ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ – ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πρόγραμμα Μεταπτυχιακών Σπουδών
«Κατανεμημένα Συστήματα, Ασφάλεια και Αναδυόμενες Τεχνολογίες Πληροφορίας»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Αξιολόγηση της ασφάλειας και της αξιοπιστίας επιταχυντών υλικού σχεδιασμένων με χρήση Σύνθεσης Υψηλού Επιπέδου** **Evaluation of the reliability and security of hardware accelerators using High-Level Synthesis**
Ονοματεπώνυμο Φοιτητή	**Καλλιόπη Ξευγένη** **Υπότροφος «ΙΔΡΥΜΑΤΟΣ ΕΥΓΕΝΙΔΟΥ-ΚΛΗΡΟΔΟΤΗΜΑΤΟΣ ΜΑΡΙΑΝΘΗΣ ΣΙΜΟΥ»**
Πατρώνυμο	**Βασίλειος**
Αριθμός Μητρώου	**ΜΠΚΣΑ20004**
Επιβλέπων	**Μιχαήλ Ψαράκης, Αναπληρωτής Καθηγητής**

ΣΕΠΤΕΜΒΡΙΟΣ 2022

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Παναγιώτης Κοτζανικολάου **Αναπληρωτής Καθηγητής**	**Μιχαήλ Ψαράκης** **Αναπληρωτής Καθηγητής**	**Αθανάσιος Παπαδημητρίου** **Επίκουρος Καθηγητής**

Περίληψη

Η πλειοψηφία των ψηφιακών μέσων που χρησιμοποιούνται για όλο το φάσμα των ανθρώπινων επικοινωνιών είτε πρόκειται για προσωπικές, κοινωνικές, επαγγελματικές, οικονομικές, εμπορικές και άλλες επικοινωνίες εμπεριέχουν σε μικρό ή μεγάλο βαθμό ψηφιακά κυκλώματα ενσωματωμένα σε διαφόρων τύπων συσκευές. Εύλογα γίνεται αντιληπτό ότι υπάρχει μεγάλη ανάγκη να εξασφαλιστεί η ασφάλεια και η αξιοπιστία των ψηφιακών κυκλωμάτων. Εξαιρετικά σημαντικό είναι το που λαμβάνει χώρα η κρυπτογράφηση, καθώς σε επίπεδο λογισμικού είναι ασύμφορη. Για την καλύτερη δυνατή απόδοση και πολλαπλάσιες επιδόσεις, η εκτέλεση αυτών των κρυπτογραφικών λειτουργιών επιλέγεται να γίνονται από κρυπτογραφικούς επιταχυντές (accelerators). Η εξέλιξη των ψηφιακών κυκλωμάτων συνεισέφερε στον να μικρύνουν σε όγκο τα ηλεκτρονικά στοιχεία και κατ' επέκταση και ο συνολικός όγκος των κυκλωμάτων, τα οποία κατασκευάζονται συνήθως υπό την μορφή ολοκληρωμένων κυκλωμάτων (Integrated Circuits, ICs) και προγραμματίζονται ώστε να πραγματοποιούν μία σειρά από πολύπλοκες λειτουργίες. Το ζητούμενο λοιπόν είναι η παραγωγή αξιόπιστου και ασφαλούς υλικού (hardware) που θα αποτελέσει το συστατικό στοιχείο για την παραγωγή αξιόπιστων και ασφαλών ψηφιακών κυκλωμάτων. Οι γλώσσες περιγραφής υλικού, όπως είναι η VHDL και η Verilog είναι γλώσσες οι οποίες είναι κατάλληλες για τη σχεδίαση ψηφιακών κυκλωμάτων και μπορούν να περιγράφουν τόσο τη συμπεριφορά όσο και τη δομή ενός κυκλώματος εντούτοις όμως είναι λιγότερο αποδοτικές από άλλες γλώσσες υψηλού επιπέδου High Level Languages (HLL) όπως είναι η C, η C++ και η System C. Για να είναι εφικτή η σύνθεση κυκλωμάτων που έχουν περιγραφεί με γλώσσες υψηλού επιπέδου δημιουργήθηκαν εργαλεία Σύνθεσης Υψηλού Επιπέδου (High Level Synthesis – HLS), τα οποία αναλαμβάνουν, από το περιγραφόμενο υλικό σε γλώσσα υψηλού επιπέδου, να συνθέσουν το αντίστοιχο κύκλωμα στο επίπεδο μεταφοράς καταχωρητών (Register Transfer Level – RTL). Η παραγόμενη περιγραφή στο RTL γίνεται σε VHDL ή/και Verilog που είναι συμβατή με την καθιερωμένη ροή σύνθεσης για την υλοποίηση του τελικού κυκλώματος (FPGA synthesis & implementation). Στην παρούσα μεταπτυχιακή διατριβή χρησιμοποιήθηκε το εργαλείο Vivado High-Level Synthesis της Xilinx το οποίο υποστηρίζει το σχεδιασμό, την προσομοίωση και την υλοποίηση (σύνθεση) ενός μοντέλου σε γλώσσα περιγραφής υλικού (HDL). Προσφέρει την αναπαράσταση της ζητούμενης λειτουργικότητας του υλικού (μέσα από την επιβολή καθορισμένων περιορισμών και βελτιστοποιήσεων) αναλαμβάνοντας το ίδιο το εργαλείο να προσομοιώσει και να δημιουργήσει αυτόματα μία σύνθεση σε γλώσσα περιγραφής υλικού, η οποία στη συνέχεια μέσω κατάλληλου εργαλείου οδηγεί στην υλοποίηση του υλικού με FPGA ή ASICs. Στο πλαίσιο της διατριβής, η σχεδίαση ενός ψηφιακού κυκλώματος αποτυπώνεται με διάφορες υλοποιήσεις, με διαφορετικές βελτιστοποιήσεις (οδηγίες-ντιρεκτίβες) ώστε να μελετηθεί ποια από όλες τις υλοποιήσεις είναι πιο ασφαλής και αξιόπιστη. Η ασφάλεια και αξιοπιστία των κυκλωμάτων αξιολογείται με την εκτέλεση εκτεταμένων πειραμάτων εισαγωγής σφαλμάτων (fault injections). Από τα αποτελέσματα των πειραμάτων καθορίζεται αν χρειάζεται η ενσωμάτωση αντιμέτρων και τεχνικών προστασίας για την βελτίωση της ανθεκτικότητας του κυκλώματος. Για αυτό το λόγο στην παρούσα διατριβή εστιάζουμε στην εισαγωγή σφαλμάτων μέσω προσομοίωσης στο RTL καθώς θα μας επιτρέψει να αξιολογήσουμε την ανθεκτικότητα των υπό μελέτη κυκλωμάτων [1] στην αρχή της σχεδιαστικής ροής.

Abstract

Most digital media used for the full range of human communications whether personal, social, professional, financial, commercial and other communications involve to a lesser or greater extent digital circuits embedded in various types of devices. It stands to reason that there is a great need to ensure the safety and reliability of digital circuits. It is extremely important where the encryption takes place, as it is unprofitable at software level. For the best possible performance, the processing of these cryptographic functions is chosen to be done by cryptographic accelerators. The evolution of digital circuits has contributed to reducing the size of electronic components and, by extension, the circuits surface area, which are usually manufactured in the form of integrated circuits (ICs) and programmed to perform a series of complex functions. The goal is the production of reliable and secure hardware that will be the component to produce reliable and secure digital circuits. Hardware description languages such as VHDL and Verilog are suitable for designing digital circuits and describe both the behavior and structure of a circuit. However, they are less efficient than high-level languages (High Level Languages (HLL) such as C, C++ and System C. For the circuit's synthesis, the High-Level Synthesis (HLS) tools, based on the hardware's description in a high-level language, synthesize the corresponding circuit at the register transfer level (Register Transfer Level - RTL). The generated RTL is in VHDL and/or Verilog compatible with the synthesis flow for the final circuit implementation (FPGA synthesis & implementation). In this thesis, the Vivado High-Level Synthesis tool from Xilinx was used, which supports the simulation and implementation (synthesis) of a model in hardware description language (HDL). It offers the performance of the requested functionality of the hardware (through the enforcement of defined constraints and optimizations) by simulating and automatically creating a composition in a hardware description language which then through a suitable tool leads to hardware implementation with FPGAs or ASICs The design of a digital circuit is modeled with various implementations, with different optimizations (instructions-directives) to study which of all implementations is the most secure and reliable. The reliability and security of the circuits are evaluated with extensive fault injection experiments. From the results, it is determined whether the integration of countermeasures and hardening techniques is needed, to improve circuit's resilience. For this reason, in this dissertation, we focus on fault injection through simulation in the RTL as it will allow us to study the robustness of the circuits [1] at the beginning of the design flow.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ειλικρινείς ευχαριστίες μου στο «ΙΔΡΥΜΑ ΕΥΓΕΝΙΔΟΥ-ΚΛΗΡΟΔΟΤΗΜΑ ΜΑΡΙΑΝΘΗΣ ΣΙΜΟΥ» για την ευκαιρία που μου έδωσε, μέσω της υποτροφίας που μου παρείχε, να εκπληρώσω ένα όνειρο ζωής φοιτώντας στο Π.Μ.Σ. «Κατανεμημένα Συστήματα, Ασφάλεια και Αναδυόμενες Τεχνολογίες Πληροφορίας», καθώς και τον κ. Γ. Σκορδίλη για την υποστήριξή του στη χορήγηση αυτής της υποτροφίας.

Επιπλέον θα ήθελα να απευθύνω θερμές ευχαριστίες στους καθηγητές μου, για τις γνώσεις που μου παρείχαν απλόχερα. Επιθυμώ να εκφράσω την ιδιαίτερη εκτίμησή μου στον καθηγητή μου κ. Μ. Ψαράκη, ο οποίος μου επέτρεψε να εκπονήσω την παρούσα μεταπτυχιακή διατριβή υπό την επίβλεψή του, παρέχοντάς μου ουσιαστική και στοχευμένη επιστημονική καθοδήγηση. Επιπρόσθετα, θα ήθελα να ευχαριστήσω τον κ. Α. Παπαδημητρίου, χάρις στην συνεργασία, στην ενθάρρυνση και στην πολύτιμη συμβολή του οποίου ολοκληρώθηκε με τον καλύτερο δυνατό τρόπο η μεταπτυχιακή μου διατριβή.

Τέλος θα ήθελα να πω ένα ξεχωριστό ευχαριστώ στην οικογένειά μου, που στάθηκε δίπλα μου καθ’ όλη τη διάρκεια των σπουδών μου και μου συμπαραστάθηκε εμπράκτως σε αυτή μου την προσπάθεια, ιδιαίτερα όμως να πω ένα μεγάλο και συγκινητικό ευχαριστώ στο γιο μου, στον οποίο αφιερώνω αυτήν τη μεταπτυχιακή διατριβή και με τον οποίο υπήρξαμε «συνοδοιπόροι» σε αυτό το εκπαιδευτικό ταξίδι.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1. Εισαγωγή

Ο τομέας της επικοινωνίας αποτελεί βασικό θεμέλιο λίθο της σύγχρονης ανθρώπινης καθημερινότητας με τις απαιτήσεις του να αυξάνονται ραγδαία κάθε χρόνο. Η πλειοψηφία των ψηφιακών μέσων που χρησιμοποιούνται για όλο το φάσμα των ανθρώπινων επικοινωνιών είτε πρόκειται για προσωπικές, κοινωνικές, επαγγελματικές, οικονομικές, εμπορικές και άλλες επικοινωνίες εμπεριέχουν σε μικρό ή μεγάλο βαθμό ψηφιακά κυκλώματα. Επίσης το εύρος της ηλικιακής ομάδας των χρηστών, των περισσοτέρων ψηφιακών μέσων, έχει διευρυνθεί αρκετά καθώς περιλαμβάνει άτομα από μικρές ηλικιακές ομάδες έως άτομα πιο προχωρημένης ηλικίας σε σχέση με τα προηγούμενα χρόνια. Αυτό συμβαίνει λόγω της αύξησης και της ποικιλομορφίας των ψηφιακών συναλλαγών.

Για την υποστήριξη της διασύνδεσης όλων αυτών των ψηφιακών εφαρμογών και μέσων υπάρχει διαθέσιμος μεγάλος αριθμός διαφόρων τύπων συσκευών (κυρίως «έξυπνων» συσκευών - smart devices), η συντριπτική πλειοψηφία των οποίων είναι συνδεδεμένες με το διαδίκτυο και τις περισσότερες φορές και μεταξύ τους (συγχρονισμένες).

Εύλογα γίνεται λοιπόν αντιληπτό ότι υπάρχει μεγάλη ανάγκη να εξασφαλιστεί η ασφάλεια και η αξιοπιστία των χρησιμοποιούμενων ψηφιακών συσκευών καθώς και των διαφόρων πληροφοριών που ανταλλάσσονται στο πλαίσιο αυτών των επικοινωνιών. Ως ψηφιακή επικοινωνία όμως δεν νοείται μόνο η επικοινωνία μεταξύ ανθρώπων αλλά και η επικοινωνία - διασύνδεση μεταξύ υποδομών και συστημάτων. Ένα αντιπροσωπευτικό παράδειγμα ψηφιακού συστήματος επικοινωνιών είναι οι «έξυπνες πόλεις», οι οποίες απαιτούν την ενιαία διασύνδεση τόσο συστημάτων («ζωντανή» απεικόνιση της οδικής κυκλοφορίας) όσο και ατόμων (απόρροια χρήσης του GPS) σε πραγματικό χρόνο. Όλα αυτά τα σύγχρονα συστήματα επικοινωνίας δομούνται από ενσωματωμένα ψηφιακά κυκλώματα τα οποία απαιτούν ταυτόχρονα και υψηλού επιπέδου ασφάλεια τόσο των πληροφοριών όσο και των υπηρεσιών. Αυτό προσφέρεται μέσω της κρυπτογραφίας με τη διαδικασία της κρυπτογράφησης.

Αν γίνει αναγωγή σε ένα ευρύτερο επίπεδο εφαρμογών υψηλών απαιτήσεων αξιοπιστίας, όπως είναι για παράδειγμα οι κρίσιμες υποδομές μιας χώρας, τα αυτόνομα οχήματα, η αεροπλοΐα και οι διαστημικές εφαρμογές, τότε η κρυπτογράφηση θα πρέπει να ακολουθεί ιδιαίτερα υψηλές απαιτήσεις που θα την καθιστούν αξιόπιστη και ασφαλή. Εδώ μπορούν να τεθούν ορισμένα ερωτήματα που εύλογα γεννιούνται, πόσο οχυρωμένα και δοκιμασμένα είναι αυτά τα ψηφιακά συστήματα και οι υποδομές από πλευράς ασφάλειας και αξιοπιστίας; Δεδομένα (ως επί το πλείστον ευαίσθητα - προσωπικά) που επεξεργάζονται και καταχωρούνται σε αυτά τα ψηφιακά προϊόντα ή διαχειρίζονται από σύγχρονες ψηφιακές υπηρεσίες επικοινωνιών πόσο ασφαλή είναι ή πόσο ευάλωτα είναι σε διαρροές; Έχουν δοκιμαστεί κατάλληλα για αυτό; Ποιες τεχνικές χρησιμοποιούνται για την προστασία τους;

Συναρτήσει όμως της αξιοπιστίας και ασφάλειας της κρυπτογράφησης θα πρέπει να συνυπολογιστεί και η απόδοση αυτών των κρυπτογραφικών μεθόδων και τεχνικών. Στην περίπτωση που ένας κοινός επεξεργαστής καταναλώνει μεγάλο ποσοστό της λειτουργίας του στο να υποστηρίζει τις κρυπτογραφικές λειτουργίες τότε θα σπαταλά πολλούς υπολογιστικούς πόρους με αποτέλεσμα να επιβραδύνεται η απόδοσή του και οι λοιπές λειτουργίες. Εξαιρετικά σημαντικό είναι το που λαμβάνει χώρα η κρυπτογράφηση, αν γίνεται δηλαδή σε επίπεδο λογισμικού (software) ή σε επίπεδο υλικού (hardware). Σε επίπεδο υλικού φαίνεται να είναι πιο κοστοβόρα. Για το λόγο αυτό και προκειμένου να υπάρχει η καλύτερη δυνατή απόδοση και πολλαπλάσιες επιδόσεις, η εκτέλεση αυτών των κρυπτογραφικών λειτουργιών επιλέγεται να γίνονται από κρυπτογραφικούς επιταχυντές (accelerators).

Με την πάροδο των χρόνων τα ψηφιακά κυκλώματα χρησιμοποιούνται ολοένα και σε πιο προηγμένες τεχνολογίες κάτι που βοηθάει στο να μικραίνουν σε όγκο τα ηλεκτρονικά στοιχεία και κατ' επέκταση και ο συνολικός όγκος των κυκλωμάτων. Τα κυκλώματα αυτά κατασκευάζονται συνήθως υπό την μορφή ολοκληρωμένων κυκλωμάτων (ICs) τα οποία προγραμματίζονται ώστε να πραγματοποιούν μία σειρά από πολύπλοκες λειτουργίες. Το ζητούμενο λοιπόν είναι να παραχθεί ένα αξιόπιστο και ασφαλές υλικό (hardware) που θα αποτελέσει το συστατικό στοιχείο για την παραγωγή αξιόπιστων και ασφαλών ψηφιακών κυκλωμάτων. Επομένως για την εκπλήρωση των προαναφερθέντων απαιτήσεων υλικού θα

πρέπει από το στάδιο της σχεδίασης έως το στάδιο της παραγωγής του, αυτό να αποδοθεί με τον καλύτερο δυνατό τρόπο.

Οι γλώσσες περιγραφής υλικού, όπως είναι η VHDL και η Verilog είναι γλώσσες οι οποίες είναι κατάλληλες για τη σχεδίαση ψηφιακών κυκλωμάτων και μπορούν να περιγράφουν τόσο τη συμπεριφορά όσο και τη δομή ενός κυκλώματος. Οι γλώσσες αυτές διευκολύνουν τη σχεδίαση σύνθετων ψηφιακών κυκλωμάτων μέσω της περιγραφής τόσο της συμπεριφοράς όσο και της δομής του υλικού, αυτό όμως τις καθιστά λιγότερο αποδοτικές από άλλες γλώσσες υψηλού επιπέδου όπως για παράδειγμα είναι η C, η C++ και η System C. Αυτό οδήγησε στη χρήση γλωσσών προγραμματισμού υψηλού επιπέδου High Level Languages (HLL) οι κώδικες των οποίων μπορούν να επαναχρησιμοποιηθούν μερικώς ή ολικώς λόγω της ευρείας και ευκολότερης χρήσης τους.

Ο συνδυασμός της σύγχρονης αρχιτεκτονικής των ψηφιακών συστημάτων αλλά και των γλωσσών HLL για την παραγωγή υλικού, έχουν βοηθήσει στην καλύτερη συνδυαστική χρήση του κώδικα που αναπτύσσεται με αποτέλεσμα την αύξηση της παραγωγικότητας αφού ο κώδικας μπορεί να διαμοιραστεί και να χρησιμοποιηθεί, εξοικονομώντας χρόνο αλλά και πόρους.

Για να είναι εφικτή η σύνθεση κυκλωμάτων που έχουν περιγραφεί με γλώσσες υψηλού επιπέδου δημιουργήθηκαν εργαλεία Σύνθεσης Υψηλού Επιπέδου (High Level Synthesis – HLS). Τα εργαλεία HLS αναλαμβάνουν, από το περιγραφόμενο υλικό σε γλώσσα υψηλού επιπέδου, να συνθέσουν το αντίστοιχο κύκλωμα στο επίπεδο μεταφοράς καταχωρητών (Register Transfer Level – RTL). Η παραγόμενη περιγραφή στο RTL γίνεται σε VHDL ή/και Verilog και με αυτό τον τρόπο το αποτέλεσμα των HLS εργαλείων είναι συμβατό με την καθιερωμένη ροή σύνθεσης για την υλοποίηση του τελικού κυκλώματος (FPGA synthesis & implementation). Στην παρούσα διπλωματική χρησιμοποιήθηκε το εργαλείο Vivado HLS.

Το Vivado High-Level Synthesis της Xilinx αποτελεί ένα εργαλείο το οποίο χρησιμοποιείται για το σχεδιασμό, την προσομοίωση και την υλοποίηση (σύνθεση) ενός έργου (project) σε γλώσσα περιγραφής υλικού (HDL). Προσφέρει την αναπαράσταση της ζητούμενης λειτουργικότητας του υλικού (μέσα από την επιβολή καθορισμένων περιορισμών και βελτιστοποιήσεων) αναλαμβάνοντας το ίδιο το εργαλείο να προσομοιώσει και να δημιουργήσει αυτόματα μία σύνθεση σε γλώσσα περιγραφής υλικού η οποία στη συνέχεια μέσω κατάλληλου εργαλείου οδηγεί στην υλοποίηση του υλικού με FPGA ή ASICs.

Κατά τη σχεδίαση ενός ψηφιακού κυκλώματος δημιουργούνται διάφορες υλοποιήσεις, με διαφορετικές βελτιστοποιήσεις (οδηγίες-ντιρεκτίβες) κάθε φορά ώστε να μελετηθεί ποια από όλες τις υλοποιήσεις πληροί καλύτερα τις απαιτούμενες προδιαγραφές. Εφόσον έχει παραχθεί το υλικό τότε μία διαδικασία μελέτης της ασφάλειας και της αξιοπιστίας του υλικού είναι η πραγματοποίηση εκτεταμένων εισαγωγών σφαλμάτων (Fault Injections) με στόχο τον χαρακτηρισμό του υπό μελέτη κυκλώματος. Έπειτα ανάλογα με τα αποτελέσματα αυτής της μελέτης συναξιολογώντας τις απαιτήσεις για ασφάλεια και αξιοπιστία καθορίζεται αν χρειάζεται η ενσωμάτωση αντιμέτρων και τεχνικών προστασίας για την βελτίωση της ανθεκτικότητας του κυκλώματος. Είναι προφανές ότι όσο πιο νωρίς γίνει η μελέτη τόσο περισσότεροι θα είναι οι σχεδιαστικοί πόροι και τα υλικά που θα εξοικονομηθούν. Για αυτό το λόγο στην παρούσα διατριβή εστιάζουμε στην εισαγωγή σφαλμάτων μέσω προσομοίωσης στο RTL καθώς θα μας επιτρέψει να μελετήσουμε την ανθεκτικότητα των υπό μελέτη κυκλωμάτων στην αρχή της σχεδιαστικής ροής.

Τα υπό μελέτη κυκλώματα που χρησιμοποιήθηκαν για την εισαγωγή σφαλμάτων αποτελούν προϊόν μίας άλλης Μεταπτυχιακής Διατριβής [1] και παρουσιάζονται στο Κεφάλαιο 8. Στα κυκλώματα αυτά, τα οποία είναι υλοποιημένα με τρεις διαφορετικές απαιτήσεις βελτιστοποιήσεων (Vivado HLS default optimization, loop unrolling, no-inline) εισήχθησαν σφάλματα σε στοιχεία μνήμης (flip flop). Τα μοντέλα σφαλμάτων που εφαρμόστηκαν είναι η εξαντλητική εισαγωγή μονών σφαλμάτων (Exhaustive Single Bit Flip), καθώς και η εισαγωγή τυχαίων πολλαπλών σφαλμάτων Multiple Bit Flip με τη χρήση στατιστικής μεθόδου η οποία αναλύεται στο Κεφάλαιο 5 [2].

Για τη μέθοδο της εξαντλητικής εισαγωγής μονών σφαλμάτων (SBF) έγινε εισαγωγή σφαλμάτων σε όλα τα flip flop για όλους τους κύκλους ρολογιού (clock cycle). Για την εισαγωγή

σφαλμάτων με τη μέθοδο Multiple Bit Flip ο αριθμός όλων των πιθανών συνδυασμών είναι τόσο μεγάλος που πρακτικά δεν είναι εφικτή η δυνατότητα εισαγωγής όλων αυτών των σφαλμάτων. Προκειμένου να ολοκληρωθεί η εισαγωγή τους σε εύλογο χρονικό διάστημα εφαρμόστηκε η στατιστική μέθοδος [2].

Τα αποτελέσματα που ελήφθησαν παραθέτονται στο Κεφάλαιο 10 όπου συγκρίνονται τόσο οι σχεδιάσεις Designs μεταξύ τους όσο και η εφαρμογή των διαφορετικών βελτιστοποιήσεων και πώς όλα αυτά συμπεριφέρονται υπό την επίδραση σφαλμάτων. Σκοπός είναι να ανευρεθεί η προτιμότερη μέθοδος εισαγωγής σφαλμάτων με γνώμονα τη μείωση των χρονοβόρων δοκιμών για την ανεύρεση τυχόν ευπαθειών υλικού. Εν τέλει με αυτό τον τρόπο δύναται να εξακριβωθεί αν η εφαρμογή επιπλέον αντιμέτρων θα επηρέαζε τα αποτελέσματα των εισαγόμενων σφαλμάτων (FI).

Στο Κεφάλαιο 2 αναλύεται η ανάγκη ασφαλών σύγχρονων επικοινωνιών και πώς η κρυπτογραφία συμβάλλει σε αυτό. Στη συνέχεια το Κεφάλαιο 3 αναφέρεται στην περιγραφή υλικού με χρήση γλωσσών υψηλού επιπέδου (HLL) και στα βήματα σχεδιασμού με χρήση εργαλείων υψηλού επιπέδου (HLS). Ακολούθως στο Κεφάλαιο 4 αναλύεται η ασφάλεια και η αξιοπιστία υλικού με τη χρήση εισαγωγής σφαλμάτων, ενώ στο Κεφάλαιο 5 περιγράφονται τα μοντέλα σφαλμάτων που μπορούν να χρησιμοποιήθηκαν για την εισαγωγή σφαλμάτων, δύο εκ των οποίων χρησιμοποιήθηκαν στην παρούσα Διατριβή. Στο Κεφάλαιο 6 παρουσιάζεται το εργαλείο Vivado HLS το οποίο χρησιμοποιήθηκε για το σχεδιασμό και την προσομοίωση εισαγωγής σφαλμάτων των υπό μελέτη σχεδιάσεων, οι οποίες ως βασικό στοιχείο έχουν την περιγραφή του κρυπτογραφικού αλγορίθμου Advanced Encryption Standard (AES), για τον οποίο μιλά το Κεφάλαιο 7. Στο Κεφάλαιο 8 γίνεται ανάλυση των υπό αξιολόγηση κυκλωμάτων στα οποία έχουν εφαρμοστεί διαφορετικά αντίμετρα. Τέλος, στο Κεφάλαιο 9 περιγράφεται ο υπολογισμός των δειγμάτων καθώς και το σύστημα εισαγωγής σφαλμάτων που εφαρμόστηκε στα υπό μελέτη κυκλώματα με τα αποτελέσματα της αξιολόγησης αυτών να παρατίθενται στο Κεφάλαιο 10. Τα συμπεράσματα και οι προοπτικές συζητούνται στο Κεφάλαιο 11.

2. Ασφαλείς Σύγχρονες Επικοινωνίες και Κρυπτογραφία

Οι συνεχώς αυξανόμενες απαιτήσεις της σύγχρονης ψηφιακής εποχής έχουν ανεβάσει κατακόρυφα τη ζήτηση, τη χρήση αλλά και τις απαιτήσεις γύρω από το κομμάτι της επικοινωνίας. Η πλειοψηφία των ψηφιακών μέσων χρησιμοποιούνται καθημερινά για όλο το φάσμα των ανθρώπινων επικοινωνιών είτε πρόκειται για προσωπικές, κοινωνικές, επαγγελματικές, εμπορικές και άλλες επικοινωνίες. Η ανάγκη αυτή έχει οδηγήσει τους χρήστες στη ραγδαία αύξηση της χρήσης διαφόρων τύπων συσκευών (κυρίως «έξυπνων» συσκευών - smart devices), η συντριπτική πλειοψηφία των οποίων είναι συνδεδεμένες με το διαδίκτυο και τις περισσότερες φορές και μεταξύ τους (συγχρονισμένες) ώστε να ανασύρονται οι πληροφορίες που απαιτεί ο χρήστης από οποιαδήποτε συσκευή οποιαδήποτε στιγμή. Οι καθημερινές απαιτήσεις μπορεί να αφορούν από τη χρήση ενός «έξυπνου» τηλεφώνου για την οποιαδήποτε μορφή επικοινωνίας μέσω τηλεφώνου, γραπτών μηνυμάτων, ηλεκτρονικού ταχυδρομείου, κοινωνικών μέσων δικτύωσης ως την παρακολούθηση της σωματικής δραστηριότητας του ατόμου ή τη διαχείριση της εξ αποστάσεως λειτουργίας των οικιακών ηλεκτρικών συσκευών. Αυτές οι δραστηριότητες συμβάλλουν στην καθημερινή διευκόλυνση απλών ή και πιο σύνθετων ζητημάτων σε ατομικό ή και συλλογικό επίπεδο.

Όταν μιλάμε για επικοινωνία φυσικά δεν εννοούμε μόνο την επικοινωνία μεταξύ ανθρώπων αλλά και την επικοινωνία - διασύνδεση μεταξύ υποδομών και συστημάτων. Οι «έξυπνες πόλεις» είναι ένα αντιπροσωπευτικό παράδειγμα επικοινωνίας συστημάτων μεταξύ τους, καθώς μέσω τις διασύνδεσής τους, πραγματοποιείται η «ζωντανή» ρύθμιση της οδικής κυκλοφορίας μιας πόλης ως απόρροια της λειτουργίας του GPS στις ατομικές «έξυπνες» συσκευές οι οποίες χρησιμοποιούνται σε πραγματικό χρόνο, δημιουργώντας μία «αυτορρύθμιση» της κυκλοφορίας του οδικού δικτύου των πόλεων μέσω της ταυτόχρονης επικοινωνίας διάφορων συστημάτων. Επίσης ο βιομηχανικός κόσμος, τραπεζικά καταστήματα, κρατικές υπηρεσίες, κρίσιμες υποδομές και δομές υγείας, συνδέονται άρρηκτα με τον τομέα των σύγχρονων συστημάτων επικοινωνίας που απαιτούν όμως ταυτόχρονα και υψηλού επιπέδου

ασφάλεια των πληροφοριών του εκάστοτε χρήστη. Όλος αυτός ο όγκος πληροφοριών και δεδομένων που βρίσκονται διασυνδεδεμένα σε συστήματα επικοινωνίας που αποθηκεύουν ευαίσθητες πληροφορίες χρήζουν απόλυτης προστασίας και εξασφάλισης της εμπιστευτικότητας, της αξιοπιστίας και της ασφάλειας των επικοινωνιών.

Οι εταιρείες παραγωγής αυτών των έξυπνων συσκευών αλλά και της παροχής υπηρεσιών επικοινωνίας έχουν σπεύσει να καλύψουν αυτές τις ανάγκες τόσο με την αναβάθμιση των προϊόντων - υπηρεσιών τους, όσο και για εμπορικούς και οικονομικούς λόγους δημιουργώντας νέα ψηφιακά προϊόντα – συσκευές αλλά και υπηρεσίες (νέφους - cloud), αναβαθμίζοντας ταυτόχρονα τις υποδομές και το δίκτυό τους. Εδώ μπορούν να τεθούν ορισμένα ερωτήματα που εύλογα γεννιούνται, πόσο οχυρωμένα και δοκιμασμένα είναι αυτά τα νέα ψηφιακά προϊόντα και οι υποδομές των παρεχόμενων υπηρεσιών σε ενδεχόμενες προσπάθειες εκμετάλλευσης δεδομένων από τρίτους; Δηλαδή δεδομένα (ως επί το πλείστων ευαίσθητα - προσωπικά) που καταχωρούνται σε αυτά τα νέα ψηφιακά προϊόντα ή διαχειρίζονται από τις παρεχόμενες σύγχρονες υπηρεσίες επικοινωνιών πόσο ασφαλή είναι ή πόσο ευάλωτα είναι σε διαρροές; Έχουν δοκιμαστεί κατάλληλα για αυτό; Ποιες τεχνικές χρησιμοποιούνται για την προστασία των δεδομένων; Διότι τα δεδομένα αυτά μπορεί να προσεγγίσουν αρνητικά το ενδιαφέρον κακόβουλων τρίτων για την απόκτησή τους με σκοπό το κέρδος, τον εκβιασμό, την απαίτηση καταβολής χρηματικών ποσών, την διαρροή τους σε τρίτους, την παραποίησή τους για την κάλυψη τυχόν έκνομων ενεργειών κ.α. Το εύρος των κινήτρων ενός κακόβουλου ατόμου είναι σαφώς μεγάλο καθώς και το φάσμα δυνατοτήτων τέλεσης κάποιας επίθεσης για την υποκλοπή επικοινωνιών και την απόκτηση πληροφοριών.

2.1 Ο ρόλος της κρυπτογράφησης στις σύγχρονες επικοινωνίες

Η διαδικασία της επικοινωνίας προϋποθέτει ότι υπάρχουν τουλάχιστον δύο μέρη – οντότητες (για παράδειγμα ένας αποστολέας και ένας παραλήπτης) οι οποίοι επιθυμούν να επικοινωνήσουν μεταξύ τους και να ανταλλάξουν πληροφορίες. Εύλογα γίνεται λοιπόν αντιληπτό ότι υπάρχει μεγάλη ανάγκη να εξασφαλιστεί η ασφάλεια των διαφόρων πληροφοριών των επικοινωνιών, κάτι το οποίο προσφέρεται μέσω της κρυπτογραφίας με τη διαδικασία της κρυπτογράφησης. Πιο συγκεκριμένα οι πληροφορίες που διακινούνται θα πρέπει να προστατεύονται ως προς την εμπιστευτικότητα (confidentiality) και την ακεραιότητά (Integrity) τους. Επίσης άλλα χαρακτηριστικά που θα πρέπει να έχει η επικοινωνία για να είναι σίγουρα ασφαλείς είναι η αυθεντικοποίηση (Authentication) των μερών που συμμετέχουν σε αυτήν, η διαθεσιμότητα (Availability) των δεδομένων – υπηρεσιών αλλά και η εξασφάλιση της ποιότητας – αξιοπιστίας (Reliability) των υπηρεσιών. Επιπλέον κανένα από τα μέρη που επικοινωνούν να μην μπορεί να αποποιηθεί την ευθύνη (Non – Repudiation) ότι απέστειλε ή έλαβε δεδομένα της επικοινωνίας [3]. Οι παραπάνω ιδιότητες αποτελούν βασικές αρχές για την διατήρηση της προστασίας των δεδομένων από τις πιο απλές έως τις πλέον σύνθετες και μυστικές επικοινωνίες.

Ως προς την εμπιστευτικότητα, οι πληροφορίες αυτές θα πρέπει απαραιτήτως να προστατευτούν και να διασφαλιστεί ότι σε περίπτωση που κάποιος τρίτος (κακόβουλος) προσπαθήσει να λάβει γνώση αυτών, δεν θα μπορέσει να κατανοήσει το περιεχόμενό τους ακόμη και αν καταφέρει και τις αποκτήσει με οποιονδήποτε τρόπο. Αυτό συμβαίνει όταν τα δεδομένα της επικοινωνίας κρυπτογραφούνται, δηλαδή μετατρέπονται σε μία «μη αναγνώσιμη» μορφή, για όποιον τρίτο δεν αποτελεί νόμιμο μέρος της επικοινωνίας. Μόνο ο δημιουργός, αποστολέας ή παραλήπτης μπορούν να αποκρυπτογραφήσουν τα δεδομένα χρησιμοποιώντας μία προ-συμφωνημένη τεχνική διαδικασία (είδος κρυπτογράφησης, κρυπτογραφικός αλγόριθμος).

Ο κρυπτογραφικός αλγόριθμος αποτελεί το μέσο για την κωδικοποίηση των δεδομένων, δηλαδή την μετατροπή τους σε μία «μη αναγνώσιμη» μορφή για όποιον κακόβουλο τρίτο προσπαθήσει να τα διαβάσει. Η κρυπτογράφηση μπορεί να είναι είτε συμμετρική δηλαδή να χρησιμοποιείται το ίδιο μυστικό κλειδί από όλα τα νόμιμα μέρη που επικοινωνούν, είτε ασύμμετρη με τη χρήση διαφορετικών κλειδιών από τους χρήστες (ζεύγη κλειδιών ήτοι ιδιωτικό – δημόσιο κλειδί). Οποιαδήποτε μορφή κρυπτογράφησης επιλεγεί, το σημαντικό είναι σε κάθε

περίπτωση να φυλάσσεται και να διατηρείται η μυστικότητα των κλειδιών που χρησιμοποιούνται. Ειδικότερα, στην περίπτωση της συμμετρικής κρυπτογραφίας θα πρέπει ο τρόπος (ηλεκτρονικά ή φυσικά) που ανταλλάσσεται το κοινό κρυπτογραφικό κλειδί να γίνεται κάτω από απόλυτη μυστικότητα και προσοχή αποφυγής διαρροών. Συχνά οι διαρροές οφείλονται στην ανθρώπινη αμέλεια. Αναφορικά με τη διαδικασία κρυπτογράφησης των δεδομένων, συνήθως γίνεται σε τμήματα δεδομένων (block), σπανιότερα ανά χαρακτήρα (bit). Η συνεχής κρυπτογράφηση ανά στοιχείο είναι ασύμφορη καθώς απαιτείται να θυσιαστεί χρόνος και μεγάλη υπολογιστική ισχύς. Ο αποστολέας και ο παραλήπτης θα πρέπει να έχουν βρει τρόπο ώστε να έχουν ανταλλάξει πριν την επικοινωνία τους ένα μυστικό κλειδί του ίδιου μήκους, κάτι ανέφικτο για τον κανόνα των επικοινωνιών [4]. Ενδεικτικά, ορισμένοι συμμετρικοί αλγόριθμοι κρυπτογράφησης είναι οι AES, DES, TripleDES, RC5, Blowfish, Twofish, IDEA, BC3 ενώ ασύμμετροι οι Diffie – Hellman, RSA, DSS, ECC κ.α.

Ως προς την ακεραιότητα των δεδομένων που μεταδίδονται αυτό επιτυγχάνεται μέσα από διάφορες μεθόδους επιβεβαίωσης μία εκ των οποίων είναι και οι συναρτήσεις κατακερματισμού (hash values). Οι παραλήπτες των δεδομένων χρησιμοποιώντας αυτές τις συναρτήσεις μπορούν να επαληθεύσουν αν τα δεδομένα που έχουν λάβει είναι εκείνα που τους έστειλε ο νόμιμος αποστολέας. Για παράδειγμα, αν έχει πραγματοποιηθεί η παραμικρή αλλαγή (όσο ασήμαντη και αν είναι) στο κείμενο που έχει αποσταλεί (plain text), τότε η τιμή της συνάρτησης κατακερματισμού θα διαφέρει αισθητά από την αναμενόμενη τιμή την οποία έχει υπολογίσει και στείλει ο αποστολέας για να μπορέσει να γίνει η επαλήθευση στο άλλο άκρο της επικοινωνίας (παραλήπτης). Υπάρχουν διάφορες συναρτήσεις κατακερματισμού όπως MD5, SHA1, SHA2 (SHA224, SHA256, SHA384, SHA512), SHA3, KDF κ.α. Ως προς την υποστήριξη της ασφάλειας της επικοινωνίας στο επίπεδο συστημάτων αυτό το προσφέρουν πρωτόκολλα όπως το SSL/TLS και PGP.

Ως προς την αυθεντικοποίηση των χρηστών αυτή επιτυγχάνεται πιστοποιώντας την ταυτότητα των χρηστών που πρόκειται να επικοινωνήσουν. Η ταυτοποίηση τους γίνεται πριν την έναρξη της μετάδοσης των δεδομένων, ώστε να έχουν σιγουρευτεί τα δύο μέρη ότι επικοινωνούν με το σωστό άτομο. Ως προς την αποποίηση ευθύνης θα πρέπει κανένα από τα εμπλεκόμενα μέρη να μην μπορεί να αρνηθεί ότι απέστειλε ή παρέλαβε δεδομένα κατά τη διάρκεια της μεταξύ τους επικοινωνίας. Η ψηφιακή υπογραφή υποστηρίζει εκτός της ακεραιότητας των δεδομένων και της αυθεντικοποίησης της ταυτότητας του χρήστη και την μη αποποίηση της ευθύνης του ατόμου που υπέγραψε. Για τη δημιουργία της ψηφιακής υπογραφής απαιτείται η υλοποίηση ασύμμετρης διαδικασίας κρυπτογράφησης με τη χρήση αλγορίθμων όπως DSA και RSA.

Τέλος όλα τα επικοινωνιακά συστήματα θα πρέπει να είναι οχυρωμένα απέναντι σε επιθέσεις που έχουν ως σκοπό την υποκλοπή δεδομένων ή τη διατάραξη της ομαλής λειτουργίας τους. Τέτοιες επιθέσεις μπορεί να στοχεύουν στο να διακόψουν εντελώς τη λειτουργία του συστήματος ή να το θέσουν προσωρινά εκτός λειτουργίας για κάποια χρονική περίοδο.

Από όλα τα προαναφερθέντα συμπεραίνεται ότι η αναγκαιότητα χρήσης της κρυπτογράφησης είναι επιβεβλημένη και η εφαρμογή της θα πρέπει να συνοδεύεται από ορισμένες δικλείδες ασφαλείας που θα εξασφαλίζουν την ορθή υλοποίηση των κρυπτογραφικών διαδικασιών. Επίσης θα πρέπει να συνυπολογιστεί και η απόδοση αυτών των κρυπτογραφικών μεθόδων και τεχνικών. Εάν ένας κοινός επεξεργαστής απασχολείται σε μεγάλο ποσοστό της λειτουργίας του στο να υποστηρίζει τις κρυπτογραφικές λειτουργίες τότε θα καταναλώνει πολλούς υπολογιστικούς πόρους με αποτέλεσμα να επιβραδύνεται η απόδοσή του και οι λοιπές λειτουργίες. Επίσης εξαιρετικά σημαντικό είναι το που λαμβάνει χώρα η κρυπτογράφηση, αν γίνεται σε επίπεδο λογισμικού (software) ή σε επίπεδο υλικού (hardware). Σε επίπεδο λογισμικού είναι πολύ πιο χρονοβόρα και κοστοβόρα. Για το λόγο αυτό και προκειμένου να υπάρχει η καλύτερη δυνατή απόδοση και πολλαπλάσιες επιδόσεις, η εκτέλεση αυτών των κρυπτογραφικών λειτουργιών επιλέγεται να γίνεται από κρυπτογραφικούς επιταχυντές (accelerators) υλικού [5].

Βασικοί μετασχηματισμοί που χρησιμοποιούνται στους κρυπτογραφικούς αλγόριθμους είναι εκείνοι της αντικατάστασης, της μετάθεσης - μετατόπισης. Προκειμένου να πραγματοποιηθούν οι υπολογισμοί που απαιτούνται για την κρυπτογράφηση εκτελούνται

λογικές πράξεις όπως AND, XOR, OR κ.α. λαμβάνοντας υπόψιν τους πίνακες αληθείας (look up tables) για τον υπολογισμό των τιμών που παίρνει κάθε στοιχείο (bit). Σε επίπεδο υλικού η υλοποίησή τους σαφώς και είναι αποδοτικότερη ειδικότερα όταν διαχειρίζονται μεγάλο όγκο δεδομένων σε πραγματικό χρόνο (real time). Τέτοιες υλοποιήσεις συναντάμε για παράδειγμα σε κυκλώματα FPGAs.

Αν γίνει αναγωγή σε ένα ευρύτερο επίπεδο εφαρμογών υψηλών απαιτήσεων αξιοπιστίας , όπως είναι για παράδειγμα οι κρίσιμες υποδομές μιας χώρας, τα αυτόνομα οχήματα, η αεροπλοΐα και οι διαστημικές εφαρμογές, τότε η κρυπτογράφηση θα πρέπει να ακολουθεί ιδιαίτερα υψηλές απαιτήσεις.

3. Περιγραφή ψηφιακών κυκλωμάτων με χρήση εργαλείων σύνθεσης υψηλού επιπέδου (HLS)

Τα ψηφιακά κυκλώματα αποτελούν κυκλώματα που είναι δομημένα από ηλεκτρονικά στοιχεία ή αλλιώς ψηφιακά κυκλώματα που ονομάζονται ψηφιακές (ή λογικές) πύλες. Τέτοιες πύλες είναι για παράδειγμα οι πύλες AND, NOR, OR, XOR, XNOR, NAND κ.α. Στην εποχή μας η χρήση των ψηφιακών κυκλωμάτων γίνεται από την καθαυτού τεχνολογική βιομηχανία, τις μνήμες, τους μικροεπεξεργαστές, έως την κάθε λογής υπολογιστική συσκευή που χρησιμοποιείται για τις καθημερινές επικοινωνίες ή τα πλέον σύγχρονα συστήματα ασφαλείας. Χαρακτηριστικό των κυκλωμάτων αυτών είναι ότι με την πάροδο των χρόνων χρησιμοποιούνται ολοένα και πολυπλοκότερες τεχνολογίες κάτι που βοηθάει στο να μικραίνουν σε όγκο τα ηλεκτρονικά στοιχεία και κατ' επέκταση και ο συνολικός όγκος των κυκλωμάτων.

Η λειτουργία των ψηφιακών κυκλωμάτων βασίζεται στη δυαδική λογική, δηλαδή σαν είσοδο και έξοδο των κυκλωμάτων τα δεδομένα αναπαρίστανται με μία ακολουθία των ψηφίων 0 ή 1. Για τον υπολογισμό των τιμών λειτουργίας ενός ψηφιακού κυκλώματος χρησιμοποιείται η δίτιμη άλγεβρα Boole με την οποία υπολογίζουμε τους πίνακες αληθείας (ή αλλιώς truth tables) της εξόδου του κυκλώματος. Τα δεδομένα επεξεργάζονται από συνδυαστικά ψηφιακά κυκλώματα και αποθηκεύονται στα επί μέρους ψηφιακά στοιχεία (π.χ. κύκλωμα μνήμης με Flip-Flop) που απαρτίζουν το κύκλωμα. Τα Flip-Flop αποτελούν συστατικό στοιχείο και των καταχωρητών (Registers) και γενικότερα όλων εκείνων των κυκλωμάτων που εμπεριέχουν κάποια μορφή μνήμης.

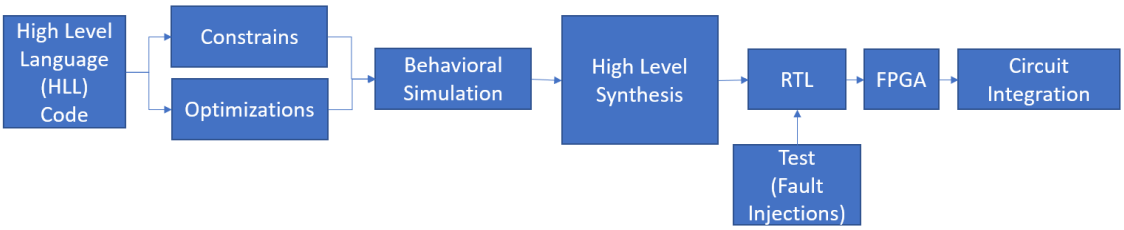
Εφόσον βρισκόμαστε σε επίπεδο υλικού είναι προφανές ότι τα σήματα που «καταλαβαίνει» ένα ψηφιακό κύκλωμα είναι σήματα τάσης με το ψηφίο '0' να αντιστοιχεί στο χαμηλό επίπεδο και το ψηφίο '1' να αντιστοιχεί στο υψηλό επίπεδο τάσης. Επιγραμματικά αναφέρεται ότι τα ψηφιακά κυκλώματα χωρίζονται σε σύγχρονα και ασύγχρονα ακολουθιακά κυκλώματα. Η διαφορά τους είναι ότι στα σύγχρονα χρησιμοποιείται ρολόι, υπάρχει δηλαδή μία γεννήτρια παλμών (τετραγωνικής μορφής) που τροφοδοτεί και οδηγεί τις εισόδους – εξόδους και τις ενδιάμεσες λειτουργίες σε συγκεκριμένο χρόνο.

3.1 Γλώσσες υψηλού επιπέδου – High Level Language (HLL)

Οι γλώσσες περιγραφής υλικού, όπως είναι η VHDL και η Verilog είναι γλώσσες οι οποίες είναι κατάλληλες για τη σχεδίαση ψηφιακών κυκλωμάτων και μπορούν να περιγράφουν τόσο τη συμπεριφορά όσο και τη δομή ενός κυκλώματος. Οι γλώσσες αυτές διευκολύνουν τη σχεδίαση σύνθετων ψηφιακών κυκλωμάτων μέσω της περιγραφής της συμπεριφοράς του υλικού αλλά το γεγονός ότι είναι κατασκευασμένες και για την περιγραφή της δομής και της λειτουργίας του υλικού τις καθιστά πιο δύσχρηστες από γλώσσες υψηλού επιπέδου όπως για παράδειγμα η C και η C++.

Έτσι η ολοένα και αυξανόμενη πολυπλοκότητα των ψηφιακών κυκλωμάτων για σύγχρονες εφαρμογές οδήγησε στη χρήση γλωσσών προγραμματισμού υψηλού επιπέδου (HLL) οι κώδικες των οποίων μπορούν να επαναχρησιμοποιηθούν μερικώς ή ολικώς. Η σύγχρονη αρχιτεκτονική των ψηφιακών συστημάτων αλλά και οι γλώσσες HLL έχουν βοηθήσει

στην καλύτερη συνδυαστική χρήση του παραγόμενου κώδικα με αποτέλεσμα την αύξηση της παραγωγικότητας αφού ο κώδικας μπορεί να διαμοιραστεί και να χρησιμοποιηθεί πάλι από άλλους ενδιαφερόμενους, εξοικονομώντας χρόνο και φυσικά πόρους. Επιπλέον βελτιώνεται η ποιότητα του κώδικα αφού τροποποιείται και εκσυγχρονίζεται κάθε φορά που επαναχρησιμοποιείται. Τέτοιες γλώσσες υψηλού επιπέδου είναι και οι γλώσσες C, C++ και System C. Για να είναι εφικτή η σύνθεση κυκλωμάτων που έχουν περιγραφεί με γλώσσες υψηλού επιπέδου δημιουργήθηκαν εργαλεία Σύνθεσης Υψηλού Επιπέδου (High Level Synthesis – HLS). Τα εργαλεία HLS αναλαμβάνουν, δεδομένης μιας περιγραφής υλικού με μια γλώσσα υψηλού επιπέδου, να συνθέσουν το περιγραφόμενο κύκλωμα στο επίπεδο μεταφοράς καταχωρητών (Register Transfer Level – RTL). Η παραγόμενη περιγραφή στο RTL γίνεται σε VHDL ή/και Verilog και με αυτό τον τρόπο το αποτέλεσμα των HLS εργαλείων είναι συμβατό με το την καθιερωμένη ροή σύνθεσης για την υλοποίηση του τελικού κυκλώματος (FPGA synthesis & implementation) (Εικόνα 1). Στην παρούσα διατριβή χρησιμοποιήθηκε το εργαλείο Vivado HLS για την σύνθεση κυκλωμάτων για Xilinx FPGAs.



Εικόνα 1Διαδικασία σύνθεσης με HLS

Στις HLL περιγραφές οι καταχωρητές δηλώνονται ως μεταβλητές και χρησιμοποιούνται γνωστές προγραμματιστικές δομές όπως while, for, if για την περιγραφή της λογικής λειτουργίας τους. Αυτό βοηθάει σε όλα τα επίπεδα προετοιμασίας, σχεδίασης, προσομοίωσης και επαλήθευσης. Έτσι οι κατασκευαστικές εταιρείες γλιτώνουν από τον κόπο υλοποίησης πολλών πειραματικών πρωτοτύπων ενώ παράλληλα μειώνονται και τα έξοδα προσομοιώσεων και δοκιμών των παραγόμενων τεχνολογικών προϊόντων.

3.2 Βήματα σχεδιασμού σε HLS

Η διαδικασία σχεδιασμού ενός σχεδιαστικού κυκλώματος σε γλώσσα HLS ακολουθεί κάποια βασικά βήματα [6] όπως:

- Η χρήση κάποιας γλώσσας υψηλού επιπέδου για την αναπαράσταση των προδιαγραφών και την αποτύπωση των λειτουργιών και των δεδομένων, ως ροή (data flow), που θα εκτελεί το σχεδιαστικό μοντέλο. Το μοντέλο κυκλώματος που περιγράφεται, οργανώνεται σε block που βοηθούν στην αναπαράσταση διαφορετικών παράλληλων λειτουργιών και των μεταξύ τους εξαρτήσεων. Σε αυτό το σημείο πραγματοποιούνται τυχόν βελτιστοποιήσεις (optimization) με στόχο τον περιορισμό των καθυστερήσεων και την ταχύτερη απόδοση του κυκλώματος. Τέτοιες τεχνικές βελτιστοποίησης είναι για παράδειγμα οι συγχωνεύσεις, οι μετασχηματισμοί βρόγχων (ακολουθιακών - loop unrolling ή ευθυγραμμισμένων loop pipelining), η απαλοιφή του κώδικα που δεν χρησιμοποιείται ή η πρόβλεψη και εξάλειψη πιθανών σφαλμάτων για ασαφή- εσφαλμένα δεδομένα.

- Στο επόμενο στάδιο γίνεται η κατανομή (allocation) των μονάδων του υλικού στις λειτουργίες που περιγράφονται στον αλγόριθμο καθώς και οι δίαυλοι συνδέσεων. Επίσης λαμβάνονται υπόψιν οι όποιοι περιορισμοί ως προς τη σχεδίαση και λειτουργία του κυκλώματος. Σε αυτό το βήμα εισάγονται και οι προαπαιτούμενες βιβλιοθήκες (τεχνικά χαρακτηριστικά των στοιχείων).

- Επακόλουθο του προηγούμενου σταδίου είναι και η διαδικασία υπολογισμού του χρονισμού του κυκλώματος, δηλαδή ο προγραμματισμός των λειτουργιών του σχεδιαστικού κυκλώματος (scheduling) ως προς τους απαιτούμενους κύκλους ρολογιού. Με άλλα λόγια κάθε κύκλος ρολογιού αντιστοιχίζεται σε κάποιες συγκεκριμένες λειτουργίες οι οποίες μπορεί να εκτελούνται διαδοχικά ή παράλληλα, εφόσον δεν υπάρχουν εξαρτώμενες λειτουργίες – περιορισμοί και επιτρέπεται η επεξεργασία δεδομένων που δεν χρησιμοποιούνται αλλού. Μπορεί τα δεδομένα εξόδου μίας λειτουργίας να αποτελούν δεδομένα εισόδου κάποιας άλλης, οπότε όλα συνυπολογίζονται. Επιλέγεται η λιγότερο δαπανηρή διαδικασία εκτέλεσης, δηλαδή εκείνη που προσφέρει την καλύτερη δυνατή απόδοση, λαμβάνοντας υπόψιν τους διαθέσιμους πόρους.

- Τα στάδια σχεδιασμού ολοκληρώνονται με τη συστοίχιση και δέσμευση (binding) όλων των ηλεκτρονικών στοιχείων και μονάδων του μοντέλου (τα οποία έχουν ήδη καθοριστεί) με τις αντίστοιχες λειτουργίες τους. “Δένει” δηλαδή όλα τα λειτουργικά στοιχεία με τις αντίστοιχες λειτουργίες για τις οποίες προορίζονται και τις βελτιστοποιεί στο σύνολό τους. Οι πόροι που απαιτούνται για την υλοποίηση του σχεδιαστικού μοντέλου καταμερίζονται αναλόγως και αν κάποιες λειτουργίες εκτελούνται σε διαφορετικούς κύκλους ρολογιού, τότε μπορεί να δεσμεύσουν σε διαφορετικό χρόνο τους ίδιους πόρους. Επίσης δημιουργούνται δεσμευτικές συνδέσεις των μεταβλητών (π.χ. καταχωρητές) στις μνήμες ανάλογα με το κάθε πότε και από ποιον χρησιμοποιούνται καθώς και το που θα αποθηκεύονται οι ενδιάμεσες τιμές των καταχωρητών που προκύπτουν. Επιπρόσθετα προδιαγράφονται οι συνδέσεις των διαύλων επικοινωνίας μεταξύ των πόρων. Όλα τα παραπάνω είναι αλληλεξαρτώμενα.

- Εν συνεχεία με την ολοκλήρωση όλων των προηγούμενων σταδίων, συντίθεται το σχεδιαστικό μοντέλο αρχιτεκτονικής register-transfer level (RTL). Το οποίο μοντελοποιεί το σχέδιο του εν δυνάμει ψηφιακού κυκλώματος με όλα τα λειτουργικά του στοιχεία και με τις προαναφερθείσες βελτιστοποιήσεις. Το μοντέλο RTL αποτελεί καλή πρακτική για τις σχεδιαστικές ανάγκες των σύγχρονων ψηφιακών κυκλωμάτων αφού αναπαρίστανται η ροή των δεδομένων (ψηφιακά σήματα) και σε ποιες λογικές λειτουργίες υποβάλλονται στα εκάστοτε στοιχεία υλικού – καταχωρητές (register).

Συνοψίζοντας με την καθιέρωση γλωσσών HLL όπως η C, μειώνεται αισθητά ο χρόνος που απαιτείται για την ανάπτυξη του προγραμματιστικού μέρους του υλικού, ενώ ο κώδικας μπορεί να ανακυκλωθεί και να επαναχρησιμοποιηθεί αν χρειαστεί. Έτσι επιτυγχάνονται γρηγορότερα και καλύτερα σχεδιαστικά μοντέλα, με λιγότερο κόπο και απλούστερη δομή. Κάτι που είναι εξαιρετικά βοηθητικό στις μετέπειτα δοκιμές επαλήθευσης των λειτουργιών του κυκλώματος πριν την τελική του μορφή.

Από την άλλη πλευρά όμως ο κώδικας που προορίζεται για υλοποίηση υλικού θα πρέπει να διαμορφώνεται όσο το δυνατόν καλύτερα για το σκοπό αυτό, ώστε να μπορεί να μεταγλωττιστεί και να αποδοθεί σωστά από τα εργαλεία υψηλού επιπέδου. Σε διαφορετική περίπτωση θα παραχθεί μίας κακής ποιότητας υλικό. Επιπροσθέτως το μοντέλο RTL που παράγεται με τη χρήση εργαλείων HLS δεν είναι εύκολα διαχειρίσιμο σε επίπεδο κώδικα από τον προγραμματιστή αν προκύψει δυσλειτουργία.

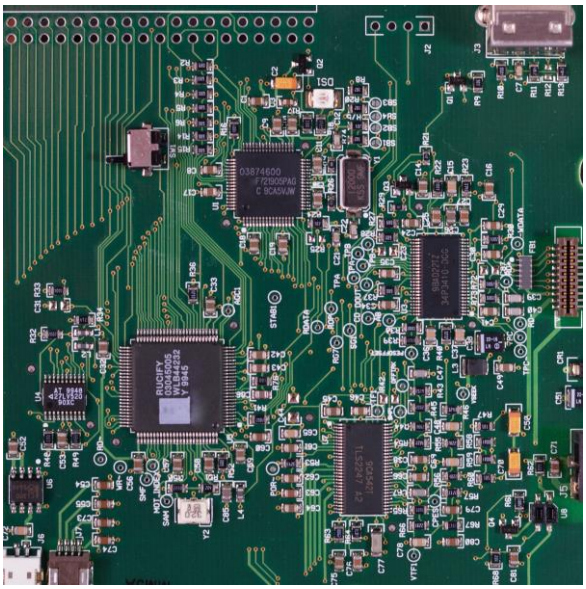
Ένα εργαλείο HLS μπορεί για παράδειγμα να εφαρμόσει βελτιστοποιήσεις στο υλικό οι οποίες έχουν να κάνουν με τη χωροτοποθέτηση στοιχείων το χρόνο εκτέλεσης των λειτουργιών ή την κατανάλωση ισχύος του υλικού. Στην κατηγορία των βελτιστοποιήσεων ανήκει για παράδειγμα το ξετύλιγμα βρόγχων (loop unrolling) το οποίο χρησιμεύει για το όταν υπάρχει στον κώδικα ένας for βρόγχος, τότε δύναται να μπορεί να “ξετυλιχθεί” ώστε να μπορέσουν οι υπολογισμοί να γίνουν ταυτόχρονα, εφόσον τα δεδομένα δεν είναι αλληλεξαρτώμενα μεταξύ τους. Άλλη μορφή βελτιστοποίησης αποτελεί η παραμετροποίηση no-inline δηλαδή η κάθε λειτουργία (function) που εκτελείται θα υλοποιηθεί ως έχει. Επιτρέπεται να γίνουν συγκεκριμένες βελτιστοποιήσεις, αν αυτό χρειάζεται, στο εσωτερικό της κάθε λειτουργίας (function) και όχι από κοινού βελτιστοποιήσεις ανάμεσα σε διαφορετικές λειτουργίες (functions).

4. Ανάλυση της ασφάλειας και αξιοπιστίας υλικού με τη χρήση εισαγωγής σφαλμάτων (Fault Injection)

Η εγγύηση της αξιοπιστίας και της ασφάλειας του υλικού (hardware) αποτελούν καίρια ζητήματα στη σύγχρονη εποχή. Παλαιότερα στον τομέα της ασφάλειας γινόταν εστίαση στην οχύρωση των συστημάτων από πλευράς λογισμικού. Οι επιθέσεις αφορούσαν στην πλειοψηφία τους το λογισμικό και έτσι ο έλεγχος της ασφάλειας των συστημάτων πραγματοποιούνταν γύρω από αυτόν τον τομέα (software security). Στην πορεία όμως των χρόνων άρχισαν να εμφανίζονται ολοένα και περισσότερες επιθέσεις που αφορούσαν στην παραβίαση της ασφάλειας του υλικού (hardware security). Επίσης, από πλευράς αξιοπιστίας ανέκαθεν το ζητούμενο ήταν η ορθή λειτουργία του υλικού και της ικανότητάς του να διατηρεί όσο το δυνατόν πιο απαρέγκλιτα τη λειτουργικότητά του ακόμα και δεδομένης της έκθεσής του σε εχθρικά περιβάλλοντα.

Ξεκινώντας την ανάλυση της ασφάλειας και της αξιοπιστίας υλικού υπό την εισαγωγή σφαλμάτων κρίνεται σκόπιμο να γίνει μία σύντομη αναφορά στο τι αναφέρεται ως υλικό, από τι απαρτίζεται, που χρησιμοποιείται, ποιος είναι ο κύκλος ζωής του, πόσο αξιόπιστο μπορεί να θεωρηθεί το υλικό, ποιες είναι οι ευπάθειες που μπορεί να προκύψουν, ποιες οι απειλές που υπάρχουν και τα ενδεχόμενα μέτρα προστασίας απέναντί τους.

Στην κατηγορία του υλικού εντάσσονται όλα τα δομικά στοιχεία τα οποία συνθέτουν ένα ηλεκτρονικό κύκλωμα. Αυτό συμπεριλαμβάνει, αναλογικά ή ηλεκτρονικά στοιχεία (όπως τρανζίστορ, πυκνωτές, ενισχυτές, αντιστάσεις), τυπωμένες πλακέτες (Printed Circuit Boards - PCBs), ολοκληρωμένα κυκλώματα (Integrated Circuits - ICs) και εξαρτήματα τα οποία περιέχουν τσιπ (chip) όπως κυκλώματα προσαρμοσμένης λειτουργίας (ASICs) ή γενικής εμπορικής χρήσης (Commercial Off the Shelf - COTS π.χ. Field-programmable gate array FPGAs), μικροελεγκτές, μικροεπεξεργαστές (Εικόνα 2). Τα ASICs κατασκευάζονται κατόπιν ειδικής απαίτησης για να εξυπηρετήσουν έναν εξειδικευμένο - συγκεκριμένο σκοπό. Τα COTS είναι ολοκληρωμένα ευρείας εμπορικής χρήσης και μπορούν να προγραμματιστούν ανάλογα με τις ανάγκες της λειτουργίας του κάθε συστήματος που εφαρμόζονται.



Εικόνα 2 Ψηφιακό κύκλωμα – [Πηγή: https://www.pexels.com/el-gr/photo/343457/]

Για τη δημιουργία ενός ολοκληρωμένου κυκλώματος (IC) ή αλλιώς τσιπ (chip) ακολουθούνται κάποια στάδια όπως η σχεδίαση, η δημιουργία της εσωτερικής διάταξης του κυκλώματος του ολοκληρωμένου και κατόπιν ορισμένων πολύπλοκων διαδικασιών, όπως η χάραξή του, παράγεται ένας δίσκος πυριτίου (wafer) που απαρτίζεται από πολλά τσιπ. Αυτό αποτελεί μια παρτίδα IC, τα οποία κόβονται και τελικά ενσωματώνονται σε διάφορα κυκλώματα. Πριν την ενσωμάτωσή τους σε κυκλώματα τα IC της παρτίδας ελέγχονται για τυχόν αστοχίες υλικού. Στη συνέχεια τα τσιπ τοποθετούνται σε μια συσκευασία (packaging) με μεταλλικές ακίδες οι οποίες χρησιμεύουν στη σύνδεσή του πάνω στην τυπωμένη πλακέτα (PCB) [7].

Ανάλογα με την αρχιτεκτονική του ολοκληρωμένου κυκλώματος καθορίζονται και οι ακροδέκτες του, οι οποίοι θα προσαρτηθούν σε υποδοχείς εξωτερικών συνδέσεων του τσιπ, που στη συνέχεια θα συγκολληθούν στην τυπωμένη πλακέτα επιτρέποντας τη σύνδεση του τσιπ με τα υπόλοιπα στοιχεία και εξαρτήματα της πλακέτας προκειμένου το υλικό να γίνει λειτουργικό. Η πλακέτα αποτελεί μία πλάκα (ειδικού υλικού όπως FR-4, fiberglass) που έχει πάνω της τυπωμένες αγώγιμες διαδρομές συνδέσεων. Ουσιαστικά αποτελεί το μέσο των ηλεκτρικών συνδέσεων μεταξύ των διαφόρων εξαρτημάτων.

Αν τα παραγόμενα τσιπ δεν ελεγχθούν επαρκώς ως προς την αξιοπιστία τους και την ασφάλειά τους τότε υπάρχει αρκετά μεγάλη πιθανότητα το τελικό προϊόν που θα παραχθεί να είναι ευάλωτο όταν λειτουργεί σε εχθρικά περιβάλλοντα και απροστάτευτο σε επιθέσεις υλικού. Κατά συνέπεια τίθεται σε κίνδυνο όλο το σύστημα στο οποίο θα ενσωματωθούν.

Για τη διαπίστωση της ασφάλειας του υλικού είθισται να πραγματοποιούνται διάφορα δοκιμαστικά τεστ, όπως η εισαγωγή σφαλμάτων (μέσω διαφόρων μεθόδων) για να εξακριβωθεί κατά πόσο το υλικό είναι ευάλωτο σε ανάλογες επιθέσεις. Αντίστοιχες δοκιμές γίνονται για να εξακριβωθεί η συμπεριφορά του υλικού κάτω από απρόβλεπτες συνθήκες λειτουργίας (εχθρικά περιβάλλοντα όπως αναφέρθηκε). Αν κατά τα αρχικά στάδια της σχεδίασης δεν ληφθούν υπόψιν οι απαιτήσεις ασφάλειας και αξιοπιστίας του υλικού τότε για την τροποποίησή του σε μεταγενέστερο στάδιο της διαδικασίας παραγωγής απαιτείται επιπλέον χρόνος μελέτης και διόρθωσης, πόροι, ανθρώπινο δυναμικό και υλικά. Αυτό ανεβάζει κατά πολύ το προκαθορισμένο κόστος παραγωγής. Ειδικά στις περιπτώσεις στις οποίες το υλικό δεν μπορεί να δεχτεί τροποποίηση και καλείται ο κατασκευαστής να διαλέξει ανάμεσα στη διατήρηση του ευάλωτου υλικού ή στον επανασχεδιασμό παραμερίζοντας το κόστος.

Ανέκαθεν η διασφάλιση της αξιοπιστίας, της ακεραιότητας, της εμπιστευτικότητας και της διαθεσιμότητας αποτελούσαν τα κύρια συστατικά της ασφάλειας ενός συστήματος. Ως ακεραιότητα νοείται η διασφάλιση ότι τα δεδομένα που επεξεργάζονται για παράδειγμα σε έναν επεξεργαστή δεν κινδυνεύουν να υποστούν οποιαδήποτε αλλοίωση από κάποιον εξωγενή και ίσως αστάθμητο παράγοντα. Αυτή η λειτουργία μπορεί να υποστηριχθεί με τη χρήση συναρτήσεων κατακερματισμού (hash functions) οι οποίες θα προσθέσουν ένα επίπεδο ασφάλειας στα δεδομένα καθώς σε οποιαδήποτε απόπειρα παραποίησής τους η τιμή της συνάρτησης κατακερματισμού (hash value) μεταβάλλεται. Ως προς την εμπιστευτικότητα αυτή μπορεί να υποστηριχθεί με τη χρήση κρυπτογραφικών αλγορίθμων ώστε τα δεδομένα να είναι μη αναγνωρίσιμα από οποιονδήποτε μη εξουσιοδοτημένο χρήστη. Με την κρυπτογράφηση οποιοσδήποτε τρίτος δεν μπορεί να έχει πρόσβαση στα δεδομένα που δεν πρέπει να τα αναγνώσει. Μόνο ο εξουσιοδοτημένος χρήστης μπορεί να τα αποκρυπτογραφήσει χρησιμοποιώντας το αντίστοιχο κρυπτογραφικό κλειδί. Ως προς τη διαθεσιμότητα, θα πρέπει να εξασφαλίζεται ότι τα δεδομένα είναι διαθέσιμα όταν ζητούνται κατά τη διάρκεια της λειτουργίας του συστήματος.

Από την σκοπιά του υλικού ως αξιοπιστία (reliability) εννοείται η ικανότητα ανταπόκρισης του υλικού στο να μπορεί να εκτελεί άρτια τις λειτουργίες για τις οποίες προορίζεται. Η λειτουργικότητα του υλικού είναι προκαθορισμένη να εκτελείται κατά τη διάρκεια μιας ορισμένης χρονικής περιόδου και υπό συγκεκριμένες εσωτερικές (δικές του) και εξωτερικές περιβαλλοντικές συνθήκες. Η έκθεση του υλικού σε αντίξοες εξωγενείς εχθρικές συνθήκες μπορεί να το οδηγήσει σε τροποποίηση της λειτουργικότητάς του ήτοι σε μία μη αποδεκτή κατάσταση λειτουργίας ή και ακόμη και στη διακοπή της. Ιδανικά το υλικό για να μπορεί να χαρακτηριστεί ως αξιόπιστο θα πρέπει να είναι καλά δοκιμασμένο σε πιθανές περιβαλλοντικές αντιξοότητες ώστε να μην επηρεάζεται ή να ξεπερνάει τυχόν σφάλματα μέσω διαδικασιών ελέγχου και αυτορρύθμισης. Ανάμεσα στα τεχνικά χαρακτηριστικά του υλικού σημαντική

πληροφορία αποτελεί η καταγραφή τυχόν γνωστών αστοχιών, η συχνότητα εμφάνισής τους καθώς και ο κύκλος ζωής του υλικού (εργοστασιακή πρόβλεψη). Η παρούσα διατριβή όσον αφορά την αξιοπιστία εστιάζει στην ανοχή σε σφάλματα τα οποία μπορούν να προκληθούν για παράδειγμα από την ακτινοβολία σωματιδίων (π.χ. βαρέα ιόντα ή πρωτόνια) υψηλής ενέργειας.

Ειδικότερα με την ασφάλεια και την αξιοπιστία του υλικού σχετίζονται τα διάφορα στάδια της παραγωγής του. Από την σχεδίαση, τη δοκιμή, την υλοποίηση μέχρι την τελική κυκλοφορία του. Το κατά πόσο εφαρμόζονται έλεγχοι κατά τη διάρκεια της κατασκευής του καθορίζει σε μεγάλο βαθμό και το ποσοστό ασφάλειας και αξιοπιστίας που θα επιτευχθεί. Όπως σε επίπεδο λογισμικού έτσι και σε επίπεδο υλικού θα πρέπει να απαγορεύεται η πρόσβαση στα κρίσιμα δεδομένα του συστήματος. Η συνθήκη αυτή χαρακτηρίζεται ως εμπιστευτικότητα του υλικού.

Η διασφάλιση της ακεραιότητας του υλικού επίσης αποτελεί πυλώνα της ασφάλειας ενός συστήματος. Είναι κρίσιμο να είναι σχεδιασμένο με τέτοιο τρόπο ώστε να αναγνωρίζει τα στοιχεία του και σε τυχόν αλλαγή τους (για παράδειγμα προσπάθεια εισαγωγής πλαστού εξαρτήματος) να ειδοποιεί και να διακόπτει τη λειτουργία του. Η διαθεσιμότητα του υλικού αποτελεί επίσης ενδεικτικό στοιχείο ασφάλειας του συστήματος. Αυτό συνεπάγεται την απρόσκοπτη πρόσβαση στα απαραίτητα δεδομένα του συστήματος καθώς και στον εσωτερικό κώδικα του υλικού από τον κάτοχο του συστήματος ή τον εξουσιοδοτημένο χρήστη. Τα δεδομένα στα οποία επιτρέπεται η πρόσβαση καθορίζονται σύμφωνα με τους τιθέμενους περιορισμούς από την πολιτική ασφαλείας του συστήματος.

Τα παραπάνω κρίνεται απαραίτητο να πληρούνται για τη διασφάλιση της εύρυθμης λειτουργίας και της παρεμπόδισης διαρροής πληροφοριών του συστήματος . Επίσης σημαντικό παράγοντα αποτελεί και η μελέτη της συνεργασίας του υλικού με το λογισμικό, καθώς το υλικό παρέχει την δομική αρχιτεκτονική του συστήματος διευκολύνοντας την εκτέλεση των λειτουργιών του και το λογισμικό συντονίζει το υλικό για την ορθή τήρηση των προκαθορισμένων κανόνων λειτουργίας του συστήματος.

Από πλευράς αξιοπιστίας του υλικού, κατά την παραγωγή του, είναι δόκιμο να χρησιμοποιούνται υλικά που οι προδιαγραφές τους εξυπηρετούν την εξασφάλιση της ανθεκτικότητας του παραγόμενου προϊόντος. Καθώς οι παράγοντες έκθεσης σε εχθρικές συνθήκες ποικίλουν, επιδιώκεται ο μετριασμός τους ώστε να μην τίθεται υπό αμφισβήτηση η αξιοπιστία του υλικού.

Για να μετριαστούν όμως οι απειλές θα πρέπει να είναι γνωστό ποιες είναι οι κυριότερες ευπάθειες του υλικού. Όπως αναφέρθηκε υπάρχουν πολλά στάδια της παραγωγικής αλυσίδας και σε όλα μπορεί να υπάρχουν τρωτά σημεία. Η αρχιτεκτονική του υλικού παίζει καθοριστικό ρόλο. Κατά τη σχεδίαση η μη εφαρμογή προκαθορισμένων και τυποποιημένων μεθόδων και εστιασμένων ελέγχων έχει ως συνέπεια τη μετέπειτα δημιουργία λειτουργικών σφαλμάτων. Τα σφάλματα αυτά διαφαίνονται μέσα από την παρατήρηση της λειτουργίας του υλικού, όπου όποιες ανωμαλίες προκύψουν, αποτελούν και τα ευαίσθητα σημεία. Η εσκεμμένη καθοδήγηση του συστήματος σε συνθήκες μη φυσιολογικής λειτουργίας μπορεί να οδηγήσει το σύστημα στη διακοπή της λειτουργίας του ή την αποκάλυψη κάποιας εσωτερικής πληροφορίας. Πολλές φορές μπορεί όμως αυτό να συμβεί και ακούσια από το σύστημα.

Άλλη πιθανή ευπάθεια μπορεί να είναι ο ελλιπής καθορισμός ρόλων και έλεγχος των φυσικών προσώπων που έχουν πρόσβαση κατά τη σχεδίαση και υλοποίηση του υλικού. Μία λανθασμένη ενέργεια μη εξουσιοδοτημένου υπαλλήλου μπορεί για παράδειγμα να οδηγήσει σε παραγωγή ελλαττωματικών τσιπ χωρίς αυτό να γίνει αντιληπτό ή και τσιπ που να εμπεριέχουν σκόπιμο σφάλμα λειτουργίας.

Ανάλογα με το είδος της ευπάθειας που μπορεί να υπάρχει σε κάποιο στοιχείο του συστήματος μπορεί μετά από αξιολόγησή του να γίνει αποδεκτό (υπολογίζοντας το ρίσκο) ή να επιλεγεί η λήψη αντιμέτρων ασφαλείας ή αντικατάστασής του για λόγους αξιοπιστίας.

Άλλο ένα κρίσιμο σημείο της αλυσίδας παραγωγής του υλικού που πρέπει να προσεχθεί είναι τα μέσα για τη δοκιμή της ορθής λειτουργίας του, των δοκιμών ανεύρεσης αστοχιών, οι μετρήσεις των επιδόσεών του ή τα όρια της εύρυθμης λειτουργίας του. Αυτά τα δοκιμαστικά τεστ μπορεί να διαρρεύσουν και να χρησιμοποιηθούν για τον αντίθετο σκοπό από αυτό της δημιουργίας τους. Παρακάτω παρουσιάζονται επιθέσεις υλικού είτε ως αυτόνομες επιθέσεις είτε σε συνδυασμό με την εισαγωγή σφαλμάτων.

4.1 Επιθέσεις βάσει Μικροαρχιτεκτονικής (Microarchitectural Attacks)

Η βιομηχανία παραγωγής των διαφόρων ηλεκτρονικών στοιχείων - εξαρτημάτων «κυνηγάει» τη γοργά αναπτυσσόμενη αγορά. Λόγω αυτής της φιλοσοφίας των επιχειρήσεων τεχνολογίας και για την γρήγορη κυκλοφορία των προϊόντων θυσιάζεται αρκετός από τον χρόνο δοκιμών ασφάλειας του νέου υλικού που παράγεται. Έτσι δημιουργούνται κενά στην ασφάλεια του υλικού και υπάρχει πρόσφορο έδαφος για την ανάπτυξη τεχνικών εκμετάλλευσής τους. Ο διαχωρισμός των σύγχρονων επιθέσεων βάσει μικροαρχιτεκτονικής έγκειται στο γεγονός ότι οι επιθέσεις αυτές εκμεταλλεύονται τη μικροαρχιτεκτονική των επεξεργαστών λόγω του ότι τα εξαρτήματα αυτά δεν έχουν υποβληθεί σε διεξοδικούς ελέγχους και δοκιμασίες ασφαλείας.

4.2 Differential Fault Analysis - DFA

Η εισαγωγή σφαλμάτων αποτελεί μία γνωστή πρακτική που χρησιμοποιείται με σκοπό την κρυπτανάλυση με εφαρμογή πάνω σε διαφορετικές κρυπτογραφικές μεθόδους. Η Διαφορική Ανάλυση Σφαλμάτων (Differential Fault Analysis - DFA) είναι η μια συχνή χρησιμοποιούμενη τεχνική βασιζόμενη στην εισαγωγή σφαλμάτων για την συστημική παρατήρηση της εξόδου ενός κρυπτογραφημένου συστήματος [8]. Η φιλοσοφία της DFA είναι η σύγκριση των τιμών της εξόδου ενός κρυπτογραφικού αλγορίθμου, υπό την εισαγωγή σφαλμάτων σε αυτό. Συγκρίνεται δηλαδή μία εσφαλμένη τιμή εξόδου (υπό εισαγωγή σφάλματος) σε σχέση με την αντίστοιχη αναμενόμενη σωστή τιμή εξόδου του κρυπτογραφικού αλγορίθμου. Με τη χρήση της DFA, είναι δυνατή η αποκάλυψη του κρυπτογραφικού κλειδιού.

Σε μία επίθεση που βασίζεται σε DFA παρακολουθείται ένα κρυπτογραφημένο σύστημα συλλέγοντας στοιχεία που αφορούν την είσοδο και έξοδό του. Η παρατήρηση και η συλλογή γίνεται για τουλάχιστον δύο δείγματα (με δεδομένο κείμενο εισόδου plaintext) και συλλέγεται η είσοδος, η έξοδος και μέσω υποθέσεων για τον κρυπτογραφικό αλγόριθμο είναι δυνατή η εύρεση του κρυπτογραφικού κλειδιού [9].

Αν για παράδειγμα γίνει μία δοκιμή εξαντλητικής εισαγωγής μονών σφαλμάτων (Single Bit Flip – SBF) στην είσοδο του τελευταίου γύρου του AES, λόγω της δομής του AES, παρατηρείται ότι το σύστημα επηρεάζεται καθώς το σφάλμα διαδίδεται στα κρυπτογραφικά δεδομένα ως ένα μονό εσφαλμένο byte. Μέσω της εύρεσης της διαφοράς εισαγωγής σφάλματος (σε σχέση με την έξοδο άνευ σφάλματος) επιδιώκεται να αποκαλυφθεί η τιμή του αντίστοιχου byte του μυστικού κλειδιού του τελευταίου γύρου του AES που εισήχθη το σφάλμα. Δοκιμάζονται όλοι οι πιθανοί συνδυασμοί των τιμών που θα μπορούσε να έχει αυτό το byte του μυστικού κλειδιού. Όταν η υπολογιζόμενη πιθανή διαφορά (μέσω δοκιμών) γίνει ίση με τη διαφορά σφάλματος του SBF τότε η δοκιμή της τιμής του κλειδιού είναι επιτυχής και άρα έχει βρεθεί η τιμή του byte που ο επιτιθέμενος έψαχνε. Με τον ίδιο τρόπο υπολογίζονται και τα υπόλοιπα byte του κρυπτογραφικού κλειδιού του τελευταίου γύρου του AES.

4.3 Side Channel Analysis (SCA)

Οι επιθέσεις πλευρικού καναλιού αποτελούν την προσπάθεια εκμετάλλευσης μιας ή παραπάνω διαρροών πληροφορίας ενός ασφαλούς κυκλώματος προκειμένου να εξαχθούν «κρυμμένες» πληροφορίες. Αυτό μπορεί να επιτευχθεί μέσω της προσπάθειας μέτρησης και ανάλυσης ορισμένων παραμέτρων (πλευρικών καναλιών) του κρυπτογραφικού αλγορίθμου όπως ο χρόνος εκτέλεσης (διάρκεια), η κατανάλωση της ενέργειας (το ρεύμα τροφοδοσίας), η θερμότητα που παράγεται ή η ποσότητα της ηλεκτρομαγνητικής ακτινοβολίας που εκπέμπεται κατά τη διάρκεια εκτέλεσης κάποιας λειτουργίας. Μία επίθεση πλευρικού καναλιού μπορεί να εφαρμοστεί είτε μετρώντας τη διαρροή μίας παραμέτρου του συστήματος κατά την εκτέλεση της κρυπτογραφικής λειτουργίας, είτε συλλέγοντας διάφορες παραμέτρους της διαρροής που παρατηρείται και στη συνέχεια μέσω εφαρμογής στατιστικών μεθόδων επιδιώκεται η ανεύρεση ευαίσθητων δεδομένων του συστήματος. Ακολούθως αυτές οι πληροφορίες μπορούν να χρησιμοποιηθούν για την αποκάλυψη μυστικών παραμέτρων ασφαλείας ενός κυκλώματος, όπως είναι για παράδειγμα ένα κρυπτογραφικό κλειδί.

Η εισαγωγή σφαλμάτων σε ένα σύστημα μπορεί να χρησιμοποιηθεί προκειμένου να ενισχυθούν οι επιθέσεις πλευρικού καναλιού (Side Channel Attacks) ή να αντιμετωπιστούν τυχόν αντίμετρα που μπορεί να έχουν εφαρμοστεί.

Επομένως ο τομέας της ασφάλειας αποτελεί μία πρόκληση στην αλυσίδα της βιομηχανίας κατασκευής υλικού. Η κατεύθυνση προσανατολισμού για τη λύση των περισσοτέρων προαναφερθέντων ζητημάτων ασφαλείας είναι η απ' αρχής σχεδίαση του υλικού με επίκεντρο την ασφάλεια. Η υιοθέτηση ανάλογων πολιτικών και ενδεχομένως και προτύπων (όπως ISO) επιβολής ελάχιστων επιπέδων ασφαλείας μπορεί να διαδραματίσει καθοριστικό ρόλο στην δημιουργία διαχρονικότερου και ποιοτικότερου υλικού. Τα υλικά καθώς και η αναβάθμιση των δοκιμαστικών τεστ, για την ανεύρεση αδυναμιών του υλικού, συμβάλουν τα μέγιστα στην παραγωγή αξιόπιστου και πιστοποιημένου ασφαλούς υλικού. Οι ποιοτικοί έλεγχοι και οι έλεγχοι γύρω από την ασφάλεια, θα πρέπει να πραγματοποιούνται από το αρχικό κιόλας στάδιό της σχεδίασης του υλικού μέχρι και το τελικό στάδιο της δοκιμής του πριν την τελική διάθεσή του για χρήση. Επίσης το κόστος παραγωγής λόγω επανασχεδιασμών ή απόσυρσης μη ασφαλών εξαρτημάτων θα μειωθεί ενώ ταυτόχρονα θα αυξηθεί και η ποιότητα των συστημάτων στα οποία θα ενσωματώνεται το νέο υλικό. Στόχος είναι η δημιουργία ποιοτικού υλικού αντοχής στις επιθέσεις και ίσως μελλοντικά υλικό που να μπορεί με μικρές μεταβολές-τροποποιήσεις-προσθήκες να προσαρμοστεί εύκολα σε νέες απαιτήσεις.

5. Μοντέλα σφαλμάτων (Fault Models)

5.1 Επιθέσεις Εισαγωγής σφαλμάτων

Οι επιθέσεις εισαγωγής σφαλμάτων εστιάζουν στη μελέτη της συμπεριφοράς για παράδειγμα ενός μικροεπεξεργαστή ή ενός κυκλώματος υπό την επίδραση εισαγωγής σφαλμάτων. Μελετάται λοιπόν σε τι βαθμό θα επηρεάσει αυτό το σφάλμα το σύστημα, αν θα αποσβεστεί ή θα διαδοθεί κατά τη διάρκεια της λειτουργίας του κυκλώματος, αν δηλαδή θα γίνει αντιληπτό και θα οδηγήσει το σύστημα σε μία λειτουργία για την οποία δεν προορίζεται ή δεν έχει προβλεφθεί ή αν θα αποσβεστεί. Η καταγραφή της συμπεριφοράς του κυκλώματος οδηγεί στο να εξαχθούν συμπεράσματα για τον τρόπο που επηρεάστηκε και έτσι να αναδειχθούν οι αδυναμίες του. Η δυνατότητα χειρισμού ενός σφάλματος από ένα κύκλωμα είναι μία πολύ σημαντική διαδικασία καθώς δείχνει κατά πόσο το σύστημα μπορεί να ανακάμψει από αυτήν την «εσφαλμένη» λειτουργία που οδηγήθηκε, τι επίδραση θα έχει το σφάλμα και πόσο χρόνο θα χρειαστεί για να ανακάμψει ή να οδηγηθεί στη διακοπή της λειτουργίας του ή αλλιώς της διαθεσιμότητάς του.

Η διαδικασία της μελέτης της ασφάλειας και της αξιοπιστίας ενός κυκλώματος υπό την επίδραση σφαλμάτων ξεκινάει με εκτεταμένες εισαγωγές σφαλμάτων με στόχο τον χαρακτηρισμό του υπό μελέτη κυκλώματος. Αυτές οι εισαγωγές σφαλμάτων μπορούν να υλοποιηθούν με διαφορετικές μεθόδους με τον ίδιο όμως στόχο που είναι η ανεύρεση ευπαθειών του κυκλώματος. Η ανάδειξη αυτών των ευπαθειών αποτελεί την αφετηρία της αξιολόγησης. Έπειτα ανάλογα με τα αποτελέσματα αυτής της μελέτης γίνεται συνεκτίμηση αυτών σε συνάρτηση με τις απαιτούμενες προδιαγραφές για την ασφάλεια και την αξιοπιστία. Στην περίπτωση που δεν ικανοποιούνται οι απαραίτητες προδιαγραφές που έχουν τεθεί καθορίζεται όπου χρειάζεται η ενσωμάτωση αντιμέτρων και τεχνικών προστασίας για την βελτίωση της ανθεκτικότητας του κυκλώματος. Αν παρά την ανάδειξη ευπαθειών του κυκλώματος επιλεγεί σχεδιαστικά να μην ενσωματωθούν αντίμετρα αυτό γίνεται με αποδοχή του όποιου κινδύνου έχει προκύψει. Δεδομένου ότι όσο πιο νωρίς γίνει η μελέτη τόσο περισσότεροι θα είναι οι σχεδιαστικοί πόροι που θα εξοικονομηθούν, οι δοκιμές ανθεκτικότητας στην εισαγωγή σφαλμάτων πρέπει να εφαρμόζονται πρώιμα. Για αυτό το λόγο στην παρούσα διπλωματική εστιάζουμε στην εισαγωγή σφαλμάτων μέσω προσομοίωσης στο RTL καθώς θα μας επιτρέψει να μελετήσουμε την ανθεκτικότητα των υπό μελέτη κυκλωμάτων στην αρχή της σχεδιαστικής ροής.

Για την αξιολόγηση ενός κυκλώματος υπό την επίδραση σφαλμάτων χρησιμοποιούνται και τα αντίστοιχα εργαλεία καθώς και ένα σαφώς καθορισμένο μοντέλο εισαγωγής σφαλμάτων

το οποίο θα προσομοιώνει όσο πιο αντιπροσωπευτικά γίνεται το σφάλμα που αναμένεται να εφαρμοστεί. Αυτό σημαίνει ότι πρέπει να είναι σαφής ο τύπος σφάλματος που θα εισαχθεί, πότε θα εισαχθεί, για πόσο χρονικό διάστημα, πόσες φορές και ποια χαρακτηριστικά του κυκλώματος που αναμένεται να επηρεαστούν. Εισαγωγή σφαλμάτων σε επίπεδο υλικού μπορεί να γίνει για παράδειγμα μέσω ηλεκτρομαγνητικών παρεμβολών καθώς επίσης και μέσω της απότομης αυξομείωσης της τροφοδοσίας προκειμένου να επιτευχθεί η διαταραχή της εύρυθμης λειτουργίας του κυκλώματος, τεχνική που μπορεί να οδηγήσει σε μη αποδεκτή λειτουργία της συσκευής. Μία ακόμη τεχνική είναι η έκθεση για παράδειγμα ενός ολοκληρωμένου κυκλώματος σε σωματίδια ακτινοβολίας υψηλής ενέργειας (κοσμική) το οποίο μπορεί να προκαλέσει αντιστροφή ενός ή πολλών bit (bit-flip) του ολοκληρωμένου. Όσον αφορά στη χρονική στιγμή της αξιολόγησης του κυκλώματος κατά τη διάρκεια της σχεδιαστικής ροής μπορούν να εισαχθούν σφάλματα πρίν την κατασκευή (στα σχέδια του κυκλώματος). Η εισαγωγή σφαλμάτων μπορεί να πραγματοποιηθεί με τη χρήση προσομοίωσης σε πληθώρα αφαιρετικών επιπέδων (πχ. RTL, Gate Level, Post Place and Route) και τη χρήση κατάλληλων σφαλματικών μοντέλων τα οποία εξομοιώνουν πειραματικές μεθόδους εισαγωγής σφαλμάτων). Σε κάθε περίπτωση η τελική αξιολόγηση θα λάβει χώρα μετά την κατασκευή του κυκλώματος μέσω εισαγωγής σφαλμάτων σε πειραματικό επίπεδο και ενώ το σύστημα βρίσκεται στην κανονική λειτουργία του.

Σύγχρονες μέθοδοι εξαγωγής πληροφοριών από επιθέσεις στο υλικό [9] είναι:

- Δυσλειτουργία τάσης (Voltage Glitch)
- Δυσλειτουργία ρολογιού (Clock Glitch)
- Ηλεκτρομαγνητικές επιθέσεις (Electromagnetic Attacks)
- Επιθέσεις με λέιζερ (Laser Attacks)

Οι επιθέσεις εισαγωγής σφαλμάτων που πραγματοποιούνται κατά τη διαδικασία παραγωγής του υλικού στοχεύουν στο να ανακαλύψουν δυσλειτουργίες στα υπό αξιολόγηση κυκλώματα. Για κάθε σχεδιαστικό κύκλωμα πραγματοποιούνται διαφορετικές υλοποιήσεις οι οποίες έχουν μικρές ή αρκετά σημαντικές διαφορές μεταξύ τους. Κάποιες σχεδιάσεις μπορεί να διαφέρουν ως προς την εφαρμογή ή μη βελτιστοποιήσεων ενώ κάποιες άλλες ως προς την υλοποίησή τους με ενσωμάτωση αντίμετρων. Η απόφαση της τελικής επιλογής μίας σχεδίασης ως προς την καταλληλόλητα και την πληρότητά της γίνεται μετά από μελέτη εισαγωγής σφαλμάτων σε όλες τις σχεδιάσεις. Η αξιολόγηση των αποτελεσμάτων θα αναδείξει τη σχεδίαση με τα λιγότερα κρίσιμα σφάλματα.

Τα υπό αξιολόγηση κυκλώματα της παρούσας εργασίας αποτελούν αντικείμενο μελέτης της εισαγωγής σφαλμάτων σε κρυπτογραφική συνάρτηση (Sbox) υλοποιημένη σε FPGA. Οι σχεδιάσεις υλοποιήθηκαν και προσομοιώθηκαν με τη χρήση του εργαλείου Vivado HLS και διαφέρουν μεταξύ τους ως προς τρεις επιλογές βελτιστοποιήσεων που είναι η εφαρμογή της προκαθορισμένων παραμέτρων σύνθεσης του εργαλείου, η εφαρμογή loop unrolling και η επιλογή no-inline.

Τα μοντέλα σφαλμάτων συνήθως εφαρμόζονται σε επίπεδο αλγορίθμου, πηγαίου κώδικα ή επίπεδο εντολών. Συνήθως οι τεχνικές κρυπτανάλυσης βασίζονται σε μοντέλα σφαλμάτων στα οποία τα σφάλματα εισάγονται κατά τη διάρκεια ροής των δεδομένων ενός προγράμματος μίας συσκευής - στόχου και σκοπό έχουν να επηρεάσουν ένα μεμονωμένο bit ή και περισσότερα (ένα byte ή μια λέξη (word)) μιας μεταβλητής που αποτελεί καίριο κομμάτι της ασφάλειας του συστήματος. Οι τρόποι που επιχειρείται να επηρεαστεί το bit – στόχος είναι με αναστροφή (flip), με αρχικοποίηση (set), με επαναφορά (reset) ή και δίνοντάς του μια τυχαία τιμή [10].

5.1.1 Εξαντλητική εισαγωγή μονών σφαλμάτων (Exhaustive Single Bit Flip)

Πριν από την εκδήλωση μίας επίθεσης υλικού ο επιτιθέμενος οργανώνει την επίθεσή του μελετώντας και καταρτίζοντας μια μεθοδολογία η οποία περιλαμβάνει πληροφορίες σχετικά με τη συσκευή – στόχο που θα προσπαθήσει να εκμεταλλευτεί, ποιο μοντέλο σφαλμάτων (Fault Model) θα ακολουθήσει, ποια τεχνική εισαγωγής σφαλμάτων (Fault Injection) και ποια μέθοδο εκμετάλλευσης των σφαλμάτων (Fault Exploitation Method) προκειμένου να "αποκωδικοποιήσει" τα αποτελέσματα της επίθεσής του [10]. Όσο καλύτερη προετοιμασία γίνει γύρω από τα τεχνικά χαρακτηριστικά και τις ιδιότητες του κυκλώματος - στόχου της επίθεσης δηλαδή μελέτη για τα σφάλματα που πρέπει να εισαχθούν, πότε και πού τόσο πιο επιτυχημένη θα είναι η έκβασή της. Αυτό πρέπει να λαμβάνεται υπόψιν κατά την αξιολόγηση των κυκλωμάτων. Για το λόγο αυτό γίνεται προσπάθεια εξαντλητικής εισαγωγής σφαλμάτων ώστε να δοκιμαστούν όλες οι πιθανές περιπτώσεις εισαγωγής σφαλμάτων και να εντοπιστούν τυχόν αδυναμίες του υλικού που θα μπορούσαν εν δυνάμει να οδηγήσουν το κύκλωμα σε μη αποδεκτή λειτουργία.

Για παράδειγμα, κατά την αξιολόγηση ενός κυκλώματος μπορεί να συνυπολογιστεί ότι κάποιος, ο οποίος διαθέτει εξοπλισμό με ισχυρές δυνατότητες, μπορεί να εισαγάγει σφάλματα σε οποιαδήποτε χρονική στιγμή (clock cycle) και να επηρεάσει με ακρίβεια ένα οποιοδήποτε flip flop τη φορά. Με γνώμονα τη δημιουργία ασφαλούς υλικού και επειδή θα ήταν ελλιπές να αφεθούν στοιχεία (flip flops) τα οποία δεν θα δοκιμαστεί η αντοχή τους στην εισαγωγή σφαλμάτων επιλέγεται η τεχνική της εξαντλητικής εισαγωγής μονών σφαλμάτων (Exhaustive Single Bit Flip), δηλαδή σφάλματα σε μεμονωμένα bit με στόχο την αναστροφή τους (flip) σε ένα ηλεκτρονικό κύκλωμα. Αυτό σημαίνει ότι θα εισαχθούν σφάλματα στο κύκλωμα στοχεύοντας κάθε φορά ένα μεμονωμένο bit με σκοπό την αντιστροφή της τιμής του, είτε από 0 σε 1 είτε από 1 σε 0. Αυτή η διαδικασία της προσπάθειας αντιστροφής της τιμής ενός bit θα επαναληφθεί με ακριβώς το ίδιο μοτίβο όσες φορές χρειάζεται προκειμένου να ληφθούν και να αξιολογηθούν τα αποτελέσματα όλων των πιθανών περιπτώσεων. Ο χαρακτηρισμός των σφαλμάτων γίνεται σύμφωνα με το ποια σφάλματα μπορούν να διαδοθούν και να επηρεάσουν την τιμή της εξόδου (και επομένως είναι κρίσιμα) και ποια φάνηκε να μην επηρεάζουν τη λειτουργικότητα του κυκλώματος.

Σε επίπεδο μικροαρχιτεκτονικής κυκλώματος το αποτέλεσμα της εισαγωγής ενός σφάλματος δημιουργεί εσφαλμένη λειτουργία του κυκλώματος και εν συνεχεία αλλαγή στις binary τιμές που αποθηκεύονται για παράδειγμα σε κυκλώματα μνήμης όπως flip-flop. Τα flip-flop με τη σειρά τους θα οδηγήσουν συνδυαστικά στοιχεία (πύλες) και ανάλογα τη δομή του κυκλώματος θα παρατηρηθεί στην έξοδο του flip-flop αν το σφάλμα θα διαδοθεί ή θα αποσβεστεί. Μέσα από την παρατήρηση αυτής της συμπεριφοράς διαπιστώνονται οι επιπτώσεις των υπό εισαγωγή σφαλμάτων τις οποίες ο επιτιθέμενος ψάχνει για την ανεύρεση μοτίβων σφαλμάτων ή συγκεκριμένες "ευαισθησίες" του υλικού. Ο ρόλος της αξιολόγησης είναι να εξουδετερώσει όσες περισσότερες αδυναμίες υλικού εντοπίσει είτε μέσω αντιμέτρων είτε μέσω επιλογής της λιγότερο επισφαλούς υλοποίησης.

Από τη σκοπιά του επιτιθέμενου, αυτός μετά την εισαγωγή σφάλματος αξιολογεί και εκμεταλλεύεται τα αποτελέσματα προκειμένου να εξάγει πληροφορίες για τη λειτουργία του συστήματος, να αποκτήσει πρόσβαση σε ευαίσθητα δεδομένα που εμπεριέχονται στο σύστημα όπως ο κρυπτογραφικός αλγόριθμος που χρησιμοποιείται (ανεύρεση του κρυπτογραφικού κλειδιού), τη δυνατότητα παράκαμψης του συστήματος ασφαλείας ή τυχόν αντιμέτρων που έχουν χρησιμοποιηθεί, την απόκτηση αυξημένων δικαιωμάτων ώστε να μπορεί να παρέμβει στις ρυθμίσεις ασφαλείας του συστήματος χωρίς να γίνει αντιληπτός κ.α. Επομένως η στρατηγική εκμετάλλευσης που θα επιλέξει ο επιτιθέμενος περιλαμβάνει τη θέση που θα εισαχθεί το σφάλμα, τη διάρκεια του σφάλματος, το πλήθος των σφαλμάτων και την αναμενόμενη επίδραση που θα έχουν.

5.1.2 Μοντέλο τυχαίων πολλαπλών σφαλμάτων (Random Multiple Bit-Flip)

Όπως αναφέρθηκε παραπάνω η εξαντλητική τεχνική εισαγωγής σφαλμάτων (Exhaustive Single Bit) στοχεύει στην εξάντληση των υποψήφιων bit στα οποία μπορεί να εισαχθεί ένα σφάλμα. Αυτό όμως αντιπροσωπεύει την εισαγωγή ενός μονού σφάλματος. Τα υπό αξιολόγηση κυκλώματα θα πρέπει να μελετηθούν και ως προς την εισαγωγή πολλαπλών σφαλμάτων. Εύκολα γίνεται αντιληπτό ότι κάτι τέτοιο μπορεί να αποβεί εξαιρετικά χρονοβόρο καθώς είναι πολύ μεγάλος ο χρόνος που απαιτείται για την εισαγωγή πολλαπλών σφαλμάτων με εξαντλητικό τρόπο. Ο αριθμός των σφαλμάτων είναι τόσο μεγάλος που πρακτικά είναι σχεδόν αδύνατη η προσπάθεια εξαντλητικής εισαγωγής. Επομένως θα πρέπει να αναζητηθούν τρόποι για το αν υπάρχει η δυνατότητα εξαγωγής ασφαλών συμπερασμάτων εφαρμόζοντας την αξιολόγηση με πολλαπλά bit (Multiple Bit-Flip) [11] με χρήση μικρότερου στατιστικού δείγματος. Ο τρόπος επιλογής των δειγμάτων (πόσα bit) στα οποία τελικά θα εισαχθούν σφάλματα είναι ένα σημαντικό πρόβλημα, καθώς θα πρέπει να καθοριστεί το κριτήριο της επιλογής (τυχαία), ο αριθμός των δειγμάτων (sample) και το εύρος της επιλογής (range). Επομένως εφόσον δεν μπορεί να πραγματοποιηθεί εξαντλητική εισαγωγή πολλαπλών σφαλμάτων, παρά μόνο μερικώς ή μέσω χρήσης στατιστικής μεθόδου θα πρέπει να οριστεί το ελάχιστο αποδεκτό σφάλμα (margin of errors) το οποίο μπορεί να γίνει αποδεκτό.

5.1.3 Μοντέλο λογικών κώνων (Cone partitioning fault model (multi-bit))

Σύμφωνα και με τα προηγούμενα η επιλογή του μοντέλου σφαλμάτων και η ακρίβεια με την οποία περιγράφει την αναμενόμενη εισαγωγή σφαλμάτων εξαιτίας μιας πειραματικής διάταξης εισαγωγής σφαλμάτων ή ενός φαινομένου αποτελεί σημαντικό παράγοντα για την της ορθότητα των αποτελεσμάτων της αξιολόγησης. Εκτός των κλασικών μοντέλων εισαγωγής σφαλμάτων Single bit flip και Multiple bit flip, όταν πρόκειται για επιθέσεις laser φαίνεται να λειτουργεί πιο αποδοτικά το μοντέλο σφαλμάτων λογικών κώνων [12].

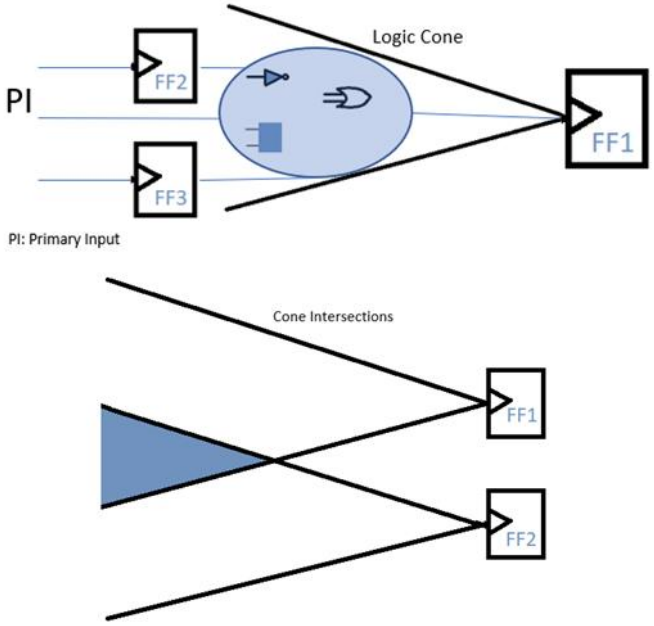
Για την περίπτωση μελέτης των επιπτώσεων μια τοπικής εισαγωγής σφαλμάτων, είτε μέσω μίας επίθεσης με χρήση laser, είτε εξαιτίας σωματιδίων υψηλής ενέργειας, στο RTL (Register Transfer Level) επίπεδο και της μελέτης της επίπτωσής της για παράδειγμα σε ένα ολοκληρωμένο κύκλωμα (από την πλευρά της ασφάλειας), θα πρέπει να συνυπολογιστούν και άλλες παράμετροι που ανεβάζουν τον πήχη πολυπλοκότητας.

Σε επίπεδο RTL δεν μπορούν να γίνουν παρά μόνο εικασίες για το που θα βρίσκεται τοποθετημένο ποιο εξάρτημα καθώς οι βελτιστοποιήσεις και η χωροτοποθέτηση (placement) λαμβάνουν χώρα πολύ αργότερα (σύνθεση). Μόνο οι καταχωρητές (registers) για τους οποίους είναι γνωστό εκ των προτέρων ότι δεν θα επηρεαστούν από τη ροή της σύνθεσης, κατά την εφαρμογή βελτιστοποιήσεων, μπορούν να προσφέρουν σχετικές πληροφορίες γύρω από την ανάλυση της επίδρασης ενός μοντέλου σφαλμάτων.

Το μοντέλο των λογικών κώνων εστιάζει στην εισαγωγή σφαλμάτων στα flip-flops ενός κυκλώματος. Ο κάθε κώνος απαρτίζεται από το ένα flip-flop το οποίο αποτελεί την αφετηρία (κορυφή) του κώνου και εκτείνεται προς τα πίσω. Ουσιαστικά ένας λογικός κώνος ορίζεται, από τις εξόδους προς τις εισόδους, από το σύνολο όλων των συνδέσεων (nets), συνδυαστικών κόμβων και πρωτευόντων εισόδων που βρίσκονται διασυνδεδεμένα στην είσοδο ενός flip flop [12]. Μέσα σε αυτό λαμβάνονται υπόψιν όλα τα στοιχεία που περικλείονται μέσα στον λογικό κώνο, ο οποίος εκτείνεται έως το σημείο όπου συναντάται άλλο flip-flop ή κάποιο σήμα αρχικοποίησης (primary input).

Η φιλοσοφία της εισαγωγής σφαλμάτων με αυτήν τη μέθοδο βρίσκεται στο ότι αντί να εισαχθούν πολλαπλά σφάλματα με τυχαίο τρόπο εισάγονται σφάλματα σε πολλαπλά στοιχεία (flip-flops) τα οποία ανήκουν στον υπό ανάλυση λογικό κώνο. Αυτό συνεπάγεται ότι πριν από την εισαγωγή σφαλμάτων καθορίζονται οι λογικοί κώνοι και διαπιστώνεται ποια flip-flops εμπεριέχονται σε κάθε έναν από αυτούς. Τα σφάλματα εισάγονται στη συνέχεια στα συγκεκριμένα flip-flops και αναμένεται να επηρεάσουν μόνο τα στοιχεία (flip-flops) που εμπεριέχονται εσωτερικά του λογικού κώνου που εισήχθησαν τα σφάλματα και όχι άλλα στοιχεία του κυκλώματος με τα οποία δεν προκύπτει καμία σύνδεση με τον υπό εξέταση λογικό κώνο. Με αυτό τον τρόπο μπορεί να επιτευχθεί η εισαγωγή πολλαπλών σφαλμάτων με

χαρακτηριστικά υψηλής τοπικότητας. Στον υπό ανάλυση λογικό κώνο περιλαμβάνονται κάθε φορά και στοιχεία τα οποία μπορεί να εμπίπτουν ταυτόχρονα σε δύο λογικούς κώνους (επηρεάζονται από τα σφάλματα και των δύο λογικών κώνων). Αυτό συμβαίνει όταν δύο λογικοί κώνοι τέμνονται μεταξύ τους όπως φαίνεται στο παρακάτω Εικόνα 3.



Εικόνα 3 Μοντέλο Λογικών Κώνων

Συνοψίζοντας το μοντέλο εισαγωγής σφαλμάτων λογικών κώνων μπορεί να εφαρμοστεί μόνο στα flip-flops που βρίσκονται σε ένα συγκεκριμένο κομμάτι του RTL σχεδίου, εξομοιώνοντας μια στοχευμένη επίθεση με laser. Τα αποτελέσματά αυτά μπορεί να είναι αντιπροσωπευτικά και μετά το στάδιο της σύνθεσης εφόσον τα συγκεκριμένα στοιχεία που περικλείονται μέσα στον λογικό κώνο εξακολουθούν να βρίσκονται διασυνδεδεμένα (λειτουργικά) μεταξύ τους.

Τα flip-flops που βρίσκονται έξωθεν του λογικού κώνου αναμένεται να μην επηρεαστούν, συμπεριλαμβανομένου των flip-flop που εμπεριέχονται σε τυχόν τομή του λογικού κώνου.

5.2 Στατιστική Μέθοδος Εισαγωγής Σφαλμάτων

Για την μέτρηση της αξιοπιστίας ενός κυκλώματος ή ειδικότερα ενός ολοκληρωμένου χρησιμοποιούνται όπως αναφέρθηκε και νωρίτερα πολλές μέθοδοι και τεχνικές εισαγωγής σφαλμάτων. Στόχος πάντα είναι να επιτευχθεί το επιθυμητό αποτέλεσμα σε ένα εύλογο χρονικό διάστημα, το οποίο όμως να επιτρέπει την εξαγωγή ασφαλών και αντιπροσωπευτικών συμπερασμάτων. Με σεβασμό στα διαφορετικά στάδια υλοποίησης επιδιώκεται κάθε φορά, κατά το δυνατόν, η συντόμευση των διαδικασιών των διαφόρων σχεδιαστικών δοκιμών.

Όσο αυξάνεται η πολυπλοκότητα των κυκλωμάτων τόσο δυσκολότερη και πιο χρονοβόρα γίνεται και η ανάλυσή τους μέσα από μοντέλα σφαλμάτων. Ο πιθανός αριθμός εισαγωγής σφαλμάτων για την κάλυψη όλων των πιθανοτήτων μπορεί να φτάσει σε δυσθεώρητα νούμερα, πράγμα που οδηγεί στην υιοθέτηση περισσότερων πρακτικών λύσεων. Για αυτό το λόγο είθισται να επιλέγεται ένα τυχαίο υποσύνολο πιθανών σφαλμάτων το οποίο αποτελεί ένα αντιπροσωπευτικό δείγμα εκ του συνόλου των πιθανών σφαλμάτων.

Ως προς την πρακτικότερη εφαρμογή εισαγωγής σφαλμάτων και την εμπιστοσύνη γύρω από τα μετρούμενα αποτελέσματα, αναφέρεται ότι έχουν πραγματοποιηθεί μελέτες οι οποίες προτείνουν μέσα από στατιστικές μεθόδους τη δυνατότητα υπολογισμού ενός ελάχιστου απαιτούμενου αριθμού εισαγωγής σφαλμάτων με δεδομένο κάποιο περιθώριο λάθους επί των

μετρούμενων αποτελεσμάτων. Η μέθοδος αυτή προσφέρει κάποιου είδους ποσοτικοποίηση όσον αφορά στα περιθώρια λάθους (margin of errors) γύρω από τα αποτελέσματα και σε τι βαθμός-διάστημα εμπιστοσύνης ανταποκρίνονται.

Δεδομένου ότι ο αριθμός των στοιχείων ενός κυκλώματος είναι ένας πεπερασμένος αριθμός (μπορεί να είναι αριθμητικά τεράστιος), εντούτοις τα N στοιχεία μνήμης που υπάρχουν εξαρτώνται από το εκάστοτε κύκλωμα και είναι αυτά στα οποία μπορεί να εκδηλωθεί μία επίθεση η οποία δύναται να τους μεταβάλλει την τιμή τους κατά την εκδήλωσή της κατά τη διάρκεια λειτουργίας του κυκλώματος σε δεδομένους κύκλους ρολογιού [13].

Ειδικότερα ο αριθμός των τυχαίων υπό εισαγωγή σφαλμάτων, μπορεί να υπολογιστεί μέσω μίας προτεινόμενης μαθηματικής εξίσωσής έχοντας τα εξής δεδομένα σύμφωνα με [13]:

$$n = \frac{N}{1 + e^2 \ * \frac{N-1}{t^2 \ * p * (1-p)}}$$

- όπου N ο αριθμός όλων των στοιχείων μνήμης του κυκλώματος για παράδειγμα flip-flops.

- όπου p ορίζεται το τυπικό σφάλμα (χαρακτηριστικό με δεδομένη πιθανότητα οδήγησης σε σφάλμα)

- όπου e το περιθώριο λάθους το οποίο πρέπει να βρίσκεται εντός διαστήματος [Peval – e ; Peval + e], ή [Peval – e*100 ; Peval + e*100] (ποσοστό).

- όπου t αντιστοιχεί στο επίπεδο εμπιστοσύνης και δείχνει την πιθανότητα η ακριβής τιμή να βρίσκεται στην πραγματικότητα εντός του διαστήματος σφάλματος. Συνήθως επιλέγεται ένα ποσοστό εμπιστοσύνης 95%.

Για το λόγο αυτό είναι σημαντικό όλα τα παραπάνω να προσμετρώνται κατά το στάδιο σχεδιασμού και να λαμβάνονται κατάλληλα και αποτελεσματικά αντίμετρα έναντι επιθέσεων βασιζόμενων σε διαφορετικές μεθόδους εισαγωγής σφαλμάτων. Η επικαιροποίηση των αντίστοιχων εργαλείων ανάλυσης με γνώμονα την ασφάλεια μπορούν να παρέχουν αποτελεσματική ανάδειξη των αδυναμιών των υπό σχεδίαση κυκλωμάτων. Οι διαδικασίες αυτές θα πρέπει να εφαρμόζονται όχι μόνο ως αντίμετρα για την αντιμετώπιση γνωστών επιθέσεων αλλά και ως μέτρα πρόληψης μελλοντικών επιθέσεων.

6. Vivado & Vivado HLS (Flow)

Το Vivado High-Level Synthesis της Xilinx αποτελεί ένα εργαλείο το οποίο χρησιμοποιείται για τον σχεδιασμό, προσομοίωση και υλοποίηση (σύνθεση) ενός έργου (project) σε γλώσσα περιγραφής υλικού (HDL). Δέχεται ως είσοδο προγράμματα σε γνωστές γλώσσες προγραμματισμού όπως C, C++ ή System C (HLL) και δημιουργεί το αντίστοιχο αρχείο υλοποίησης Register Transfer Level (RTL) σε επίπεδο λογικής ροής (λογικών πυλών). Το παραγόμενο αρχείο RTL είναι σε μορφή VHDL ή Verilog. Επόμενο στάδιο, μετά την παραγωγή του RTL αρχείου, είναι η σύνθεση για τον προγραμματισμό ενσωματωμένων συστημάτων Field Programmable gate array (FPGA), τα οποία λόγω των χαρακτηριστικών της ιδιότητας της μαζικής παράλληλης αρχιτεκτονικής του FPGA, προσφέρουν καλύτερες επιδόσεις. Οι επιδόσεις αυτές μπορεί να αφορούν στο επίπεδο της ενέργειας (κατανάλωση ισχύος) της ταχύτητας, στο επίπεδο του κόστους καθώς και εν γένει στην αύξηση της παραγωγικότητας μέσα από τη μείωση του χρόνου υλοποίησης μεταξύ των διάφορων σταδίων υλοποίησης και της επαναχρησιμοποίησης του κώδικα. Στην συγκεκριμένη εργασία χρησιμοποιήθηκε η έκδοση Vivado 2020.1 (web pack).

Η ροή σχεδίασης που ακολουθείται σε αυτό το εργαλείο επιτρέπει στους εκάστοτε σχεδιαστές να καταφέρνουν να παράγουν ένα αξιόλογο και σύμφωνα με τις προδιαγραφές υλικό χρησιμοποιώντας ένα κώδικα γλώσσας υψηλού επιπέδου που έχει ενσωματωμένη όμως μία λογική υλικού. Σε κάθε project μπορεί να συμπεριληφθούν περιορισμοί (constrains) σχετικά με την επιθυμητή υλοποίηση. Οι περιορισμοί αυτοί μπορεί να αφορούν για παράδειγμα την περίοδο του ρολογιού ή την επιλογή του τύπου (μοντέλου) του FPGA. Επίσης μεγάλο πλεονέκτημα του Vivado HLS είναι οι οδηγίες ή αλλιώς ντιρεκτίβες (directives) που εισάγονται στο project από τον σχεδιαστή για να προσδιοριστούν ποιες και αν επιθυμείται να εφαρμοστούν βελτιστοποιήσεις (optimizations) πάνω στο σχεδιασμό. Τα παραπάνω μπορούν να εισαχθούν είτε μέσα από το γραφικό περιβάλλον του Vivado είτε μέσα από τη γραμμή εντολών του [14].

Κατά το στάδιο της σύνθεσης παρουσιάζεται ο “λογικός” σχεδιασμός του κώδικα σε C με συγκεκριμένο τρόπο. Οι αλγόριθμοι της σύνθεσης λαμβάνοντας υπόψη τους περιορισμούς και τις παραμετροποιήσεις προσπαθούν να παράξουν το καλύτερο δυνατό αποτελέσματα για τον σχεδιασμό του κάθε project. Ένα έργο (project) Vivado HLS περιέχει τον κώδικα σε γλώσσα υψηλού επιπέδου, τους περιορισμούς (constrains), τις οδηγίες (directives) και ένα αρχείο testbench για την προσομοίωση του κώδικα σε C πριν από τη σύνθεση και για την επαλήθευση της εξόδου RTL.

Συνοψίζοντας αυτό το ευέλικτο εργαλείο Vivado HLS προσφέρει την αναπαράσταση της ζητούμενης λειτουργικότητας του υλικού (μέσα από την επιβολή καθορισμένων περιορισμών και βελτιστοποιήσεων) αναλαμβάνοντας τελικά το ίδιο το εργαλείο να προσομοιώσει και να δημιουργήσει αυτόματα μία σύνθεση σε γλώσσα περιγραφής υλικού καθώς και την μετέπειτα υλοποίηση του υλικού σε FPGA.

Τα στάδια του σχεδιασμού του Vivado HLS είναι ο χρονοπρογραμματισμός (Scheduling) όπου καθορίζεται ποιες λειτουργίες λαμβάνουν χώρα κατά τη διάρκεια ποιου κύκλου ρολογιού (clock cycle), η συστοίχιση (binding) όλων των ηλεκτρονικών στοιχείων όπου καθορίζεται ποιος πόρος υλικού υλοποιεί ποια προγραμματισμένη λειτουργία. Τέλος η εξαγωγή λογικής ελέγχου για τη δημιουργία μιας μηχανής πεπερασμένης κατάστασης (FSM) που είναι συμβατή με τις πράξεις στο σχέδιο RTL.

Αναλυτικότερα τα πλεονεκτήματα σχεδιασμού μέσα από τη χρήση του Vivado HLS είναι:

- Η βελτίωση της παραγωγικότητας, καθώς οι σχεδιαστές του υλικού μπορούν να προβούν σε δοκιμαστικές σχεδιάσεις σε ένα ανώτερο επίπεδο αφαίρεσης και να εφαρμόζουν αλλαγές καθ’ όλη τη διάρκεια έως την τελική υλοποίηση. Μειωμένος χρόνος ανάπτυξης υλικού.

- Παραγωγή αποδοτικότερου υλικού, με καλύτερες προδιαγραφές και επιδόσεις. Η βελτίωση των επιδόσεων ακολουθεί και εν γένει τα συστήματα που ενσωματώνεται το παραγόμενο υλικό.

- Ποιοτικότερο υλικό, το οποίο συνεπάγεται καλύτερες επιδόσεις και του λογισμικού (software) αφού παρέχεται στους προγραμματιστές η δυνατότητα γρηγορότερης εκτέλεσης των χρονοβόρων αλγοριθμικών υπολογισμών μέσα από τη χρήση FPGA.

- Γρηγορότερη επαλήθευση της λειτουργικότητας σε επίπεδο γλώσσας υψηλού επιπέδου C, σε σχέση με τις κλασικές γλώσσες περιγραφής υλικού.

- Έλεγχος των βελτιστοποιήσεων της σχεδίασης μέσω των καθορισμένων οδηγιών (directives). Δημιουργία υλικού συγκεκριμένων επιδόσεων και προδιαγραφών εξ αρχής προκαθορισμένων.

- Από τον ίδιο αρχικό κώδικα C δημιουργία πολλαπλών υλοποιήσεων και δυνατότητα προσομοίωσης και καταγραφής της καλύτερης επίδοσης (αναλόγως των βελτιστοποιήσεων).

- Δοκιμές της χωροταξικής τοποθέτησης των υλικών σχεδιασμού με σκοπό την εύρεση της βέλτιστης υλοποίησης.

- Επαναχρησιμοποιούμενος συμβατός κώδικας C για πολλαπλές υλοποιήσεις και μοντέλα FPGA. Ενσωμάτωσή του σε νέους σχεδιασμούς και συσκευές.

Εν κατακλείδι οι σχεδιαστές υλικού καταβάλουν λιγότερη προσπάθεια και χρόνο σχεδιασμού και επανασχεδιασμού καθώς έχουν εικόνα του σχεδιαστικού αποτελέσματος και αμεσότερη ανατροφοδότηση για τον αν το παραγόμενο υλικό πληροί τις απαιτούμενες προδιαγραφές του έργου (project). Μέσα από τις παραπάνω περιγραφόμενες διαδικασίες ενισχύεται ο τομέας της ασφάλειας κατά τη διάρκεια της παραγωγής και ως εκ τούτου το υλικό

που προκύπτει είναι σαφώς πιο βελτιωμένο και ασφαλές. Η επιβολή ή μη βελτιστοποιήσεων δεν επηρεάζουν την ποιότητα του υλικού.

Συγκρατείται ότι ανάλογα με το απαραίτητο επίπεδο ανθεκτικότητας από τη σκοπιά της ασφάλειας θα πρέπει να γίνεται η ανάλογη χρήση κρυπτογραφικών αλγορίθμων με κατάλληλα αντίμετρα και τεχνικές μετριασμού των γνωστών ευπαθειών του υλικού. Τα μέτρα προστασίας θα πρέπει να εφαρμόζονται στο υψηλότερο επίπεδο αφαίρεσης, από τα πρώτα στάδια σχεδιασμού, ώστε να ενισχύεται όπως προαναφέρθηκε η παραγωγικότητα, η επαναχρησιμοποίηση αλλά και η επεκτασιμότητα του ήδη δοκιμασμένου υλικού.

7. AES – Advanced Encryption Standard

Ο αλγόριθμος Advanced Encryption Standard (AES), παλαιότερα αλγόριθμος Rijndael, είναι ένας συμμετρικός αλγόριθμος κρυπτογράφησης μπλοκ, όπου τα δεδομένα που κρυπτογραφούνται ή αποκρυπτογραφούνται είναι χωρισμένα σε μπλοκ των 128-bit (πίνακας συστοιχίας 4x4 των 16 byte) με το μέγεθος των κρυπτογραφημένων δεδομένων να παραμένει σταθερά το ίδιο ανεξαρτήτως του διαχωρισμού των δεδομένων.

Συνοπτικά αναφέρεται ότι κάθε μπλοκ δεδομένων τροποποιείται κατά τη διάρκεια εκτέλεσης πολλαπλών γύρων επεξεργασίας. Αρχικά γίνεται του διαίρεση των δεδομένων σε block (και την επέκτασή τους αν είναι απαραίτητα για το ορθό μήκος) και η δημιουργία από το αρχικό κλειδί των κλειδιών που θα χρησιμοποιηθούν στους επόμενους γύρους. Κάθε γύρος έχει ως είσοδο το μπλοκ του προηγούμενου γύρου και περιλαμβάνει τα επαναλαμβανόμενα βήματα, της αντικατάστασης των δεδομένων από τον πίνακα S-box, την μετατόπιση των γραμμών (συνήθως αριστερά), το ανακάτεμα των στηλών του μπλοκ, την προσθήκη του μυστικού κλειδιού του γύρου στα δεδομένα (Data XOR Key). Οι δέκα γύροι είναι πανομοιότυποι εκτός από τον τελευταίο γύρο ο οποίος δεν εκτελεί το ανακάτεμα των στηλών.

Ο AES υποστηρίζει τρία μεγέθη κλειδιών μήκους 128-bit, 192-bit και 256-bit με τους αντίστοιχους αριθμούς γύρων να είναι 10 γύροι (εννέα γύροι κρυπτογράφησης και ένας τελικός γύρος), 12 γύροι (έντεκα γύροι κρυπτογράφησης και ένας τελικός γύρος) ή 14 γύροι (δεκατρείς γύροι κρυπτογράφησης και ένας τελικός γύρος), αντίστοιχα. Η διαφορά στο μέγεθος του κλειδιού έγκειται στην απαιτούμενη υπολογιστική ισχύ κάθε φορά. Πρακτικά όσο μεγαλύτερο το μήκος κλειδιού τόσο μεγαλύτερη ασφάλεια παρέχεται, με κόστος όμως την κατανάλωση μεγαλύτερης υπολογιστικής ισχύος.

Όσον αφορά τώρα στο αποτέλεσμα της εξόδου του AES μετά τη συμμετρική κρυπτογράφηση, αυτό που λαμβάνεται είναι ένα σύνολο δεδομένων τα οποία είναι δυσνόητα για όποιον δεν κατέχει το σωστό κλειδί κρυπτογράφησης – αποκρυπτογράφησης. Το μπλοκ των κρυπτογραφημένων δεδομένων που προκύπτει κάθε φορά έχει ακριβώς το ίδιο μήκος 128-bit με το αντίστοιχο μη κρυπτογραφημένο μπλοκ (plaintext).

Αναφορικά με υλοποιήσεις του S-box μπορεί κανείς να ανατρέξει σε πολλές προτάσεις που έχουν γίνει για την επίτευξη διάφορων στόχων του AES. Πιο συγκεκριμένα, συμπαγείς υλοποιήσεις παρουσιάστηκαν αρχικά από τους Satoh et al [16] και Mentens et al [17] και εν συνεχεία από τον Canright D. [15], ο οποίος απέδωσε μία πιο βελτιωμένη εκδοχή του compact S-box των προαναφερόμενων [16].

Η υλοποίηση του προαναφερόμενου συμπαγούς (compact) S-Box του Canright βοήθησε σε ορισμένες περιπτώσεις στην επίτευξη βελτιωμένων επιδόσεων του S-Box σε ποσοστό 20% [15]. Αποτέλεσμα αυτής της βελτιωμένης επίδοσης ήταν η περαιτέρω υποβοήθηση εφαρμογών υλικού που απαιτούσαν περιορισμό ή και μείωση του χώρου υλοποίησης. Επίσης όσο περισσότερο συμπαγές είναι το S-box τόσο εξυπηρετεί στην καλύτερη απόδοση του παραλληλισμού του εκάστοτε κυκλώματος.

8. Ανάλυση των υπό αξιολόγηση κυκλωμάτων

Στην παρούσα εργασία προκειμένου να πραγματοποιηθεί η εισαγωγή των σφαλμάτων με τη χρήση του Vivado Simulator χρησιμοποιήθηκαν τα έτοιμα σχεδιαστικά κυκλώματα (designs)

που υλοποιήθηκαν σύμφωνα με [1]. Τα κυκλώματα αυτά αποτελούν μία υλοποίηση του αλγόριθμου υπολογισμού του S-box για το AES σύμφωνα με τον Canright D. [18].

Αναλυτικότερα, τα ήδη υλοποιημένα κυκλώματα που χρησιμοποιήθηκαν από την εργασία [1], αντιπροσωπεύουν την υλοποίηση του αλγόριθμου του S-Box του Canright [18] σε γλώσσα C και παραγωγή του αντίστοιχου υλικού με χρήση υψηλού επιπέδου (HLS). Οι έτοιμες υπό μελέτη σχεδιάσεις [1] που μελετήθηκαν διαφέρουν μεταξύ τους ως προς τις παραμέτρους του χρόνου (clock cycle) και του χώρου (place) του υλικού, παράγοντες οι οποίοι αποτελούν μείζον ζήτημα κατά τη διαδικασίας της σχεδίασης ενός ενσωματωμένου κυκλώματος. Η όσο το δυνατόν καλύτερη απόδοση αυτών των παραμέτρων, στις εκάστοτε προσαρμοσμένες απαιτήσεις κάθε σχεδίασης, είναι που θα προσφέρουν μεγαλύτερη ταχύτητα και εξοικονόμηση χώρου. Αυτό συνεπάγεται στο να δημιουργηθεί ένα καλύτερο τελικό προϊόν κατασκευαστικά, το οποίο όμως θα πληροί τις απαιτούμενες προϋποθέσεις με τον καλύτερο δυνατό τρόπο. Όλα τα παραπάνω θα πρέπει να γίνονται λαμβάνοντας υπόψη τόσο την ασφάλεια όσο και την αξιοπιστία των ενσωματωμένων κυκλωμάτων ώστε να μη χρειάζεται η επανάληψη της σχεδίασης σε μεταγενέστερο στάδιο υλοποίησης. Άρα σε αρχικό στάδιο ακόμη της σχεδίασης απαιτείται η πρακτική δοκιμή των κυκλωμάτων [1] για τη διαπίστωση της αντοχής τους τόσο σε εχθρικά περιβάλλοντα όσο και στην εσκεμένη εισαγωγή σφαλμάτων. Μετά την αξιολόγηση των αποτελεσμάτων των δοκιμών μπορούν να υπάρξουν διορθώσεις – βελτιώσεις όπως για παράδειγμα η ενσωμάτωση αντιμέτρων για την καλύτερη οχύρωση από πλευράς ασφάλειας ή την αντικατάσταση στοιχείων περισσότερο ανθεκτικών κατά την έκθεσή τους σε επιβλαβείς συνθήκες. Οι πληροφορίες που συγκεντρώνονται σε αυτό το στάδιο είναι αρκετά σημαντικές για τη συνολική τελική ποιότητα των ενσωματωμένων κυκλωμάτων.

Με το εργαλείο Vivado HLS, όπως ήδη αναφέρθηκε, έγινε στις διάφορες σχεδιάσεις [1] η προσομοίωση της εισαγωγής των σφαλμάτων σε συμπεριφορικό (Behavioral) επίπεδο. Κάθε σχεδίαση έχει διαφορετικές προκαθορισμένες απαιτήσεις και περιορισμούς. Αυτές οι διαφορετικές παραμετροποιήσεις έχουν εφαρμοστεί με τη χρήση ντιρεκτίβων του εργαλείου Vivado HLS και σκοπός της παρούσας εργασίας είναι να εξετάσει ποια σχεδίαση φαίνεται να είναι περισσότερο ικανοποιητική από θέμα ασφάλειας και αξιοπιστίας και με ποιο τρόπο φαίνεται να επιδρά αυτή η χρήση ντιρεκτίβων σε κάθε σχεδίαση.

Όπως προαναφέρθηκε η γενική υλοποίηση των σχεδιάσεων που χρησιμοποιήθηκαν από [1] αφορά στον αλγόριθμο του S-Box του Canright D. [18]. Η κάθε σχεδίαση προσομοιώνεται με τη χρήση του εργαλείου Vivado HLS όπου μελετάται το αποτέλεσμα της επίδρασης της εισαγωγή σφαλμάτων στα κυκλώματα της εκάστοτε σχεδίασης. Ειδικότερα, κάθε σχεδίαση παρουσιάζεται σε τρείς επιμέρους υποπεριπτώσεις, που έχουν όμως ως είσοδο την ίδια σχεδίαση (design), οι οποίες η καθεμία προκαθορίζει διαφορετικές απαιτήσεις υλοποίησης κάθε κυκλώματος. Ουσιαστικά επιβάλει ή όχι στο εργαλείο Vivado HLS την παραγωγή του αντίστοιχου υλικού με συγκεκριμένους κανόνες, οι οποίοι πρέπει να πληρούνται. Αυτοί οι κανόνες αποτελούν τις παραμετροποιήσεις και έχουν εφαρμοστεί με τη χρήση ντιρεκτίβων του εργαλείου Vivado HLS.

Τα τρία διαφορετικά κυκλώματα που χρησιμοποιήθηκαν σύμφωνα με [1], μελετώνται για προκειμένου να διαπιστωθεί κατά πόσο επηρεάζονται αν σε αυτά γίνει επιβολή διαφορετικών οδηγιών - ντιρεκτίβων (π.χ. inline, no-inline, loop unrolling). Επομένως επιδιώκεται η σύγκριση της συμπεριφοράς του ίδιου κυκλώματος δηλαδή της ίδιας σχεδίασης (Design) όταν σε αυτό εφαρμοστούν διαφορετικές ντιρεκτίβες (Solution 1, Solution 2, Solution 3). Η εργασία αυτή παραθέτει κατά πόσο επηρεάζεται κάθε σχεδίαση (ανά Solution) και κατ' επέκταση η συμπεριφορά αυτής όταν το εκάστοτε κύκλωμά της υποβάλλεται σε διαδικασία εισαγωγής σφαλμάτων (fault injection) αλλά επιπρόσθετα και η αξιολόγηση των αποτελεσμάτων των διαφορετικών μεταξύ τους σχεδιάσεων.

Αρχικά αποθηκεύεται στην μεταβλητή gold_out η αναμενόμενη ορθή τιμή εξόδου του κυκλώματος. Οσο αφορά τη διεπαφή του κυκλώματος, με το σήμα ap_return επιστρέφεται η τιμή εξόδου του S-box, με το σήμα ap_done μπορούμε να γνωρίζουμε (όταν πάρει την τιμή 1) πότε η έξοδος είναι έτοιμη να διαβαστεί (έχουν ολοκληρωθεί οι λειτουργίες του κυκλώματος). Με βάση αυτά τα σήματα και τις μεταβλητές μπορεί να αξιολογηθεί η σωστή λειτουργία και η αξιοπιστία των κυκλωμάτων και να διαπιστωθεί κατά πόσο αυτή επηρεάζεται όταν κατά τη διάρκεια της λειτουργίας τους πραγματοποιηθεί εισαγωγή σφαλμάτων.

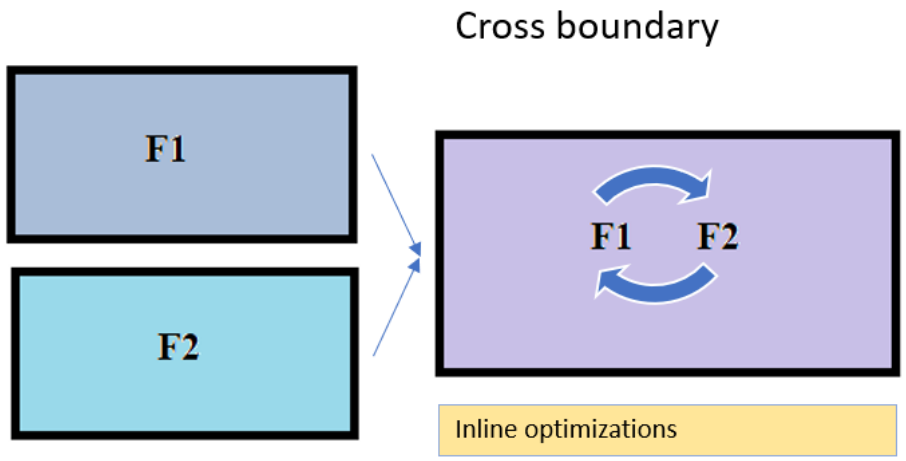
8.1 Σχεδίαση 1 – Simple Canright (Design 1 ή d1)

Η λειτουργικότητα της συγκεκριμένης σχεδίασης d1 είναι ότι πραγματοποιεί υπολογιστικά το S-Box χωρίς να εμπεριέχονται στη σχεδίαση αντίμετρα. Οι υπολογισμοί αυτοί περιλαμβάνουν μία αντιστροφή πολυωνύμου εβδόμου βαθμού που πραγματοποιείται στο πεδίο GF (28) και έναν εξαρτώμενο μετασχηματισμό (affine transformation). Χρησιμοποιούνται δηλαδή οι κανονικοί υπολογισμοί του S-bytes οι οποίοι γίνονται πάνω στα δεδομένα εισόδου που δίνονται τη στιγμή εκείνη (on the fly) και όχι έτοιμοι υπολογισμοί από πίνακες αληθείας (look up tables). Με την δυνατότητα παραλληλοποίησης σε επίπεδο υλικού επιτυγχάνεται η αύξηση της ταχύτητας υπολογισμού πολλαπλών S-Box..

8.1.1 Υποπερίπτωση 1 - Design 1 Solution 1 (Sol1)

Η συγκεκριμένη υποπερίπτωση Solution 1 της σχεδίασης Design 1 επιτρέπει την εφαρμογή της προκαθορισμένης σύνθεσης υψηλού επιπέδου του Vivado HLS. Αν για παράδειγμα υπάρχει κάποια λειτουργία (function) ή μικρός βρόγχος (loop) που μπορεί να δεχτεί βελτιστοποίηση εφαρμόζει τις παραμετροποιήσεις inline ή unroll. Επίσης αν είναι read_only οι πίνακες που χρησιμοποιούνται δεσμεύει read_only στοιχεία μνήμης (δηλαδή ram, όχι bram ή lut).

Η παραμετροποίηση inline (Εικόνα 4) που αναφέρθηκε επιτρέπει κατά την εκτέλεση του κώδικα αν υπάρχει κλήση κάποιας εμφωλευμένης συνάρτησης (function) κατά την εκτέλεση του κυρίως (main) κώδικα, τότε μεταφέρει την κλήση αυτής της εμφωλευμένης συνάρτησης μέσα στον αρχικό κώδικα επιτρέποντας βελτιστοποιήσεις στη νέα λειτουργικότητα συνολικά. Οι βελτιστοποιήσεις γίνονται ανάμεσα σε διαφορετικές functions που έχουν οριστεί μέσα στον κώδικα ή και στο εσωτερικό της καθεμιάς.



Εικόνα 4 Inline Optimizations

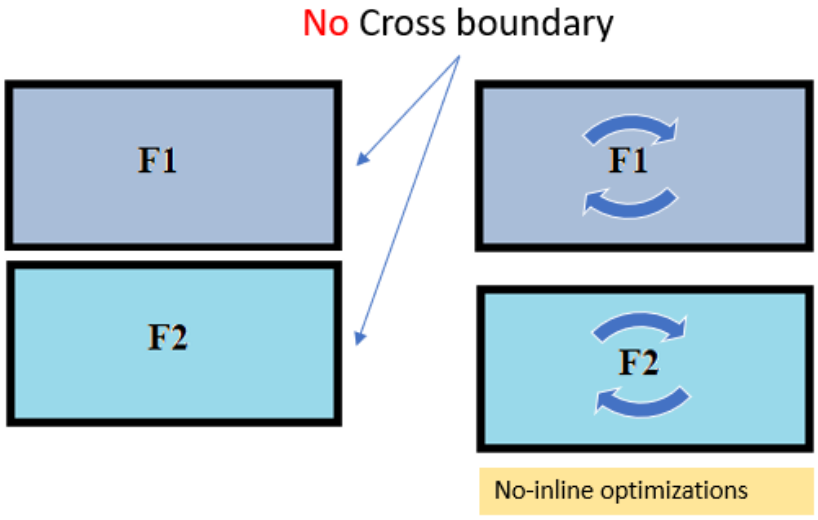
Η παραμετροποίηση unroll επιτρέπει κατά την εκτέλεση του κώδικα αν έχει ένα βρόγχο (loop) for στο υλικό, μπορεί να εφαρμόσει το λεγόμενο unroll στον βρόγχο ώστε να μπορούν να γίνουν οι υπολογισμοί ταυτόχρονα αν δεν υπάρχουν εξαρτήσεις δεδομένων (data dependencies), δηλαδή τα δεδομένα πρέπει να είναι ανεξάρτητα μεταξύ τους. Έτσι επιτυγχάνεται παραλληλοποίηση των υπολογισμών.

8.1.2 Υποπερίπτωση 2 – Design 1 Solution 2 (Sol2)

Με τη συγκεκριμένη υποπερίπτωση Solution 2 της σχεδίασης Design 1 επιβάλλεται στο εργαλείο Vivado HLS η εφαρμογή του loop unrolling στη σχεδίαση υψηλού επιπέδου. Εδώ ο ορισμός του unroll έγινε με το όρισμα factor=8. Το οκτώ (8) επιλέχθηκε διότι τόσες είναι και οι επαναλήψεις των δύο βρόγχων που υπάρχουν μέσα στον κώδικα. Σε αυτήν την περίπτωση έχουμε πλήρη παραλληλοποίηση (full unroll) ενώ αντίθετα αν το νούμερο αυτό είναι μικρότερο (πολλαπλάσιο) των επαναλήψεων έχουμε μερική παραλληλοποίηση (partial unroll). Επιπρόσθετα για την πλήρη παραλληλοποίηση χρειάζεται τα δεδομένα να είναι ανεξάρτητα μεταξύ τους.

8.1.3 Υποπερίπτωση 3 – Design 1 Solution 3 (Sol3)

Με τη συγκεκριμένη υποπερίπτωση Solution 3 της σχεδίασης Design 1 επιβάλλεται στο εργαλείο Vivado HLS να μην εφαρμόσει παραμετροποίηση inline δηλαδή επιβάλλονται ρυθμίσεις no-inline (Εικόνα 5). Επομένως δεν γίνονται βελτιστοποιήσεις στη σχεδίαση υψηλού επιπέδου. Έτσι η κάθε λειτουργία (function) θα υλοποιηθεί ως έχει. Επιτρέπεται να γίνουν συγκεκριμένες βελτιστοποιήσεις, αν αυτό χρειάζεται, στο εσωτερικό της κάθε λειτουργίας (function) και όχι από κοινού βελτιστοποιήσεις ανάμεσα σε διαφορετικές λειτουργίες (functions). Διατυπώνοντάς το λίγο διαφορετικά δεν μπορεί να συμβεί διασταύρωση ορίων (cross boundary) όπως φαίνεται και στο παρακάτω σχεδιάγραμμα. Επίσης γίνεται καθορισμός του τύπου της μνήμης που θα χρησιμοποιηθεί από τους πίνακες, οι οποίο ορίζονται μέσα στον κώδικα του solution (sol5). Στους πίνακες αυτούς γράφονται οι οδηγίες (directives).

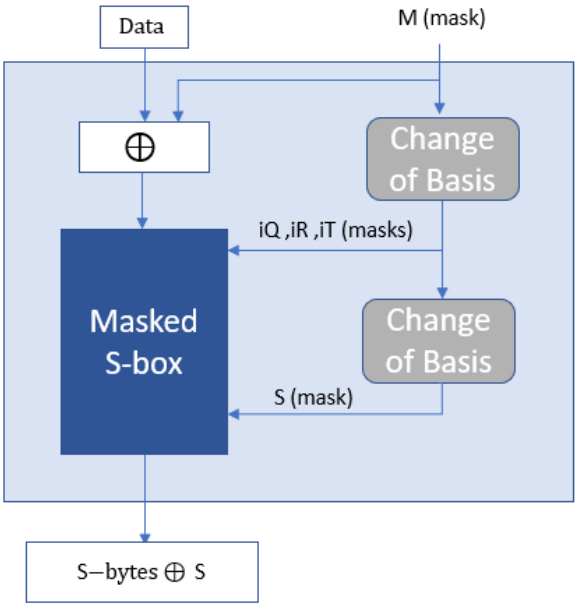


Εικόνα 5 No-inline optimizations

8.2 Σχεδίαση 2 – Masked Canright HLS (Design 2 ή d2)

Σε αυτή τη σχεδίαση χρησιμοποιείται πάλι το κύκλωμα που χρησιμοποιήθηκε και στην προηγούμενη σχεδίαση Simple Canright (Design 1), χρησιμοποιούνται δηλαδή οι κανονικοί υπολογισμοί του S-bytes οι οποίοι γίνονται πάνω στα δεδομένα εισόδου που δίνονται τη στιγμή εκείνη (on the fly). Η διαφορά αυτής της σχεδίασης έγκειται στην εφαρμογή ενός επιπέδου ασφάλειας ενάντια σε επιθέσεις πλευρικού καναλιού (Side Channel Analysis attacks). Η ασφάλεια αυτή επιτυγχάνεται χρησιμοποιώντας τυχαίες μάσκες (masking) [1]. Διευκρινιστικά αναφέρεται ότι οι τυχαίες τιμές των μασκών αυτών, για την αξιολόγηση του κυκλώματος υπό εισαγωγή σφαλμάτων, δεν αλλάζουν σε κάθε επεξεργασία δεδομένων, δηλαδή σε κάθε γύρω του S-box ή σε κάθε εισαγωγή σφάλματος. Η τιμή της κάθε μάσκας παραμένει σταθερή κατά τη

διάρκεια εκτέλεσης του κώδικα. Συνολικά υπάρχουν πέντε μάσκες οι οποίες είναι η S, iQ, iR, iT και η M. Η μία μάσκα M χρησιμοποιείται για τα δεδομένα και οι υπόλοιπες τέσσερις μάσκες S, iQ, iR, iT για τους ενδιάμεσους υπολογισμούς του S-box. Ο στόχος είναι να επιτευχθεί η αλλαγή της συσχέτισης των ενδιάμεσων τιμών που υπολογίζονται κατά τη διάρκεια εκτέλεσης του κώδικα σε σχέση με την παραγόμενη διαρροή πλευρικού καναλιού (side channel leakage). Λόγω του ότι οι ενδιάμεσες τιμές υπολογισμού του S-box φέρουν μυστική πληροφορία, καθώς περιέχουν κομμάτι αυτούσιων των δεδομένων, επιχειρείται να γίνει “καμουφλάρισμα” των ενδιάμεσων τιμών ώστε σε περίπτωση επίθεσης πλευρικού καναλιού (side channel attack) η μέτρηση της διαρροής (leakage) να εμπεριέχει “καμουφλαρισμένα” τα δεδομένα. Το τελικό αποτέλεσμα S-bytes περιέχει πολλαπλές πράξεις xor των αρχικών δεδομένων με τις μάσκες (Εικόνα 6)



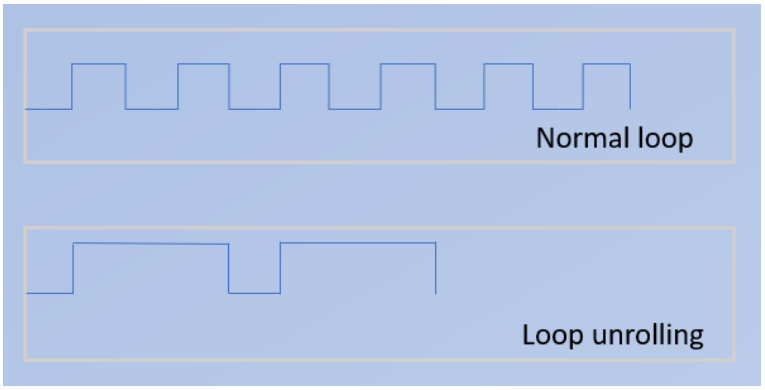
Εικόνα 6 Υλοποίηση S-Box με Masked αντίμετρο [1]

8.2.1 Υποπερίπτωση 1 – Design 2 Solution 1 (Sol1)

Όπως και στην προηγούμενη σχεδίαση (Design 2 Simple Canright), έτσι και εδώ στο Design 4 η συγκεκριμένη υποπερίπτωση Solution 1 διέπεται από τις ίδιες παραμετροποιήσεις που εφαρμόστηκαν στο αντίστοιχο Solution 1 της προηγούμενης σχεδίασης. Αυτές είναι η εφαρμογή της προκαθορισμένη λειτουργίας (default) του εργαλείου Vivado HLS. Με την προεπιλεγμένη αυτή λειτουργιά επιτρέπεται στο εργαλείο να προβεί σε μικρές αναγκαίες βελτιστοποιήσεις. Υπενθυμίζεται ότι οι παραμετροποιήσεις inline ή unroll (αναλυτικότερη επεξήγηση έχει δοθεί στην προηγούμενη σχεδίαση Design 2 Solution 1) αποτελούν τέτοιου είδους βελτιστοποιήσεις. Σκοπός είναι επίσης η μείωση του χρόνου εκτέλεσης του κώδικα, επομένως αύξηση της ταχύτητας, και η εξοικονόμηση κατανάλωσης ισχύος. Σε κάθε περίπτωση όμως, το υπό εξέταση Solution 1 αφορά στην σχεδίαση Masked Canright HLS. Δηλαδή εφαρμόζονται κατά την εκτέλεση του κώδικα οι προαναφερόμενες επιλεγμένες μάσκες S, iQ, iR, iT και M κατά τους υπολογισμούς των ενδιάμεσων τιμών του S-box.

8.2.2 Υποπερίπτωση 2 – Design 2 Solution 2 (Sol2)

Με τη συγκεκριμένη υποπερίπτωση Solution 2 της σχεδίασης Design 2 επιβάλλεται στο εργαλείο Vivado HLS η παραμετροποίηση loop unrolling. Με αυτό το "ξετύλιγμα" του βρόγχου (loop unrolling) [21] επιδιώκεται η αύξηση του όγκου των εντολών που εκτελούνται μέσα σε έναν βρόγχο, με συνέπεια τη μείωση του αριθμού των φορών που θα εκτελεστεί τελικά ο βρόγχος. Η τεχνική εφαρμόζεται λαμβάνοντας υπόψιν τη δομής του βρόγχου δηλαδή τη λογική εκτέλεσή του. Έτσι επιτυγχάνεται μείωση της επιβάρυνσης του βρόγχου και άρα ταχύτερη επίδοση του κώδικα (Εικόνα 7). Συνήθως για το σκοπό αυτό δημιουργούνται πολλαπλά αντίγραφα (αύξηση του αριθμού εντολών του κώδικα) του εσωτερικού του βρόγχου και τροποποιείται ανάλογα ο τερματισμός του. Επίσης με το loop unrolling χρησιμοποιούνται περισσότεροι καταχωρητές. Επομένως το loop unrolling πρέπει να επιλέγεται όταν, υπάρχουν αρκετοί διαθέσιμοι πόροι για την παραλληλοποίηση της λειτουργίας του αλγορίθμου.



Εικόνα 7 Βελτιστοποίηση Loop unrolling

Πάλι εφαρμόζονται οι επιλεγμένες μάσκες S, iQ, iR, iT και M ως αντίμετρο ασφαλείας. Άρα πάνω στην ίδια σχεδίαση που μελετάται εφαρμόζονται διαφορετικές οδηγίες (directives) που υπαγορεύουν στο εργαλείο τι να κάνει και πώς να το κάνει. Εδώ έχουμε πλήρη παραλληλοποίηση (full unroll).

8.2.3 Υποπερίπτωση 3 – Design 2 Solution 3 (Sol3)

Στη συγκεκριμένη υποπερίπτωση Solution 3 της σχεδίασης Design 2 τίθεται περιορισμός στις βελτιστοποιήσεις που μπορεί να κάνει το εργαλείο Vivado HLS. Του επιβάλλεται μέσα από τον κώδικα να εφαρμόσει παραμετροποίηση no-inline (σχήμα 2). Επομένως δεν γίνονται βελτιστοποιήσεις ανάμεσα στις συναρτήσεις (no cross boundary). Έτσι η κάθε συνάρτηση (function) θα υλοποιηθεί ως έχει. Οι μόνες βελτιστοποιήσεις που επιτρέπεται να γίνουν, εφόσον είναι εφικτό, είναι στο εσωτερικό της κάθε συνάρτησης (function).

8.3 Σχεδίαση 3– Cng (Design 3 ή d3)

Στη σχεδίαση αυτή έχει υλοποιηθεί πάλι το κύκλωμα της σχεδίασης Simple Canright (Design 1), με μία διαφορά όμως στο μήκος των δεδομένων. Ενώ στην προηγούμενη σχεδίαση τα δεδομένα ήταν μήκους 8 bit, στο Design 3 όπως δόθηκε σύμφωνα με [1] γίνεται επέκταση της αλληλουχίας (concatenation) και η πληροφορία είναι διπλάσιου μήκους ήτοι 16 bit. Γίνεται λοιπόν μία εισαγωγή πλεονασμού (redundancy) στο data path. Όλη η λειτουργικότητα του κώδικα και όλες οι μεταβλητές του χρησιμοποιούν μήκος πληροφορίας 16 bit. Οτιδήποτε στον κώδικα είναι σταθερό (στατικό) επεκτείνεται κατά 8 bit αντιγράφοντας τα αρχικά 8 bit του εαυτού του. Ο υπολογισμός του S-box πραγματοποιείται εις διπλούν (με το ίδιο κλειδί σε κάθε γύρο) και κατά την ολοκλήρωση των υπολογισμών γίνεται σύγκριση των δύο 8 bit αποτελεσμάτων μεταξύ τους. Γίνονται δύο υπολογισμοί της λειτουργίας του S-box και επομένως στην περίπτωση που

δεν υπάρχει κάποια παρέμβαση (εισαγωγή σφάλματος) θα πρέπει και τα δύο 8 bit αποτελέσματα που καταχωρούνται στο ap_return να είναι ίδια μεταξύ τους και ταυτόχρονα να συμφωνούν και με την αναμενόμενη ορθή τιμή εξόδου (gold_out).

Στην τρέχουσα σχεδίαση ουσιαστικά το συγκεκριμένο αντίμετρο πραγματοποιεί μια τεχνική πλεονασμού υλικού (redundancy) και αντί να χρησιμοποιηθεί ως αντίμετρο ανάλυσης πλευρικού καναλιού (side channel analysis) μετατρέπεται σε αντίμετρο ανίχνευσης σφαλμάτων (fault detection).

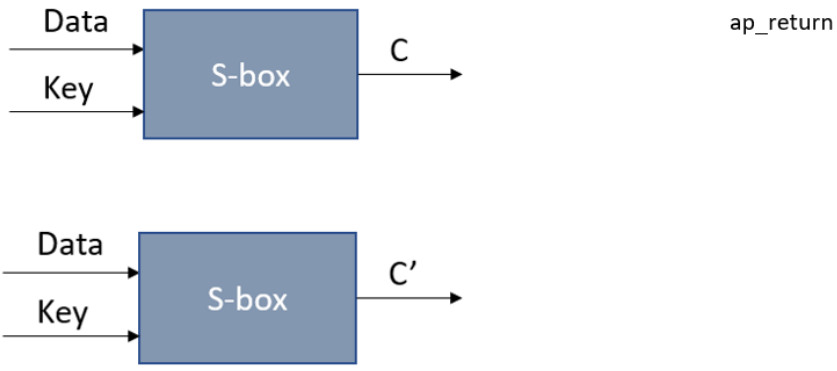
Αναλυτικότερα σχετικά με τα υπό μελέτη κυκλώματα:

- Τα αποτελέσματα των S-box μας επιστρέφονται στη μεταβλητή ap_return, η οποία είναι 32 bit. Τα τελευταία 16 bit είναι εκείνα που μας ενδιαφέρουν. Η σύγκριση των αποτελεσμάτων των S-box γίνεται ανάμεσα στο C και C' όπως φαίνονται και στο παρακάτω (Εικόνα 8).
- Όταν η μεταβλητή ap_done γίνει ίση με 1 τότε μπορούν να διαβαστούν οι έξοδοι των S-box. Στην περίπτωση που το ap_done δεν γίνεται ίσο με ένα αυτό σημαίνει ότι το κύκλωμα δεν κατάφερε να ολοκληρώσει επιτυχώς τη λειτουργία του, επομένως έχει οδηγηθεί σε μία μη αναμενόμενη συμπεριφορά (Hang). Σε αυτήν την κατηγορία εντάσσονται και τα αποτελέσματα των εξόδων όταν αυτές λάβουν αδιευκρίνιστες τιμές εξόδους (U,X κ.α).

- Αν οι δύο έξοδοι C και C' είναι ίδιες μεταξύ τους (C = C') και η τιμή τους συμφωνεί με την αναμενόμενη ορθή τιμή εξόδου τότε δεν υπάρχει σφάλμα ή εισήχθη αλλά δεν επηρέασε τα S-box (Silent error).

- Αν όμως ισχύει C = C' και η τιμή εξόδου διαφέρει από την αναμενόμενη τιμή εξόδου τότε υπάρχει σφάλμα το οποίο επηρέασε και τα δύο S-box κάτι που είναι σοβαρό και αντιστοιχεί σε ένα κρίσιμο σφάλμα (Critical error).

- Στην περίπτωση που ισχύει C ≠ C' τότε το σφάλμα γίνεται αντιληπτό και είναι ανιχνεύσιμο (Detected error).



Εικόνα 8 S-box Fault Detection αντίμετρο

8.3.1 Υποπερίπτωση 1 – Design 3 Solution 1 (Sol1)

Όπως και στις προηγούμενες σχεδιάσεις, έτσι και εδώ στο Design 3 η συγκεκριμένη υποπερίπτωση Solution 1 χαρακτηρίζεται από τις ίδιες παραμετροποιήσεις που εφαρμόστηκαν και στα προηγούμενα Solution 1. Επιτρέπεται η εφαρμογή της προκαθορισμένη λειτουργίας (default) του Vivado HLS. Με αυτήν την προεπιλεγμένη λειτουργιά το εργαλείο Vivado HLS προχωράει όπου είναι αναγκαίο σε μικρές αναγκαίες βελτιστοποιήσεις. Υπενθυμίζεται ότι οι παραμετροποιήσεις inline ή unroll (αναλυτικότερη επεξήγηση έχει δοθεί στην προηγούμενη σχεδίαση Design 1 Solution 1) είναι βελτιστοποιήσεις οι οποίες δύναται να εφαρμοστούν. Σκοπός είναι επίσης η αύξηση της ταχύτητας, και η εξοικονόμηση κατανάλωσης ισχύος. Σε κάθε περίπτωση όμως, το υπό εξέταση Solution 1 αφορά στην σχεδίαση Cng. Ισχύει επομένως αυτό που αναλύθηκε νωρίτερα δηλαδή η επέκταση των παραμέτρων του κώδικα σε 16 bit. Άρα και το αποτέλεσμα της εξόδου που θα διαβαστεί στο ap_return όταν το ap_done = 1 θα είναι της μορφής πχ. 00007070.

8.3.2 Υποπερίπτωση 2 – Design 3 Solution 2 (Sol2)

Στη συγκεκριμένη υποπερίπτωση Solution 2 επιβάλλεται επίσης στο εργαλείο Vivado HLS η παραμετροποίηση loop unrolling. Με αυτό το “ξετύλιγμα” του βρόγχου (loop unrolling) [21] επιδιώκεται η αύξηση του μεγέθους των εντολών που εκτελούνται μέσα σε έναν βρόγχο, με συνέπεια τη μείωση του αριθμού των φορών που θα εκτελεστεί τελικά ο βρόγχος. Επιπρόσθετα αναφέρεται ότι για να επιτευχθεί πλήρη παραλληλοποίηση, δηλαδή να γίνει “ξετύλιγμα” όλων των βρόγχων, χρειάζεται τα δεδομένα να είναι ανεξάρτητα μεταξύ τους.

8.3.3 Υποπερίπτωση 3 – Design 3 Solution 3 (Sol3)

Στη συγκεκριμένη υποπερίπτωση Solution 5 της σχεδίασης Design 5 επιβάλλεται περιορισμός των βελτιστοποιήσεων που επιτρέπεται να γίνουν από το εργαλείο Vivado HLS. Η ντιρεκτίβα που εφαρμόζεται είναι no-inline (σχήμα 2). Επομένως στη σχεδίαση υψηλού επιπέδου δεν επιτρέπονται από κοινού βελτιστοποιήσεις ανάμεσα σε διαφορετικές λειτουργίες (functions). Η κάθε λειτουργία (function) θα υλοποιηθεί ως έχει, εκτός αν στο εσωτερικό της και μόνο είναι αναγκαίο να εφαρμοστεί κάποια βελτιστοποίηση. Απαγορεύεται δηλαδή να συμβεί διασταύρωση των ορίων (cross boundary) των λειτουργιών (functions).

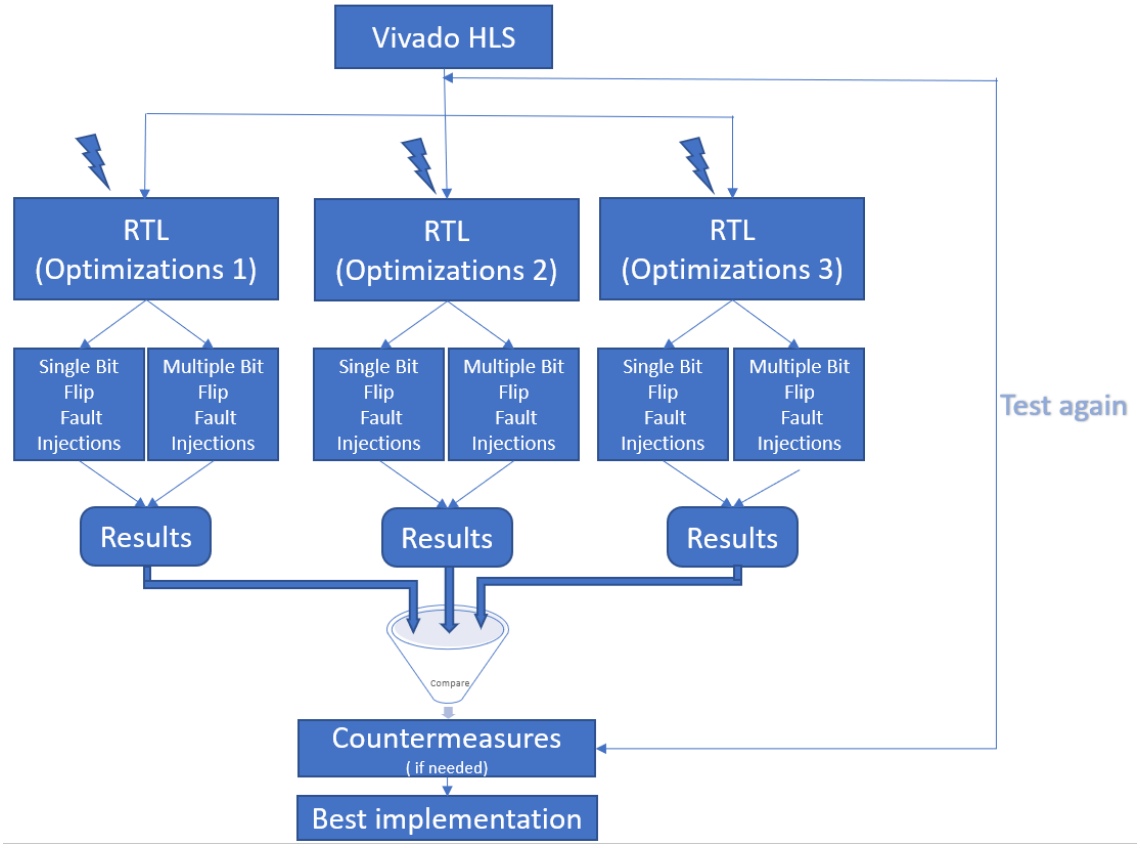
Παρακάτω δίνεται πίνακας (Εικόνα 9) ο οποίος παρουσιάζει το πλήθος των Flip Flop (ανά σχεδίαση), στο οποίο πραγματοποιήθηκε εισαγωγή σφαλμάτων.

FF	Solution 1	Solution 2	Solution 5
Design 1 Simple Canright	293	18	109
Design 2 Masked Canright HLS	583	169	373
Design 3 Cng1	212	50	207

Εικόνα 9 Flip Flops ανά υλοποίηση (Solution)

9. Σύστημα εισαγωγής σφαλμάτων με προσομοίωση

Στην παρούσα εργασία όπως ήδη έχει προαναφερθεί χρησιμοποιήθηκε το εργαλείο Vivado HLS 2020.1 προκειμένου να γίνει εξομοίωση του εκάστοτε σχεδιασμού (design) - κυκλωμάτων τα οποία δόθηκαν έτοιμα σύμφωνα με το προηγούμενο Κεφάλαιο 8 και εν συνεχεία πραγματοποιήθηκε η εισαγωγή σφαλμάτων μέσω διαφορετικών μοντέλων σφαλμάτων (Exhaustive Single Bit Flip, Multiple Bit Flip) (Εικόνα 10).



Εικόνα 10 Εισαγωγή SBF και MBF σφαλμάτων σε RTL

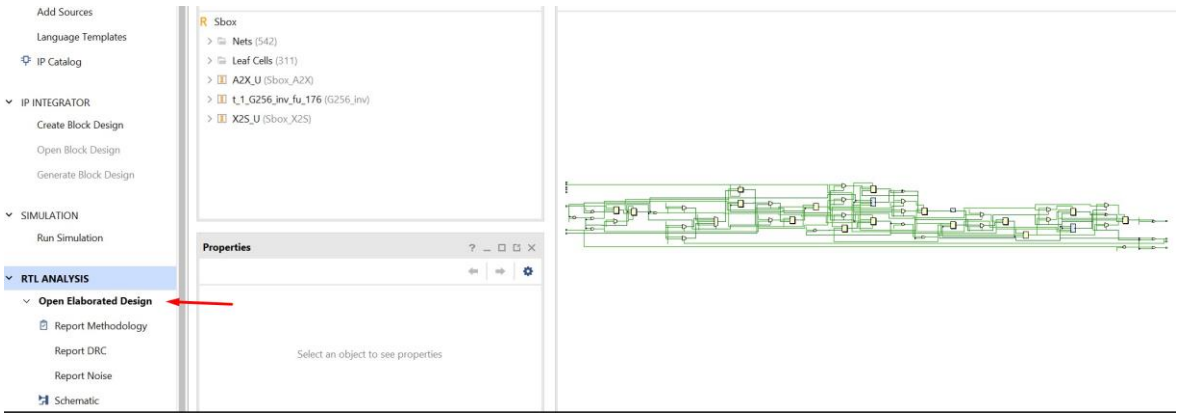
9.1 Δημιουργία RTL

Για τη δημιουργία του κάθε RTL project πραγματοποιήθηκαν οι απαιτούμενες ρυθμίσεις (Εικόνα 11) όπως η επιλογή του τύπου FPGA (έγινε η ίδια επιλογή σε όλα τα project), της HLL, η εισαγωγή των αρχείων κώδικα κάθε σχεδίασης, η επιλογή του φακέλου αποθήκευσης κ.α.

Εικόνα 11Δημιουργία RTL project

Η γλώσσα που χρησιμοποιήθηκε για τη δημιουργία του script εισαγωγής των σφαλμάτων ήταν η TCL η οποία είναι μια αποδοτική γλώσσα scripting ανοιχτού κώδικα και χρησιμοποιείται εκτενώς λόγω της αυξημένης δυνατότητας επαναχρησιμοποίησης. Επίσης απαιτεί μειωμένο χρόνο ανάπτυξης, έχει απλό user interface και τρέχει οπουδήποτε (Windows, Mac OS X και σχεδόν σε κάθε πλατφόρμα Unix) [22].

Μετά την δημιουργία του project από το μενού του Vivado HLS έγινε άνοιγμα του elaborated design (Εικόνα 12) προκειμένου να εξαχθεί κάθε φορά η λίστα όλων των flip-flops που εμπεριέχονται στο συγκεκριμένο σχεδιασμό.



Εικόνα 12 Vivado HLS - Elaborated Design

9.1 Υπολογισμός δείγματος SBF

Μετά την εξαγωγή των flip-flops παρατηρήθηκε ότι το Vivado HLS προσέθετε μία επέκταση "_reg" στην κατάληξη κάθε flip-flop η οποία και έπρεπε να αφαιρεθεί προκειμένου το εργαλείο να μπορεί να αναγνωρίσει σωστά τα flip-flops κατά την εισαγωγή των σφαλμάτων. Για να ολοκληρωθεί η διαδικασία δημιουργίας των δειγμάτων μελετήθηκε κάθε σχεδίαση (solution) ξεχωριστά έτσι ώστε να βρεθούν σημαντικές πληροφορίες γύρω από τη λειτουργικότητα κάθε κυκλώματος. Για παράδειγμα για να δημιουργηθεί σωστά η λίστα των προς εισαγωγή σφαλμάτων χρειάζεται να είναι γνωστή η παράμετρος clock_cycle, δηλαδή οι παλμοί του ρολογιού, διάστημα μέσα στο οποίο γίνονται οι υπολογισμοί του Sbox. Η τιμή του ανευρίσκεται μετά από αρχικοποίηση των εσωτερικών σημάτων του κυκλώματος και μέτρησης των

απαιτούμενων κύκλων ρολογιού που χρειάζονται προκειμένου το Sbox να ολοκληρώσει τους υπολογισμούς του και να φτάσει η έξοδος να λάβει τιμή δηλαδή να γίνει το ap_done=1. Η εν λόγω μεταβλητή μας δείχνει πότε είναι έτοιμη η τιμή στην έξοδο του Sbox να διαβαστεί. Αν για τον οποιοδήποτε λόγο η τιμή του ap_done ≠ 1 τότε σημαίνει ότι το κύκλωμα δεν πραγματοποίησε σωστά τους υπολογισμούς και δεν έχει λάβει σωστή τιμή η έξοδος του Sbox. Αυτό μπορεί να προκληθεί από την εισαγωγή ενός σφάλματος το οποίο θα οδηγήσει σε μία μη αποδεκτή λειτουργία.

Μετά την εύρεση της τιμής της παραμέτρου clock_cycle, με τη βοήθεια ενός script calc_single_sample.m που δημιουργήθηκε στο Matlab, καταρτίζεται το δείγμα του Exhaustive Single Bit Flip (SBF) το οποίο αποθηκεύεται σε ένα αρχείο κειμένου test_sample.txt.

Για το μοντέλο σφαλμάτων Exhaustive Single Bit Flip το δείγμα αποτελείται από όλα τα flip-flops για όλα τα clock_cycle. Για παράδειγμα στο project Simple Canright το solution 1 έχει 293 flip-flops και η τιμή του latency είναι 34 (34 clock cycles) (σχήμα 3). Επομένως κατά την προσομοίωση θα εισαχθούν 293*34 = 9962 σφάλματα (Fault Injections).

9.2 Υπολογισμός δείγματος Multiple Bit Flip

Για τη δημιουργία των λιστών των προς εισαγωγή σφαλμάτων με τη μέθοδο Multiple Bit Flip χρησιμοποιήθηκε η στατιστική μέθοδος [23]. Το αριθμητικό δείγμα υπολογίζεται με τη βοήθεια ενός script στο Matlab το οποίο υπολογίζει πόσα σφάλματα αριθμητικά (κατ΄ ελάχιστον) θα πρέπει να εισαχθούν σύμφωνα με τη στατιστική μέθοδο που αναλύθηκε στο ΚΕΦ.5 [23]. Καθώς επιθυμείται η επίτευξη μικρού ποσοστού περιθωρίου λάθους και μεγάλου ποσοστού αξιοπιστίας των αποτελεσμάτων ο καθορισμός των μεταβλητών έγινε για όλα τα solution ως εξής:

- Περιθώριο λάθους (margin of error) ίσο με 1% ή αλλιώς e=0,01

- t = 2,5758 τιμή η οποία αντιστοιχεί σε ποσοστό εμπιστοσύνης 99% , δείχνει την πιθανότητα η ακριβής τιμή να βρίσκεται στην πραγματικότητα εντός του διαστήματος σφάλματος.

- p = 0,5 είναι το τυπικό σφάλμα.

- όπου N ο συνολικός αριθμός όλων των στοιχείων μνήμης, δηλαδή των flip-flops.

- Στην παρούσα εργασία η πολλαπλότητα σφάλματος M ορίζεται κάθε φορά από 0 έως 5 ανάλογα με την πολλαπλότητα των δειγμάτων. Για παράδειγμα για M=2 γίνεται τυχαία επιλογή δύο flip-flop στα οποία θα εισαχθεί σφάλμα στο ίδιο clock_cycle. Για M=3 γίνεται τυχαία επιλογή τριών flip-flop στα οποία θα εισαχθεί σφάλμα στο ίδιο clock_cycle κ.ο.κ.

Το script επιστρέφει έναν αριθμό ανάλογα με τις ορισθείσες παραμέτρους ο οποίος δείχνει το ελάχιστο αριθμητικό δείγμα που θα πρέπει να χρησιμοποιηθεί για να υπάρχει ένα ασφαλές αποτέλεσμα με 1% περιθωρίου λάθους.

Κατόπιν, δημιουργήθηκε ένα άλλο script στο Matlab το οποίο για κάθε solution επιλέγει τυχαία μέσα από τη λίστα του συνόλου των flip-flops που έχει ήδη εξαχθεί, τον απαιτούμενο αριθμό δειγμάτων για τις διαφορετικές πολλαπλότητες M=2, M=3, M=4 και M=5. Τονίζεται ότι για την ορθή επιλογή των δειγμάτων θα πρέπει αυτά να είναι μοναδικά, δηλαδή να μην υπάρχουν δείγματα εις διπλούν ανάμεσα στον πληθυσμό των δειγμάτων που έχουν επιλεγεί, ούτε σε έκαστο δείγμα πολλαπλότητας 2,3,4 ή 5 να εμπεριέχεται ένα flip-flop πάνω από μία φορά. Ενδεικτικά παρατίθεται ένα δείγμα πολλαπλότητας M=3 (Εικόνα 13). Στο τέλος του κάθε δείγματος επιλέγεται επίσης τυχαία ο κύκλος ρολογιού (μέσα από το εύρος της τιμής του clock_cycle) στο οποίο θα πραγματοποιηθεί η εισαγωγή του σφάλματος.

```
grp_G256_inv_fu_64/a_reg_1093[10] t_reg_114[2] t_reg_114[13] 5
grp_G256_inv_fu_64/b_reg_1098[8] grp_G256_inv_fu_64/a_reg_1093[11] t_reg_114[11] 4
grp_G256_inv_fu_64/b_reg_1098[11] grp_G256_inv_fu_64/a_reg_1093[1] grp_G256_inv_fu_64/a_reg_1093[5] 4
ap_CS_fsm[2] ap_CS_fsm[0] t_reg_114[3] 7
grp_G256_inv_fu_64/tmp_37_reg_1113[0] grp_G256_inv_fu_64/a_reg_1093[8] t_reg_114[13] 7
grp_G256_inv_fu_64/or_ln124_2_reg_1108[2] grp_G256_inv_fu_64/a_reg_1093[4] t_reg_114[8] 6
t_reg_114[3] t_reg_114[11] ap_CS_fsm[1] 6
t_reg_114[9] t_reg_114[10] t_reg_114[13] 5
grp_G256_inv_fu_64/ap_CS_fsm[1] grp_G256_inv_fu_64/a_reg_1093[10] grp_G256_inv_fu_64/a_reg_1093[11] 6
t_reg_114[0] t_reg_114[10] t_reg_114[13] 4
grp_G256_inv_fu_64/a_reg_1093[0] t_reg_114[10] ap_CS_fsm[1] 4
```

Design d5_sol2 Δείγμα για MM=3

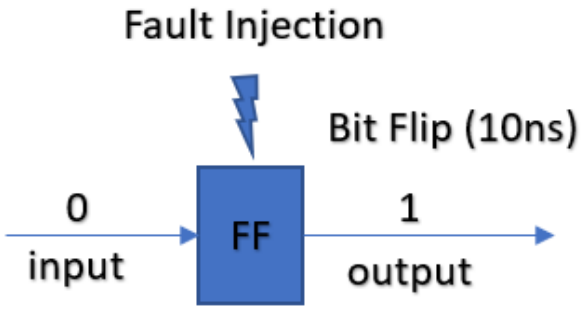
Εικόνα 13 Sample for FI (Multiplicity 3)

Μετά την ολοκλήρωση της δημιουργίας των αρχείων κειμένου των δειγμάτων test_sample.txt για Exhaustive Single Bit Flip και Multiple Bit Flip, με τη χρήση του Vivado HLS πραγματοποιήθηκε η προσομοίωση της εισαγωγής των σφαλμάτων στο κύκλωμα του καθενός solution ανά design ξεχωριστά, ξεκινώντας από την εισαγωγή σφαλμάτων σύμφωνα με το μοντέλο Exhaustive Single Bit Flip και εν συνεχεία σύμφωνα με το Multiple Bit Flip.

9.3 Προσομοίωση σφαλμάτων

Η εισαγωγή σφαλμάτων στο κύκλωμα του κάθε solution ανά σχεδίαση έγινε μέσα από ένα script γραμμένο σε γλώσσα TCL με το οποίο αρχικά γίνονται οι απαραίτητες αρχικοποιήσεις των εσωτερικών σημάτων, καθορισμός της υπό εξέτασης μεταβλητής ap_return στην οποία επιστρέφεται η έξοδος του Sbox και κατόπιν υπολογίζεται η μεταβλητή gold_value η τιμή της οποίας αντιστοιχεί στην ορθή έξοδο του Sbox, χωρίς να έχει εισαχθεί οποιοδήποτε σφάλμα. Ακολούθως γίνεται ανάγνωση μέσα από το αρχείο κειμένου test_sample.txt του δείγματος, καταχώρηση των τιμών στις αντίστοιχες μεταβλητές FlipFLop και ClockCycle, και στη συνέχεια εκτελέστηκε η εισαγωγή των σφαλμάτων.

Η λογική της εισαγωγής των σφαλμάτων κατά την εκτέλεση του script είναι να εξετάζει κάθε φορά την τιμή που είναι καταχωρημένη στο flip-flop στο οποίο επιθυμείται να εισαχθεί το σφάλμα κατά το συγκεκριμένο clock_cycle. Αν η τιμή του είναι 1 το αναγκάζει σε αντιστροφή του, δηλαδή να γίνει 0 για 10 ns και αντίστροφα αν είναι 0 να γίνει 1 για 10 ns. Αυτό πραγματοποιείται τόσες φορές όσα και τα δείγματα που εμπεριέχονται στο test_sample.txt (Εικόνα 14). Στη συνέχεια στο αρχείο κειμένου out.txt εγγράφεται το δείγμα και μετά το αποτέλεσμα της εξόδου του Sbox στη μεταβλητή ap_return, η οποία είναι ασφαλής να αναγνωσθεί όταν το ap_done = 1.



Εικόνα 14 Εισαγωγή σφάλματος σε Flip Flop

9.3.1 Κατηγοριοποίηση σφαλμάτων

Μετά την ολοκλήρωση της εισαγωγής όλων των σφαλμάτων και της λήψης των αποτελεσμάτων σειρά είχε η αξιολόγησή τους. Η κατηγοριοποίηση των αποτελεσμάτων έγινε συγκρίνοντας το αποτέλεσμα του κάθε Fault Injection με την τιμή gold_value σύμφωνα με τα κατωτέρω:

Silent error: Η τιμή εξόδου (ap_return) είναι ίδια με την τιμή gold_value. Αυτό σημαίνει ότι το σφάλμα που εισήλθε δεν επέφερε μεταβολή στην τιμή της εξόδου του Sbox. Δηλαδή το σφάλμα παρότι ενδεχομένως να επηρέασε τις τιμές κάποιου ή κάποιων flip-flop εν τέλη αποσβέστηκε και δεν έγινε αντιληπτό στην έξοδο του κυκλώματος.

Critical error: Η τιμή εξόδου (ap_return) είναι διαφορετική από την τιμή gold_value. Σε αυτήν τη κατηγορία κατατάσσονται τα σφάλματα τα οποία εισήχθησαν σε ένα ή περισσότερα flip-flop στο κύκλωμα και αυτό επέφερε αλλαγές έως και την έξοδό του. Επομένως το σφάλμα αυτό είναι κρίσιμο διότι επηρεάζει τη λειτουργία του κυκλώματος.

Hang error: Η τιμή εξόδου (ap_return) είναι απροσδιόριστη. Μπορεί να πάρει την τιμή U ή X ή κάποια άλλη ασαφή τιμή. Αυτό συνεπάγεται ότι το κύκλωμα έχει οδηγηθεί σε μία μη προβλεπόμενη λειτουργία η οποία μπορεί να επιφέρει άγνωστες συνέπειες στη λειτουργικότητα του κυκλώματος. Σε αυτήν την κατηγορία εντάσσονται όλα τα αποτελέσματα τα οποία λαμβάνονται χωρίς να έχει καταστεί εφικτό το ap_done να λάβει την τιμή 1, δηλαδή για όλα σφάλματα τα αποτελέσματα των οποίων συλλέχθηκαν ενώ το ap_done ≠1. Κατά τις δοκιμές της εκτέλεσης των προσομοιώσεων παρατηρήθηκε ότι στην περίπτωση που δεν αρχικοποιηθούν όλα τα εσωτερικά σήματα του κυκλώματος και αυτά έχουν απροσδιόριστη τιμή U για παράδειγμα, τότε η έξοδος του κυκλώματος είναι πιθανό να οδηγηθεί σε Hang error.

Detected errors: Καθώς στη σχεδίαση Design 5 Cng 1 όπως αναφέρθηκε και στο Κεφάλαιο 8 γίνεται μία εισαγωγή πλεονασμού (redundancy) και όλη η λειτουργικότητα και όλες οι μεταβλητές του κώδικα χρησιμοποιούν πληροφορία μήκους 16 bit, προστίθεται η κατηγορία των Detected errors. Η σύγκριση των αποτελεσμάτων των S-box γίνεται ανάμεσα στο C (8 bit) και C’ (8 bit) των αποτελεσμάτων που λαμβάνονται. Αν οι δύο έξοδοι C και C’ είναι ίδιες μεταξύ τους (C = C’) και η τιμή τους συμφωνεί με την αναμενόμενη ορθή τιμή gold_value (16 bit) τότε δεν υπάρχει σφάλμα ή εισήχθη αλλά δεν επηρέασε τα S-box (Silent error). Στην περίπτωση όμως που ισχύει C ≠ C’ τότε το σφάλμα γίνεται αντιληπτό και είναι ανιχνεύσιμο (Detected error).

9.3.2 Καταγραφή αποτελεσμάτων

Για την εξαγωγή των στατιστικών των αποτελεσμάτων δημιουργήθηκε ένα Matlab script το οποίο διαβάζοντας το αρχείο κειμένου των αποτελεσμάτων out.txt δημιουργεί ένα αρχείο Log.txt το οποίο περιέχει πληροφορίες σχετικά με την τιμή gold_value, την μεταβλητή που αποθηκεύεται η έξοδος του κυκλώματος, ο αριθμός των flip-flop που έχει το solution, πόσα σφάλματα εισήχθησαν (flip-flop*clock_cycle όταν πρόκειται για Exhaustive Single Bit Flip ή τον αριθμό των δειγμάτων όταν πρόκειται για Multiple Bit Flip καθώς και την ποσοστιαία κατηγοριοποίηση των σφαλμάτων (Εικόνα 15).

```
Vivado - output analyser
Kalliopi


Gold Variable : /sbox/ap_return
Gold Value    : 00000070
Prefix        : /sbox/

Reading Data...
Header Confirmed (/sbox/ap_return).


Found 293 Flip Flops
Found 9962 Fault Injections
Exporting to CSV
DONE


92.591849 % Total Silent Errors
6.585023 % Total Critical Errors
0.823128 % Total Hang Errors
```

Εικόνα 15 Log αρχείο αποτελεσμάτων FI

Επίσης δημιουργεί ένα αρχείο τύπου excel FaultInjectionLog_.csv (σχήμα 7) στο οποίο γίνεται καταγραφή των αποτελεσμάτων δίνοντας πληροφορίες σχετικά με τα Critical errors και τα Hang errors. Συγκεκριμένα παρέχονται πληροφορίες για το σε ποιο flip-flop εντοπίζεται το κάθε σφάλμα, ποια εσφαλμένη τιμή έχει λάβει η έξοδος του κυκλώματος και σε ποια θέση έχει πραγματοποιηθεί αντιστροφή bit (ήτοι bit flip).

Επιπρόσθετα δημιουργείται ακόμη ένα αρχείου τύπου excel (σχήμα 8) στο οποίο καταγράφονται αναλυτικά ανά flip flop πόσα σφάλματα εντοπίστηκαν ανά κατηγορία. Οι πληροφορίες αφορούν στα σφάλματα ανά κατηγορία (Silent, Critical και Hang errors). Από το σύνολο των σφαλμάτων που εισήχθησαν γίνεται διάκριση για κάθε flip-flop σχετικά με το πόσα σφάλματα πέρασαν ως Silent και πόσα ως Critical ή Hang παρέχοντας έτσι πληροφορίες για το ποιο flip flop φαίνεται να είναι πιο ευάλωτο σε Critical ή Hang errors. Στην τελευταία στήλη γίνεται άθροισμα των Critical και Hang errors.

Κατόπιν συγκέντρωσης όλων των ανωτέρω αποτελεσμάτων εισαγωγής σφαλμάτων (Fault Injection), εν λόγω αποτελέσματα καταγράφηκαν σε πίνακες και σχεδιάστηκαν σε γραφήματα με τη χρήση του προγράμματος excel τα οποία παρατίθενται στο Κεφάλαιο 10.

Μετά την ολοκλήρωση των ανωτέρω προσομοιώσεων έγινε μελέτη για το αν είναι εφικτή η αντίστοιχη εισαγωγή σφαλμάτων με το μοντέλο των λογικών κώνων (Logical Cones). Δημιουργήθηκε ένα script σε γλώσσα Tcl το οποίο εξάγει από το Vivado τους λογικούς κώνους του σχεδιασμού Design 3 solution 1. Αφού αφαιρέθηκε το σήμα ap_clk εισήχθησαν οι λογικοί κώνοι στο Matlab όπου ελέγχθηκαν για διπλοεγγραφές που υπήρχαν (ορισμένοι κώνοι αποτελούσαν υποσύνολα άλλων), αφαιρέθηκαν τα υποσύνολα και στη συνέχεια εξήχθησαν σε αρχείο κειμένου τα σετ των flip-flop που απέμειναν. Για την εισαγωγή σφαλμάτων για κάθε σετ flip flop που υπέδειξε η μεθοδολογία, δημιουργήθηκαν τα αναγκαία αρχεία δειγμάτων τόσο για Exhaustive Single Bit Flip όσο και για Multiple bit flip σύμφωνα με τη στατιστική μέθοδο που παρουσιάστηκε και στα προηγούμενα μοντέλα σφαλμάτων. Ο όγκος των προς εισαγωγή σφαλμάτων για όλα τα Solution όλων των Design που προέκυψε ήταν αρκετά μεγάλος. Προκειμένου να υπάρξουν αποτελέσματα τα οποία θα είναι συγκρίσιμα με τα αντίστοιχα των προηγούμενων μοντέλων σφαλμάτων, θα πρέπει να πληρούνται οι ίδιες απαιτήσεις (π.χ. margin of errors 1%), για το λόγο αυτό δεν δύναται να ολοκληρωθούν όλες οι προσομοιώσεις εισαγωγής σφαλμάτων (για το μοντέλο των λογικών κώνων) για όλα τα Solution όλων των

Design για όλα τα Multiplicities (2,3,4 και 5) στο διάστημα της συγγραφής της παρούσας εργασίας. Δύναται όμως να πραγματοποιηθούν σε ενδεχόμενη μελλοντική εργασία.

10. Αποτελέσματα αξιολόγησης των κυκλωμάτων μέσω εισαγωγής σφαλμάτων (Fault Injections)

Στη συνέχεια παρατίθενται ανά σχεδίαση τα αποτελέσματα των Fault Injections που πραγματοποιήθηκαν σύμφωνα με τα υπό αξιολόγηση κυκλώματα τα οποία περιγράφηκαν στο Κεφάλαιο 8. Η εισαγωγή σφαλμάτων γίνεται μόνο στα στοιχεία μνήμης των υπό εξέταση κυκλωμάτων και όχι στα εσωτερικά σήματα του κυκλώματος. Εισάγονται τα σφάλματα με τρία διαφορετικά μοντέλα σφαλμάτων ήτοι με Exhaustive Single Bit Flip, με Multiple Bit Flip και με το μοντέλο των λογικών κώνων (Logical Cones).

Όσον αφορά το χρόνο που χρειάστηκε για την εισαγωγή των σφαλμάτων αυτός ήταν διαφορετικός ανά σχεδίαση. Οι παράμετροι που επηρέασαν για το πόσος χρόνος θα χρειαστεί κάθε φορά ήταν σαφώς το πλήθος των Flip-Flops που διέθετε το κάθε κύκλωμα, όσο περισσότερα Flip-Flops, τόσο περισσότερα τα εν δυνάμει σφάλματα και τόσος περισσότερος χρόνος για την εισαγωγή τους. Επίσης η διάρκεια των κύκλων ρολογιού κάθε σχεδίασης έπαιξαν καθοριστικό ρόλο. Οι κύκλοι του ρολογιού δεν επηρέασαν μόνο τον χρόνο αλλά και τα αποτελέσματα καθ' αυτά αφού σε ορισμένες σχεδιάσεις οι οποίες είχαν μικρό κύκλο ρολογιού για παράδειγμα 1 clock cycle παρατηρήθηκε ότι επηρεάστηκε καταλυτικά η λειτουργικότητα του κυκλώματος.

Επίσης χρησιμοποιώντας το εργαλείο προσομοίωσης Vivado HLS σε διαφορετικούς υπολογιστές με διαφορετικούς υπολογιστικούς πόρους διαπιστώθηκε ότι ο χρόνος προσομοίωσης των σφαλμάτων διαφέρει αρκετά ανάλογα με τα τεχνικά χαρακτηριστικά του υπολογιστή προσομοίωσης αλλά και τον επιλεγμένο τρόπο (mode) εκτέλεσης των πειραμάτων.

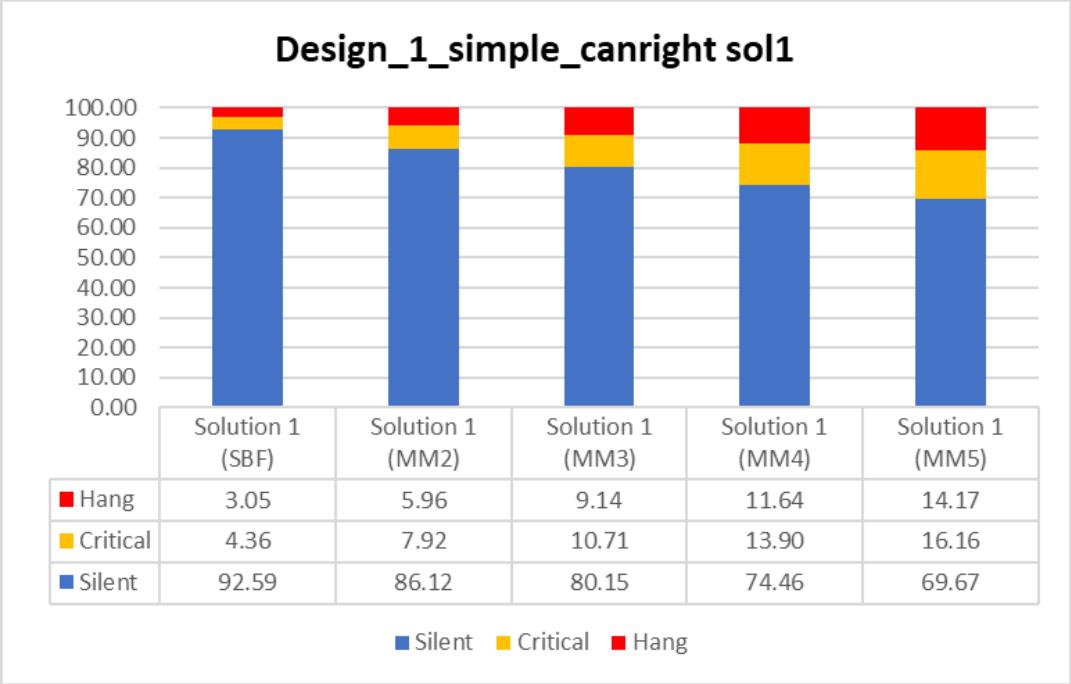
Για την παρούσα εργασία σύμφωνα με τα πειράματα που εκτελέστηκαν ο ρυθμός εισαγωγής σφαλμάτων με τη χρήση του προσομοιωτή είναι της τάξης των 25 σφαλμάτων ανά δευτερόλεπτο.

Στη συνέχεια παρατίθενται τα αποτελέσματα που προέκυψαν, ανά σχεδιασμό Design.

10.1 DESIGN 1 – SIMPLE CANRIGHT

10.1.1 Solution 1 (default Vivado HLS settings)

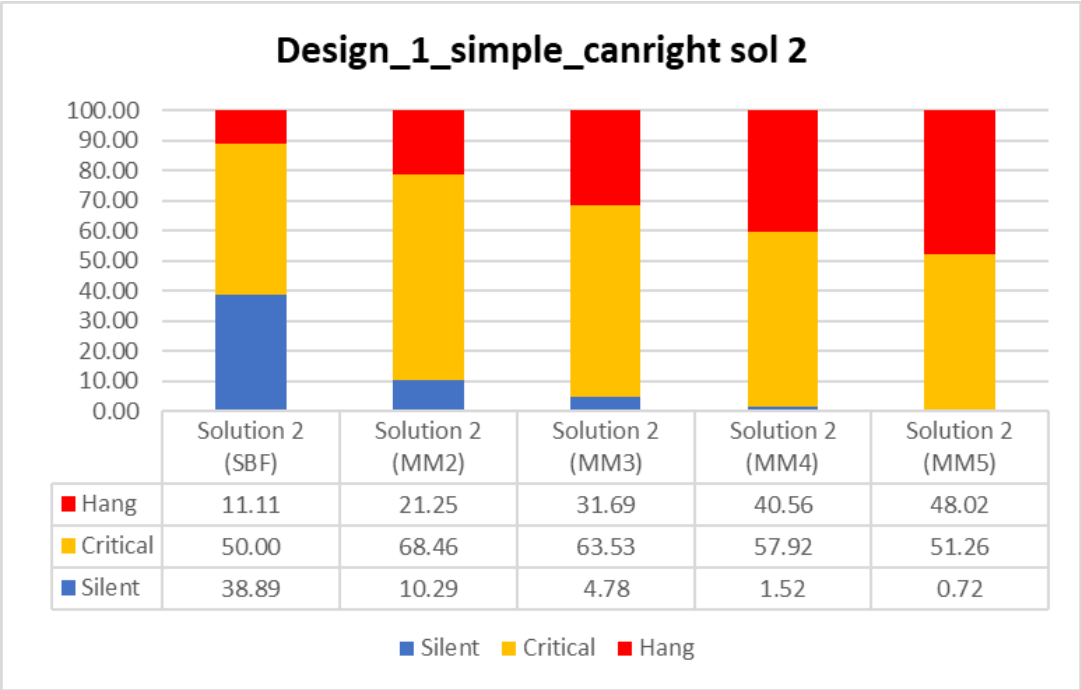
Solution 1 (default Vivado HLS settings): Στην Εικόνα 16 παρουσιάζονται τα αποτελέσματα του Solution 1 στα οποία φαίνεται ότι το μεγαλύτερο ποσοστό σφαλμάτων κατηγοριοποιείται ως Silent. Αυτό σημαίνει ότι φαίνεται ότι τα περισσότερα σφάλματα που εισήχθησαν δεν διαδόθηκαν έως την έξοδο του Sbox, παρότι ενδεχομένως να επηρέασαν προσωρινά τις τιμές κάποιου ή κάποιων flip-flop εν τέλη αποσβέστηκαν και δεν έγιναν αντιληπτά στην έξοδο του κυκλώματος. Επίσης αυτό που παρατηρείται είναι ότι όσο μεγαλώνει το multiplicity (η εισαγωγή πολλαπλών σφαλμάτων) μειώνεται το ποσοστό των Silent σφαλμάτων και αυξάνονται τα ποσοστά των Critical και Hang σφαλμάτων. Μάλιστα τα Critical errors, αυτά δηλαδή που φτάνουν να επηρεάσουν την τιμή της εξόδου του Sbox φαίνεται ότι υπερτερούν αριθμητικά σε σχέση με τα Hang. Μεταξύ των Multiple Bit Flip η μεγαλύτερη ποσοστιαία διαφορά των Critical errors παρατηρείται για M=4, ενώ των errors Hang για M=3.



Εικόνα 16 Design 1 Solution 1

10.1.2 Solution 2 (Loop unrolling)

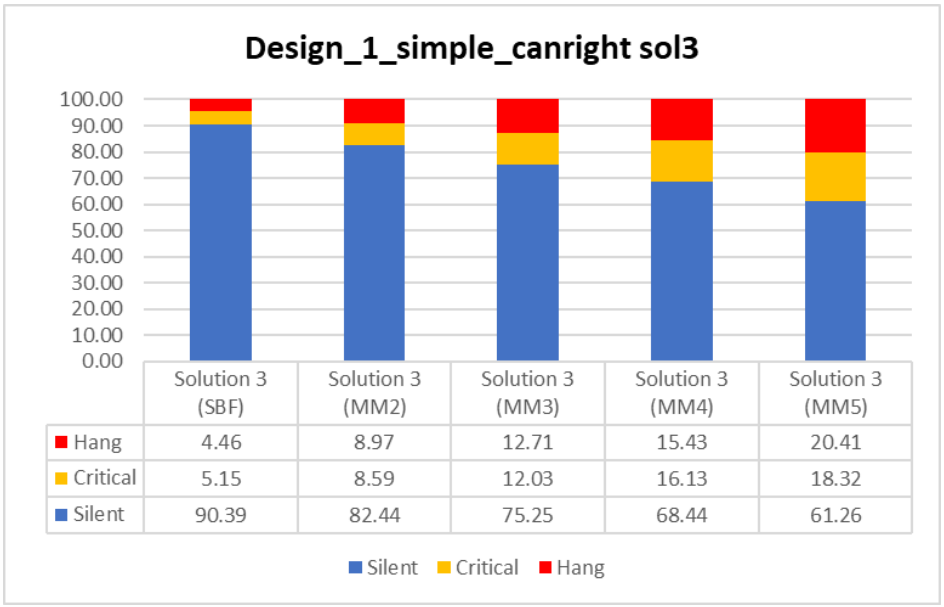
Solution 2 (Loop unrolling): Στην Εικόνα 17 απεικονίζονται τα αποτελέσματα του Solution 2 στα οποία φαίνεται ότι το μεγαλύτερο ποσοστό σφαλμάτων κατηγοριοποιείται ως Critical. Αυτό σημαίνει ότι φαίνεται ότι τα περισσότερα σφάλματα που εισήχθησαν διαδόθηκαν έως την έξοδο του Sbox. Παρατηρείται ότι όσο αυξάνεται η πολλαπλότητα του Multiple Bit Flip τα Silent errors τείνουν να μηδενιστούν πράγμα που σημαίνει ότι όσο αυξάνεται το multiplicity, οι πιθανότητες το σφάλμα να επηρεάσει το κύκλωμα είτε οδηγώντας το σε Critical είτε σε Hang errors είναι σχεδόν βέβαιη. Αν επομένως εισαχθεί το σφάλμα σε flip-flop το οποίο περιέχει data θα οδηγήσει σε Critical error, ενώ αν εισαχθεί σε flip-flop ελέγχου της μηχανής πεπερασμένων καταστάσεων θα οδηγήσει σε Hang. Επίσης παρατηρείται ότι τα Critical errors κυμαίνονται από 50% έως 68.46% είτε με τη μέθοδο εισαγωγής σφαλμάτων Exhaustive Single Bit Flip είτε με την μέθοδο Multiple Bit Flip, ενώ τα Hang φαίνεται να αυξάνονται αλματωδώς όσο αυξάνεται το Multiplicity.



Εικόνα 17 Design 1 Solution 2

10.1.3 Solution 3 (No inline)

Solution 3 (No inline): Στην Εικόνα 18 τα αποτελέσματα του Solution 3 παρομοιάζουν με τα αποτελέσματα που ελήφθησαν με το Solution 1 επιστρέφοντας πολλά Silent σφάλματα, τα οποία όσο μεγαλώνει το Multiplicity μειώνονται. Κάτι επιπλέον που σημειώνεται είναι ότι ενώ για πολλαπλότητα M=2 τα Critical και τα Hang errors φαίνεται να είναι σχεδόν ίδια, όσο αυξάνεται το Multiplicity τα Hang errors υπερτερούν έναντι των Critical. Η μεταξύ τους διαφορά φαίνεται να ανοίγει στο M=5.

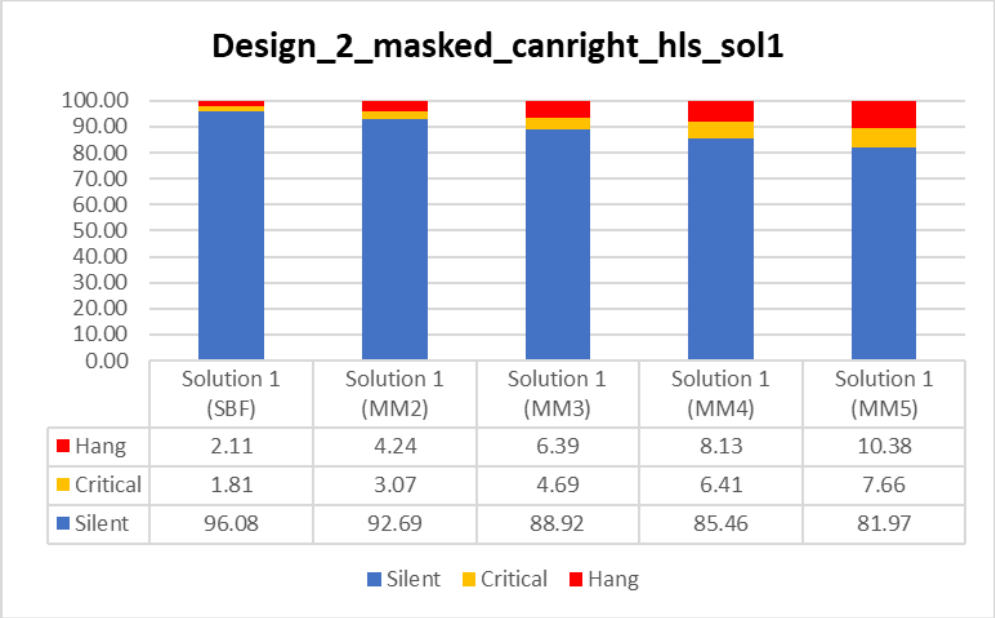


Εικόνα 18 Design 1 Solution 3

10.2 DESIGN 2 – MASKED CANRIGHT HLS

10.2.1 Solution 1 (default Vivado HLS settings)

Solution 1 (default Vivado HLS settings): Στην Εικόνα 19 παρουσιάζονται τα αποτελέσματα του Solution 1 και σημειώνεται ότι μετά την εισαγωγή των σφαλμάτων τα περισσότερα κατατάσσονται στα Silent. Αυξάνοντας την πολλαπλότητα των σφαλμάτων παρατηρείται ότι τα Silent εξακολουθούν να κατέχουν το μεγαλύτερο ποσοστό, ενώ τα Critical και Hang errors φαίνεται ότι μεταβάλλονται ανάλογα μεταξύ των διαφορετικών Multiplicities.

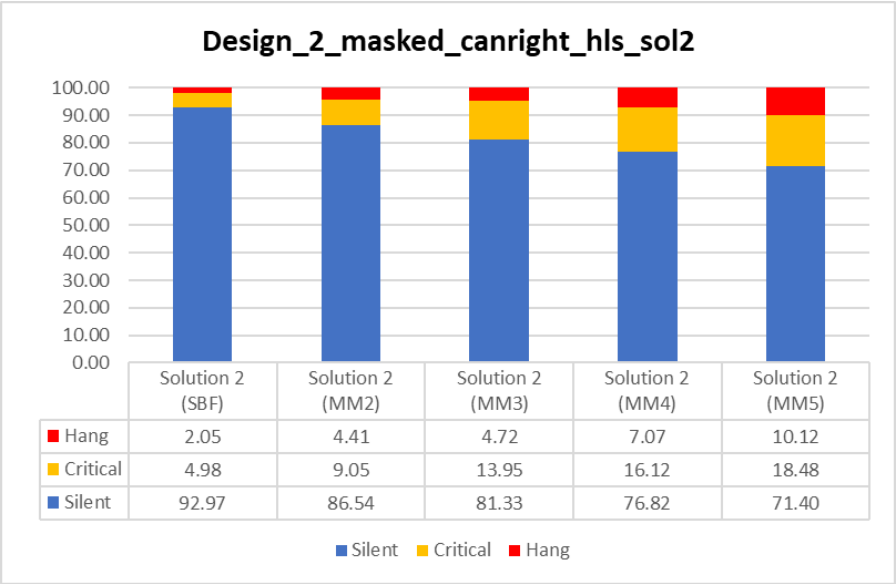


	Solution 1 (SBF)	Solution 1 (MM2)	Solution 1 (MM3)	Solution 1 (MM4)	Solution 1 (MM5)
Hang	2.11	4.24	6.39	8.13	10.38
Critical	1.81	3.07	4.69	6.41	7.66
Silent	96.08	92.69	88.92	85.46	81.97

Εικόνα 19 Design 2 Solution 1

10.2.2 Solution 2 (Loop unrolling)

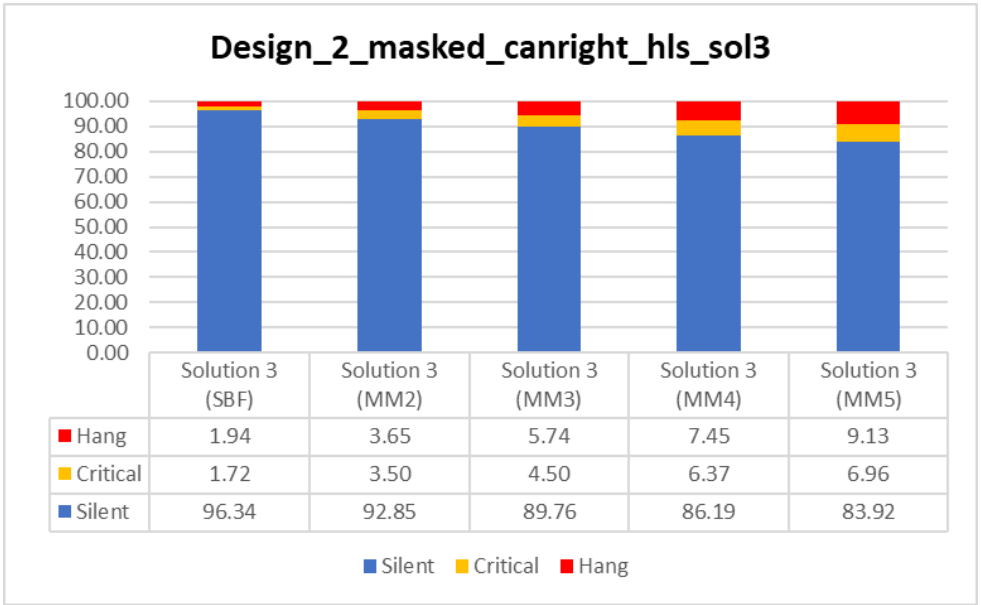
Solution 2 (Loop unrolling): Στην Εικόνα 20 διαφαίνεται μέσα από τα αποτελέσματα ότι τα περισσότερα σφάλματα είναι επίσης Silent ενώ τα Critical errors σε σχέση με τα υπόλοιπα δύο Solution το ίδιου Design. Λόγω του μικρότερου κύκλου ρολογιού της σχεδίασης, είναι πιθανόν το σφάλμα που θα εισαχθεί σε ένα flip-flop να χρησιμοποιηθεί αμέσως μετά. Μεταξύ των Critical και Hang errors και όσο αυξάνονται τα Multiplicities φαίνεται να υπερτερούν τα Critical.



Εικόνα 20 Design 2 Solution 2

10.2.3 Solution 3 (No inline)

Solution 3 (No inline): Στην Εικόνα 21 τα αποτελέσματα του Solution 3 παρομοιάζουν με τα αποτελέσματα που ελήφθησαν με το Solution 1 επιστρέφοντας πολλά Silent. Η υλοποίηση αυτή έχει ελάχιστες διαφορές με τα αποτελέσματα του Solution 1.

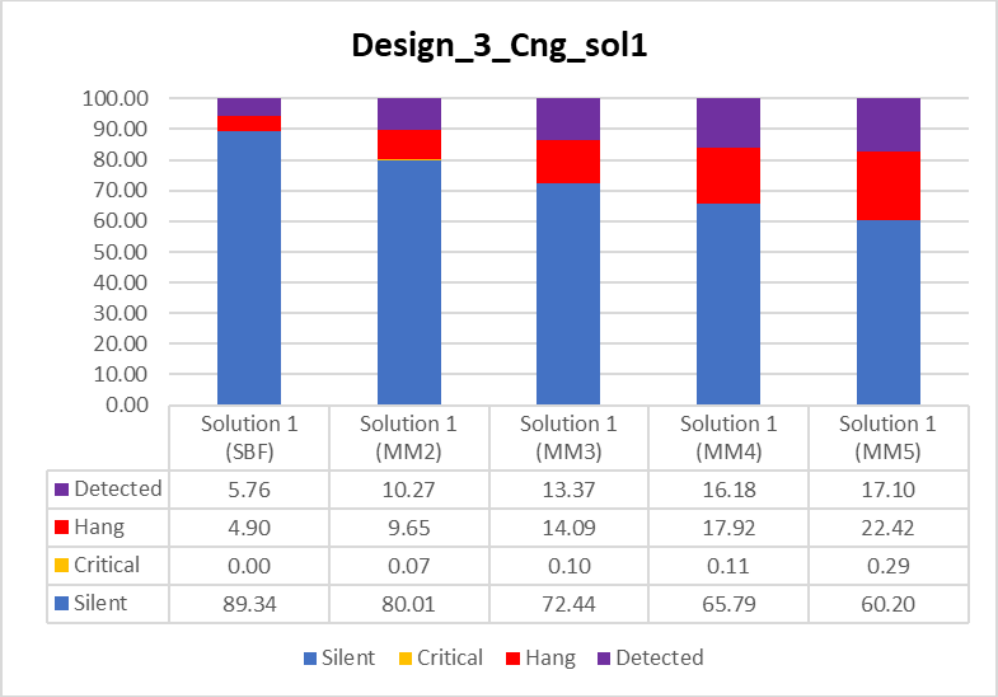


Εικόνα 21 Design 2 Solution 3

10.3 DESIGN 3 – CNG CANRIGHT HLS

10.3.1 Solution 1 (default Vivado HLS settings)

Solution 1 (default Vivado HLS settings): Στην Εικόνα 22 αποτυπώνονται τα αποτελέσματα του Solution 1 και παρατηρείται ότι τα Critical errors είναι σχεδόν μηδενικά. Αυτό σημαίνει ότι η παρούσα υλοποίηση με τη χρήση του redundancy δουλεύει σχεδόν άριστα αφήνοντας περιθώριο μόνο για Hang errors. Όσο μικρότερο είναι το Multiplicity αντίστοιχα χαμηλό είναι και το Hang error. Το redundancy αντίμετρο, επηρεάζειωτα αποτελέσματα, πράγμα αναμενόμενο.

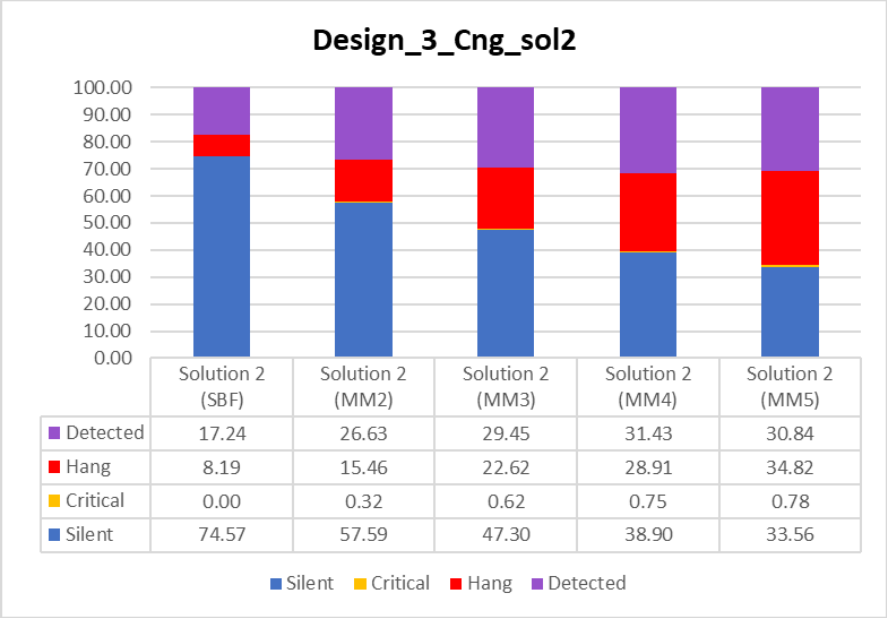


	Solution 1 (SBF)	Solution 1 (MM2)	Solution 1 (MM3)	Solution 1 (MM4)	Solution 1 (MM5)
Detected	5.76	10.27	13.37	16.18	17.10
Hang	4.90	9.65	14.09	17.92	22.42
Critical	0.00	0.07	0.10	0.11	0.29
Silent	89.34	80.01	72.44	65.79	60.20

Εικόνα 22 Design 3 Solution 1

10.3.2 Solution 2 (Loop unrolling)

Solution 2 (Loop unrolling): Στην Εικόνα 23 του εν λόγω σχεδιασμού παρατηρείται ότι αυξάνονται τα σφάλματα που δίνουν αποτέλεσμα Detected σε σχέση με τα άλλα δύο Solution. Αυτό συμβαίνει γιατί αντίστοιχα μειώνονται τα Silent λόγω του μικρότερου κύκλου ρολογιού και επομένως υπάρχει μεγαλύτερη πιθανότητα εισάγοντας ένα σφάλμα που εμπεριέχει μέσα του αποθηκευμένα data να χρησιμοποιηθεί σε επόμενο υπολογισμό της υλοποίησης και να γίνει αντιληπτό. Τα Critical errors εξακολουθούν να είναι σχεδόν μηδενικά.

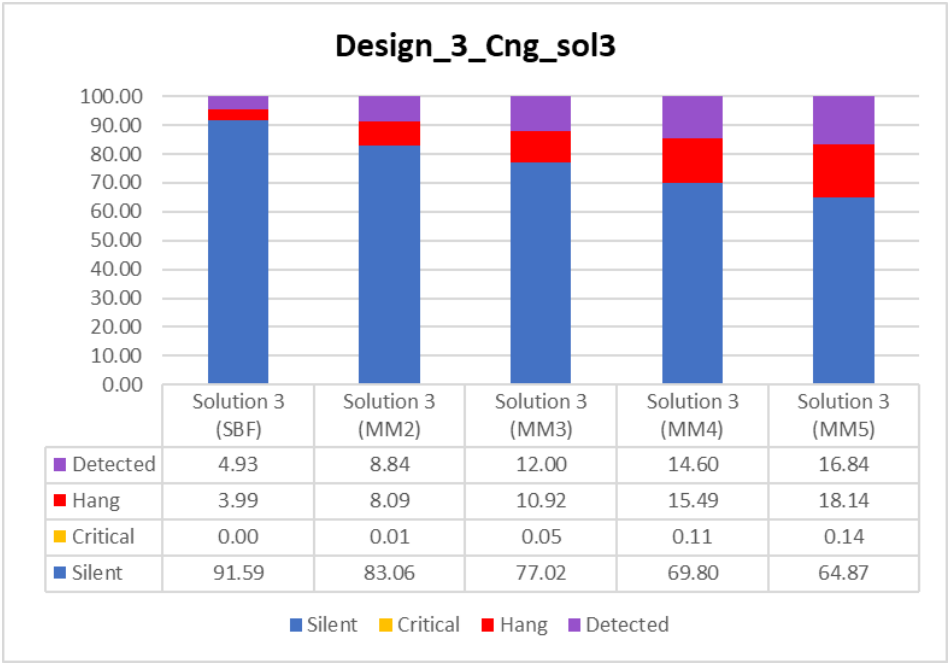


	Solution 2 (SBF)	Solution 2 (MM2)	Solution 2 (MM3)	Solution 2 (MM4)	Solution 2 (MM5)
Detected	17.24	26.63	29.45	31.43	30.84
Hang	8.19	15.46	22.62	28.91	34.82
Critical	0.00	0.32	0.62	0.75	0.78
Silent	74.57	57.59	47.30	38.90	33.56

Εικόνα 23 Design 3 Solution 2

10.3.3 Solution 3 (No inline)

Solution 3 (No inline): Στο σχήμα 9 τα αποτελέσματα του Solution 3 παρομοιάζουν με τα αποτελέσματα που ελήφθησαν με το Solution 1 με λίγο υψηλότερα ποσοστά Silent. Τα Critical errors αγγίζουν σχεδόν το μηδέν. Αυτό συνεπάγεται ότι είναι πιθανόν τα περισσότερα σφάλματα που θα εισαχθούν να μην γίνουν αντιληπτά.



	Solution 3 (SBF)	Solution 3 (MM2)	Solution 3 (MM3)	Solution 3 (MM4)	Solution 3 (MM5)
Detected	4.93	8.84	12.00	14.60	16.84
Hang	3.99	8.09	10.92	15.49	18.14
Critical	0.00	0.01	0.05	0.11	0.14
Silent	91.59	83.06	77.02	69.80	64.87

Εικόνα 24 Design 3 Solution 3

Αυτό που παρατηρείται από τα αποτελέσματα είναι ότι για όλα τα Solution του Design3 Cng για τα σφάλματα που εισήχθησαν εξαντλητικά (Exhaustive Single Bit Flip) έχουν μηδενιστεί τα Critical errors, ενώ όσο αυξάνεται το Multiplicity η αύξησή τους είναι σχεδόν αμελητέα. Φαίνεται δηλαδή ότι σε αυτή την High Level Synthesis σχεδίαση τα redundant block διατηρούνται διαχωρισμένα και έτσι όλα τα σφάλματα SBF είναι ανιχνεύσιμα. Επίσης όσο αυξάνεται το Multiplicity τα Silent errors μειώνονται σε πολύ μεγάλο ποσοστό, ενώ τα Detected errors αντίθετα αυξάνονται γεγονός που σημαίνει ότι η σχεδίαση λειτουργεί καλά.

Η πιο περίπλοκη σχεδίαση από όλες είναι το Design 2 Masked Canright HLS. Συγκρίνοντας τα αποτελέσματα μεταξύ των διαφορετικών Solution παρατηρείται ότι για το Solution 2 όταν μεγαλώνει το Multiplicity τα ποσοστά των Silent errors μειώνονται αρκετά ενώ αυξάνονται τα Critical errors έχοντας αισθητή διαφορά με εκείνα των άλλων Solution της ίδιας σχεδίασης. Τα Hang δείχνουν να κυμαίνονται στα ίδια περίπου ποσοστά για όλα τα Solution. Φαίνεται ότι η βελτιστοποίηση που εφαρμόστηκε (loop unrolling) δεν απέδωσε τα οφέλη της. Αυτό μπορεί να οφείλεται λόγω του μικρότερου κύκλου ρολογιού που έχει το Solution 2 σε σχέση με τα Solution 1 και 3, επομένως όταν εισαχθεί ένα σφάλμα είναι πιο πιθανό να επηρεάσει ένα flip-flop το οποίο θα χρησιμοποιηθεί αμέσως μετά. Ανάλογα αποτελέσματα παρατηρούνται και στο Solution 2 του Design 1 που πάλι λόγω μικρού κύκλου ρολογιού τα σφάλματα ειδικά όταν αυξάνεται το Multiplicity είναι σχεδόν βέβαιο ότι θα οδηγήσουν σε Critical ή Hang errors. Τα Solution 1 και 3 του Design 1 δεν παρουσιάζουν μεγάλες διαφορές. Συνοψίζοντας σημειώνεται ότι οι βελτιστοποιήσεις που εφαρμόστηκαν στις διαφορετικές υλοποιήσεις φαίνεται ότι επηρεάζουν την ανθεκτικότητα είτε των προστατευμένων είτε των μη προστατευμένων κυκλωμάτων έναντι επιθέσεων εισαγωγής σφαλμάτων (FI).

11. Συμπεράσματα – Προοπτικές

Στην παρούσα διπλωματική μελετήθηκαν οι επιπτώσεις της χρήσης μιας ροής HLS στην ασφάλεια προστατευμένων και μη προστατευμένων κρυπτογραφικών υλοποιήσεων έναντι επιθέσεων εισαγωγής σφαλμάτων. Τα υπό εισαγωγή σφάλματα εισήχθησαν σε υλοποιήσεις του αλγορίθμου Canright SBOX, οι δύο εκ των οποίων ενσωματώνουν αντίμετρα απόκρυψης και κάλυψης χρησιμοποιώντας το Vivado HLS. Μετά την ολοκλήρωση της εισαγωγής των σφαλμάτων και την καταγραφή των αποτελεσμάτων παρατηρήθηκε ότι μπορούν να εξαχθούν πληροφορίες και συμπεράσματα γύρω από μοτίβα που να καταδεικνύουν πώς πρόκειται να συμπεριφερθεί ένα κύκλωμα ανάλογα με το είδος και τα χαρακτηριστικά της επίθεσης.

Από τα αποτελέσματα φαίνεται ότι οι σχεδιαστές ασφαλών και αξιόπιστων κυκλωμάτων (υλικού) θα πρέπει να λαμβάνουν υπόψιν τους τις βελτιστοποιήσεις όταν χρησιμοποιούν μια ροή HLS καθώς φαίνεται να επηρεάζουν την ανθεκτικότητα είτε των προστατευμένων είτε των μη προστατευμένων σχεδίων έναντι επιθέσεων SCA ή Fault Injections (FI). Η υιοθέτηση αντιμέτρων αντικατοπτρίζεται στα αποτελέσματα, όπως για παράδειγμα στο Cng με το redundancy αντίμετρο.

Θεωρητικά η ισχυρότερη υλοποίηση είναι το Design 3 CNG Canright HLS γιατί επιτυγχάνει τα καλύτερα αποτελέσματα σε σχέση με τα Critical errors για όλα τα Solution. Επίσης όσο αυξάνεται το Multiplicity τα Silent errors μειώνονται σε πολύ μεγάλο ποσοστό, ενώ τα Detected errors αντίθετα αυξάνονται γεγονός που σημαίνει ότι το αντίμετρο του CNG που βασίζεται σε διπλασιασμούς διατηρεί τη θεωρητική του ικανότητα να ανιχνεύει όλες τις αναστροφές ενός bit. Επιπλέον, λόγω του ότι φαίνεται τα αποτελέσματα να επηρεάζονται ανάλογα την υλοποίηση, διαφαίνεται ότι υπάρχει ανάγκη για περαιτέρω έρευνα και ανάπτυξη της βελτίωση των αλγορίθμων HLS, προκειμένου να λαμβάνεται υπόψη η ανάγκη για ασφαλείς και αξιόπιστους επιταχυντές υλικού.

Η οχύρωση του υλικού (hardware) από άποψη ασφαλείας κρίνεται επιτακτική. Για το λόγο αυτό και προς εξοικονόμηση πολύτιμου χρόνου και πόρων κατά τη διαδικασία σχεδίασης και παραγωγής του, μπορούν να εφαρμοστούν κατάλληλες τεχνικές ώστε το υλικό να δοκιμαστεί και να αναδειχθούν τυχόν αδυναμίες με σκοπό τη λήψη αντιμέτρων.

Ως επέκταση της παρούσας Διπλωματικής θα μπορούσε να γίνει προσομοίωση της εισαγωγής σφαλμάτων σύμφωνα με τη μέθοδο των λογικών κώνων για όλες τις πολλαπλότητες Multiple Bit Flip (M=2,3,4 και 5), τα δείγματα των οποίων έχουν υπολογιστεί σύμφωνα με την στατιστική μέθοδο που παρουσιάστηκε στο Κεφάλαιο 5. Ο υπολογισμός των απαιτούμενων δειγμάτων χαρακτηρίζεται από τις ίδιες απαιτήσεις (margin of errors κ.λ.π.) με σκοπό να ανευρεθεί η προτιμότερη μέθοδος εισαγωγής σφαλμάτων ώστε να ελαχιστοποιούνται οι χρονοβόρες δοκιμές για την ανεύρεση ευπαθειών υλικού και να εξακριβωθεί αν η εφαρμογή περισσοτέρων αντιμέτρων θα επηρέαζε τα αποτελέσματα των FI.

12. Βιβλιογραφία

[1] Κουφοπούλου Αμαλία-Άρτεμις, Φεβρουάριος 2022, Πανεπιστήμιο Πειραιά, Τμήμα Πληροφορικής, ΠΜΣ «Κατανεμημένα Συστήματα, Ασφάλεια και Αναδυόμενες Τεχνολογίες», Μεταπτυχιακή Διατριβή: Development of hardware countermeasures for embedded systems security using High-Level Synthesis.

[2] Leveugle, R., Calvez, A., Maistri, P., & Vanhauwaert, P. (2009, April). Statistical fault injection: Quantified error and confidence. In 2009 Design, Automation & Test in Europe Conference & Exhibition (pp. 502-506). IEEE.

[3] Σημειώσεις Ασφάλεια Δικτύων και επικοινωνιών, Κοτζανικολάου, 2021.

[4] Boneh, Shoup, V. A Graduate Course in Applied Cryptography, Version 0.5, Jan. 2020, p.8-12, https://crypto.stanford.edu/~dabo/cryptobook/BonehShoup_0_5.pdf
Προσπελάστηκε: Μάϊος, 08, 2022

[5] https://www.ibm.com/docs/en/ztpf/2019?topic=concepts-hardware-cryptography.
Προσπελάστηκε: Ιούνιος 24, 2022

[6] An Introduction to High-Level Synthesis, 2. Analyzing Security Vulnerabilities Induced by High-level Synthesis

[7] Hardware Security A Hands-on Learning Approach by Mark Tehranipoor, Swarup Bhunia (z-lib.org) p.6-9,315

[8] Yuce, B., Schaumont, P., & Witteman, M. (2018). Fault attacks on secure embedded software: Threats, design, and evaluation. Journal of Hardware and Systems Security, 2(2), 111-130.

[9] Breier, J., & Hou, X. (2022). How Practical are Fault Injection Attacks, Really?. Cryptology ePrint Archive.

[10] Yuce, B., Schaumont, P., & Witteman, M. (2018). Fault attacks on secure embedded software: Threats, design, and evaluation. Journal of Hardware and Systems Security, 2(2), 111-130 (p.9).

[11] Sangchoolie, B., Pattabiraman, K., & Karlsson, J. (2017, June). One bit is (not) enough: An empirical study of the impact of single and multiple bit-flip errors. In 2017 47th annual IEEE/IFIP international conference on dependable systems and networks (DSN) (pp. 97-108). IEEE.

[12] Papadimitriou, A., Hély, D., Beroulle, V., Maistri, P., & Leveugle, R. (2014, March). A multiple fault injection methodology based on cone partitioning towards RTL modeling of laser attacks. In 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE) (pp. 1-4). IEEE.

[13] Leveugle, R., Calvez, A., Maistri, P., & Vanhauwaert, P. (2009, April). Statistical fault injection: Quantified error and confidence. In 2009 Design, Automation & Test in Europe Conference & Exhibition (pp. 502-506). IEEE.

[14]https://www.xilinx.com/content/dam/xilinx/support/documents/sw_manuals/xilinx2020_2/ug902-vivado-high-level-synthesis.pdf#nameddest=xApplyingOptimizationDirectives
Προσπελάστηκε: Ιούνιος, 19, 2022

[15] Canright, D. (2005, August). A very compact S-box for AES. In International Workshop on Cryptographic Hardware and Embedded Systems (pp. 441-455). Springer, Berlin, Heidelberg
[16] A. Satoh, S. Morioka, K. Takano, and Seiji Munetoh. A compact Rijndael hardware architecture with S-box optimization. In Advances in Cryptology – ASIACRYPT 2001, volume 2248 of Lecture Notes in Computer Science, pages 239–254. Springer,2001.

[17] Nele Mentens, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. A systematic evaluation of compact hardware implementations for the Rijndael S-box. In CTRSA, volume 3376 of Lecture Notes in Computer Science, page 323333. Springer, 2005.

[18] Canright, D. (2005, August). A very compact S-box for AES. In International Workshop on Cryptographic Hardware and Embedded Systems (pp. 441-455). Springer, Berlin, Heidelberg

[19] A. Satoh, S. Morioka, K. Takano, and Seiji Munetoh. A compact Rijndael hardware architecture with S-box optimization. In Advances in Cryptology – ASIACRYPT 2001, volume 2248 of Lecture Notes in Computer Science, pages 239–254. Springer,2001.

[20]. Nele Mentens, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. A systematic evaluation of compact hardware implementations for the Rijndael S-box. In CTRSA, volume 3376 of Lecture Notes in Computer Science, page 323333. Springer, 2005.

[21] https://www.eetimes.com/tutorial-programming-high-performance-dsps-part-2/

[22] TCL: https://www.tutorialspoint.com/tcl-tk/index.htm

[23] Leveugle, R., Calvez, A., Maistri, P., & Vanhauwaert, P. (2009, April). Statistical fault injection: Quantified error and confidence. In 2009 Design, Automation & Test in Europe Conference & Exhibition (pp. 502-506). IEEE.