102] *IEEE Standard 1076-2002: VHDL Language Reference Manual*, IEEE-SA Standards Board, 2002.

[103] *IEEE Standard 1076-2008: VHDL Language Reference Manual*, IEEE-SA Standards Board, 2009.

[104] *IEEE-SA Standards Board Operations Manual*, The Institute of Electrical and Electronics Engineers, Inc., Mar. 2015.

[105] *IEEE Standard 1076.1-1999: VHDL Analog and Mixed-Signal Extensions*, IEEE-SA Standards Board, 1999.

[106] *IEEE Standard 1076.1-2007: VHDL Analog and Mixed-Signal Extensions*, IEEE-SA Standards Board, 2007.

[107] *IEEE Standard 1364-1995: Verilog Hardware Description Language*, IEEE Standards Board, 1996.

[108] *IEEE Standard 1364-2001: Verilog Hardware Description Language*, IEEE-SA Standards Board, 2001.

[109] *IEEE Standard 1364-2005: Verilog Hardware Description Language*, IEEE-SA Standards Board, 2006.

[110] D. Gajski and R. Kuhn, "Guest Editors' Introduction: New VLSI Tools," *Computer*, vol. 16, no. 12, pp. 11–14, Dec 1983.

[111] R. Walker and D. Thomas, "A Model of Design Representation and Synthesis," in *Design Automation, 1985. 22nd Conference on*, June 1985, pp. 453–459.

[112] *IEEE Standard 1666-2011: Standard SystemC Language Reference Manual*, IEEE-SA Standards Board, 2012.

[113] *MaxCompiler White Paper*, Maxeler Technologies, Feb. 2011.

[114] *Xilinx User Guide UG902 v2014.3: Vivado Design Suite User Guide - High-Level Synthesis*, Xilinx, Inc., Oct. 2014.

[115] G. Lehmann, B. Wunder, and M. Selz, *Schaltungsdesign mit VHDL*. Franzis-Verlag GmbH, Poing, 1994.

[116] *Xilinx User Guide UG695 v14.1: ISE In-Depth Tutorial*, Xilinx, Inc., Apr. 2012.

[117] V. G. Gudise and G. K. Venayagamoorthy, "FPGA Placement and Routing Using Particle Swarm Optimization," in *VLSI, 2004. Proceedings. IEEE Computer society Annual Symposium on*, Feb 2004, pp. 307–308.

[118] E. Stott, Z. Guan, J. Levine, J. Wong, and P. Cheung, "Variation and Reliability in FPGAs," *Design Test, IEEE*, vol. 30, no. 6, pp. 50–59, Dec 2013.

[119] E. L. Lawler, J. K. Lenstra, A. R. Kan, and D. B. Shmoys, *The traveling salesman problem: a guided tour of combinatorial optimization*. Wiley New York, 1985, vol. 3.

[120] G. Frederickson, M. S. Hecht, and C. E. Kim, "Approximation Algorithms for some Routing Problems," in *Foundations of Computer Science, 1976., 17th Annual Symposium on*, Oct 1976, pp. 216–227.

[121] C. Brunelli, F. Garzia, J. Nurmi, F. Campi, and D. Picard, "Reconfigurable hardware: The holy grail of matching performance with programming productivity," in *Field Programmable Logic and Applications, 2008. FPL 2008. International Conference on*, Sept 2008, pp. 409–414.

[122] J. Barreiros and E. Costa, "Global routing for lookup-table based fpgas using genetic algorithms," in *Field Programmable Logic and Application*. Springer, 2003, pp. 141–150.

[123] R. Rubinstein, "The cross-entropy method for combinatorial and continuous optimization," *Methodology And Computing In Applied Probability*, vol. 1, no. 2, pp. 127–190, 1999.

[124] J. S. Kilby, "The integrated circuit's early history," *Proceedings of the IEEE*, vol. 88, no. 1, pp. 109–111, Jan 2000.

[125] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolic, *Digital integrated circuits*. Prentice hall Englewood Cliffs, 2002, vol. 2.

[126] *Intel Datasheet: 4004 Single Chip 4-bit P-Channel Microprocessor*, Intel Corporation, 1977.

[127] *Xilinx Product Specification DS054 v2.5: XC9500XL High-Performance CPLD Family Data Sheet*, Xilinx, Inc., May 2009.

[128] P. Alfke, I. Bolsens, B. Carter, M. Santarini, and S. Trimberger, "It's an FPGA!" *Solid-State Circuits Magazine, IEEE*, vol. 3, no. 4, pp. 15–20, Fall 2011.

[129] H. Engel and U. Kebschull, "Common read-out receiver card for alice run2," *Journal of Instrumentation*, vol. 8, no. 12, p. C12016, 2013.

[130] *Xilinx Product Specification DS112 v3.1: Virtex-4 Family Overview*, Xilinx, Inc., Aug. 2010.

[131] *Xilinx Product Specification DS100 v5.0: Virtex-5 Family Overview*, Xilinx, Inc., Feb. 2009.

[132] *Xilinx Product Specification DS190 v1.7: Zynq-7000 All Programmable SoC Overview*, Xilinx, Inc., Oct. 2014.

[133] *Altera Arria 10 Device Overview*, Altera Corporation, Sep. 2014.

[134] Q. Wang, R. Kassa, W. Shen, N. Ijih, B. Chitlur, M. Konow, D. Liu, A. Sheiman, and P. Gupta, "An FPGA Based Hybrid Processor Emulation Platform," in *Field Programmable Logic and Applications (FPL), 2010 International Conference on*, Aug 2010, pp. 25–30.

[135] A. DeHon, "The Density Advantage of Configurable Computing," *Computer*, vol. 33, no. 4, pp. 41–49, Apr 2000.

[136] *Intel Atom Processor E6x5C Series Product Preview Datasheet Rev. 001US*, Intel Corporation, Dec. 2010.

[137] W.-J. Huang, N. Saxena, and E. McCluskey, "A Reliable LZ Data Compressor on Reconfigurable Coprocessors," in *Field-Programmable Custom Computing Machines, 2000 IEEE Symposium on*, 2000, pp. 249–258.

[138] *Xilinx Product Specification DS031 v4.0: Virtex-II Platform FPGAs: Complete Data Sheet*, Xilinx, Inc., Apr. 2014.

[139] *Xilinx User Guide UG500 v1.0: Programmable Logic Design - Quick Start Guide*, Xilinx, Inc., May 2008.

[140] P. Graham, M. Caffrey, J. Zimmerman, D. E. Johnson, P. Sundararajan, and C. Patterson, "Consequences and Categories of SRAM FPGA Configuration SEUs," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, 2000.

[141] *Xilinx User Guide UG702 v14.1: Partial Reconfiguration User Guide*, Xilinx, Inc., Apr. 2012.

[142] J.-B. Note and E. Rannaud, "From the Bitstream to the Netlist," in *Proceedings of the 16th International ACM/SIGDA Symposium on Field Programmable Gate Arrays*. New York, NY, USA: ACM, 2008, pp. 264–264.

[143] *Xilinx User Guide UG768 v14.7: Xilinx 7 Series FPGA and Zynq-7000 All Programmable SoC Libraries Guide for HDL Designs*, Xilinx, Inc., Oct. 2013.

[144] K. Chapman, *Xilinx Application Note XAPP864 v2.0: SEU Strategies for Virtex-5 Devices*, Apr. 2010.

[145] X. Iturbe, M. Azkarate, I. Martinez, J. Perez, and A. Astarloa, "A novel SEU, MBU and SHE handling strategy for Xilinx Virtex-4 FPGAs," in *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*, Aug 2009, pp. 569–573.

[146] *Xilinx User Guide UG474 v1.7: 7 Series FPGAs Configurable Logic Block User Guide*, Xilinx, Inc., Nov. 2014.

[147] *Xilinx Product Specification DS003 v4.0: Virtex 2.5V Field Programmable Gate Arrays*, Xilinx, Inc., Mar. 2013.

[148] C. Bobda, Ed., *Introduction to Reconfigurable Computing: Architectures, Algorithms, and Applications*, 1st ed. Springer Publishing Company, Incorporated, 2007.

[149] F. Grüll, "Acceleration of Biomedical Image Processing and Reconstruction with FPGAs," Ph.D. dissertation, Goethe University Frankfurt, 2014.

[150] *Xilinx User Guide UG070 v2.6: Virtex-4 FPGA User Guide*, Xilinx, Inc., Dec. 2008.

[151] *Xilinx User Guide UG475 v1.13: 7 Series FPGAs Packaging and Pinout*, Xilinx, Inc., Nov. 2014.

[152] *Xilinx User Guide UG471 v1.4: 7 Series FPGAs SelectIO Resources*, Xilinx, Inc., Nov. 2014.

[153] *Xilinx User Guide UG479 v1.8: 7 Series DSP48E1 Slice User Guide*, Xilinx, Inc., Nov. 2014.

[154] C. Carmichael, *Xilinx Application Note XAPP197 v1.0.1: Triple Module Redundancy Design Techniques for Virtex FPGAs*, Jul. 2006.

[155] P. Graham, M. Caffrey, D. Johnson, N. Rollins, and M. Wirthlin, "SEU Mitigation for Half-Latches in Xilinx Virtex FPGAs," *Nuclear Science, IEEE Transactions on*, vol. 50, no. 6, pp. 2139–2146, Dec 2003.

[156] P. Graham, H. Quinn, and J. Moore, "Xilinx Virtex FPGA Design Guide for Space," Los Alamos National Laboratory, Tech. Rep., Jul. 2008, lA-UR-08-04992.

[157] *Xilinx User Guide UG156 v2.2: TMRTool Software Version 9.2i*, Xilinx, Inc., Sep. 2007.

[158] E. C. of Technological Safety, "Interaction of Radiation with Matter," Nov. 2012.

[159] A. H. Compton, "Secondary Rediations produced by X-rays, and some of their applications to physical problems," *Bulletin of the National Research Council*, vol. 4, no. 20, p. 56, Oct. 1922.

[160] M. Lovellette, K. Wood, and J. Beall, "High Energy Gamma-rays and Modern Electronics," in *Aerospace Conference, 2008 IEEE*, March 2008, pp. 1–7.

[161] W. R. Leo, *Techniques for Nuclear and Particle Physics Experiments: A How-to Approach*, 2nd ed. Springer-Verlag Berlin Heidelberg GmbH, 1994.

[162] V. Zajic, "Calculation Tool: Energy vs. LET vs. Range calculator v1.24," Dec. 2001.

[163] D. Grahn, Ed., *HZE-Particle Effects in Manned Spaceflight*. National Academy of Sciences, May 1973.

[164] K. Olive and P. D. Group, "Review of Particle Physics," *Chinese Physics C*, vol. 38, no. 9, 2014.

[165] H. Nikjoo, S. Uehara, and D. Emfietzoglou, *Interaction of Radiation with Matter*. CRC Press, 2012.

[166] F. Bloch, "Über die quantenmechanik der elektronen in kristallgittern," *Zeitschrift für Physik*, vol. 52, no. 7-8, pp. 555–600, 1929.

[167] C. Kittel and P. McEuen, *Introduction to Solid State Physics*. Wiley New York, 1976, vol. 8.

[168] C. A. Klein, "Bandgap Dependence and Related Features of Radiation Ionization Energies in Semiconductors," *Journal of Applied Physics*, vol. 39, no. 4, pp. 2029–2038, Mar 1968.

[169] G. Lutz, *Semiconductor Radiation Detectors: Device Physics*, 1st ed. Springer-Verlag Berlin Heidelberg, 2007.

[170] J. R. Schwank, "Space and Military Radiation Effects in Silicon-on-Insulator Devices," Sandia National Laboratories, Albuquerque, New Mexico, Tech. Rep., 1996.

[171] F. Emery and T. Rabson, "Temperature Dependence of Average Energy Per Pair in Semiconductor Detectors," *Nuclear Science, IEEE Transactions on*, vol. 13, no. 1, pp. 48–52, Feb 1966.

[172] W. Zimmerman, "Experimental Verification of the Shockley–Read–Hall Recombination Theory in Silicon," *Electronics Letters*, vol. 9, no. 16, pp. 378–379, August 1973.

[173] A. Schenk, "A Model for the Field and Temperature Dependenceof Shockley-Read-Hall Lifetimes in Silicon," *Solid-State Electronics*, vol. 35, no. 11, pp. 1585–1596, 1992.

[174] R. P. Mertens, R. J. van Overstraeten, and H. J. de Man, "Heavy Doping Effects in Silicon," *Advances in Electronics and Electron Physics*, vol. 55, pp. 77–118, 1981.

[175] J. F. Ziegler and J. P. Biersack, "Calculation Tool: SRIM & TRIM - The Stopping and Range of Ions in Matter v2013.00," 2013.

[176] A. Fassò, A. Ferrari, J. Ranft, and P. R. Sala, "Calculation Tool: FLUKA 2011.2c.0," Oct. 2014.

[177] D. G. Mavis and P. H. Eaton, "SEU and SET Mitigation Techniques for FPGA Circuit and Configuration Bit Storage Design," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, 2000.

[178] L. Gunderson and J. Tepper, *Clinical Radiation Oncology*. Elsevier Health Sciences, 2011.

[179] S. Buchner, N. Kanyogoro, D. McMorrow, C. Foster, P. O'Neill, and K. Nguyen, "Variable Depth Bragg Peak Method for Single Event Effects Testing," *Nuclear Science, IEEE Transactions on*, vol. 58, no. 6, pp. 2976–2982, Dec 2011.

[180] J. Adams, R. Silberberg, and C. Tsao, "Cosmic ray effects on microelectronics," *Nuclear Science, IEEE Transactions on*, vol. 29, no. 1, pp. 169–172, Feb 1982.

[181] E. Petersen, *Single Event Effects in Aerospace*. John Wiley & Sons, 2011.

[182] M. McKee, "Nature News: Cosmic rays originate from supernova shockwaves," Feb. 2013.

[183] C. Blume, K. Rabbertz, and S. Tapprogge, "Die starke Seite des LHC," *Physik Journal, Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim*, vol. 11, no. 4, pp. 45–49, Apr. 2012.

[184] M. Ackermann, M. Ajello, A. Allafort, L. Baldini, J. Ballet, G. Barbiellini, M. G. Baring, D. Bastieri, K. Bechtol, R. Bellazzini, R. D. Blandford, E. D. Bloom, E. Bonamente, A. W. Borgland, E. Bottacini, T. J. Brandt, J. Bregeon, M. Brigida, P. Bruel, R. Buehler, G. Busetto, S. Buson, G. A. Caliandro, R. A. Cameron,

P. A. Caraveo, J. M. Casandjian, C. Cecchi, . Çelik, E. Charles, S. Chaty, R. C. G. Chaves, A. Chekhtman, C. C. Cheung, J. Chiang, G. Chiaro, A. N. Cillis, S. Ciprini, R. Claus, J. Cohen-Tanugi, L. R. Cominsky, J. Conrad, S. Corbel, S. Cutini, F. D'Ammando, A. de Angelis, F. de Palma, C. D. Dermer, E. do Couto e Silva, P. S. Drell, A. Drlica-Wagner, L. Falletti, C. Favuzzi, E. C. Ferrara, A. Franckowiak, Y. Fukazawa, S. Funk, P. Fusco, F. Gargano, S. Germani, N. Giglietto, P. Giommi, F. Giordano, M. Giroletti, T. Glanzman, G. Godfrey, I. A. Grenier, M.-H. Grondin, J. E. Grove, S. Guiriec, D. Hadasch, Y. Hanabata, A. K. Harding, M. Hayashida, K. Hayashi, E. Hays, J. W. Hewitt, A. B. Hill, R. E. Hughes, M. S. Jackson, T. Jogler, G. Jóhannesson, A. S. Johnson, T. Kamae, J. Kataoka, J. Katsuta, J. Knödlseder, M. Kuss, J. Lande, S. Larsson, L. Latronico, M. Lemoine-Goumard, F. Longo, F. Loparco, M. N. Lovellette, P. Lubrano, G. M. Madejski, F. Massaro, M. Mayer, M. N. Mazziotta, J. E. McEnery, J. Mehault, P. F. Michelson, R. P. Mignani, W. Mitthumsiri, T. Mizuno, A. A. Moiseev, M. E. Monzani, A. Morselli, I. V. Moskalenko, S. Murgia, T. Nakamori, R. Nemmen, E. Nuss, M. Ohno, T. Ohsugi, N. Omodei, M. Orienti, E. Orlando, J. F. Ormes, D. Paneque, J. S. Perkins, M. Pesce-Rollins, F. Piron, G. Pivato, S. Rainò, R. Rando, M. Razzano, S. Razzaque, A. Reimer, O. Reimer, S. Ritz, C. Romoli, M. Sánchez-Conde, A. Schulz, C. Sgrò, P. E. Simeon, E. J. Siskind, D. A. Smith, G. Spandre, P. Spinelli, F. W. Stecker, A. W. Strong, D. J. Suson, H. Tajima, H. Takahashi, T. Takahashi, T. Tanaka, J. G. Thayer, J. B. Thayer, D. J. Thompson, S. E. Thorsett, L. Tibaldo, O. Tibolla, M. Tinivella, E. Troja, Y. Uchiyama, T. L. Usher, J. Vandenbroucke, V. Vasileiou, G. Vianello, V. Vitale, A. P. Waite, M. Werner, B. L. Winer, K. S. Wood, M. Wood, R. Yamazaki, Z. Yang, and S. Zimmer, "Detection of the Characteristic Pion-Decay Signature in Supernova Remnants," *Science*, vol. 339, no. 6121, pp. 807–811, 2013.

[185] D. E. Gary, "Solar Radio Burst Effects on Wireless Systems," in *Electromagnetic Compatibility (EMC), 2011 IEEE International Symposium on*, Aug 2011, pp. 661–664.

[186] G. Summers, M. Xapsos, and E. Burke, "Application of Extreme Value Statistics to the Prediction of Solar Flare Proton Effects on Solar Cells," in *Photovoltaic Specialists Conference, 1996., Conference Record of the Twenty Fifth IEEE*, May 1996, pp. 289–292.

[187] K. Scherer, H. Fichtner, B. Heber, and U. Mall, Eds., *Space Weather: The Physics Behind a Slogan*, ser. Lecture Notes in Physics. Springer, 2005.

[188] G. W. Philips, D. J. Nagel, and T. Coffey, "A Primer on the Detection of Nuclear and Radiological Weapons," Center for Technology and National Security Policy, National Defense University, Tech. Rep., Jul. 2005.

[189] Scientific Committee on the Effects of Atomic Radiation, "Unscear 2000 report vol. i, annex b: Exposures from natural radiation sources," United Nations, Tech. Rep., 2000.

[190] D. P. Jackson and M. T. Welker, "Measuring and modeling cosmic ray showers with an MBL system: An undergraduate project," *American Journal of Physics*, vol. 69, no. 8, pp. 896–900, Aug. 2001.

[191] A. Bonardi, M. Aglietta, G. Bruno, W. Fulgione, and A. A. B. Machado, "Direct measurement of the atmospheric neutron flux in the energy range 10–500 MeV," *Astroparticle Physics*, vol. 34, no. 4, pp. 225 – 229, 2010.

[192] R. Baumann, "Radiation-Induced Soft Errors in Advanced Semiconductor Technologies," *Device and Materials Reliability, IEEE Transactions on*, vol. 5, no. 3, pp. 305–316, Sept 2005.

[193] J. Barth, C. Dyer, and E. Stassinopoulos, "Space, Atmospheric, and Terrestrial Radiation Environments," *Nuclear Science, IEEE Transactions on*, vol. 50, no. 3, pp. 466–482, June 2003.

[194] S. M. Barbero, S. K. Höffgen, G. Berger, and H. Guerrero, *Compendium of International Irradiation Test Facilities*, 1st ed. RADECS Conference, Sep. 2011.

[195] C. Borcea, S. Buono, P. Cennini, M. Dahlfors, V. Dangendorf, A. Ferrari, G. Garcia-Munoz, Y. Kadi, V. Lacoste, R. Nolte, E. Radermacher, C. Rubbia, F. Saldana, V. Vlachoudis, M. Weierganz, and L. Zanini, "The Neutron Time Of Flight Facility at CERN," *Journal of Nuclear Science and Technology*, vol. 39, no. sup2, pp. 653–656, 2002.

[196] Kommunikationsgruppe, "CERN FAQ - LHC - Ein Leitfaden," CERN, Tech. Rep., Oct. 2009, cERN-Brochure-2009-003-Ger.

[197] ALICE Collaboration, "Performance of the ALICE experiment at the CERN LHC," *International Journal of Modern Physics A*, vol. 29, no. 24, p. 1430044, 2014.

[198] Press Service of the Russian Federal Space Agency, "Commission report about causes of the abnormal situation which has arisen during the flight of spacecraft Phobos-Grunt (translated from Russian)," Feb. 2012. [Online]. Available: http://www.roscosmos.ru/18126

[199] R. Blair, "Surface Effects of Radiation on Transistors," *Nuclear Science, IEEE Transactions on*, vol. 10, no. 5, pp. 35–44, Nov 1963.

[200] D. Peck, R. Blair, W. Brown, and F. Smits, "Surface effects of radiation on transistors," *Bell System Technical Journal, The*, vol. 42, no. 1, pp. 95–129, Jan 1963.

[201] B. Kuehne, "Surface Effects of Radiation on MOS Transistors," in *Physics of Failure in Electronics, 1966. Fifth Annual Symposium on the*, Nov 1966, pp. 311–327.

[202] E. Snow, A. Grove, and D. Fitzgerald, "Effects of Ionizing Radiation on Oxidized Silicon Surfaces and Planar Devices," *Proceedings of the IEEE*, vol. 55, no. 7, pp. 1168–1185, July 1967.

[203] D. Fleetwood, "Total Ionizing Dose Effects in MOS and Low-Dose-Rate-Sensitive Linear-Bipolar Devices," *Nuclear Science, IEEE Transactions on*, vol. 60, no. 3, pp. 1706–1730, June 2013.

[204] A. Johnston, "Radiation Effects in Advanced Microelectronics Technologies," *Nuclear Science, IEEE Transactions on*, vol. 45, no. 3, pp. 1339–1354, Jun 1998.

[205] A. Ochoa, W. Dawes, and D. Estreich, "Latch-Up Control in CMOS Integrated Circuits," *Nuclear Science, IEEE Transactions on*, vol. 26, no. 6, pp. 5065–5068, Dec 1979.

[206] J. Rafi, A. Mercha, E. Simoen, C. Claeys, and A. Mohammadzadeh, "Radiation-induced back channel leakage in 60 MeV-proton-irradiated 0.10 μm-CMOS partially depleted SOI MOSFETs," in *Radiation and Its Effects on Components and Systems, 2003. RADECS 2003. Proceedings of the 7th European Conference on*, Sept 2003, pp. 425–432.

[207] G. Brucker, R. Ohanian, and E. Stassinopoulos, "Successful Large-Scale Use of CMOS Devices on Spacecraft Traveling Through Intense Radiation Belts," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. AES-12, no. 1, pp. 23–31, Jan 1976.

[208] B. Povh, M. Lavelle, K. Rith, C. Scholz, and F. Zetsche, *Particles and Nuclei: An Introduction to the Physical Concepts*. Springer, 2008.

[209] D. Hiemstra, F. Chayab, and Z. Mohammed, "Single Event Upset Characterization of the Virtex-4 Field Programmable Gate Array Using Proton Irradiation," in *Radiation Effects Data Workshop, 2006 IEEE*, July 2006, pp. 105–108.

[210] H. Quinn, P. Graham, J. Krone, M. Caffrey, and S. Rezgui, "Radiation-Induced Multi-Bit Upsets in SRAM-Based FPGAs," *Nuclear Science, IEEE Transactions on*, vol. 52, no. 6, pp. 2455–2461, Dec 2005.

[211] K. M. Warren, "Sensitive Volume Models For Single Event Upset Analysis and Rate Prediction for Space, Atmospheric, and Terrestrial Radiation Environments," Ph.D. dissertation, Vanderbilt University, Jun. 2010.

[212] L. Edmonds, "SEU Cross Sections Derived from a Diffusion Analysis," *Nuclear Science, IEEE Transactions on*, vol. 43, no. 6, pp. 3207–3217, Dec 1996.

[213] D. Hiemstra and E. Blackmore, "LET Spectra of Proton Energy Levels From 50 to 500 MeV and Their Effectiveness for Single Event Effects Characterization of Microelectronics," *Nuclear Science, IEEE Transactions on*, vol. 50, no. 6, pp. 2245–2250, Dec 2003.

[214] J. George, R. Koga, O. Swift, G. Allen, C. Carmichael, and C. Tseng, "Single Event Upsets in Xilinx Virtex-4 FPGA Devices," in *Radiation Effects Data Workshop, 2006 IEEE*, July 2006, pp. 109–114.

[215] M. J. Wirthlin, H. Takai, and A. Harding, "Soft error rate estimations of the Kintex-7 FPGA within the ATLAS Liquid Argon (LAr) Calorimeter," *Journal of Instrumentation*, vol. 9, 2014.

[216] P. Adell and G. Allen, "Assessing and Mitigating Radiation Effects in Xilinx FPGAs," Jet Propulsion Laboratory, National Aeronautics and Space Administration, Tech. Rep., Feb. 2008, jPL Publication 08-9.

[217] P. Adell, G. Allen, G. Swift, and S. McClure, "Assessing and Mitigating Radiation Effects in Xilinx SRAM FPGAs," in *Radiation and Its Effects on Components and Systems (RADECS), 2008 European Conference on*, Sept 2008, pp. 418–424.

[218] E. Peterson, "The Relationship of Proton and Heavy Ion Upset Thresholds," *Nuclear Science, IEEE Transactions on*, vol. 39, no. 6, pp. 1600–1604, Dec 1992.

[219] L. Edmonds, "Proton SEU Cross Sections Derived from Heavy-Ion Test Data," *Nuclear Science, IEEE Transactions on*, vol. 47, no. 5, pp. 1713–1728, Oct 2000.

[220] E. Petersen, "Radiation Induced Soft Fails in Space Electronics," *Nuclear Science, IEEE Transactions on*, vol. 30, no. 2, pp. 1638–1641, April 1983.

[221] A. L. Joseph Joe Fabula, "The NSEU Response of Static Latch Based FPGAs," in *Cosmic Radiation Single Event Effects and Avionics, 2005. The IEE Seminar on (Ref. No. 2005/11270)*, Dec 2005, pp. 4–4/25.

[222] K. Castellani-Coulie, B. Sagnes, F. Saigne, J.-M. Palau, M.-C. Calvet, P. Dodd, and F. Sexton, "Comparison of nmos and pmos transistor sensitivity to seu in srams by device simulation," *Nuclear Science, IEEE Transactions on*, vol. 50, no. 6, pp. 2239–2244, Dec 2003.

[223] I. de Paul, J. Merino, G. Torrens, C. de Benito, J. Verd, J. Segura, and S. Bota, "Top-side pulsed laser induced single event upsets in highly-scaled SRAM devices," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013, pp. 1–4.

[224] D. Savchenkov, A. Chumakov, A. Petrov, A. Pechenkin, A. Egorov, O. Mavritskiy, and A. Yanenko, "Study of SEL and SEU in SRAM using different laser techniques," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013, pp. 1–5.

[225] T. R. Oldham, "Scaling and Single Event Effects (SEE) Sensitivity," in *IEEE Nuclear and Space Radiation Effects Conference (NSREC) Short Course*, Jul. 2003.

[226] D. Binder, E. Smith, and A. Holman, "Satellite Anomalies from Galactic Cosmic Rays," *Nuclear Science, IEEE Transactions on*, vol. 22, no. 6, pp. 2675–2680, Dec 1975.

[227] T. May and M. H. Woods, "Alpha-Particle-Induced Soft Errors in Dynamic Memories," *Electron Devices, IEEE Transactions on*, vol. 26, no. 1, pp. 2–9, Jan 1979.

[228] D. Aspinall and W. Clark, *The Microprocessor and its Application: An Advanced Course*. Cambridge University Press, 1978.

[229] C. Hsieh, P. Murley, and R. O'Brien, "A Field-funneling Effect on the Collection of Alpha-Particle-Generated Carriers in Silicon Devices," *Electron Device Letters, IEEE*, vol. 2, no. 4, pp. 103–105, April 1981.

[230] F. McLean and T. Oldham, "Charge Funneling in N- and P-Type Si Substrates," *Nuclear Science, IEEE Transactions on*, vol. 29, no. 6, pp. 2017–2023, Dec 1982.

[231] O. Musseau, J.-L. Leray, V. Ferlet, A. Umbert, Y. Coic, and P. Hesto, "Charge Collection Mechanisms in MOS/SOI Transistors irradiated by energetic Heavy Ions," *Nuclear Science, IEEE Transactions on*, vol. 38, no. 6, pp. 1226–1233, Dec 1991.

[232] V. Ferlet-Cavrois, P. Paillet, D. McMorrow, N. Fel, J. Baggio, S. Girard, O. Duhamel, J. Melinger, M. Gaillardin, J. Schwank, P. Dodd, M. Shaneyfelt, and J. Felix, "New Insights Into Single Event Transient Propagation in Chains of Inverters - Evidence for Propagation-Induced Pulse Broadening," *Nuclear Science, IEEE Transactions on*, vol. 54, no. 6, pp. 2338–2346, Dec 2007.

[233] H. Nakamura, K. Tanaka, T. Uemura, K. Takeuchi, T. Fukuda, and S. Kumashiro, "Measurement of Neutron-induced Single Event Transient Pulse Width Narrower than 100ps," in *Reliability Physics Symposium (IRPS), 2010 IEEE International*, May 2010, pp. 694–697.

[234] G. Wirth, F. L. Kastensmidt, and I. Ribeiro, "Single Event Transients in Logic Circuits - Load and Propagation Induced Pulse Broadening," *Nuclear Science, IEEE Transactions on*, vol. 55, no. 6, pp. 2928–2935, Dec 2008.

[235] J. Ahlbin, T. Loveless, D. Ball, B. Bhuva, A. Witulski, L. Massengill, and M. Gadlage, "Double-Pulse-Single-Event Transients in Combinational Logic," in *Reliability Physics Symposium (IRPS), 2011 IEEE International*, April 2011, pp. 3C.5.1–3C.5.6.

[236] J. R. Ahlbin, N. M. Atkinson, M. J. Gadlage, D. R. Ball, B. L. Bhuva, and L. W. Massengill, "Angular Dependence of Double-Pulse-Single-Event Transients in Bulk CMOS Technology," in *Radiation and Its Effects on Components and Systems (RADECS), 2012 13th European Conference on*, Sept 2012, pp. 1–4.

[237] M. Gadlage, J. Ahlbin, B. Narasimham, B. Bhuva, L. Massengill, R. Reed, R. Schrimpf, and G. Vizkelethy, "Scaling Trends in SET Pulse Widths in Sub-

100 nm Bulk CMOS Processes," *Nuclear Science, IEEE Transactions on*, vol. 57, no. 6, pp. 3336–3341, Dec 2010.

[238] H. Pahlevanzadeh and Q. Yu, "Systematic Analyses for Latching Probability of Single-Event Transients," in *Quality Electronic Design (ISQED), 2014 15th International Symposium on*, March 2014, pp. 442–449.

[239] K. Røed, "Single event upsets in sram fpga based readout electronics for the time projection chamber in the alice experiment," Ph.D. dissertation, University of Bergen, 2009.

[240] M. Mager, L. Musa, A. Rehman, and A. Szczepankiewicz, "Measurement of single event upsets in the ALICE-TPC front-end electronics," *Applied Mechanics and Materials*, vol. 110, pp. 4505–4511, 2012.

[241] K. Røed, J. Alme, D. Fehlker, C. Lippmann, and A. Rehman, "First measurement of single event upsets in the readout control FPGA of the ALICE TPC detector," *Journal of Instrumentation*, vol. 6, no. 12, p. C12022, 2011.

[242] J. Alme, T. Alt, L. Bratrud, P. Christiansen, F. Costa, E. David, T. Gunji, T. Kiss, R. Langøy, J. Lien, C. Lippmann, A. Oskarsson, A. U. Rehman, K. Røed, D. Röhrich, A. Tarantola, C. Torgersen, I. N. Torsvik, K. Ullaland, A. Velure, S. Yang, C. Zhao, H. Appelshäuser, and L. Österman, "RCU2 - The ALICE TPC readout electronics consolidation for Run2," *Journal of Instrumentation*, vol. 8, no. 12, p. C12032, 2013. [Online]. Available: http://stacks.iop.org/1748-0221/8/i=12/a=C12032

[243] O. Musseau, F. Gardic, P. Roche, T. Corbiere, R. Reed, S. Buchner, P. McDonald, J. Melinger, L. Tran, and A. Campbell, "Analysis of multiple bit upsets (MBU) in CMOS SRAM," *Nuclear Science, IEEE Transactions on*, vol. 43, no. 6, pp. 2879–2888, Dec 1996.

[244] G. Gasiot, D. Giot, and P. Roche, "Multiple Cell Upsets as the Key Contribution to the Total SER of 65 nm CMOS SRAMs and Its Dependence on Well Engineering," *Nuclear Science, IEEE Transactions on*, vol. 54, no. 6, pp. 2468–2473, Dec 2007.

[245] P. Roche, G. Gasiot, S. Uznanski, J.-M. Daveau, J. Torras-Flaquer, S. Clerc, and R. Harboe-Sorensen, "A Commercial 65nm CMOS Technology for Space Applications: Heavy Ion, Proton and Gamma Test Results and Modeling," in *Ra-*

diation and Its Effects on Components and Systems (RADECS), 2009 European Conference on, Sept 2009, pp. 456–464.

[246] S. Uznanski, G. Gasiot, P. Roche, C. Tavernier, and J. Autran, "Single Event Upset and Multiple Cell Upset Modeling in Commercial Bulk 65 nm CMOS SRAMs and Flip-Flops," in *Radiation and Its Effects on Components and Systems (RADECS), 2009 European Conference on*, Sept 2009, pp. 194–200.

[247] H. Quinn, K. Morgan, P. Graham, J. Krone, M. Caffrey, and K. Lundgreen, "Domain Crossing Errors: Limitations on Single Device Triple-Modular Redundancy Circuits in Xilinx FPGAs," *Nuclear Science, IEEE Transactions on*, vol. 54, no. 6, pp. 2037–2043, Dec 2007.

[248] F. Wrobel, J.-M. Palau, M.-C. Calvet, O. Bersillon, and H. Duarte, "Simulation of nucleon-induced nuclear reactions in a simplified SRAM structure: scaling effects on SEU and MBU cross sections," *Nuclear Science, IEEE Transactions on*, vol. 48, no. 6, pp. 1946–1952, Dec 2001.

[249] A. Dixit and A. Wood, "The Impact of New Technology on Soft Error Rates," in *Reliability Physics Symposium (IRPS), 2011 IEEE International*, April 2011, pp. 5B.4.1–5B.4.7.

[250] *JEDEC Standard No. 89A (JESD89A): Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices*, JEDEC Solid State Technology Association, Oct. 2006.

[251] B. Bridgford, C. Carmichael, and C. W. Tseng, *Xilinx Application Note XAPP987 v1.0: Single-Event Upset Mitigation Selection Guide*, Mar. 2008.

[252] G. Allen, G. Madias, E. Miller, and G. Swift, "Recent Single Event Effects Results in Advanced Reconfigurable Field Programmable Gate Arrays," in *Radiation Effects Data Workshop (REDW), 2011 IEEE*, July 2011, pp. 1–6.

[253] *White Paper 1207: Understanding Single Event Functional Interrupts in FPGA Designs*, Altera Corporation, Sep. 2013.

[254] G. Johnson, J. H. Hohl, R. Schrimpf, and K. Galloway, "Simulating Single-Event Burnout of n-Channel Power MOSFET's," *Electron Devices, IEEE Transactions on*, vol. 40, no. 5, pp. 1001–1008, May 1993.

[255] M. D'Alessio, C. Poivey, V. Ferlet-Cavrois, H. Evans, R. Harboe-Sorensen, A. Keating, I. Lopez-Calle, F. Guerre, F. Lochon, S. Santandrea, A. Zadeh, M. Muschitiello, M. Markgraf, O. Montenbruck, A. Grillenberger, N. Fleurinck, K. Puimege, D. Gerrits, and P. Matthijs, "SRAMs SEL and SEU In-Flight Data From PROBA-II Spacecraft," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013, pp. 1–8.

[256] D. White, *Xilinx White Paper WP402 v1.0.1: Considerations Surrounding Single Event Effects in FPGAs, ASICs, and Processors*, Mar. 2002.

[257] M. Hargrove, S. Voldman, R. Gauthier, I. Brown, K. Duncan, and W. Craig, "Latchup in CMOS Technology," in *Reliability Physics Symposium Proceedings, 1998. 36th Annual. 1998 IEEE International*, March 1998, pp. 269–278.

[258] S. Kuboyama, K. Sugimoto, S. Shugyo, S. Matsuda, and T. Hirao, "Single-Event Burnout of Epitaxial Bipolar Transistors," *Nuclear Science, IEEE Transactions on*, vol. 45, no. 6, pp. 2527–2533, Dec 1998.

[259] S. Liu, M. Boden, D. Girdhar, and J. Titus, "Single-event burnout and avalanche characteristics of power dmosfets," *Nuclear Science, IEEE Transactions on*, vol. 53, no. 6, pp. 3379–3385, Dec 2006.

[260] R. Koga, W. Crain, K. Crawford, D. Lau, S. Pinkerton, B. Yi, and R. Chitty, "On the suitability of non-hardened high density SRAMs for space applications," *Nuclear Science, IEEE Transactions on*, vol. 38, no. 6, pp. 1507–1513, Dec 1991.

[261] C. Dufour, P. Garnier, T. Carriere, J. Beaucour, R. Ecoffet, and M. Labrunee, "Heavy ion induced single hard errors on submicronic memories," *Nuclear Science, IEEE Transactions on*, vol. 39, no. 6, pp. 1693–1697, Dec 1992.

[262] G. M. Swift, D. J. Padgett, and A. H. Johnston, "A new class of single event hard errors," *Nuclear Science, IEEE Transactions on*, vol. 41, no. 6, pp. 2043–2048, Dec 1994.

[263] N. Battezzati, L. Sterpone, and M. Violante, *Reconfigurable Field Programmable Gate Arrays for Mission-Critical Applications*. Springer, 2010.

[264] I. Mouret, M.-C. Calvet, P. Calvel, P. Tastet, M. Allenspach, K. LaBel, J. Titus, C. Wheatley, R. Schrimpf, and K. Galloway, "Experimental Evidence of the Temperature and Angular Dependence in SEGR," in *Radiation and its Effects*

on Components and Systems, 1995. RADECS 95., Third European Conference on, Sep 1995, pp. 313–320.

[265] T. F. Wrobel, "On Heavy Ion Induced Hard-Errors in Dielectric Structures," *Nuclear Science, IEEE Transactions on*, vol. 34, no. 6, pp. 1262–1268, Dec 1987.

[266] A. Johnston, G. Swift, and B. Rax, "Total Dose Effects in Conventional Bipolar Transistors and Linear Integrated Circuits," *Nuclear Science, IEEE Transactions on*, vol. 41, no. 6, pp. 2427–2436, Dec 1994.

[267] S. Duzellier, S. Bourdarie, R. Velazco, B. Nicolescu, and R. Ecoffet, "SEE In-Flight Data for two Static 32KB Memories on High Earth Orbit," in *Radiation Effects Data Workshop, 2002 IEEE*, 2002, pp. 1–6.

[268] D. Habing and B. Shafer, "Room Temperature Annealing of Ionizatton-Induced Damage in CMOS Circuits," *Nuclear Science, IEEE Transactions on*, vol. 20, no. 6, pp. 307–314, Dec 1973.

[269] J. Srour, S. Chen, S. Othmer, and R. Hartmann, "Radiation Damage Coefficients for Silicon Depletion Regions," *Nuclear Science, IEEE Transactions on*, vol. 26, no. 6, pp. 4783–4791, Dec 1979.

[270] L.-J. Cheng and J. W. Corbett, "Defect Creation in Electronic Materials," *Proceedings of the IEEE*, vol. 62, no. 9, pp. 1208–1214, Sept 1974.

[271] J. I. Frenkel, "Über die wärmebewegung in festen und flüssigen körpern," *Zeitschrift für Physik*, vol. 35, no. 8-9, pp. 652–669, 1926.

[272] R. Wunstorf, "Systematische Untersuchungen zur Strahlenresistenz von Silizium-Detektoren für die Verwendung von Hochenergie-Experimenten," Ph.D. dissertation, Universität Hamburg, 1992.

[273] Y. Shi, D. Shen, I. Wu, and K. Cheng, "A numerical study of cluster center formation in neutron-irradiated silicon," *Journal of Applied Physics*, vol. 67, no. 2, pp. 1116–1118, Jan 1990.

[274] A. Holmes-Siedle and L. Adams, *Handbook of Radiation Effects*, ser. Oxford science publications. OUP Oxford, 2002.

[275] S. Messenger, E. Burke, G. Summers, M. Xapsos, R. Walters, E. Jackson, and B. Weaver, "Nonionizing Energy Loss (NIEL) for Heavy Ions," *Nuclear Science, IEEE Transactions on*, vol. 46, no. 6, pp. 1595–1602, Dec 1999.

[276] J. Srour and J. Palko, "A Framework for Understanding Displacement Damage Mechanisms in Irradiated Silicon Devices," *Nuclear Science, IEEE Transactions on*, vol. 53, no. 6, pp. 3610–3620, Dec 2006.

[277] I. Perić, "Design and Realisation of Integrated Circuits for the Readout of Pixel Sensors in High-Energy Physics and Biomedical Imaging," Ph.D. dissertation, Bonn University, 2004.

[278] J. Gorelick, R. Ladbury, and L. Kanchawa, "The Effects of Neutron Irradiation on Gamma Sensitivity of Linear Integrated Circuits," *Nuclear Science, IEEE Transactions on*, vol. 51, no. 6, pp. 3679–3685, Dec 2004.

[279] X.-M. Jin, R.-Y. Fan, W. Chen, Y. Liu, D.-S. Lin, and S.-C. Yang, "The accelerated TID degradation induced by neutron irradiation on CMOS microprocessor," in *Solid-State and Integrated Circuit Technology (ICSICT), 2010 10th IEEE International Conference on*, Nov 2010, pp. 662–664.

[280] V. Berland, A. Touboul, and P. Crevel, "Comparison of Low Frequency Noise Evolution with Oxide Trapped Charge in Irradiated N-MOS Transistors," *Nuclear Science, IEEE Transactions on*, vol. 41, no. 3, pp. 561–564, Jun 1994.

[281] I. S. Esqueda, "Modeling of Total Ionizing Dose Effects in Advanced Complementary Metal-Oxide-Semiconductor Technologies," Ph.D. dissertation, Arizona State University, 2011.

[282] J. M. McGarrity, "Considerations for hardening mos devices and circuits for low radiation doses," *Nuclear Science, IEEE Transactions on*, vol. 27, no. 6, pp. 1739–1744, Dec 1980.

[283] T. Oldham and F. McLean, "Total ionizing dose effects in mos oxides and devices," *Nuclear Science, IEEE Transactions on*, vol. 50, no. 3, pp. 483–499, June 2003.

[284] R. Kjar and J. Peel, "Radiation induced leakage current in N-channel SOS transistors," *Nuclear Science, IEEE Transactions on*, vol. 21, no. 6, pp. 208–210, Dec 1974.

[285] D. Neamen, W. Shedd, and B. Buchanan, "Radiation induced charge trapping at the silicon sapphire substrate interface," *Nuclear Science, IEEE Transactions on*, vol. 21, no. 6, pp. 211–216, Dec 1974.

[286] H. Engel, "Development of a Fault Tolerant Softcore CPU for SRAM based FPGAs," Master's thesis, Heidelberg University, Jun. 2009.

[287] J. Schwank, P. Dodd, M. Shaneyfelt, J. Felix, G. Hash, V. Ferlet-Cavrois, P. Paillet, J. Baggio, P. Tangyunyong, and E. Blackmore, "Issues for Single-Event Proton Testing of SRAMs," *Nuclear Science, IEEE Transactions on*, vol. 51, no. 6, pp. 3692–3700, Dec 2004.

[288] K. A. LaBel and L. M. Cohn, "Radiation Testing on State-of-the-Art CMOS: Challenges, Plans, and Preliminary Results," in *Proceedgins of GOMAC 2009*, 2009.

[289] G. Gildenblat, X. Li, W. Wu, H. Wang, A. Jha, R. van Langevelde, G. Smit, A. Scholten, and D. Klaassen, "PSP: An Advanced Surface-Potential-Based MOSFET Model for Circuit Simulation," *Electron Devices, IEEE Transactions on*, vol. 53, no. 9, pp. 1979–1993, Sept 2006.

[290] J. J. Fabula, J. L. DeJong, A. Lesea, and W.-L. Hsieh, "The Total Ionizing Dose Performance of Deep Submicron CMOS Processes," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, 2008.

[291] M. J. Wirthlin, H. Takai, and A. Harding, "Soft Error Rate Estimations of the Kintex-7 FPGA within the ATAS Liquid Argon Calorimeter," in *TWEPP 2013 - Topical Workshop on Electronics for Particle Physics*, Sep. 2013.

[292] U. S. D. of Defense, "MIL-STD 883, Method 1019.5: Ionization Radiation (Total Dose) Test Procedure," Dec. 1997.

[293] G. M. Swift, "Xilinx Single Event Effects, 1st Consortium Report, Virtex-II Static SEU Characterization," Jet Propulsion Laboratory, National Aeronautics and Space Administration, Tech. Rep., Jan. 2006.

[294] L. Sterpone and M. Violante, "A new reliability-oriented place and route algorithm for SRAM-based FPGAs," *Computers, IEEE Transactions on*, vol. 55, no. 6, pp. 732–744, June 2006.

[295] *Xilinx User Guide UG470 v1.9: 7 Series FPGAs Configuration User Guide*, Xilinx, Inc., Oct. 2013.

[296] *Xilinx User Guide UG472 v1.11.1: 7 Series FPGAs Clocking Resources*, Xilinx, Inc., Mar. 2015.

[297] P. Maillard, L. W. Massengill, W. T. Holman, T. D. Loveless, Y. Chen, N. Roche, J. Warner, S. Buchner, and D. McMorrow, "SET Characterization of Two 90-nm Voltage Controlled Delay Line Topologies," in *Radiation and Its Effects on Components and Systems (RADECS), 2012 13th European Conference on*, Sept 2012, pp. 1–4.

[298] S. Chellappa, L. Clark, and K. Holbert, "A 90-nm Radiation Hardened Clock Spine," in *Radiation and Its Effects on Components and Systems (RADECS), 2011 12th European Conference on*, Sept 2011, pp. 479–484.

[299] R. Monreal and G. Swift, "Upset Manifestations in Embedded Digital Signal Processors (DSPs) due to Single Event Effects (SEE)," in *Radiation and Its Effects on Components and Systems (RADECS), 2012 13th European Conference on*, Sept 2012, pp. 1–5.

[300] C. Hu and S. Zain, *Xilinx Application Note XAPP1073 v1.0: NSEU Mitigation in Avionics Applications*, May 2010.

[301] *Xilinx User Guide UG473 v1.11: 7 Series FPGAs Memory Resources*, Xilinx, Inc., Nov. 2014.

[302] *Xilinx Product Guide PG058 v8.2: LogiCORE IP Block Memory Generator*, Xilinx, Inc., Apr. 2015.

[303] *Xilinx Product Specification DS192 v1.4: Radiation-Hardened, Space-Grade Virtex-5QV Family Overview*, Xilinx, Inc., Nov. 2014.

[304] J. Fabula, J. Moore, and A. Ware, "Understanding neutron single-event phenomena in FPGAs," *Military Embedded Systems*, Mar. 2007.

[305] F. Serrano, J. Clemente, and H. Mecha, "A study of the robustness against SEUs of digital circuits implemented with FPGA DSPs," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013, pp. 1–4.

[306] A.-D. Oancea, "Design of a fault-tolerant configuration and scrubbing chain for the CBM ROC," Master's thesis, Heidelberg University, 2013.

[307] L. Z. Scheick, S. M. Guertin, and G. M. Swift, "Analysis of Radiation Effects on Individual DRAM Cells," *Nuclear Science, IEEE Transactions on*, vol. 47, no. 6, pp. 2534–2538, Dec 2000.

[308] G. Allen, G. Swift, and G. Miller, "Upset Characterization and Test Methodology of the PowerPC405 Hard-Core Processor Embedded in Xilinx Field Programmable Gate Arrays," in *Radiation Effects Data Workshop, 2007 IEEE*, vol. 0, July 2007, pp. 167–171.

[309] T. Armbrusher and M. Krieger, *SPADIC 1.0 Specification Data Sheet v0.30*, University of Heidelberg, May 2014.

[310] T. Armbruster, P. Fischer, and I. Peric, "SPADIC - A Self-Triggered Pulse Amplification and Digitization ASIC," in *Nuclear Science Symposium Conference Record (NSS/MIC), 2010 IEEE*, Oct 2010, pp. 1358–1362.

[311] *eASIC Nextreme Device Handbook v1.3*, eASIC Corporation, Feb. 2010.

[312] U.S. Department of State - Directorate of Defense Trade Controls, "Federal Regulations (CFR), Title 22 (Foreign Relations), Chapter I (Department of State), Subchapter M (International Traffic in Arms Regulations), §121.1 (The United States Munitions List), Category XV (Spacecraft and Related Articles), Section (d)," Federal Register, April 2014.

[313] ——, "Amendments to the ITAR - 79 FR 27180," Federal Register Vol. 79, No. 92, May 2014.

[314] ——, "Federal Regulations (CFR), Title 15 (Commerce and Foreign Trade), Chapter VII (Bureau of Industry and Security, Department of Commerce), Subchapter C (Export Administration Regulations), §774 (The Commerce Control List) - Supplement No. 1," Federal Register, May 2014.

[315] ——, "Federal Regulations (CFR), Title 15 (Commerce and Foreign Trade), Chapter VII (Bureau of Industry and Security, Department of Commerce), Subchapter C (Export Administration Regulations), §772.1 (Definitions of Terms)," Federal Register, May 2014.

[316] "GTC Product Classification Lookup," Xilinx, Inc., jul 2015. [Online]. Available: http://www.xilinx.com/support/ecn

[317] *Xilinx Product Specification DS124 v2.0: QPro Virtex-II 1.5V Radiation-Hardened QML Platform FPGAs*, Xilinx, Inc., Apr. 2014.

[318] *Xilinx Product Specification DS653 v2.1: Space-Grade Virtex-4QV Family Overview*, Xilinx, Inc., Nov. 2014.

[319] U. D. of State Directorate of Defense Trade Controls, "Commodity Jurisdiction Final Determinations," Federal Register, Sep 2012, model Name: Precision Hi-Rel, Manufacturer: Mentor Graphics Corporation, Description: FPGA Synthesis software for High-Reliability applications, Final Determination: USML XV(f), Determination Date: 09/07/2012.

[320] *Precision Hi-Rel Datasheet*, Mentor Graphics Corporation, May 2011.

[321] "Personal Communication with Victor Kronberg, Corporate Marketing Engineer, Mentor Graphics," jul 2015.

[322] The European Commision, "Commission Delegated Regulation (EU) No 1382/2014 - Community regime for the control of exports, transfer, brokering and transit of dual-use items," dec 2014.

[323] M. Gaillardin, M. Raine, O. Duhamel, S. Girard, P. Paillet, D. McMorrow, J. Warner, F. Andrieu, S. Barraud, and O. Faynot, "Impact of mechanical strain on the charge collection mechanisms of nanometer scaled SOI devices under heavy ion and pulsed laser irradiation," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013, pp. 1–7.

[324] S. Sayil, A. B. Akkur, and N. Gaspard, "Single Event crosstalk shielding for CMOS logic," *Microelectronics Journal*, vol. 40, no. 6, pp. 1000–1006, 2009.

[325] P. Reviriego, J. A. Maestro, and C. Cervantes, "Reliability Analysis of Memories Suffering Multiple Bit Upsets," *Device and Materials Reliability, IEEE Transactions on*, vol. 7, no. 4, pp. 592–601, Dec 2007.

[326] M. Vogel, "Nur die härtesten schaltkreise dürfen ins all," *Physik Journal, Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim*, vol. 12, no. 8/9, pp. 76–77, 2013.

[327] C. Urbina-Ortega, G. Furano, G. Magistrati, K. Marinis, A. Menicucci, and D. Merodio-Codinachs, "Flash-based FPGAs in space, design guidelines and trade-off for critical applications," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013, pp. 1–8.

[328] A. B. Chilton, J. K. Shultis, and R. E. Faw, *Principles of radiation shielding*. Prentice-Hall, 1984.

[329] V. Harish, N. Nagaiah, and H. Harish Kumar, "Lead oxides filled isophthalic resin polymer composites for gamma radiation shielding applications," *Indian Journal of Pure and Applied Physics*, vol. 50, no. 11, pp. 847–850, 2012.

[330] W. Fan, C. Drumm, S. B. Roeske, and G. Scrivner, "Shielding Considerations for Satellite Microelectronics," *Nuclear Science, IEEE Transactions on*, vol. 43, no. 6, pp. 2790–2796, Dec 1996.

[331] R. C. Carrington, "Description of a Singular Appearance seen in the Sun on September 1, 1859," *Monthly Notices of the Royal Astronomical Society*, vol. 20, pp. 13–15, 1860.

[332] P. Jiggens, M.-A. Chavy-Macdonald, G. Santin, A. Menicucci, H. Evans, and A. Hilgers, "The magnitude and effects of extreme solar particle events," *Journal of Space Weather and Space Climate*, vol. 4, no. A20, p. 16, April 2014.

[333] H. Evans, E. Daly, P. Nieminen, G. Santin, and C. Erd, "Jovian Radiation Belt Models, Uncertainties and Margins," *Nuclear Science, IEEE Transactions on*, vol. 60, no. 4, pp. 2397–2403, Aug 2013.

[334] D. Rapp, "Radiation Effects and Shielding Requirements in Human Missions to the Moon and Mars," *MARS - The International Journal of Mars Science and Exploration*, vol. 2, pp. 46–71, September 2006.

[335] D. Estreich, A. Ochoa, and R. Dutton, "An Analysis of Latch-up Prevention in CMOS IC's using an Epitaxial-buried Layer Process," in *Electron Devices Meeting, 1978 International*, vol. 24, 1978, pp. 230–234.

[336] R. Stahlbush, G. Campisi, J. McKitterick, W. Maszara, P. Roitman, and G. Brown, "Electron and Hole Trapping in Irradiated SIMOX, ZMR and BE-SOI Buried Oxides," *Nuclear Science, IEEE Transactions on*, vol. 39, no. 6, pp. 2086–2097, Dec 1992.

[337] F. Faccio, "Design Hardening Methodologies for ASICs," in *Radiation Effects on Embedded Systems*, R. Velazco, P. Fouillat, and R. Reis, Eds. Springer Netherlands, 2007, pp. 143–160.

[338] N. Gaspard, S. Jagannathan, Z. Diggins, T. Reece, S.-J. Wen, R. Wong, K. Lilja, M. Bounasser, T. Loveless, W. Holman, B. Bhuva, and L. Massengill, "Angled

Flip-Flop Single-Event Cross Sections for Submicron Bulk CMOS Technologies," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013, pp. 1–4.

[339] S. Mitra, N. Seifert, M. Zhang, Q. Shi, and K. S. Kim, "Robust System Design with Built-In Soft-Error Resilience," *Computer*, vol. 38, no. 2, pp. 43–52, Feb 2005.

[340] M. Zhang, S. Mitra, T. Mak, N. Seifert, N. Wang, Q. Shi, K. S. Kim, N. Shanbhag, and S. Patel, "Sequential Element Design With Built-In Soft Error Resilience," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 14, no. 12, pp. 1368–1378, Dec 2006.

[341] L.-T. Wang, N. Touba, Z. Jiang, S. Wu, J.-L. Huang, and J.-M. Li, "CSER: BISER-Based Concurrent Soft-Error Resilience," in *VLSI Test Symposium (VTS), 2010 28th*, April 2010, pp. 153–158.

[342] D. Bessot and R. Velazco, "Design of SEU-hardened CMOS memory cells: the HIT cell," in *Radiation and its Effects on Components and Systems, 1993.,RADECS 93., Second European Conference on*, Sep 1993, pp. 563–570.

[343] G. Maki and S. Quan, "New design techniques for SEU immune circuits," in *Proceedings of 9th NASA Symposium on VLSI Design*, Nov 2000.

[344] G. Maki, K. Haas, S. Quan, and J. Murguia, "Conflict free radiation tolerant storage cell," Jun. 2003, uS Patent US6573773, University Of New Mexico.

[345] T. Calin, M. Nicolaidis, and R. Velazco, "Upset hardened memory design for submicron CMOS technology," *Nuclear Science, IEEE Transactions on*, vol. 43, no. 6, pp. 2874–2878, Dec 1996.

[346] T. Calin, R. Velazco, M. Nicolaidis, S. Moss, S. LaLumondiere, V. Tran, R. Koga, and K. Clark, "Topology-related upset mechanisms in design hardened storage cells," in *Radiation and Its Effects on Components and Systems, 1997. RADECS 97. Fourth European Conference on*, Sep 1997, pp. 484–488.

[347] R. Naseer and J. Draper, "The DF-dice storage element for immunity to soft errors," in *Circuits and Systems, 2005. 48th Midwest Symposium on*, Aug 2005, pp. 303–306 Vol. 1.

[348] ——, "DF-DICE: A Scalable Solution for Soft Error Tolerant Circuit Design," in *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, May 2006.

[349] S. Jahinuzzaman and R. Islam, "TSPC-DICE: A single phase clock high performance SEU hardened flip-flop," in *Circuits and Systems (MWSCAS), 2010 53rd IEEE International Midwest Symposium on*, Aug 2010, pp. 73–76.

[350] K. Kubota, M. Masuda, J. Furuta, Y. Manzawa, S. Kanda, K. Kobayashi, and H. Onodera, "A low-power and area-efficient radiation-hard redundant flip-flop, DICE ACFF, in a 65 nm thin-BOX FD-SOI," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013, pp. 1–6.

[351] B. Matush, T. Mozdzen, L. Clark, and J. Knudsen, "Area-Efficient Temporally Hardened by Design Flip-Flop Circuits," *Nuclear Science, IEEE Transactions on*, vol. 57, no. 6, pp. 3588–3595, Dec 2010.

[352] J. Knudsen and L. Clark, "An Area and Power Efficient Radiation Hardened by Design Flip-Flop," *Nuclear Science, IEEE Transactions on*, vol. 53, no. 6, pp. 3392–3399, Dec 2006.

[353] J. E. Souza, F. Almeida, E. Chielle, T. Dejoinville, S. Breard, and F. L. Kastensmidt, "Using Configurable Temporal Filtering to Reduce Soft Error Rate," in *Radiation and Its Effects on Components and Systems (RADECS), 2012 13th European Conference on*, Sept 2012, pp. 1–6.

[354] G. Anelli, M. Campbell, M. Delmastro, F. Faccio, S. Floria, A. Giraldo, E. Heijne, P. Jarron, K. Kloukinas, A. Marchioro, P. Moreira, and W. Snoeys, "Radiation Tolerant VLSI Circuits in Standard Deep Submicron CMOS Technologies for the LHC Experiments: Practical Design Aspects," *Nuclear Science, IEEE Transactions on*, vol. 46, no. 6, pp. 1690–1696, Dec 1999.

[355] S. Z., P. V., and S. G., "Fault-tolerant ASIC: Design and Implementation," *Facta universitatis - series: Electronics and Energetics*, vol. 26, no. 3, pp. 175–186, 2013.

[356] F. Hatori, T. Sakurai, K. Nogami, K. Sawada, M. Takahashi, M. Ichida, M. Uchida, I. Yoshii, Y. Kawahara, T. Hibi, Y. Saeki, H. Muroga, A. Tanaka, and K. Kanzaki, "Introducing Redundancy in Field Programmable Gate Arrays,"

in *Custom Integrated Circuits Conference, 1993., Proceedings of the IEEE 1993*, May 1993, pp. 7.1.1–7.1.4.

[357] J. J. Wang, "Radiation Effects in FPGAs," *Proceedings of the 9th Workshop on Electronics for LHC Experiments*, pp. 34–43, Sep 2003.

[358] *RTSX-SU Radiation-Tolerant FPGAs (UMC) Datasheet Rev. 9*, Microsemi Corporation, Mar. 2012.

[359] *RTAX-S/SL and RTAX-DSP Radiation-Tolerant FPGAs Datasheet Rev. 16*, Microsemi Corporation, Jan. 2013.

[360] *Test Report: Radiation-Tolerant ProASIC3 Single-Event Latch-Up*, Actel Corporation, Apr. 2010.

[361] *Microsemi Product Brief PB0051 Revision 1.5: RTG4 FPGAs*, Microsemi Corporation, May 2015.

[362] B. Bancelin, "Atmel ATF280E Rad Hard SRAM Based FPGA," ESA Estec Workshop on Fault Injection & Fault Tolerance in Space FPGAs, Sep. 2009.

[363] G. Swift, G. Madias, Y. Wang, R. Kent, M. Shoga, E. Miller, and R. M. Monreal, "Measured single-event characteristics of the virtex-5qv rhbd space-grade 65nm fpga," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013.

[364] G. Mantelet, M. Briet, G. Rouxel, and S. Hachad, "ATMEL ATF280 Radiation Hardened SRAM Based reprogrammable FPGA SEE test results," in *Radiation and Its Effects on Components and Systems (RADECS), 2010 European Conference on*, Sept 2010.

[365] *Datasheet 7750E: ATF280F - Rad-Hard Reprogrammable FPGA*, Atmel Corporation, may 2012.

[366] K. O'Neill, "Antifuse FPGA Technology: Best Option for Satellite Applications," *COTS - The Journal of Military Electronics & Computing*, no. 12, Dec 2003.

[367] T. Speers, J. Wang, B. Cronquist, J. McCollum, H. Tseng, R. Katz, and I. Kleyner, "0.25 µm FLASH Memory Based FPGA for Space Applications," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, 1999.

[368] F. Irom and D. N. Nguyen, "Single Event Effect Characterization of High Density Commercial NAND and NOR Nonvolatile Flash Memories," *Nuclear Science, IEEE Transactions on*, vol. 54, no. 6, pp. 2547–2553, Dec 2007.

[369] N. Battezzati, L. Sterpone, and M. Violante, "Reprogrammable FPGAs for Mission-Critical Applications," in *Reconfigurable Field Programmable Gate Arrays for Mission-Critical Applications*. Springer New York, 2011, pp. 179–186.

[370] C. Poivey, M. Grandjean, and F.-X. Guerre, "Radiation Characterization of Microsemi ProASIC3 Flash FPGA Family," in *Radiation Effects Data Workshop (REDW), 2011 IEEE*, July 2011, pp. 1–5.

[371] M. Grandjean, "Single Event Effects Test Report Actel A3PE3000L ProASIC3L Flash Based FPGA," Hirex Engineering, Tech. Rep. 3, Aug. 2011.

[372] M. Bagatin, G. Cellere, S. Gerardin, A. Paccagnella, A. Visconti, S. Beltrami, and M. Maccarrone, "Single Event Effects in 1Gbit 90nm NAND Flash Memories under Operating Conditions," in *On-Line Testing Symposium, 2007. IOLTS 07. 13th IEEE International*, July 2007, pp. 146–151.

[373] G. Mantelet, M. Briet, G. Rouxel, S. Hachad, B. Bancelin, and D. de saint Roman, "ATMEL ATF280E Rad Hard SRAM Based reprogrammable FPGA SEE test results," in *Radiation and Its Effects on Components and Systems (RADECS), 2009 European Conference on*, Sept 2009, pp. 606–608.

[374] G. Allen, G. Swift, and C. Carmichael, "Virtex-4VQ Static SEU Characterization Summary," Jet Propulsion Laboratory, National Aeronautics and Space Administration, Tech. Rep., Apr. 2008.

[375] P. Alfke and R. Padovani, "Radiation Tolerance of High-Density FPGAs," *ELECTRON ENG LONDON*, vol. 72, no. 880, p. 2, 2000.

[376] G. Lara, C. Carmichael, and G. Swift, "Taking Designs to New Heights with Space-Grade Virtex-4QV FPGAs," *Xilinx Xcell Journal*, no. 65, pp. 22–26, Jul. 2008.

[377] G. Swift, G. Allen, C. W. Tseng, C. Carmichael, G. Miller, and J. George, "Static Upset Characteristics of the 90nm Virtex-4QV FPGAs," in *Radiation Effects Data Workshop, 2008 IEEE*, July 2008, pp. 98–105.

[378] I. Xilinx, "Xilinx Press Release 412: Xilinx chips land on Mars," online, Jan. 2004. [Online]. Available: http://www.xilinx.com/prs_rls/design_win/0412_marsrover.htm

[379] E. Blansett, "Single Event Upset Xilinx-Sandia Experiment (SEUXSE) on the International Space Station," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, 2008.

[380] W. A. Howes, "On-orbit fpga seu mitigation and measurement experiments on the cibola flight experiment satellite," Master's thesis, Brigham Young University, 2011.

[381] J. D. Cressler and H. A. Mantooth, Eds., *Extreme Environment Electronics*. CRC Press, 2013, ch. 56: Field Programmable Gate Arrays, pp. 641–656.

[382] P. Gasnier and J. Bertrand, "CNES feedback on the ATF280E FPGA and Space FPGA Designer," ESA Estec Workshop on Fault Injection & Fault Tolerance in Space FPGAs, Sep. 2009.

[383] C. Gordon, "The Single Event Immune Reconfigurable FPGA (SIRF) Program: Development of the Radiation Hardened V5QV," in *Proceedings of Infotech@Aerospace Conference 2011*. American Institute of Aeronautics and Astronautics, Mar 2011.

[384] G. Swift and G. Allen, "Virtex-5QV Static SEU Characterization Summary," Xilinx, Inc., Tech. Rep., Jul. 2012.

[385] Y. C. Wang, *Xilinx Application Note XAPP588 v1.0: Virtex-5QV FPGA External Configuration Management*, Jan. 2012.

[386] M. Bajracharya, M. Maimone, and D. Helmick, "Autonomy for Mars Rovers: Past, Present, and Future," *Computer*, vol. 41, no. 12, pp. 44–50, Dec 2008.

[387] R. Winder, "COSMAC - A COS/MOS microprocessor," in *Solid-State Circuits Conference. Digest of Technical Papers. 1974 IEEE International*, vol. XVII, Feb 1974, pp. 64–65.

[388] *LEON3-FT SPARC V8 Processor LEON3FT-RTAX Data Sheet and User's Manual v1.9*, Aeroflex Gaisler, Jan. 2013.

[389] *UT699: 32-bit Fault-Tolerant SPARCTM V8/LEON 3FT Processor Data Sheet*, Aeroflex Gaisler, Jul. 2013.

[390] T. Austin, "DIVA: a reliable substrate for deep submicron microarchitecture design," in *Microarchitecture, 1999. MICRO-32. Proceedings. 32nd Annual International Symposium on*, 1999, pp. 196–207.

[391] K. Bowman, J. Tachanz, N. Kim, J. Lee, C. Wilkerson, S. Lu, T. Karnlk, and V. De, "Sequential circuit with error detection," Oct. 2012, uS Patent US8301970, Intel Corporation.

[392] J. Tschanz, K. Bowman, S.-L. Lu, P. Aseron, M. Khellah, A. Raychowdhury, B. Geuskens, C. Tokunaga, C. Wilkerson, T. Karnik, and V. De, "On-Line Detection and Correction of Errors Due to Fast, Dynamic Voltage Droop Events," in *SELSE - Silicon Errors in Logic - System Effects - 6th Workshop Proceedings*, Mar. 2010.

[393] A. Raychowdhury, J. Tschanz, K. Bowman, S.-L. Lu, P. Aseron, M. Khellah, B. Geuskens, C. Tokunaga, C. Wilkerson, T. Karnik, and V. De, "Error Detection and Correction in Microprocessor Core and Memory Due to Fast Dynamic Voltage Droops," *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, vol. 1, no. 3, pp. 208–217, Sept 2011.

[394] *TMS570LS Series Technical Reference Manual (SPNU489C)*, Texas Instruments Incorporated, Aug. 2011.

[395] *TMS570LS Series 16/32-BIT RISC Flash Microcontroller Data Sheet (SPNS141F)*, Texas Instruments Incorporated, Jul. 2011.

[396] D. Baumeister, S. Anderson, and T. Wooten, "Evaluation of Device-Level Irradiation Effects in a 32-bit Safety Micro Controller for Automotive Braking Applications," in *SELSE - Silicon Errors in Logic - System Effects - 8th Workshop Proceedings*, Mar. 2012.

[397] *Quad Core LEON4 SPARC V8 Processor GR740 Data Sheet and User's Manual v1.0*, Aeroflex Gaisler, Apr. 2015.

[398] I. Koren and C. M. Krishna, *Fault-Tolerant Systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007.

[399] S. Liu, G. Sorrenti, P. Reviriego, F. Casini, J. A. Maestro, and M. Alderighi, "A Fault Tolerant Adaptive Equalizer Implemented with Reconfigurable SRAM-based FPGAs," in *Radiation and Its Effects on Components and Systems (RADECS), 2010 11th European Conference on*, Sept 2010.

[400] R. Kshirsagar and S. Sharma, "Comparative Study of the proposed Shifting Algorithm for Fault Tolerance in FPGA," in *Advances in Engineering and Technology Research (ICAETR), 2014 International Conference on*, Aug 2014, pp. 1–5.

[401] M. Jose, Y. Hu, R. Majumdar, and L. He, "Rewiring for robustness," in *Design Automation Conference (DAC), 2010 47th ACM/IEEE*, June 2010, pp. 469–474.

[402] Z. Feng, N. Jing, G. Chen, Y. Hu, and L. He, "IPF: In-Place X-Filling to Mitigate Soft Errors in SRAM-Based FPGAs," in *Field Programmable Logic and Applications (FPL), 2011 International Conference on*, Sept 2011, pp. 482–485.

[403] J. A. Cheatham, J. M. Emmert, and S. Baumgart, "A Survey of Fault Tolerant Methodologies for FPGAs," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 11, no. 2, pp. 501–533, Apr. 2006.

[404] O. Mencer, K. H. Tsoi, S. Craimer, T. Todman, W. Luk, M. Y. Wong, and P. Leong, "Cube: A 512-FPGA cluster," in *Programmable Logic, 2009. SPL. 5th Southern Conference on*, April 2009, pp. 51–57.

[405] *Xilinx Product Specification DS099 v3.1: Spartan-3 FPGA Family*, Xilinx, Inc., Jun. 2013.

[406] L. Longden, C. Thibodeau, R. Hiliman, P. Layton, and M. Dowd, "Designing A Single Board Computer For Space Using The Most Advanced Processor and Mitigation Technologies," in *Proceedings of the European Space Components Conference, ESCCON 2002, 24-27 September 2002, Toulouse, France*. ESA Publications Division, 2002, pp. 313–316.

[407] J. von Neumann, "Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components," in *Automata Studies*, C. E. Shannon and J. McCarthy, Eds. Princeton University Press, 1956, no. 34, pp. 43–98.

[408] D. P. Siewiorek and R. S. Swarz, *The Theory and Practice of Reliable System Design*. Digital Press, 1982.

[409] ——, *Reliable Computer Systems - Design and Evaluation*, 2nd ed. Digital Press, 1992, rev. ed. of: The Theory and Practice of Reliable System Design, Digital Press, 1982.

[410] F. de Lima Kastensmidt, G. Neuberger, R. Hentschke, L. Carro, and R. Reis, "Designing Fault-Tolerant Techniques for SRAM-Based FPGAs," *Design Test of Computers, IEEE*, vol. 21, no. 6, pp. 552–562, Nov 2004.

[411] S. Mitra and E. McCluskey, "Word-voter: a new voter design for triple modular redundant systems," in *VLSI Test Symposium, 2000. Proceedings. 18th IEEE*, 2000, pp. 465–470.

[412] M. Wirthlin, N. Rollins, M. Caffrey, and P. Graham, "Hardness By Design Techniques for Field Programmable Gate Arrays," in *Proceedings of the 11th Annual NASA Symposium on VLSI design*, 2003.

[413] M. Berg, "Complexity management and design optimization regarding a variety of triple modular redundancy schemes through automation," Mar. 2010, mARLUG - Mid-Atlantic Region Local Users Group Meeting 2010.

[414] F. Kastensmidt, L. Sterpone, L. Carro, and M. Reorda, "On the Optimal Design of Triple Modular Redundancy Logic for SRAM-based FPGAs," in *Design, Automation and Test in Europe, 2005. Proceedings*, March 2005, pp. 1290–1295 Vol. 2.

[415] S. Golshan and E. Bozorgzadeh, "SEU-aware resource binding for modular redundancy based designs on FPGAs," in *Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09.*, April 2009, pp. 1124–1129.

[416] S. Habinc, "Functional Triple Modular Redundancy (FTMR)," *Design and Assessment Report*, pp. 1–56, Dec. 2002.

[417] P. K. Samudrala, J. Ramos, and S. Katkoori, "Selective Triple Modular Redundancy for SEU Mitigation in FPGAs," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, 2003.

[418] M. Berg, H. Kim, M. Friendlich, C. Perez, and C. Seidlick, "Actel proasic field programmable gate array single event effects (see) high-speed test - phase i," NASA Goddard Space Flight Center, Tech. Rep., Dec. 2010.

[419] ——, "Actel proasic a3pe3000l-pq208 field programmable gate array single event effects (see) high-speed test plan - phase ii," NASA Goddard Space Flight Center, Tech. Rep., Sep. 2011.

[420] K. Morgan, D. McMurtrey, B. Pratt, and M. Wirthlin, "A Comparison of TMR With Alternative Fault-Tolerant Design Techniques for FPGAs," *Nuclear Science, IEEE Transactions on*, vol. 54, no. 6, pp. 2065–2072, Dec 2007.

[421] N. Rollins, M. J. Wirthlin, M. Caffrey, and P. Graham, "Evaluating TMR techniques in the presence of single event upsets," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, 2003.

[422] S. Mitra and E. McCluskey, "Which concurrent error detection scheme to choose ?" in *Test Conference, 2000. Proceedings. International*, 2000, pp. 985–994.

[423] C. Gauer, B. LaMeres, and D. Racek, "Spatial Avoidance of Hardware Faults using FPGA Partial Reconfiguration of Tile-Based Soft Processors," in *Aerospace Conference, 2010 IEEE*, March 2010, pp. 1–11.

[424] D. Mavis and P. Eaton, "Soft Error Rate Mitigation Techniques for Modern Microcircuits," in *Reliability Physics Symposium Proceedings, 2002. 40th Annual*, 2002, pp. 216–225.

[425] P. Franco and E. McCluskey, "On-Line Delay Testing of Digital Circuits," in *VLSI Test Symposium, 1994. Proceedings., 12th IEEE*, Apr 1994, pp. 167–173.

[426] M. Berg, H. Kim, M. Friendlich, C. Perez, C. Seidleck, K. LaBel, and R. Ladbury, "SEU Analysis of Complex Circuits Implemented in Actel RTAX-S FPGA Devices," *Nuclear Science, IEEE Transactions on*, vol. 58, no. 3, pp. 1015–1022, June 2011.

[427] S. Rezgui, J. Wang, Y. Sun, D. D'Silva, B. Cronquist, and J. McCollum, "SET Characterization and Mitigation in RTAX-S Antifuse FPGAs," in *Aerospace conference, 2009 IEEE*, March 2009, pp. 1–14.

[428] J. F. Wakerly, *Error Detecting Codes, Self-Checking Circuits and Applications*, ser. Computer Design and Architecture. Elsevier North-Holland, 1978, vol. 3.

[429] S. S. Gorshe, "Concurrent error detection," Ph.D. dissertation, Oregon State University, Apr. 2002.

[430] N. Touba and E. McCluskey, "Logic synthesis of multilevel circuits with concurrent error detection," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 16, no. 7, pp. 783–789, Jul 1997.

[431] S. Tam, *Xilinx Application Note XAPP645 v2.2: Single Error Correction and Double Error Detection*, Aug. 2006.

[432] A. Dutta and N. Touba, "Multiple Bit Upset Tolerant Memory Using a Selective Cycle Avoidance Based SEC-DED-DAEC Code," in *VLSI Test Symposium, 2007. 25th IEEE*, May 2007, pp. 349–354.

[433] *Intel Application Note AP-726 v1.0: E7500 Chipset MCH Intel x4 SDDC*, Intel Corporation, Aug. 2002.

[434] M. D. Sika, A. Yazdanbakhsh, B. Kiddie, J. R. Ahlbin, M. Bajura, M. Fritze, J. Damoulakis, and J. Granacki, "Applying Residue Arithmetic Codes to Combinational Logic to Reduce Single Event Upsets," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013.

[435] S. Baloch, T. Arslan, and A. Stoica, "Design of a Novel Soft Error Mitigation Technique for Reconfigurable Architectures," in *Aerospace Conference, 2006 IEEE*, 2006, p. 9.

[436] ——, "Design of a Single Event Upset (SEU) Mitigation Technique for Programmable Devices," in *Quality Electronic Design, 2006. ISQED '06. 7th International Symposium on*, March 2006, p. 4.

[437] ——, "Design of a Single Event Effect Mitigation Technique for Reconfigurable Architectures," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, 2005.

[438] F. Lima, L. Carro, and R. Reis, "Reducing Pin and Area Overhead in Fault-tolerant FPGA-based Designs," in *Proceedings of the 2003 ACM/SIGDA Eleventh International Symposium on Field Programmable Gate Arrays*. New York, NY, USA: ACM, 2003, pp. 108–117.

[439] ——, "Designing Fault Tolerant Systems into SRAM-based FPGAs," in *Design Automation Conference, 2003. Proceedings*, June 2003, pp. 650–655.

[440] F. Abate, L. Sterpone, and M. Violante, "A New Mitigation Approach for Soft Errors in Embedded Processors," *Nuclear Science, IEEE Transactions on*, vol. 55, no. 4, pp. 2063–2069, Aug 2008.

[441] M. Reorda, M. Violante, C. Meinhardt, and R. Reis, "A Low-Cost SEE Mitigation Solution for Soft-Processors Embedded in Systems On Programmable Chips," in *Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09.*, April 2009, pp. 352–357.

[442] J. Gebelein and U. Kebschull, "Investigation of SRAM FPGA based Hamming FSM encoding in beam test," in *Radiation and Its Effects on Components and Systems (RADECS), 2012 European Conference on*, Sep. 2012.

[443] *Xilinx User Guide UG687 v14.1: XST User Guide for Virtex-6, Spartan-6, and 7 Series Devices*, Xilinx, Inc., Apr. 2012.

[444] G. Burke and S. Taft, "Fault Tolerant State Machines," in *Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, 2004.

[445] L. Frigerio and F. Salice, "RAM-based fault tolerant state machines for FPGAs," in *Defect and Fault-Tolerance in VLSI Systems, 2007. DFT '07. 22nd IEEE International Symposium on*, Sept 2007, pp. 312–320.

[446] *Xilinx User Guide UG909 v2014.4: Vivado Design Suite User Guide - Partial Reconfiguration*, Xilinx, Inc., Nov. 2014.

[447] N. Abel, *Design and Implementation of an Object-Oriented Framework for Dynamic Partial Reconfiguration*. Mensch & Buch, apr 2011.

[448] K. F. Ackermann, "Dynamische Selbstrekonfiguration in der digitalen Bildverarbeitung: Methoden und Architekturen." Ph.D. dissertation, TU Darmstadt, 2011. [Online]. Available: http://tuprints.ulb.tu-darmstadt.de/2418/

[449] N. Abel, F. Grüll, N. Meier, A. Beyer, and U. Kebschull, "Parallel Hardware Objects for Dynamically Partial Reconfiguration," in *Field Programmable Logic and Applications, 2008. FPL 2008. International Conference on*, Sept 2008, pp. 563–566.

[450] M. Straka, J. Kastil, and Z. Kotasek, "Modern fault tolerant architectures based on partial dynamic reconfiguration in FPGAs," in *Design and Diagnostics of Electronic Circuits and Systems (DDECS), 2010 IEEE 13th International Symposium on*, April 2010, pp. 173–176.

[451] *Xilinx User Guide UG001 v1.0: Virtex Configuration Guide*, Xilinx, Inc., Sep. 2000.

[452] *Xilinx User Guide UG002 v2.2: Virtex-II Platform FPGA User Guide*, Xilinx, Inc., Nov. 2007.

[453] Y. C. Wang, *Xilinx Application Note XAPP989 v1.0: Correcting Single-Event Upsets with a Self-Hosting Configuration Management Core*, Apr. 2008.

[454] M. Berg, C. Poivey, D. Petrick, D. Espinosa, A. Lesea, K. LaBel, M. Friendlich, H. Kim, and A. Phan, "Effectiveness of Internal Versus External SEU Scrubbing Mitigation Strategies in a Xilinx FPGA: Design, Test, and Analysis," in *Radiation and Its Effects on Components and Systems, 2007. RADECS 2007. 9th European Conference on*, Sept 2007, pp. 1–8.

[455] *Xilinx Product Guide PG036 v4.1: LogiCORE IP Soft Error Mitigation Controller*, Xilinx, Inc., Apr. 2014.

[456] *Xilinx Product Guide PG036 v3.4: LogiCORE IP Soft Error Mitigation Controller*, Xilinx, Inc., Dec. 2012.

[457] *Xilinx Product Specification DS796 v3.1: LogiCORE IP Soft Error Mitigation Controller v3.1*, Xilinx, Inc., Oct. 2011.

[458] *Xilinx User Guide UG380 v2.3: Spartan-6 FPGA Configuration User Guide*, Xilinx, Inc., Jul. 2011.

[459] *Xilinx User Guide UG071 v1.11: Virtex-4 FPGA Configuration User Guide*, Xilinx, Inc., Jun. 2009.

[460] *Xilinx User Guide UG191 v3.11: Virtex-5 FPGA Configuration User Guide*, Xilinx, Inc., Oct. 2012.

[461] *Xilinx User Guide UG360 v3.0: Virtex-6 FPGA Configuration User Guide*, Xilinx, Inc., Jan. 2010.

[462] A. Ebrahim, T. Arslan, and X. Iturbe, "On enhancing the reliability of internal configuration controllers in fpgas," in *Adaptive Hardware and Systems (AHS), 2014 NASA/ESA Conference on*, July 2014, pp. 83–88.

[463] A. Gabrielli, S. Bonacini, K. Kloukinas, A. Marchioro, P. Moreira, A. Ranieri, and D. De Robertis, "The GBT-SCA, a radiation tolerant ASIC for detector control applications in SLHC experiments," in *TWEPP 2009 - Topical Workshop on Electronics for Particle Physics*, sep 2009, pp. 557–560.

[464] A. Oancea, C. Stüllein, J. Gebelein, and U. Kebschull, "A Resilient, Flash-Free Soft Error Mitigation Concept for the CBM-ToF Read-Out Chain via GBT-SCA," in *Field Programmable Logic and Applications (FPL), 2015 International Conference on*, Sep 2015.

[465] C. Pilotto, J. R. Azambuja, and F. L. Kastensmidt, "Synchronizing Triple Modular Redundant Designs in Dynamic Partial Reconfiguration Applications," in *Proceedings of the 21st Annual Symposium on Integrated Circuits and System Design*, ser. SBCCI '08. ACM, 2008, pp. 199–204.

[466] C. Bolchini, D. Quarta, and M. D. Santambrogio, "SEU Mitigation for SRAM-Based FPGAs through Dynamic Partial Reconfiguration," in *Proceedings of the 17th ACM Great Lakes Symposium on VLSI*. New York, NY, USA: ACM, 2007, pp. 55–60.

[467] G. Miller, C. Carmichael, and G. Swift, *Xilinx Application Note XAPP962 v1.1: Single-Event Upset Mitigation for Xilinx FPGA Block Memories*, Mar. 2008.

[468] C. Bolchini, A. Miele, F. Salice, and D. Sciuto, "A model of soft error effects in generic IP processors," in *Defect and Fault Tolerance in VLSI Systems, 2005. DFT 2005. 20th IEEE International Symposium on*, Oct 2005, pp. 334–342.

[469] N. Saxena and E. McCluskey, "Dependable adaptive computing systems- the roar project," in *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, vol. 3, Oct 1998, pp. 2172–2177 vol.3.

[470] E. Rotenberg, "AR-SMT: A Microarchitectural Approach to Fault Tolerance in Microprocessors," in *Fault-Tolerant Computing, 1999. Digest of Papers. Twenty-Ninth Annual International Symposium on*, June 1999, pp. 84–91.

[471] N. Saxena, S. Fernandez-Gomez, W.-J. Huang, S. Mitra, S.-Y. Yu, and E. McCluskey, "Dependable Computing and Online Testing in Adaptive and Configurable Systems," *Design Test of Computers, IEEE*, vol. 17, no. 1, pp. 29–41, Jan 2000.

[472] S. Mukherjee, M. Kontz, and S. Reinhardt, "Detailed Design and Evaluation of Redundant Multithreading Alternatives," in *Computer Architecture, 2002. Proceedings. 29th Annual International Symposium on*, 2002, pp. 99–110.

[473] S. Campagna, M. Hussain, and M. Violante, "Hypervisor-Based Virtual Hardware for Fault Tolerance in COTS Processors Targeting Space Applications," in *Defect and Fault Tolerance in VLSI Systems (DFT), 2010 IEEE 25th International Symposium on*, Oct 2010, pp. 44–51.

[474] A. Avizienis, "The N-Version Approach to Fault-Tolerant Software," *Software Engineering, IEEE Transactions on*, vol. SE-11, no. 12, pp. 1491–1501, Dec 1985.

[475] M. Rebaudengo, M. Reorda, and M. Violante, "A new Software-based technique for low-cost Fault-Tolerant application," in *Reliability and Maintainability Symposium, 2003. Annual*, 2003, pp. 25–28.

[476] M. Rebaudengo, M. Reorda, M. Torchiano, and M. Violante, "Soft-error Detection through Software Fault-Tolerance techniques," in *Defect and Fault Tolerance in VLSI Systems, 1999. DFT '99. International Symposium on*, Nov 1999, pp. 210–218.

[477] G. Reis, J. Chang, N. Vachharajani, R. Rangan, and D. August, "SWIFT: Software Implemented Fault Tolerance," in *Code Generation and Optimization, 2005. CGO 2005. International Symposium on*, March 2005, pp. 243–254.

[478] G. A. Reis, J. Chang, N. Vachharajani, R. Rangan, D. I. August, and S. S. Mukherjee, "Software-Controlled Fault Tolerance," *ACM Trans. Archit. Code Optim.*, vol. 2, no. 4, pp. 366–396, Dec. 2005.

[479] S. Srinivasan and N. K. Jha, "CRAFT: Criticality based fault tolerance for real-time distributed systems with resource constraints," in *Parallel and Distributed Computing Systems, 1995. ICPADS1995. International Conference on*, Sep 1995.

[480] S. Srinivasan and N. Jha, "Hardware-Software Co-Synthesis of Fault-Tolerant Real-Time Distributed Embedded Systems," in *Design Automation Conference, 1995, with EURO-VHDL, Proceedings EURO-DAC '95., European*, Sep 1995, pp. 334–339.

[481] N. Oh, P. Shirvani, and E. McCluskey, "Error Detection by Duplicated Instructions in Super-Scalar Processors," *Reliability, IEEE Transactions on*, vol. 51, no. 1, pp. 63–75, Mar 2002.

[482] ——, "Control-flow checking by software signatures," *Reliability, IEEE Transactions on*, vol. 51, no. 1, pp. 111–122, Mar 2002.

[483] J. Azambuja, F. Sousa, L. Rosa, and F. Kastensmidt, "The Limitations of Software Signature and Basic Block Sizing in Soft Error Fault Coverage," in *Test Workshop (LATW), 2010 11th Latin American*, March 2010, pp. 1–8.

[484] A. Mahmood and E. McCluskey, "Concurrent Error Detection Using Watchdog Processors - A Survey," *Computers, IEEE Transactions on*, vol. 37, no. 2, pp. 160–174, Feb 1988.

[485] D. Lu, "Watchdog processors and structural integrity checking," *Computers, IEEE Transactions on*, vol. C-31, no. 7, pp. 681–685, July 1982.

[486] J. R. Azambuja, M. Altieri, F. L. Kastensmidt, M. Hübner, and J. Becker, "HETA: Hybrid Error-detection Technique through Assertions," in *Radiation and Its Effects on Components and Systems (RADECS), 2012 13th European Conference on*, Sept 2012, pp. 1–6.

[487] J. Azambuja, M. Altieri, J. Becker, and F. Kastensmidt, "HETA: Hybrid Error-Detection Technique Using Assertions," *Nuclear Science, IEEE Transactions on*, vol. 60, no. 4, pp. 2805–2812, Aug 2013.

[488] J. Azambuja, S. Pagliarini, M. Altieri, F. Kastensmidt, M. Hübner, J. Becker, G. Foucard, and R. Velazco, "Non-Intrusive Reconfigurable HW/SW Fault Tolerance Approach to Detect Transient Faults in Microprocessor Systems," in *Radiation and Its Effects on Components and Systems (RADECS), 2011 12th European Conference on*, Sept 2011, pp. 643–648.

[489] L. Parra, A. Lindoso, M. Portela, L. Entrena, F. Restrepo-Calle, S. Cuenca-Asensi, and A. Martinez-Alvarez, "Efficient Mitigation of Data and Control

Flow Errors in Microprocessors," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sept 2013, pp. 1–4.

[490] N. Oh, S. Mitra, and E. McCluskey, "ED4I: Error Detection by Diverse Data and Duplicated Instructions," *Computers, IEEE Transactions on*, vol. 51, no. 2, pp. 180–199, Feb 2002.

[491] O. Goloubeva, M. Rebaudengo, M. Reorda, and M. Violante, "Soft-error detection using control flow assertions," in *Defect and Fault Tolerance in VLSI Systems, 2003. Proceedings. 18th IEEE International Symposium on*, Nov 2003, pp. 581–588.

[492] M. Lovellette, K. Wood, D. Wood, J. Beall, P. Shirvani, N. Oh, and E. McCluskey, "Strategies for Fault-Tolerant, Space-Based Computing: Lessons Learned from the ARGOS Testbed," in *Aerospace Conference Proceedings, 2002. IEEE*, vol. 5, 2002, pp. 5–2109–5–2119 vol.5.

[493] W. Torres-Pomales, "Software Fault Tolerance: A Tutorial," Langley Research Center, Hampton, Virginia, Tech. Rep., Oct. 2000.

[494] G. K. Saha, "Software Based Fault Tolerance: A Survey," *Ubiquity*, vol. 2006, no. July, Jul. 2006.

[495] P. Samudrala, J. Ramos, and S. Katkoori, "Selective Triple Modular Redundancy (STMR) Based Single-Event Upset (SEU) Tolerant Synthesis for FPGAs," *Nuclear Science, IEEE Transactions on*, vol. 51, no. 5, pp. 2957–2969, Oct 2004.

[496] *Xilinx Sell Sheet: TMRTool*, Xilinx, Inc., Jul. 2009.

[497] *Xilinx Product Brief: TMRTool*, Xilinx, Inc., Mar. 2015.

[498] B. Pratt, M. Caffrey, P. Graham, K. Morgan, and M. Wirthlin, "Improving FPGA Design Robustness with Partial TMR," in *Reliability Physics Symposium Proceedings, 2006. 44th Annual., IEEE International*, March 2006, pp. 226–232.

[499] L. Sterpone, M. Reorda, and M. Violante, "RoRA: a reliability-oriented place and route algorithm for SRAM-based FPGAs," in *Research in Microelectronics and Electronics, 2005 PhD*, vol. 1, July 2005, pp. 173–176 vol.1.

[500] L. Sterpone, *Electronics System Design Techniques for Safety Critical Applications*, ser. Lecture Notes in Electrical Engineering. Springer Netherlands, 2009, vol. 26, ch. Reliability-Oriented Place and Route Algorithm, pp. 71–83.

[501] D. G. Gutiérrez, *Single Event Upsets Simulation Tool – Functional Description*, 1st ed., European Space Agency, Jul 2004.

[502] L. Sterpone and M. Violante, "Static and Dynamic Analysis of SEU effects in SRAM-based FPGAs," in *Test Symposium, 2007. ETS '07. 12th IEEE European*, May 2007, pp. 159–164.

[503] M. Alderighi, *FLIPPER – Executive Summary*, 2nd ed., European Space Agency, Jan 2009.

[504] J. Napoles, H. Guzman, M. Aguirre, J. Tombs, F. Munoz, V. Baena, A. Torralba, and L. Franquelo, "Radiation Environment Emulation for VLSI Designs: A Low Cost Platform based on Xilinx FPGA's," in *Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on*, June 2007, pp. 3334–3338.

[505] J. Napoles, H. Guzman-Miranda, M. Aguirre, J. Tombs, J. Mogollon, R. Palomo, and A. Vega-Leal, "A Complete Emulation System for Single Event Effects Analysis," in *Programmable Logic, 2008 4th Southern Conference on*, March 2008, pp. 213–216.

[506] L. Sterpone, V. F. Cavrois, D. M. Codinachs, and C. Poivey, "SETA: A New Analytical Tool for Single Event Transient Analysis on Flash-based FPGAs," in *Radiation and Its Effects on Components and Systems (RADECS), 2012 13th European Conference on*, Sept 2012, pp. 1–4.

[507] E. Johnson, M. Caffrey, P. Graham, N. Rollins, and M. Wirthlin, "Accelerator Validation of an FPGA SEU Simulator," *Nuclear Science, IEEE Transactions on*, vol. 50, no. 6, pp. 2147–2157, Dec 2003.

[508] V. Asenek, C. Underwood, R. Velazco, S. Rezgui, M. Oldfield, P. Cheynet, and R. Ecoffet, "SEU Induced Errors Observed in Microprocessor Systems," *Nuclear Science, IEEE Transactions on*, vol. 45, no. 6, pp. 2876–2883, Dec 1998.

[509] J. S. Monson, M. Wirthlin, and B. Hutchings, "Fault Injection Results of Linux Operating on an FPGA Embedded Platform," in *Reconfigurable Computing and FPGAs (ReConFig), 2010 International Conference on*, Dec 2010, pp. 37–42.

[510] ——, "A Fault Injection Analysis of Linux Operating on an FPGA-embedded Platform," *Int. J. Reconfig. Comput.*, vol. 2012, Jan. 2012.

[511] N. Abel, C. Garcia, J. Gebelein, S. Manz, and U. Kebschull, "SysCore3 - a new board for the Universal ROC," in *CBM Progress Report 2011*. GSI Helmholtzzentrum für Schwerionenforschung GmbH, 2012, p. 63.

[512] J. Gebelein, D. Gottschalk, G. May, and U. Kebschull, "SysCore3 – A universal Read-Out Controller and Data Processing Board," in *CBM Progress Report 2012*. GSI Helmholtzzentrum für Schwerionenforschung GmbH, 2013, p. 87.

[513] J. Gebelein, G. May, and U. Kebschull, "SysCore v3.1 – A universal Read Out Controller and Data Processing Board," in *CBM Progress Report 2013*. GSI Helmholtzzentrum für Schwerionenforschung GmbH, 2014, p. 88.

[514] W. Heidergott, "SEU tolerant device, circuit and processor design," in *Design Automation Conference, 2005. Proceedings. 42nd*, June 2005, pp. 5–10.

[515] R. Lauwereins, "System Level Design Technology for Realizing an Ambient Intelligent Environment," in *Design Automation Conference, 2004. Proceedings of the ASP-DAC 2004. Asia and South Pacific*, Jan 2004, pp. 1–3.

[516] C. Carmichael and C. W. Tseng, *Xilinx Application Note XAPP988 v1.0: Correcting Single-Event Upsets in Virtex-4 Platform FPGA Configuration Memory*, Mar. 2008.

[517] ——, *Xilinx Application Note XAPP1088 v1.0: Correcting Single-Event Upsets in Virtex-4 FPGA Configuration Memory*, Oct. 2009.

[518] F. Irom and D. N. Nguyen, "Radiation Tests of Highly Scaled High Density Commercial Nonvolatile NAND Flash Memories – Update 2011," Jet Propulsion Laboratory, National Aeronautics and Space Administration, Tech. Rep., Oct. 2011.

[519] F. Benz, A. Seffrin, and S. Huss, "BIL: A tool-chain for bitstream reverse-engineering," in *Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on*, Aug 2012, pp. 735–738.

[520] N. Roßkopf, "BRAM-Scrubber für den Spartan-6 FPGA," Jan. 2011, abschlussbericht Projektpraktikum Informatik.

[521] *Microsemi Handbook: CoreEDAC v2.7*, Microsemi Corporation, Jul. 2015.

[522] P. Gudge, "BSc Thesis: Evaluation and Preparation of Microcontrollers for use in Ionizing Radiation Environments," May 2013.

[523] C. Huck, "Fehlertolerante RS232-Schnittstelle," Feb. 2010, abschlussbericht Projektpraktikum Informatik.

[524] D. Hengstler, "Fehlertolerante Ethernetschnittstelle," Feb. 2010, abschlussbericht Projektpraktikum Informatik.

[525] R. Herveille, *Specification for the WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores, Revision B.3*, OpenCores Organization, Sep. 2002.

[526] *VHDL IP Stack Documentation*, School of Computer Science and Electrical Engineering, University of Queensland, Brisbane, Australia, Feb. 2001.

[527] Christoph Grimm, "VHDL-93, VHDL-AMS (IEEE 1076.1) Grammar for JavaCC," online, Aug. 1997, http://www.ti.informatik.uni-frankfurt.de/grimm/hybrid.html.

[528] Sun Microsystems, Inc., "Java Compiler Compiler™(JavaCC™) - The Java Parser Generator v5.0," online, Aug. 2009, https://javacc.java.net/.

[529] P. Sanda, J. Kellington, P. Kudva, R. Kalla, R. McBeth, J. Ackaret, R. Lockwood, J. Schumann, and C. Jones, "Soft-error resilience of the IBM POWER6 processor," *IBM Journal of Research and Development*, vol. 52, no. 3, pp. 275–284, May 2008.

[530] G. Tompa and J. Cuchiaro, "Microelectronic radiation detector," Feb. 2004, patent US 20040227094 A1.

[531] C. Stüllein, A. Oancea, J. Gebelein, S. Manz, and U. Kebschull, "Design, assembly and test of a positioning system for beam tests," in *CBM Progress Report 2014*. GSI Helmholtzzentrum für Schwerionenforschung GmbH, 2015, p. 106.

[532] J. Gebelein and U. Kebschull, "A three-dimensional FPGA array beam detector for ionizing radiation experiments," in *Radiation and Its Effects on Components and Systems (RADECS), 2011 European Conference on*, Sep. 2011.

[533] R. Harboe-Sørensen, F.-X. Guerre, and A. Roseng, “Design, Testing and Calibration of a "Reference SEU Monitor" System,” in *Radiation Effects on Components and Systems (RADECS 2005)*, September 2005, pp. B3–1–B3–7.

[534] R. Harboe-Sørensen, C. Poivey, F.-X. Guerre, A. Roseng, F. Lochon, G. Berger, W. Hajdas, A. Virtanen, H. Kettunen, and S. Duzellier, “From the Reference SEU Monitor to the Technology Demonstration Module On-Board PROBA-II,” *IEEE Transactions on Nuclear Science*, vol. 55, no. 6, pp. 3082–3087, December 2008.

[535] S. Hoeffgen, M. Durante, V. Ferlet-Cavrois, R. Harboe-Sorensen, W. Lennartz, T. Kuendgen, J. Kuhnhenn, C. LaTessa, M. Mathes, A. Menicucci, S. Metzger, P. Nieminen, R. Pleskac, C. Poivey, D. Schardt, and U. Weinand, “Investigations of Single Event Effects With Heavy Ions of Energies up to 1.5 GeV/n,” *Nuclear Science, IEEE Transactions on*, vol. 59, no. 4, pp. 1161–1166, Aug 2012.

[536] *Xilinx Application Note XAPP426 v1.3: Implementing Xilinx Flip-Chip BGA Packages*, Xilinx, Inc., Mar. 2006.

[537] F. Irom and G. R. Allen, “Radiation Tests of Highly Scaled, High-Density, Commercial, Nonvolatile NAND Flash Memories - Update 2012,” Jet Propulsion Laboratory, National Aeronautics and Space Administration, Tech. Rep., Dec. 2012, jPL Publication 12-19.

[538] *ANSI/VITA 57.1 FPGA Mezzanine Card (FMC) Standard*, American National Standards Institute, Inc., Feb. 2010.

[539] V. Erokhin, “OpenCores: Hierarchical Integer Multiplier Unit,” •, mar 2009.

[540] J. Gebelein, H. Engel, and U. Kebschull, “An approach to system-wide fault tolerance for FPGAs,” in *Radiation and Its Effects on Components and Systems (RADECS), 2009 European Conference on*, Sep. 2009.

[541] H. Quinn, P. Graham, J. Krone, M. Caffrey, S. Rezgui, and C. Carmichael, “Radiation-Induced Multi-Bit Upsets in SRAM-Based FPGAs,” *IEEE Transactions on Nuclear Science*, vol. 52, no. 6, pp. 2455–2462, December 2005.

[542] H. Quinn, K. Morgan, P. Graham, J. Krone, and M. Caffrey, “Static Proton and Heavy Ion Testing of the Xilinx Virtex-5 Device,” in *Radiation Effects Data Workshop*, October 2007, pp. 177–184.

[543] A. Senger, "FLUKA calculations for CBM," GSI Helmholtzzentrum für Schwerionenforschung GmbH, Tech. Rep., Sep. 2011, 18th CBM Collaboration Meeting.

[544] CERN Radiation Working Group (RadWG), "CERN Radiation Test Reports," online, http://radwg.web.cern.ch/RadWG.

[545] R. Erdmann, J. Gebelein, and U. Kebschull, "Beamtest results of Lattice LFE2M20E FPGA," in *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, Sep. 1993.

[546] J. Gebelein, C. Stüllein, and U. Kebschull, "Evaluation of FRAM for use in Radiation Environments," in *CBM Progress Report 2014*. GSI Helmholtzzentrum für Schwerionenforschung GmbH, 2015, p. 107.

[547] *Data Sheet: PTH05010: 15 A, 5-V Input Non-Isolated Wide-Output Adjust Power Modules [SLTS204,C]*, Texas Instruments, Incorporated, Dec. 2003.

[548] E. A. Stott, J. S. Wong, P. Sedcole, and P. Y. Cheung, "Degradation in FPGAs: Measurement and Modelling," in *Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays*. New York, NY, USA: ACM, 2010, pp. 229–238.

[549] S. Mitra, T. Karnik, N. Seifert, and M. Zhang, "Logic Soft Errors in Sub-65nm Technologies Design and CAD Challenges," in *Design Automation Conference, 2005. Proceedings. 42nd*, June 2005, pp. 2–4.

[550] A. Pellegrini, V. Bertacco, and T. Austin, "Fault-Based Attack of RSA Authentication," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, March 2010, pp. 855–860.

[551] C. Beckhoff, D. Koch, and J. Torresen, "Design Tools for Implementing Self-Aware and Fault-Tolerant Systems on FPGAs," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 7, no. 2, pp. 14:1–14:23, Jul. 2014.

[552] *Xilinx Product Specification DS160 v1.7: Spartan-6 Family Overview*, Xilinx, Inc., Mar. 2011.

[553] *Xilinx Product Specification DS150 v2.4: Virtex-6 Family Overview*, Xilinx, Inc., Jan. 2012.

Erklärung zur selbständigen Verfassung

Ich versichere, dass ich die vorliegende Dissertation selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Bei dem Design der fehlertoleranten Komponenten und Analysen wurde ich durch Praktikanten sowie Diplomanden unterstützt. Implementierung und Test der CPU erfolgte durch Heiko Engel [286] im Rahmen einer Diplomarbeit. Serielle Schnittstelle, Ethernet Schnittstelle und Xilinx BRAM Scrubber wurden durch Christian Huck [523], Daniel Hengstler [524] sowie Niko Roßkopf [520] im Rahmen von Projektpraktika implementiert. Die technische Umsetzung des Texas Instruments TMS570 OCM Scrubbers erfolgte durch Pritesh Gudge [522] im Rahmen eines Auslandspraktikums. Die Vorbereitungen, Tests und Auswertung der Lattice FPGA Strahltests wurden von Ralph Erdmann [545] durchgeführt. All diese Arbeiten wurden von Prof. Dr. Udo Kebschull und mir selbst betreut.