Εικόνα που περιέχει κείμενο, δωμάτιο, καζίνο

Περιγραφή που δημιουργήθηκε αυτόματα

**Πανεπιστήμιο Ιωαννίνων**

**Πολυτεχνική Σχολή**

**Τμήμα Μηχανικών Η/Υ και Πληροφορικής**

**Προπτυχιακό Μάθημα: «Δοκιμή και Αξιοπιστία Ηλεκτρονικών Συστημάτων»**

**Δεύτερη Εργαστηριακή Άσκηση**

**Όνομα Φοιτητή – Α.Μ.:**

**Γεώργιος Κρομμύδας – 3260**

Εικόνα που περιέχει κείμενο, ηλεκτρονικές συσκευές

Περιγραφή που δημιουργήθηκε αυτόματα

*ΙΩΆΝΝΙΝΑ,*

*2021*

Πίνακας περιεχομένων

[Μέρος ~ 1ο: Περιγραφή Εργασίας: 3](#_Toc71665857)

[Μέρος ~ 2ο: Υλοποίηση Κυκλωμάτων: 3](#_Toc71665858)

[Μέρος ~ 3ο: Fault Injections: 13](#_Toc71665859)

[APPENDIX: 15](#_Toc71665860)

[Άσκηση – 2.1: 15](#_Toc71665861)

[Άσκηση – 2.2: 17](#_Toc71665862)

[Άσκηση – 2.3: 18](#_Toc71665863)

[Άσκηση – 2.4: 20](#_Toc71665864)

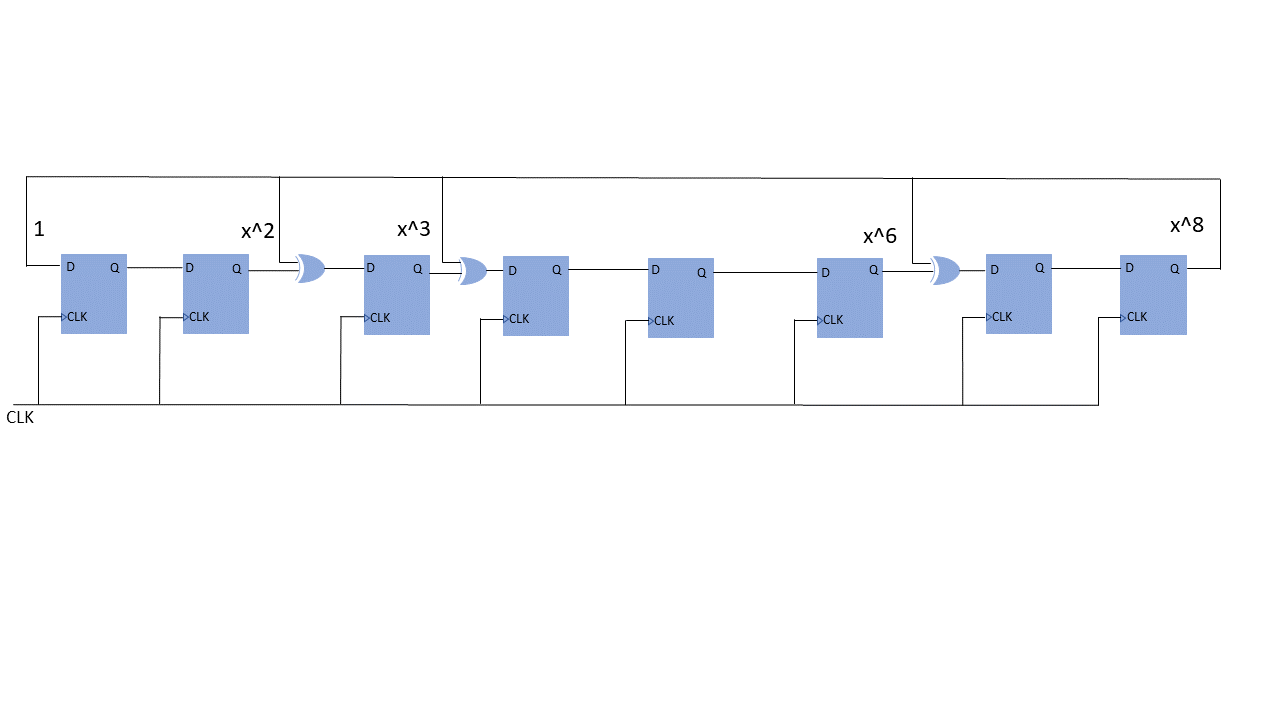
# Μέρος ~ 1ο: Περιγραφή Εργασίας:

Σε αυτή την εργασία είχαμε να υλοποιήσουμε δύο κυκλώματα, τα ***TRCUTwithLSFR*** και ***TRCUTwithMISR***. Χρησιμοποιώντας το μοντέλο ***TRCUT*** της προηγούμενης εργασίας και τα κυκλώματα ***LSFR*** και ***MISR*** θα δημιουργηθούν τα αντίστοιχα κυκλώματα. Η σειριακή σάρωση θα γίνει μέσω ψευδοτυχαίων διανυσμάτων που παράγονται από το ***LSFR*** και εισάγονται σειριακά στην είσοδο της σειριακής αλυσίδας του υποκυκλώματος ***TRCUT***. Στη συνέχεια η έξοδος της σειριακής αλυσίδας αποτελεί είσοδο στο ***MISR***. Η έξοδος του κυκλώματος αυτού ονομάζεται υπογραφή (***sign***) και από αυτήν ανιχνεύεται η σωστή λειτουργία ενός κυκλώματος. Γενικά, το κύκλωμα αυτό ονομάζεται ***BIST(Built-In Self-Test)***.

# Μέρος ~ 2ο: Υλοποίηση Κυκλωμάτων:

Το πρώτο κύκλωμα είναι το ***TRCUTwithLSFR***. Αρχικά, χρησιμοποιώντας το ***LSFRtestbench*** σχεδιάστηκε το κύκλωμα ***LSFR*** 8ου βαθμού. Το ***LSFR*** είναι ένα κύκλωμα παραγωγής ψευδοτυχαίων διανυσμάτων τα οποία δημιουργούνται βάσει του σήματος ρολογιού και της αρχικοποίησης μίας αρχικής κατάστασης. Το συγκεκριμένο κύκλωμα έχει χαρακτηριστικό πολυώνυμο:

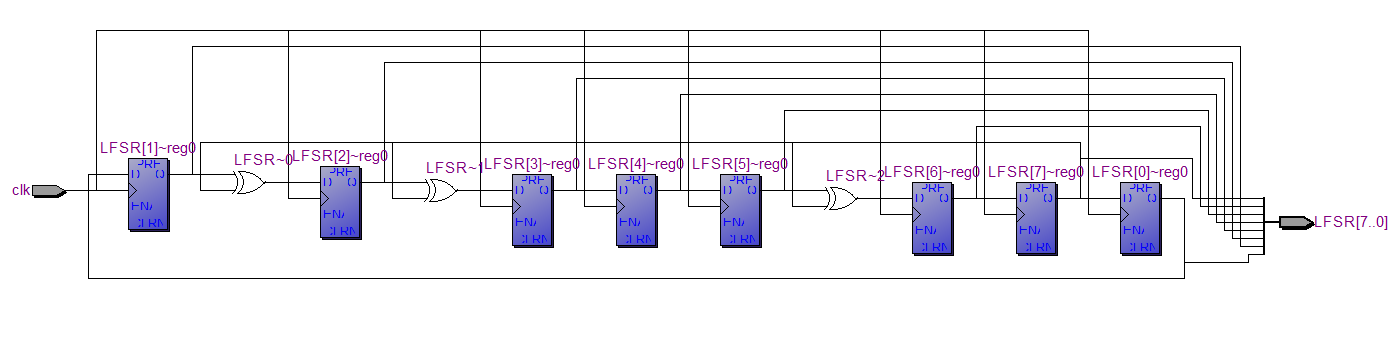
Το κύκλωμα έχει σχεδιαστεί με την μέθοδο ***internal (or modular) LSFR***. Οι ***xor*** πύλες τοποθετήθηκαν σε αυτά τα σημεία για να μπορούν να παραχθούν 255 ψευδοτυχαία διανύσματα. Επιπλέον, δημιουργείται ανάδραση για να ανατροφοδοτεί την είσοδο των ***flip-flop*** παράλληλα με τις πύλες. Έτσι, θα μπορέσουν να σχηματιστούν τα διανύσματα ελέγχου για το κύκλωμα υπό δοκιμή. Παρακάτω φαίνεται το κύκλωμα:



Εικόνα 1: Κύκλωμα LSFR 8ου Βαθμού

Η υλοποίηση αυτού του κυκλώματος βασίζεται στην ***RTL*** περιγραφή του κώδικα **Κ1** του κεφαλαίου **APPENDIX**. Το κύκλωμα έχει ως είσοδο το σήμα του ρολογιού και ως έξοδο τα αποτελέσματα των καταχωρητών τα οποία θα παράγουν τα διανύσματα. Επιπλέον, υπάρχει και το *wire* **feedback**, το οποίο οδηγείται στην είσοδο του πρώτου ***flip-flop*** και στις πύλες ***xor***.

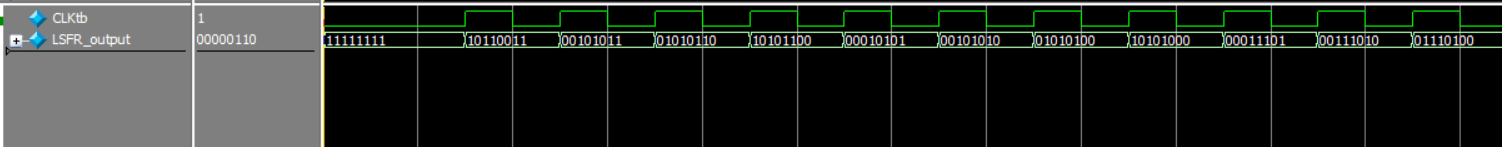
Η σχεδίαση αυτή μπορεί να επαληθευτεί και από τον ***RTL – Viewer*** ως εξής:



Εικόνα 2: Κύκλωμα LSFR Quartus

Το επόμενο στάδιο είναι η προσομοίωση του παραπάνω κυκλώματος μέσω ενός ***testbench***. Έχει δημιουργηθεί με βάση του κώδικα **Κ2** του κεφαλαίου **APPENDIX**. Αρχικά έχουμε τα σήματα ***CLKtb*** και ***LSFR\_output***  όπου είναι η είσοδος και η έξοδος του κυκλώματος. Στην συνέχεια δημιουργείται το μπλοκ παραγωγής ρολογιού το οποίο έχει περίοδο 20 *ns*. Στη συνέχεια καλείται το instance του κυκλώματος ως:

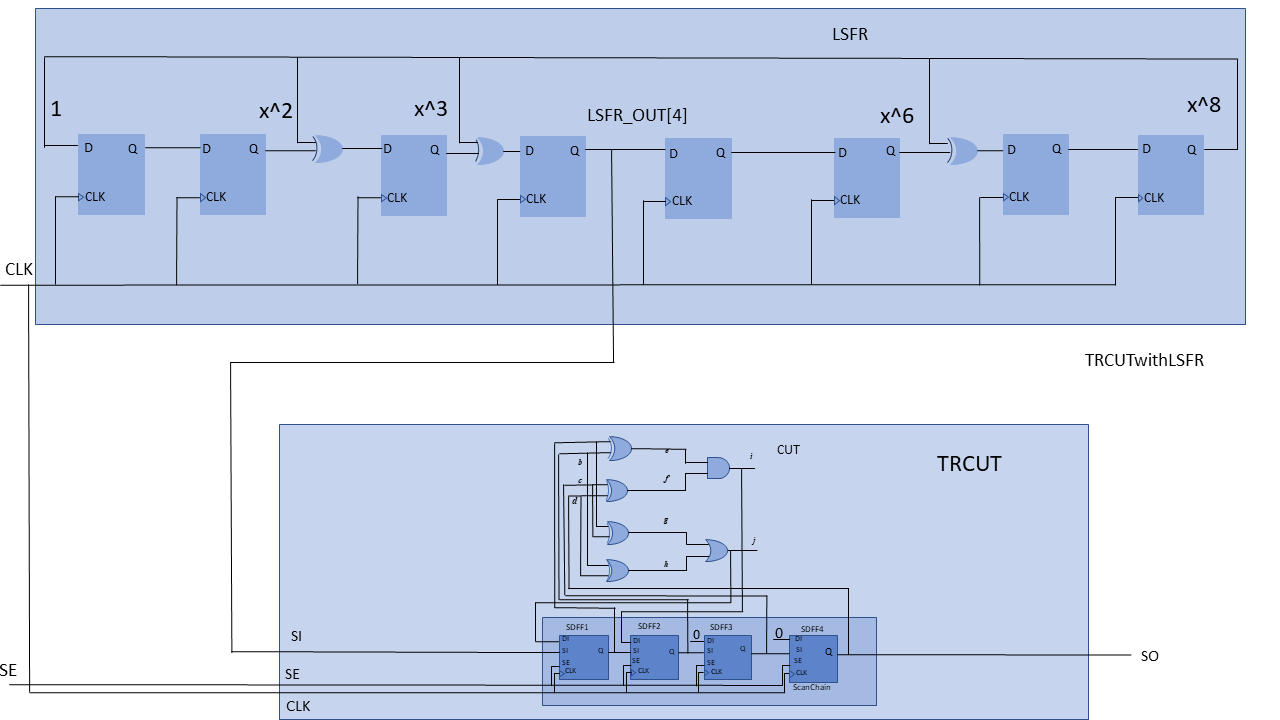
Έτσι, θα γίνει σωστά η προσομοίωση του παραπάνω κυκλώματος και θα παραχθούν σωστά όλα τα διανύσματα. Παρακάτω φαίνεται το αποτέλεσμα της προσομοίωσης:



Εικόνα 3: Προσομοίωση Κυκλώματος LSFR

Ξεκινά το κύκλωμα πρώτα με την αρχική τιμή που δόθηκε που είναι το 255. Στη συνέχεια, κατά την έναρξη λειτουργίας οι τιμές αλλάζουν και παράγεται στον επόμενο κύκλο ρολογιού. Αυτή η διαδικασία θα συνεχιστεί μέχρις ότου παραχθούν όλα τα διανύσματα δοκιμής του ***LSFR***.

Το επόμενο κομμάτι είναι η σχεδίαση του κυκλώματος ***TRCUTwithLSFR*** το οποίο χρησιμοποιεί το παραπάνω κύκλωμα ως υποκύκλωμα μαζί με το ***TRCUT***, το οποίο είναι το κύκλωμα υπό δοκιμή. Αρχικά, η έξοδος του ***LSFR*** θα είναι από το τέταρτο ***flip-flop***, η οποία θα αποτελεί είσοδο στην σειριακή αλυσίδα του κυκλώματος. Δηλαδή, τα ψευδοτυχαία διανύσματα θα εισάγονται σειριακά στην είσοδο ***SI*** του ***TRCUT***. Έτσι θα μπορούμε να ελέγξουμε την ορθή λειτουργία του κυκλώματος ***CUT***. Θα μεταδίδονται τα διανύσματα μέσα στην σειριακή αλυσίδα και έπειτα στο κύκλωμα. Οι αποκρίσεις του κυκλώματος θα επιστρέφουν στην σειριακή αλυσίδα. Το αποτέλεσμα της αλυσίδας το οποίο είναι το σήμα ***SO*** θα πρέπει να είναι ίδιο με αυτό της εισόδου. Το κύκλωμα γενικά αποτελείται από δύο εισόδους τα οποία είναι το ρολόι ***CLK*** και το σήμα ***SE(Scan Enable)*** και μία έξοδο την ***SO(Scan Output)***.

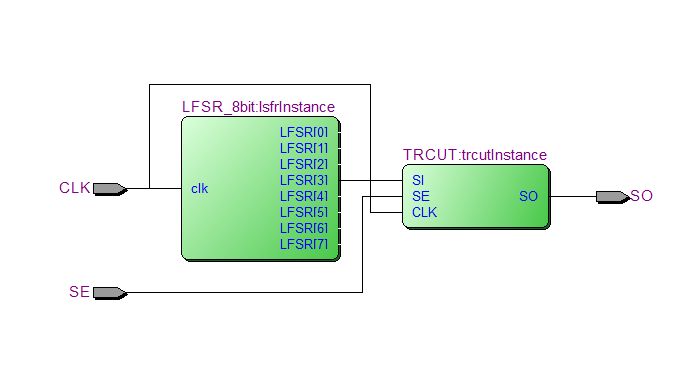


Εικόνα 4: Κύκλωμα TRCUTwithLSFR

Η παραπάνω αρχιτεκτονική σχεδιάστηκε βάσει του κώδικα Κ3 του κεφαλαίου **APPENDIX**. Έχουμε τις εισόδους μας και τις εξόδους μας και τις κάνουμε και *wire* για να μπορούν να συνδεθούν με τα αντίστοιχα υποκυκλώματα. Επιπλέον, έχουμε και ένα *wire* το οποίο αποτελεί τον δίαυλο με τις εξόδους του ***LSFR*** το οποίο είναι το **lsfr\_out**. Στη συνέχεια καλούμε το υποκύκλωμα ***LSFR*** ως εξής:

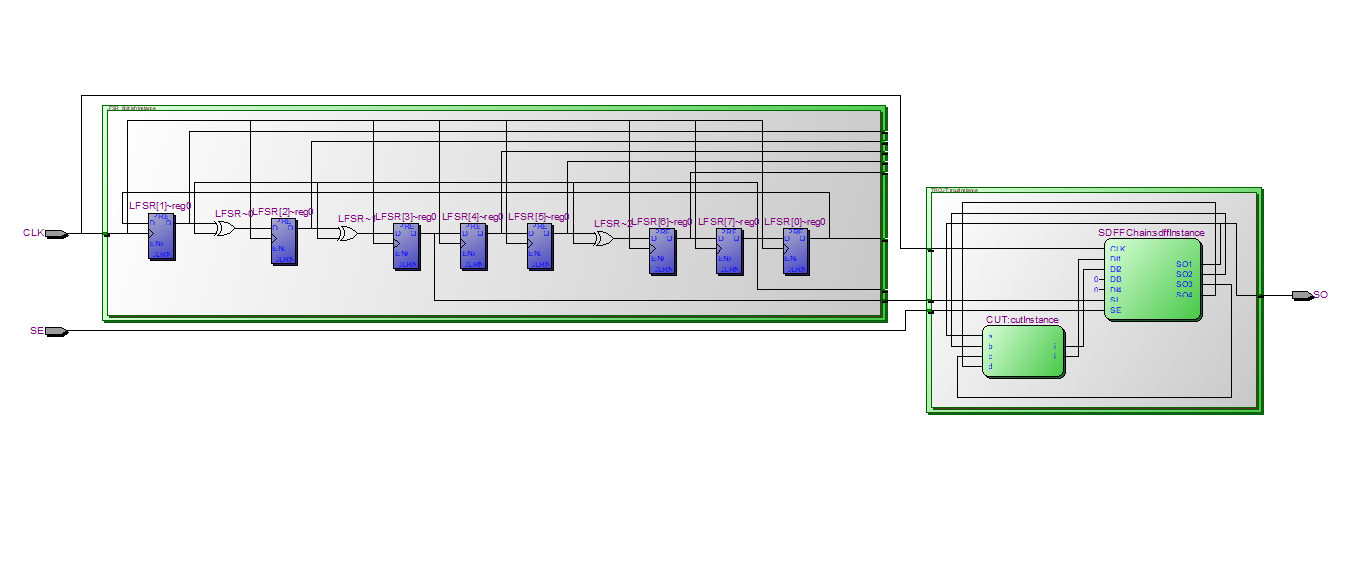
Στη συνέχεια παίρνουμε μία έξοδο του κυκλώματος την οποία την αναθέτουμε στο *wire* **lsfr\_output**. Συγκεκριμένα έχει επιλεχθεί η έξοδος του τέταρτου ***flip-flop***. Συνεπώς, η έξοδος αυτή θα αποτελεί την σειριακή είσοδο στην ***Scan Chain*** του κυκλώματος ***TRCUT***. Κατ΄ επέκταση, τα διανύσματα που παράγονται θα εισάγονται από εκεί για τον έλεγχο λειτουργίας. Επιπλέον, στη σειριακή αλυσίδα του ***TRCUT*** έχουν τοποθετηθεί οι είσοδοι **DI3** και **DI4** στην τιμή ‘0’, για να μην είναι στον αέρα και δέχεται το κύκλωμα τιμές ‘X’ ή ‘Z’. Έπειτα καλείται το κύκλωμα δοκιμής:

Η επαλήθευση από το ***Quartus*** φαίνεται παρακάτω:

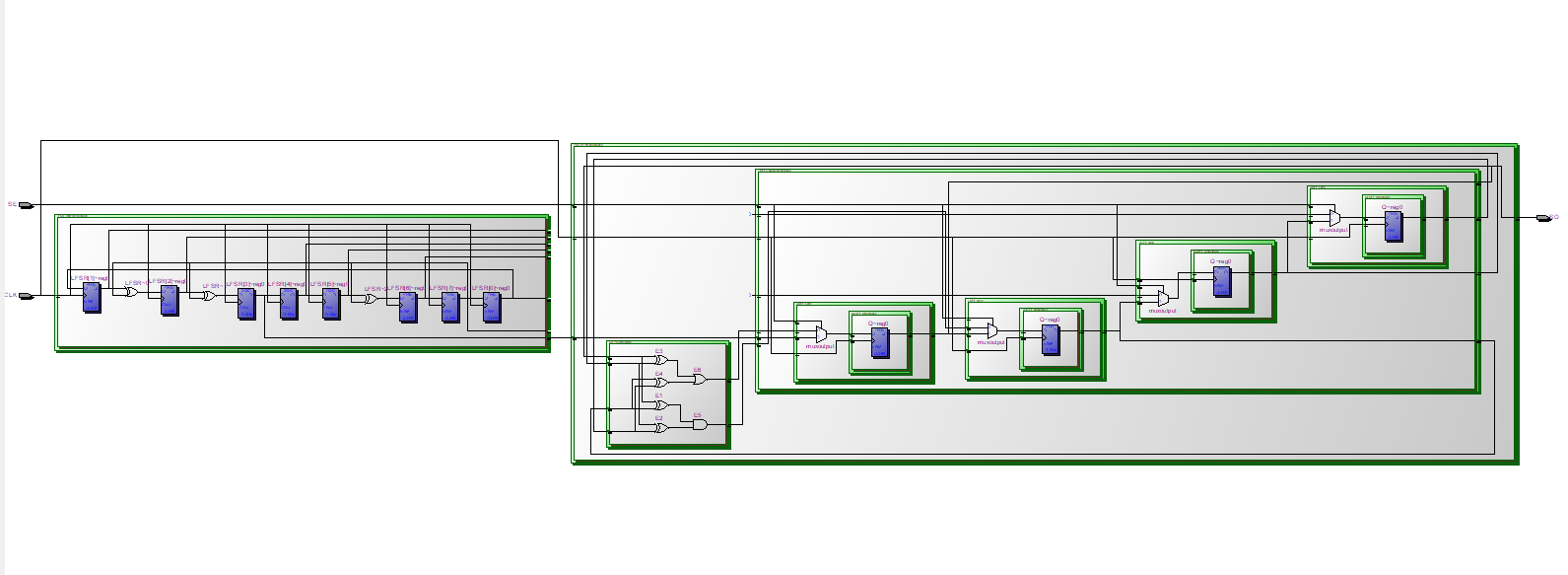


Εικόνα 5: Κύκλωμα TRCUTwithLSFR

Παρατηρούμε πως η έξοδος του τέταρτου ***flip-flop*** αποτελεί είσοδο στην σειριακή αλυσίδα. Επιπλέον, εάν κάνουμε και flatten το σχήμα αλλά και show contents στα αντίστοιχα υποκυκλώματα, βλέπουμε σε επίπεδο πυλών τις συνδέσεις.



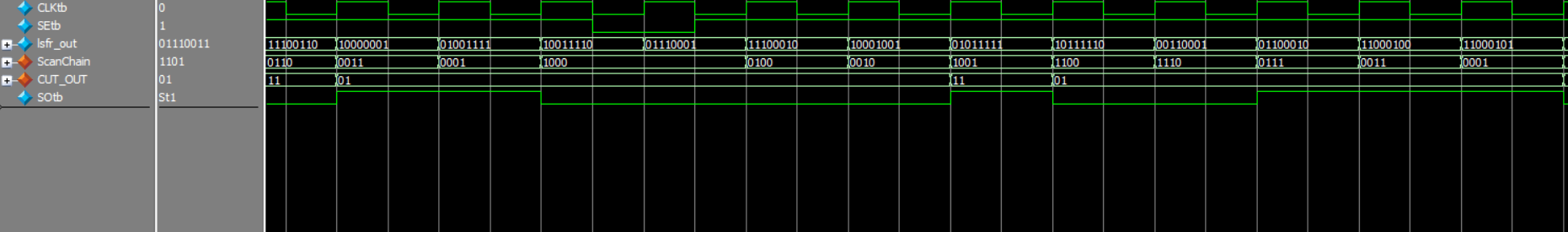
Εικόνα 6: Κύκλωμα TRCUTwithLSFR



Εικόνα 7: Κύκλωμα TRCUTwithLSFR

Το επόμενο βήμα είναι να προσομοιώσουμε το παραπάνω κύκλωμα και να δούμε τις αποκρίσεις του κυκλώματος ως προς δοκιμή. Έχει φτιαχτεί το ***testbench*** του κώδικα Κ4 του κεφαλαίου **APPENDIX**. Πρώτα αρχικοποιούνται ως καταχωρητές τα σήματα ***CLKtb*** και ***SEtb*** για να μπορούν να αποθηκεύουν τιμές και να τις μεταδίδουν στα κυκλώματα. Επιπλέον παρατηρούμε την έξοδο στο *wire* ***SOtb*** και βλέπουμε εάν είναι σωστές οι αποκρίσεις. Στη συνέχεια καλείται το instance του παραπάνω κυκλώματος με εισόδους και έξοδο τα παραπάνω σήματα. Η δήλωση γίνεται ως εξής:

Το επόμενο βήμα είναι η παραγωγή παλμών ρολογιού. Ο κάθε παλμός έχει περίοδο 20 *ns*. Ανά 10 *ns* έχουμε άνοδο και πτώση αντίστοιχα ρου παλμού. Τέλος, έχουμε ένα μικρό λουπ στο οποίο ανατίθεται τιμή στο σήμα ***Scan Enable*** για να ξεκινήσει η σειριακή αλυσίδα να λειτουργεί. Αρχικά θέτουμε την τιμή στον ‘1’ για να ξεκινήσουν τα διανύσματα να εισάγονται στην αλυσίδα. Από την στιγμή που παράγονται συνολικά 255 διαφορετικά διανύσματα, τότε έχουμε και 32 διανύσματα που χρειαζόμαστε. Ανά μία περίοδο εισάγεται το μισό διάνυσμα μέσα στην αλυσίδα. Έτσι θα χρειαστεί για το μισό διάνυσμα 4 κύκλους και αντίστοιχα άλλους 4 για το επόμενο μισό. Αυτό ισχύει διότι η σειριακή αλυσίδα έχει μήκος ***L = 4***. Στη συνέχεια έχουμε ***Capture Mode*** για έναν κύκλο ρολογιού για παρατηρούμε την απόκριση του κυκλώματος στην σειριακή αλυσίδα. Έπειτα ξανά αλλάζει η λειτουργία του κυκλώματος και εισάγονται περαιτέρω τα υπόλοιπα διανύσματα. Για να περαστούν και 32 διανύσματα από το κύκλωμα θα χρειαστούμε:

Εικόνα που περιέχει κείμενο, εσωτερικό

Περιγραφή που δημιουργήθηκε αυτόματαΠαρακάτω φαίνονται οι αποκρίσεις από τις κυματορμοφές που παράγει το ***testbench***.

Capture Mode

vector

half vector

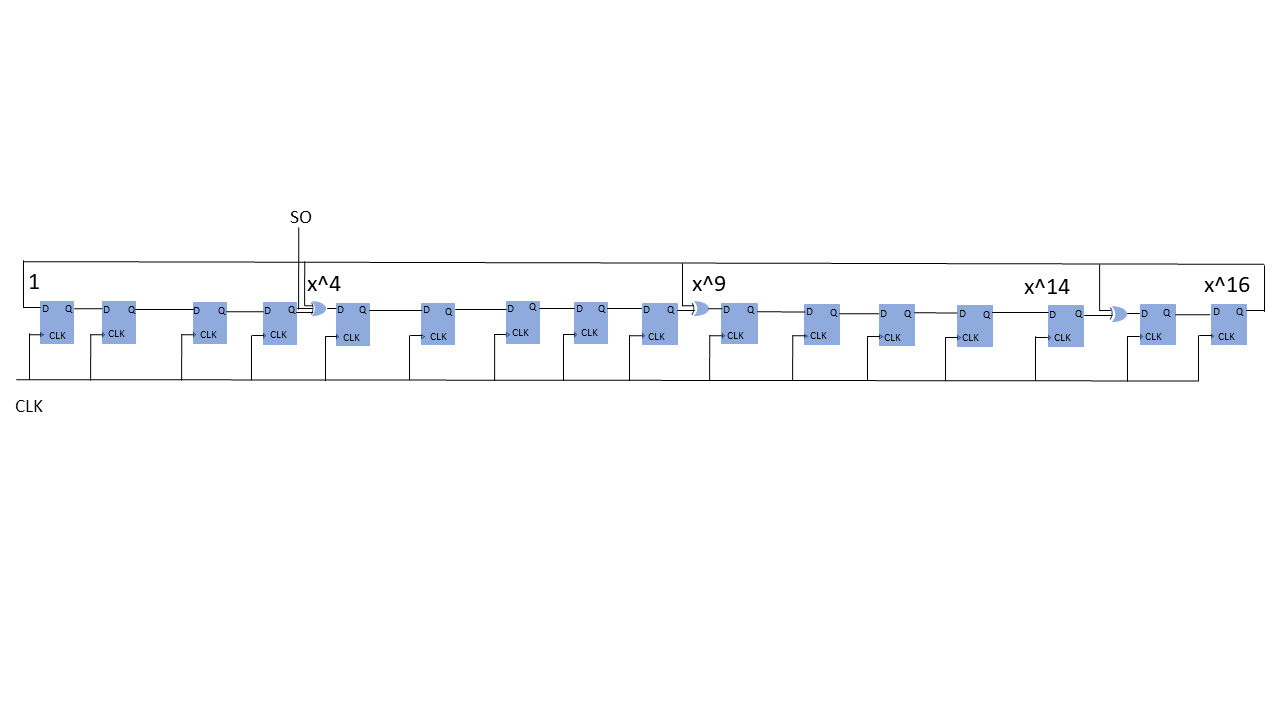
Εικόνα 8: Κυματομορφή 2 Κυκλώματος TRCUTwithLSFR

Εικόνα 9: Κυματομορφή 1 Κυκλώματος TRCUTwithLSFR

Οι αποκρίσεις του κυκλώματος ***CUT*** είναι σωστές, ωστόσο υπάρχουν προβλήματα στα ψευδοτυχαία διανύσματα. Προλαβαίνει σε κάποιες περιπτώσεις να πάρει και τα δύο κομμάτια του διανύσματος. Όμως, σε κάποιες άλλες περιπτώσεις παίρνει το μισό διάνυσμα και σε κάποιες δεν εισάγεται καθόλου. Αυτό οφείλεται στο γεγονός πως το μήκος της αλυσίδας δεν είναι ίδιο μα το μήκος των διανυσμάτων. Επιπλέον χρειάζεται 4 επιπλέον κύκλους ρολογιού για να μεταδώσει το μισό διάνυσμα μέσα στην αλυσίδα, με συνέπεια κάποια διανύσματα να χάνονται.

Το επόμενο κύκλωμα είναι το ***TRCUTwithMISR***. Αυτό το κύκλωμα αποτελεί γενίκευση του προηγουμένου με την μόνη διαφορά ότι υπάρχει και ένα επιπλέον κύκλωμα ***MISR***. Αυτό το κύκλωμα στην ουσία είναι ένα ***internal (***ή ***modular) LSFR*** με την μόνη διαφορά πως υπάρχει μία είσοδο στο κύκλωμα που συνδέεται με μία πύλη ***xor***. Για την άσκηση την συγκεκριμένη έχει δημιουργηθεί ένα ***MISR*** 16ου βαθμού το οποίο έχει χαρακτηριστικό πολυώνυμο:

Δηλαδή, έχει πύλες στις εξόδους των ***flip-flops*** που βρίσκεται ο εκθέτης, εκτός από την μονάδα που δηλώνει ανάδραση. Το παραπάνω πολυώνυμο έχει την δυνατότητα παραγωγής 65535 διανυσμάτων υπογραφής. Παρακάτω φαίνεται το κύκλωμα:



Εικόνα 10: Κύκλωμα MISR 16ου Βαθμού

Γενικά το κύκλωμα έχει δύο εισόδους και μία έξοδο. Είναι ένα σύγχρονο ακολουθιακό κύκλωμα με θετική ακμοπυροδότηση. Η μία είσοδος είναι ο παλμός του ρολογιού και η άλλη είσοδος θα είναι η έξοδος από την σειριακή αλυσίδα του κυκλώματος υπό δοκιμή Η υλοποίηση αυτού του κυκλώματος βασίζεται στην ***RTL*** περιγραφή του κώδικα Κ5 του κεφαλαίου **APPENDIX**. Το κύκλωμα έχει είσοδο το σήμα ρολογιού ***CLK*** και την είσοδο ***misr\_input*** όπου θα εισάγονται τα δεδομένα από την σειριακή αλυσίδα του κυκλώματος ***CUT***. Ως έξοδο έχει τις τιμές των 15 καταχωρητών που αποτελούν έναν δίαυλο. Η έξοδος αρχικά έχει τιμή 65535 και με την πάροδο του χρόνου δημιουργούνται και άλλα διανύσματα. Η επαλήθευση από το ***Quartus*** είναι η εξής:

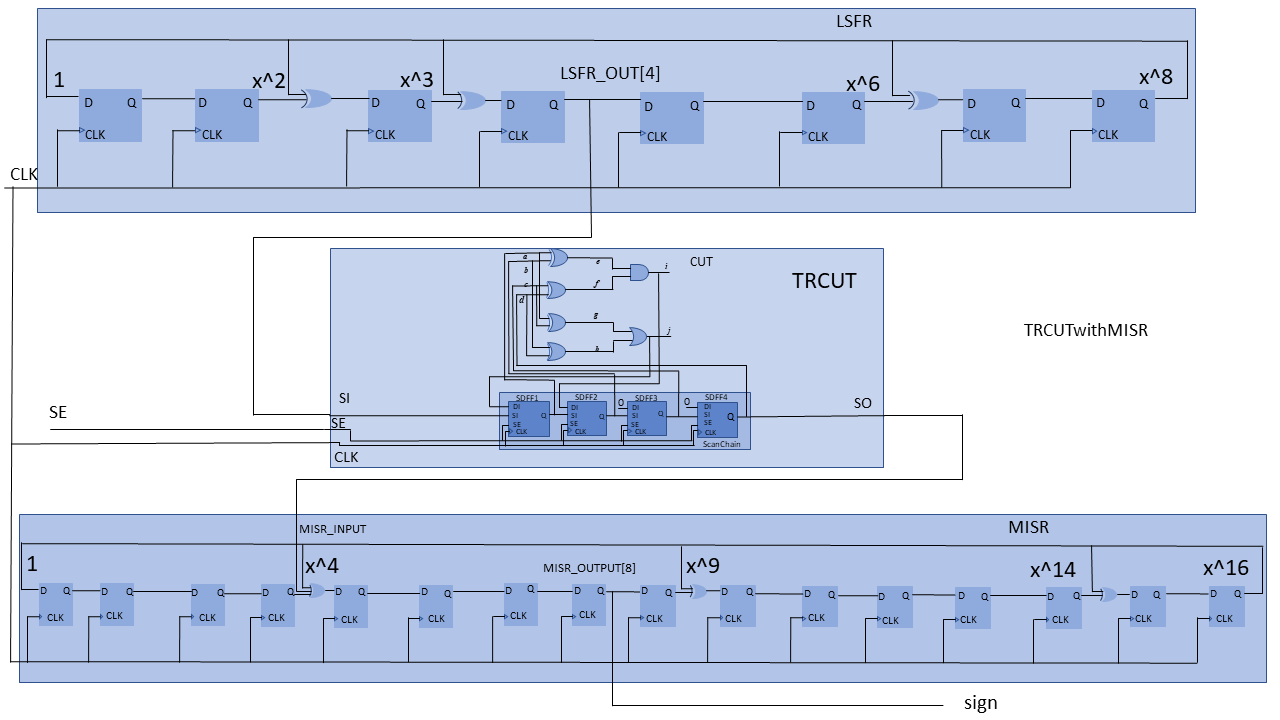
Εικόνα που περιέχει κείμενο, κεραία, στιγμιότυπο οθόνης

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα 11: Κύκλωμα MISR Quartus

Το επόμενο και τελικό στάδιο είναι να ενσωματωθεί το κύκλωμα ***MISR*** με το κύκλωμα ***TRCUTwithLSFR*** και να δημιουργηθεί το συνολικό κύκλωμα ***TRCUTwithMISR***. Το κύκλωμα αυτό αποτελείται από δύο εισόδους τις ***CLK*** και ***SE***. Το σήμα ***CLK*** χρησιμοποιείται για τον συγχρονισμό των κυκλωμάτων ***LSFR***, ***MISR*** και την σειριακή αλυσίδα. Ενώ το σήμα ***SE*** είναι μόνο για την σειριακή αλυσίδα. Αν πρόκειται για εισαγωγή δεδομένων σειριακά ή από τις εισόδους ***DI***. Η σύνδεση του κυκλώματος ***MISR*** με την σειριακή αλυσίδα γίνεται μέσω της ***SO*** και της ***misr\_input***.

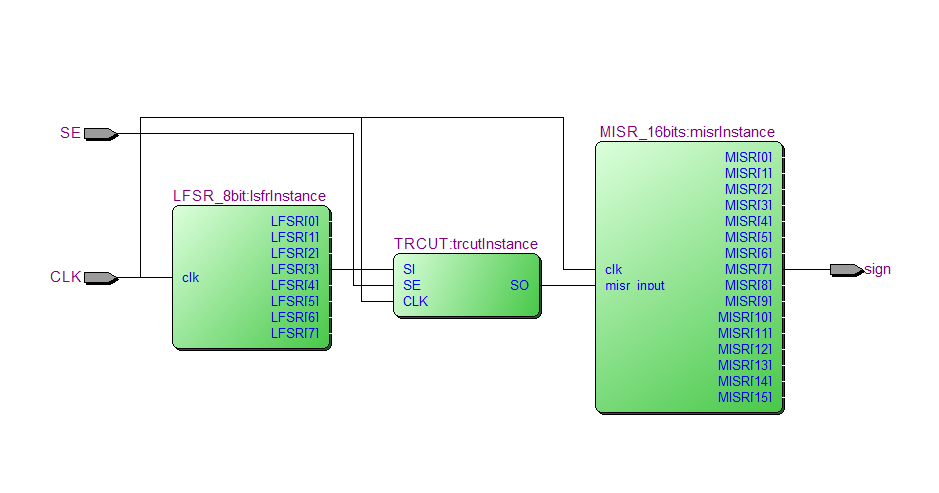
Επιπλέον, το κύκλωμα έχει ως έξοδο το ***sign*** η οποία είναι η υπογραφή του κυκλώματος υπό δοκιμή. Η υπογραφή είναι ένα διάνυσμα το οποίο προσδιορίζει την ορθότητα του ελέγχου. Δηλαδή, ένα διάνυσμα το οποίο έχει εισαχθεί μέσα στην σειριακή αλυσίδα και αποτελεί έξοδο και στην έξοδο υπογραφής, τότε το κύκλωμα λειτουργεί σωστά και δεν έχει σφάλματα. Παρακάτω φαίνεται το κύκλωμα:



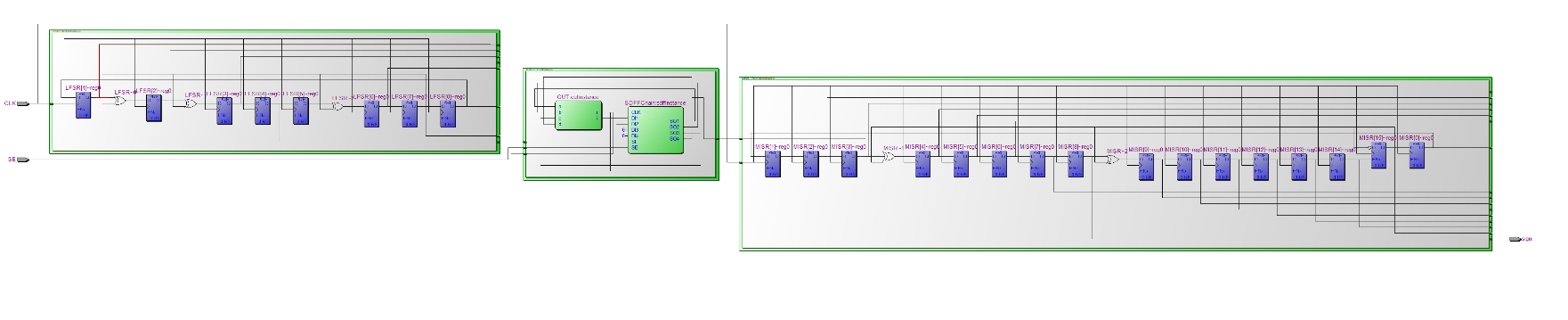
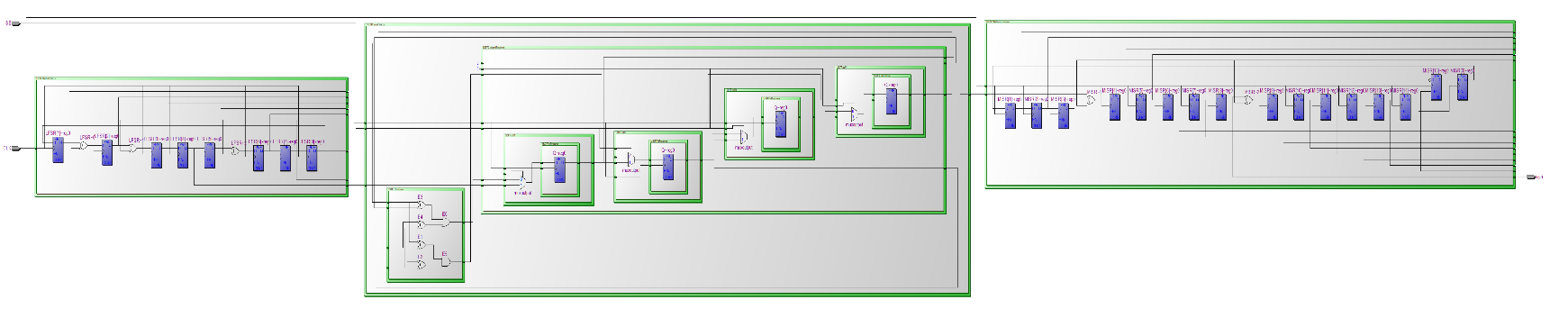
Εικόνα 12: Κύκλωμα TRCUTwithMISR

Η υλοποίηση αυτού του κυκλώματος βασίζεται στην ***RTL*** περιγραφή του κώδικα Κ6 του κεφαλαίου **APPENDIX**. Ο κώδικας αυτός είναι πανομοιότυπος με τον κώδικα Κ3 του κεφαλαίου **APPENDIX** με την μόνη διαφορά ότι καλείται και το κύκλωμα ***MISR***. Γίνεται μέσω της εντολής:

Επιπλέον, έχει δημιουργηθεί και ένας δίαυλος ***misr\_output*** που βγάζει τα αποτελέσματα των καταχωρητών. Επίσης, η έξοδος ***sign*** αποτελεί την έξοδο του κυκλώματος και βγάζει την υπογραφή του. Αποτελεί την έξοδο του όγδοου ***flip-flop***, όπου το ***misr\_out*** αναθέτεται στο σήμα ***sign***. Η επαλήθευση από το ***Quartus*** είναι η εξής:



Εικόνα 13: Κύκλωμα TRCUTwithMISR



Εικόνα 14:Κύκλωμα TRCUTwithMISR Quartus

Εικόνα 15: Κύκλωμα TRCUTwithMISR Quartus

Το επόμενο βήμα είναι να προσομοιώσουμε το παραπάνω κύκλωμα και να δούμε τις αποκρίσεις του κυκλώματος ως προς δοκιμή. Έχει φτιαχτεί το ***testbench*** του κώδικα Κ7 του κεφαλαίου **APPENDIX**. Ο κώδικας αυτός είναι παρόμοιος με αυτόν του Κ5, με την μόνη διαφορά ότι ως έξοδο θα έχουμε την απόκριση από το κύκλωμα ***MISR***. Η έξοδος αποτελεί την υπογραφή του κυκλώματος η οποία θα πρέπει να έχει ίδιες τιμές με την έξοδο του κυκλώματος ***LSFR***. Έτσι θα γνωρίζουμε την ορθή λειτουργία του κυκλώματος υπό δοκιμή. Σε περίπτωση που κάποιο bit έχει αλλοιωθεί, τότε θα ξέρουμε πως έχει υπάρξει κάποιο σφάλμα στο κύκλωμα. Παρακάτω φαίνονται οι κυματομορφές Εικόνα που περιέχει κείμενο, πίνακας αποτελεσμάτων, επίπεδος

Περιγραφή που δημιουργήθηκε αυτόματαΕικόνα που περιέχει κείμενο

Περιγραφή που δημιουργήθηκε αυτόματαμε τα 32 διανύσματα που παράχθηκαν:

Εικόνα 16: Κυματομορφή Κυκλώματος TRCUTwithMISR 2

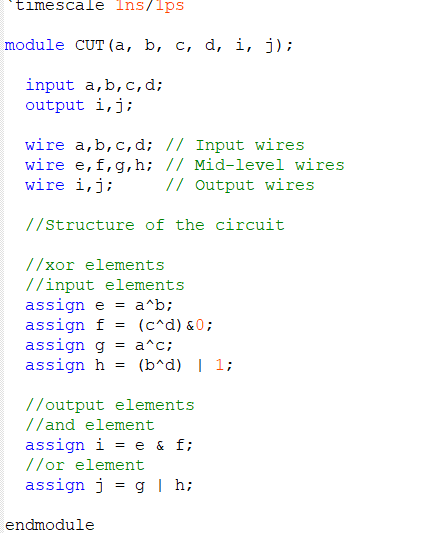
Εικόνα 17: Κυματομορφή Κυκλώματος TRCUTwithMISR

Παρατηρούμε πως οι τιμές του σήματος ***SO*** εισάγονται μέσω της πύλης ***xor*** και έτσι θα δημιουργηθεί η υπογραφή που περιμένουμε από το κύκλωμα. Ωστόσο, παρατηρείται ότι η υπογραφή καταστρέφεται λόγω των άγνωστων τιμών που έχει αρχικά το κύκλωμα υπό δοκιμή.

Σε γενικές γραμμές για να πάρουμε και τα 32 διανύσματα που παράχθηκαν θα χρειαστούμε τους 164 κύκλους που υπολογίστηκαν προηγουμένως, συν ακόμα 16 κύκλους που είναι το μήκος του ***MISR***. Οπότε το κύκλωμα συνολικά θα χρειαστεί 180 κύκλους ρολογιού, έτσι ώστε να εξακριβώσουμε την ορθή του λειτουργία.

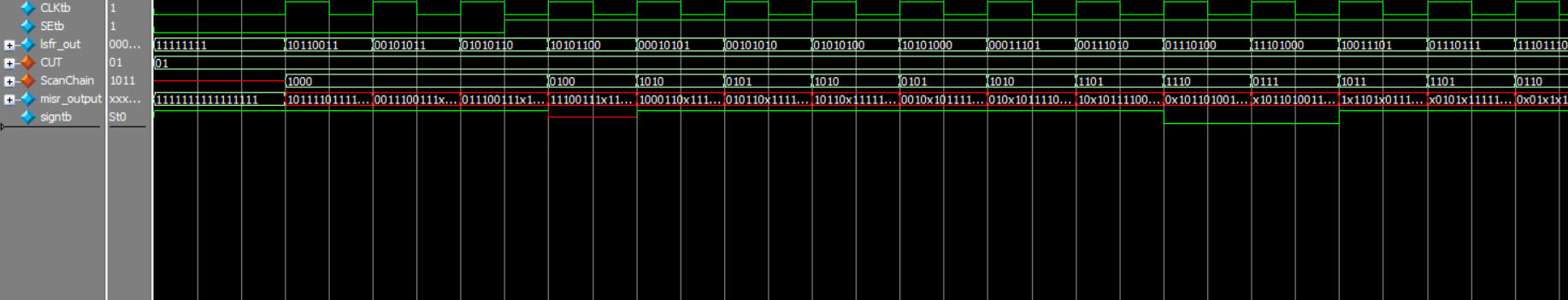
# Μέρος ~ 3ο: Fault Injections:

Το επόμενο κομμάτι είναι να εξετάσουμε την συμπεριφορά του κυκλώματος σε ***fault injections***. Αρχικά, έστω ότι κάνουμε ***assign*** στην έξοδο της δεύτερης πύλης ***xor*** την τιμή ‘0’ και στην έξοδο της πύλης της τέταρτης πύλης ***xor*** την τιμή ‘1’. Έτσι θα έχουμε μόνιμα την έξοδο ***(i,j) = (0,1)*** από το κύκλωμα. Τώρα ας δούμε την συμπεριφορά που θα έχει το κύκλωμα.



Εικόνα 18: Νέο Κύκλωμα CUT

Αφού έγινε η αλλαγή στο κύκλωμα, τώρα θα πρέπει να δούμε και την προσομοίωση του. Παρακάτω φαίνεται η κυματομορφή:



Εικόνα 19: Κυματομορφή Fault Injection

Παρατηρούμε πως η υπογραφή που παίρνει δεν είναι ίδια, καθώς οφείλεται στο γεγονός ότι η έξοδος του κυκλώματος είναι μόνιμα σταθερή. Ανεξαρτήτως των διανυσμάτων που εισάγονται στο κύκλωμα η απόκριση θα είναι πάντα λανθασμένη.

# APPENDIX:

Παρακάτω φαίνονται οι RTL περιγραφές των κυκλωμάτων με τα αντίστοιχα testbench.

## Άσκηση – 2.1:

Κ1: Κώδικας Κυκλώματος LSFR 8 bits:

`timescale 1ns/1ps

module LFSR\_8bit(

input clk,

output reg [7:0] LFSR = 255

);

wire feedback = LFSR[7];

always @(posedge clk)

begin

LFSR[0] <= feedback;

LFSR[1] <= LFSR[0];

LFSR[2] <= LFSR[1] ^ feedback;

LFSR[3] <= LFSR[2] ^ feedback;

LFSR[4] <= LFSR[3];

LFSR[5] <= LFSR[4];

LFSR[6] <= LFSR[5] ^ feedback;

LFSR[7] <= LFSR[6];

end

endmodule

Κ2: Κώδικας Testbench Κυκλώματος LSFR 8 bits:

`timescale 1ns/1ps

module LFSR\_8bittb();

reg CLKtb;

wire [7:0] LSFR\_output;

// Testbench starts here

//Block for clock generation

initial begin

CLKtb=0;

#20

forever begin

#10 CLKtb=!CLKtb;

end

end

//Instance model for lsfr

LFSR\_8bit lsfrInstance(CLKtb, LSFR\_output);

endmodule

Κ3: Κώδικας κυκλώματος TRCUTwithLSFR:

`timescale 1ns/1ps

module TRCUTwithLSFR(SE,CLK,SO);

// input signals

input SE, CLK;

// output signal

output SO;

// Necessary wires for connections between instances

wire SE, CLK, SO;

// Bus wire for lsfr circuit

wire [7:0] lsfr\_out;

// LSFR instance

LFSR\_8bit lsfrInstance(CLK, lsfr\_out);

// Single output from the LSFR as a serial input in TRCUT circuit

wire lsfr\_output = lsfr\_out[4];

// TRCUT instance

TRCUT trcutInstance(lsfr\_output, SE, CLK, SO);

endmodule

## Άσκηση – 2.2:

Κ4: Κώδικας Testbench κυκλώματος TRCUTwithLSFR:

`timescale 1ns/1ps

module TRCUTwithLSFRtb();

reg CLKtb, SEtb;

wire SOtb;

TRCUTwithLSFR dut(SEtb,CLKtb,SOtb);

//Block for clock generation

initial begin

CLKtb = 0;

#20

forever begin

#10 CLKtb =! CLKtb;

end

end

// Testbench starts here

initial begin

SEtb = 1;

#60 SEtb = 0;

#20 SEtb = 1;

#1200 SEtb = 0;

#20 SEtb = 1;

#1200 SEtb = 0;

#20 SEtb = 1;

#1200 SEtb = 0;

#20 SEtb = 1;

end

endmodule

## Άσκηση – 2.3:

Κ5: Κώδικας κυκλώματος MISR 16 bits:

`timescale 1ns/1ps

module MISR\_16bits(

input clk,

input misr\_input,

output reg [15:0] MISR = 65535

);

wire feedback = MISR[15];

always @(posedge clk)

begin

MISR[0] <= feedback;

MISR[1] <= MISR[0];

MISR[2] <= MISR[1];

MISR[3] <= MISR[2];

MISR[4] <= MISR[3] ^ feedback ^ misr\_input;

MISR[5] <= MISR[4];

MISR[6] <= MISR[5];

MISR[7] <= MISR[6];

MISR[8] <= MISR[7];

MISR[9] <= MISR[8] ^ feedback;

MISR[10] <= MISR[9];

MISR[11] <= MISR[10];

MISR[12] <= MISR[11];

MISR[13] <= MISR[12];

MISR[14] <= MISR[13] ^ feedback;

MISR[15] <= MISR[14];

end

endmodule

Κ6: Κώδικας κυκλώματος TRCUTwithMISR:

`timescale 1ns/1ps

module TRCUTwithMISR(SE,CLK,sign);

// input signals

input SE, CLK;

// output signal

output sign;

// Necessary wires for connections between instances

wire SE, CLK, SO, sign;

// Bus wire for lsfr circuit and misr circuit

wire [7:0] lsfr\_out;

wire [15:0] misr\_output;

// LSFR instance

LFSR\_8bit lsfrInstance(CLK, lsfr\_out);

// Single output from the LSFR as a serial input in TRCUT circuit

wire lsfr\_output = lsfr\_out[3];

// TRCUT instance

TRCUT trcutInstance(lsfr\_output, SE, CLK, SO);

// MISR instance

MISR\_16bits misrInstance(CLK, SO, misr\_output);

wire misr\_out = misr\_output[7];

assign sign = misr\_out;

endmodule

## Άσκηση – 2.4:

Κ7: Κώδικας Testbench κυκλώματος TRCUTwithMISR:

timescale 1ns/1ps

module TRCUTwithMISRtb();

reg CLKtb, SEtb;

wire signtb;

TRCUTwithMISR dut(SEtb,CLKtb,signtb);

//Block for clock generation

initial begin

CLKtb = 0;

#100

forever begin

#50 CLKtb =! CLKtb;

end

end

// Testbench starts here

initial begin

SEtb = 0;

#400 SEtb = 1;

#1400 SEtb = 0;

#100 SEtb = 1;

#1400 SEtb = 0;

#100 SEtb = 1;

#1400 SEtb = 0;

#100 SEtb = 1;

end

endmodule