Παναγιώτου Δήμητρα

AEM:2595

ECE 333

LAB 1 REPORT

Ο στόχος της πρώτης εργαστηριακής εργασίας είναι η υλοποίηση ενός οδηγού των τεσσάρων ενδείξεων 7-τμημάτων LED της πλακέτας ARTIX-7, για να επιτευχθεί η περιστροφική παρουσίαση ενός μηνύματος μεγέθους 16 χαρακτήρων.

PartA

LEDdecoder

ΣΚΟΠΟΣ: Στο πρώτο μέρος της εργασίας υλοποιήθηκε ένας decoder , ο οποίος για κάθε είσοδο char 4 bit παράγει μία έξοδο LED 7 bit η οποία οδηγεί κατάλληλα τα σήματα A,B,C,D,E,F,G του 7-segment-display.

ΥΛΟΠΟΙΗΣΗ:Η υλοποίηση του περιλαμβάνει ένα case statement το οποίο αναθέτει κατάλληλα τις τιμές όταν παρατηρείται αλλαγή στην τιμή του char.

PartB

Στο δεύτερο μέρος της εργασίας υλοποιήθηκε μία ολοκληρωμένη διάταξη FourDigitLEDdriver η οποία προβάλει ένα σταθερό μήνυμα τεσσάρων χαρακτήρων. Για να επιτευχθεί η λειτουργία αυτή δημιουργήθηκαν τέσσερα modules τα οποία με την σειρά είναι:

- ResetSynchronizer

  ΣΚΟΠΟΣ : Το κύκλωμα αυτό επιτυγχάνει το συγχρονισμό των σημάτων του clock και του reset.

  ΥΛΟΠΟΙΗΣΗ: Η υλοποίηση του περιλαμβάνει δύο always statements έτσι ώστε σε κάθε κύκλο ρολογιού να γίνεται ανάθεση του reset σε μια προσωρινή μεταβλητή και έπειτα στον επόμενο κύκλο να ανατίθεται η προσωρινή μεταβλητή στη νέα τιμή του reset. (Η προσωρινή μεταβλητή κρίνεται απαραίτητη για να αποφευχθούν φαινόμενα μεταστάθειας)

- Debouncer

  ΣΚΟΠΟΣ:Η διάταξη αυτή είναι υπεύθυνη για την τροφοδοσία του συστήματος μας με ακέραια συνεχή σήματα που προέρχονται από το reset καθώς και για την απόρριψη σημάτων που μπορεί να οφείλονται σε θόρυβο.

ΥΛΟΠΟΙΗΣΗ:Για την πραγματοποίηση αυτής της λειτουργίας χρησιμοποιείται ένας counter ο οποίος μετρά ένα ελάχιστο αριθμών κύκλων ρολογιού για τους οποίους ο παλμός πρέπει να κρατά την τιμή 1 .Όταν ο counter φτάσει στον ελάχιστο αριθμό κύκλων ρολογιού που ζητάμε τότε δημιουργούμε έναν νέο παλμό που περνάμε στα υπόλοιπα κυκλώματα/

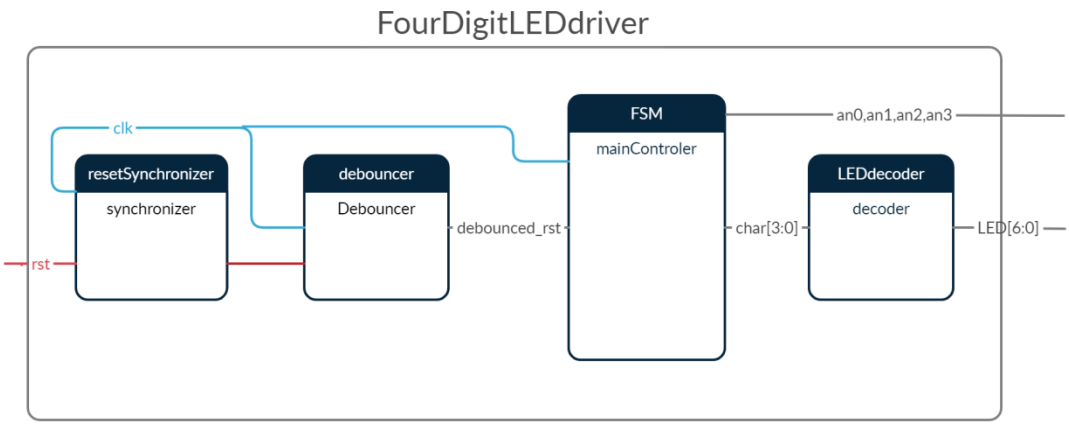
- FSM

ΣΚΟΠΟΣ: Σκοπός της fsm είναι να οδηγήσει το σύστημα στο κατάλληλο στάδιο για να επιτευχθεί μια λειτουργία. Για την ορθή λειτουργία του FourDigitLEDdriver είναι απαραίτητη η οδήγηση των ανόδων στο 0 (μετά το πέρας συγκεκριμένου χρόνου) ,η επιστροφή τους στην τιμή 1(όπου παραμένουν σβηστές) , αλλά και η αποθήκευση των δεδομένων που θα προβληθούν από την αντίστοιχη άνοδο. Για τον σκοπό αυτόν δημιουργείται ένα σήμα clk_enable με περίοδο 0.1ms.Σε μία πλήρη περίοδο αυτού του σήματος ανατίθονται δεδομένα στο char ή αλλάζουν οι τιμές των ανόδων για να επιτευχθεί η σωστή λειτουργία του συστήματος. Επίσης η fsm είναι υπεύθυνη για την αρχικοποίηση του συστήματος όταν το reset=1.

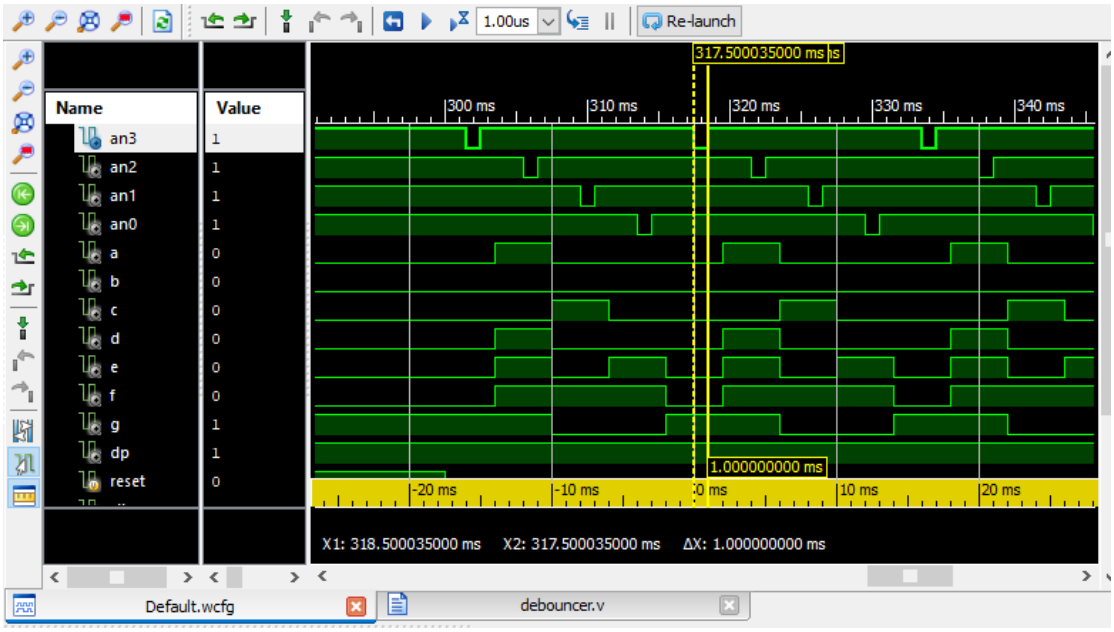
ΥΛΟΠΟΙΗΣΗ:Η υλοποίηση της fsm περιλαμβάνει ένα always statement που δημιουργεί το σήμα clk_enable με την βοήθεια ενός state_counter . Η μέγιστη τιμή που φτάνει ο state_counter στην συγκεκριμένη περίπτωση είναι ίση με 10000 εξαιτίας των προδιαγραφών της πλακέτας για την οδήγηση των ανόδων(απαιτείται ελάχιστο διάστημα 0,1ms και διαθέτουμε ρολόι με περίοδο 10ns).Επιπλέον χρησιμοποιείται ένα ακόμα always statement που ενεργοποιείται σε κάθε θετική ακμή του clk_enable και εμφωλευμένα περιλαμβάνεται ένα case statement που περιλαμβάνει τις αναθέσεις που πρέπει να κάνει το σύστημα στο κάθε στάδιο.

*Σημειώνεται ότι και στα δύο always statements εσωτερικά ελέγχεται η τιμή του reset και σε περίπτωση θετικού reset αρχικοποιούνται κατάλληλα οι μεταβλητές.

- LEDdecoder Όπως στο μέρος Α.



Dataflow diagram partB



Behavioral simulation partB

PartC

Στο τρίτο μέρος της εργασίας ,παρόμοια με το μέρος Β , υλοποιήθηκε μία ολοκληρωμένη διάταξη FourDigitLEDdriver η οποία προβάλει ένα μήνυμα το οποίο αλλάζει με το πάτημα ενός κουμπιού. Για να επιτευχθεί η λειτουργία αυτή δημιουργήθηκαν τέσσερα modules τα οποία είναι:

- ResetSynchronizer

ΣΚΟΠΟΣ: Το κύκλωμα αυτό επιτυγχάνει το συγχρονισμό των σημάτων του clock και του reset.

ΥΛΟΠΟΙΗΣΗ: Η υλοποίηση του περιλαμβάνει δύο always statements έτσι ώστε σε κάθε κύκλο ρολογιού να γίνεται ανάθεση του reset σε μια προσωρινή μεταβλητή και έπειτα στον επόμενο κύκλο να ανατίθεται η προσωρινή μεταβλητή στη νέα τιμή του reset. (Η προσωρινή μεταβλητή κρίνεται απαραίτητη για να αποφευχθούν φαινόμενα μεταστάθειας)

- Debouncer

ΣΚΟΠΟΣ: Η διάταξη αυτή είναι υπεύθυνη για την τροφοδοσία του συστήματος μας με ακέραια συνεχή σήματα που προέρχονται από το button και για την απόρριψη σημάτων που μπορεί να οφείλονται σε θόρυβο.

ΥΛΟΠΟΙΗΣΗ: Για την πραγματοποίηση αυτής της λειτουργίας χρησιμοποιείται ένας counter ο οποίος μετρά έναν ελάχιστο αριθμών κύκλων ρολογιού για τους οποίους ο παλμός πρέπει να κρατά την τιμή 1 .Όταν ο counter φτάσει τον μέγιστο αριθμό κύκλων ρολογιού που ζητάμε τότε δημιουργούμε ένα νέο παλμό που περνάμε στα υπόλοιπα κυκλώματα .Διατηρήσουμε τον νέο παλμό όσους κύκλους ρολογιού ήταν και ο αρχικός, μειώνοντας απλά την τιμή του μετρητή.

- FSM

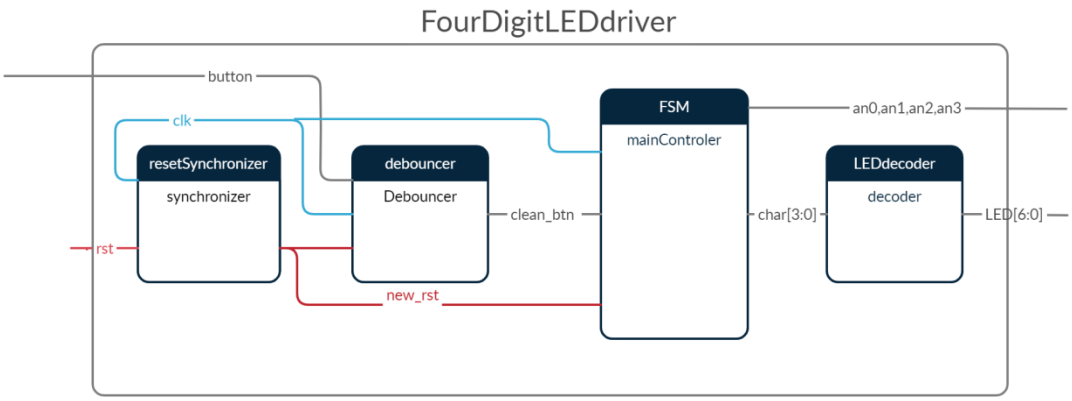
  ΣΚΟΠΟΣ: Σκοπός της fsm είναι να οδηγήσει το σύστημα στο κατάλληλο στάδιο για να επιτευχθεί μια λειτουργία. Για το FourDigitLEDdriver είναι απαραίτητο να δημιουργείται ένα σήμα clk_enable (με περίοδο που εξαρτάται από τα χαρακτηριστικά της πλακέτας).Σε μία πλήρη περίοδο αυτού του σήματος αναθέτονται δεδομένα στο char ή αλλάζουν οι τιμές των ανόδων για να οδηγηθεί σωστά το σύστημα. Επίσης η fsm είναι υπεύθυνη για την αρχικοποίηση του συστήματος όταν το reset=1. Στο στάδιο αυτό είναι απαραίτητο να εξασφαλίσουμε την αλλαγή των δεδομένων που αναθέτονται στο reg char , όταν πατάμε το button, για τον λόγο αυτό χρησιμοποιούμε 4 μεταβλητές indexAn3, indexAn2, indexAn1, indexAn0 η οποίες δείχνουν στα κατάλληλα στοιχεία της μνήμης.

  ΥΛΟΠΟΙΗΣΗ: Η υλοποίηση της fsm περιλαμβάνει ένα always statement που δημιουργεί το σήμα clk_enable με την βοήθεια ενός state_counter όπως χρησιμοποιήθηκε στο μέρος Β. Στο ίδιο statement για τις περιπτώσεις που το reset είναι 0 γίνεται ένας έλεγχος για να ανιχνευθούν μεταβολές στην τιμή του button . Στην περίπτωση που εντοπιστεί θετικός παλμός η τιμές των μεταβλητών tempAn_ αυξάνονται. Παράλληλα , η υλοποίηση του module απαιτεί την χρήση ενός ακόμα always statement που ενεργοποιείται σε κάθε θετική ακμή του clk_enable και περιέχει εσωτερικά ένα case statement που

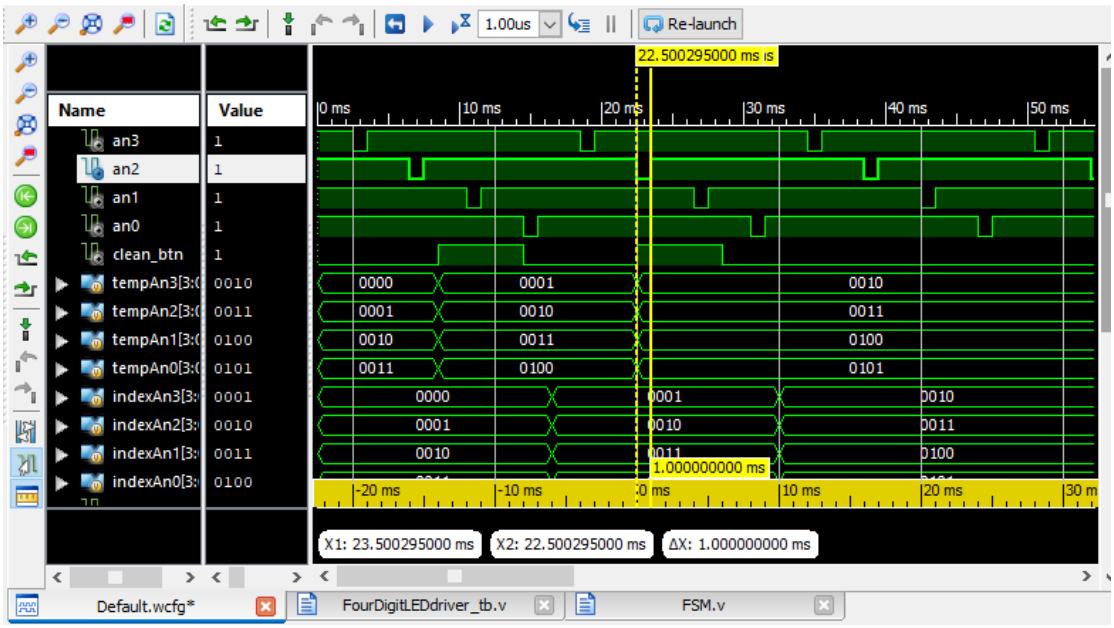
περιλαμβάνει τις αναθέσεις που πρέπει να κάνει το σύστημα στο κάθε στάδιο.

*Το σήμα clk_enable υλοποιείται με ένα μέτρητη που για τις μισές τιμές του κρατάει τον παλμό χαμηλά και για τις υπόλοιπες τον ανεβάζει.
**Σημειώνεται ότι και στα δύο always statements εσωτερικά ελέγχεται η τιμή του reset και σε περίπτωση θετικού reset αρχικοποιούνται κατάλληλα οι μεταβλητές.
***Για να αποφύγουμε την προβολή αριθμών που δεν βρίσκονται σε διαδοχικές θέσεις , συμπεριφορά που οφείλεται στην αλλαγή των index values σε κάποια τυχαία τιμή του state counter χρησιμοποιούμε προσωρινές μεταβλητές οι οποίες μεταβάλλονται με το πάτημα του κουμπιού αλλά αναθέτονται στα indexAn_ όταν ο state_counter πάρει την τιμή 0(initial state).

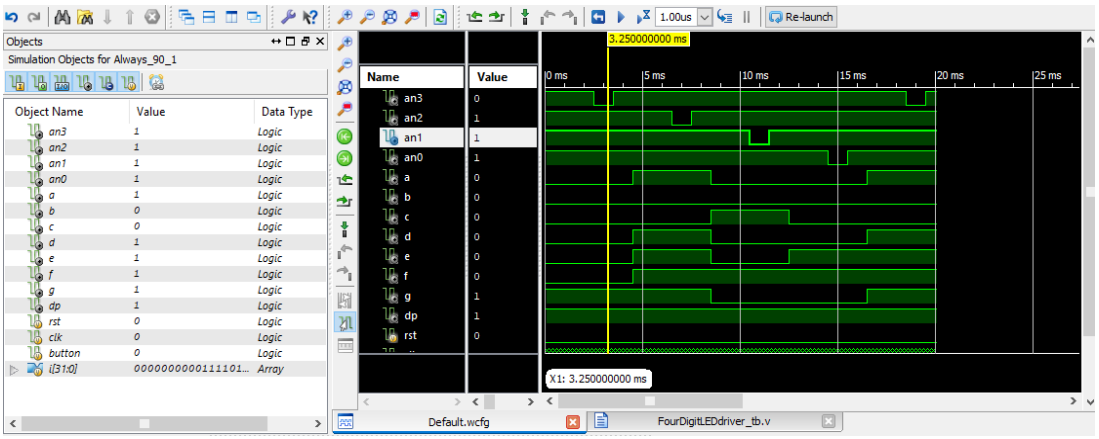
- LEDdecoder
  Όπως στο μέρος Α.



Dataflow diagram partC



Behavioral simulation partC



Post and route simulation partC

PartD

Στο τέταρτο μέρος της εργασίας ,παρόμοια με το μέρος Β,Γ , υλοποιήθηκε μία ολοκληρωμένη διάταξη FourDigitLEDdriver η οποία προβάλει ένα μήνυμα το οποίο αλλάζει μετά από κάποιο χρόνο . Για να επιτευχθεί η λειτουργία αυτή δημιουργήθηκαν τέσσερα modules τα οποία είναι:

- ResetSynchronizer
  ΣΚΟΠΟΣ : Το κύκλωμα αυτό επιτυγχάνει το συγχρονισμό των σημάτων του clock και του reset.

ΥΛΟΠΟΙΗΣΗ: Η υλοποίηση του περιλαμβάνει δύο always statements έτσι ώστε σε κάθε κύκλο ρολογιού να γίνεται ανάθεση του reset σε μια προσωρινή μεταβλητή και έπειτα στον επόμενο κύκλο να ανατίθεται η προσωρινή μεταβλητή στη νέα τιμή του reset. (Η προσωρινή μεταβλητή κρίνεται απαραίτητη για να αποφευχθούν φαινόμενα μεταστάθειας)

- Debouncer

  ΣΚΟΠΟΣ: Η διάταξη αυτή είναι υπεύθυνη για την τροφοδοσία του συστήματος μας με ακέραια συνεχή σήματα που προέρχονται από το reset και για την απόρριψη σημάτων που μπορεί να οφείλονται σε θόρυβο.

  ΥΛΟΠΟΙΗΣΗ:Για την πραγματοποίηση αυτής της λειτουργίας χρησιμοποιείται ένας counter ο οποίος μετρά ένα ελάχιστο αριθμών κύκλων ρολογιού για τους οποίους ο παλμός πρέπει να κρατά την τιμή 1 .Όταν ο counter φτάσει τον ελάχιστο αριθμό κύκλων ρολογιού που ζητάμε τότε δημιουργούμε έναν νέο παλμό τον οποίο και χρησιμοποιούμε στα υπόλοιπα κυκλώματα .

- FSM

  ΣΚΟΠΟΣ: Σκοπός της fsm είναι να οδηγήσει το σύστημα στο κατάλληλο στάδιο για να επιτευχθεί μια λειτουργία. Για το FourDigitLEDdriver είναι απαραίτητο να δημιουργείται ένα σήμα clk_enable (με περίοδο που εξαρτάται από τα χαρακτηριστικά της πλακέτας).Σε μία πλήρη περίοδο αυτού του σήματος αναθέτονται δεδομένα στο char ή αλλάζουν οι τιμές των ανόδων για να οδηγηθεί σωστά το σύστημα. Επίσης η fsm είναι υπεύθυνη για την αρχικοποίηση του συστήματος όταν το reset=1. Στο στάδιο αυτό είναι απαραίτητο να εξασφαλίσουμε την αλλαγή των δεδομένων που αναθέτονται στο reg char ,μετά από πεπερασμένο χρονικό διάστημα .

  ΥΛΟΠΟΙΗΣΗ: Η υλοποίηση της fsm περιλαμβάνει ένα always statement που δημιουργεί το σήμα clk_enable με την βοήθεια ενός state_counter όπως χρησιμοποιήθηκε στο μέρος Β , Γ. Στο ίδιο statement για τις περιπτώσεις που το reset είναι 0 χρησιμοποιείται ένας μετρητής del_counter ο οποίος μετράει μέχρι έναν συγκεκριμένο αριθμό(22'b1) και έπειτα αλλάζει τις τιμές των tempAn_ ώστε να δείχνου στα κατάλληλα στοιχεία του mem. Παράλληλα , η υλοποίηση του module απαιτεί την χρήση ενός ακόμα always statement που ενεργοποιείται σε κάθε θετική ακμή του clk_enable και περιέχει εσωτερικά ένα case statement που περιλαμβάνει τις αναθέσεις που πρέπει να κάνει το σύστημα στο κάθε στάδιο.

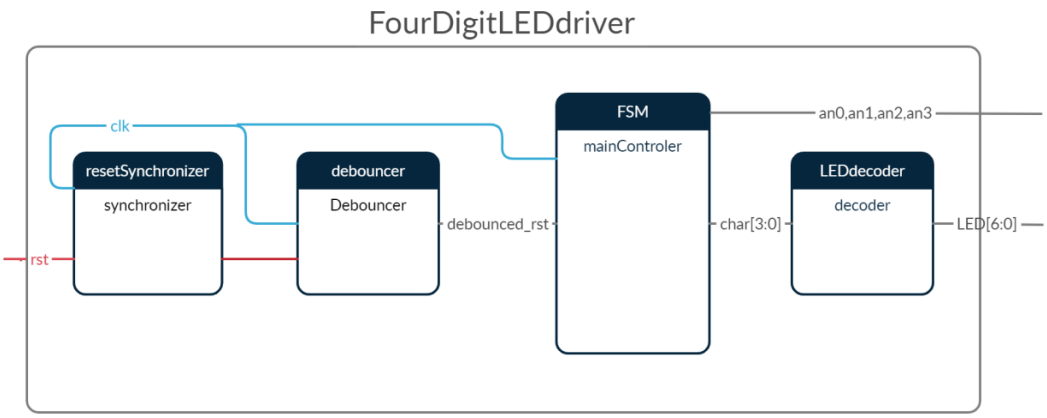
  *Το σήμα clk_enable υλοποιείται με ένα μέτρητη που για τις μισές τιμές του κρατάει τον παλμό χαμηλά και για τις υπόλοιπες τον ανεβάζει.

**Σημειώνεται ότι και στα δύο always statements εσωτερικά ελέγχεται η τιμή του reset και σε περίπτωση θετικού reset αρχικοποιούνται κατάλληλα οι μεταβλητές.

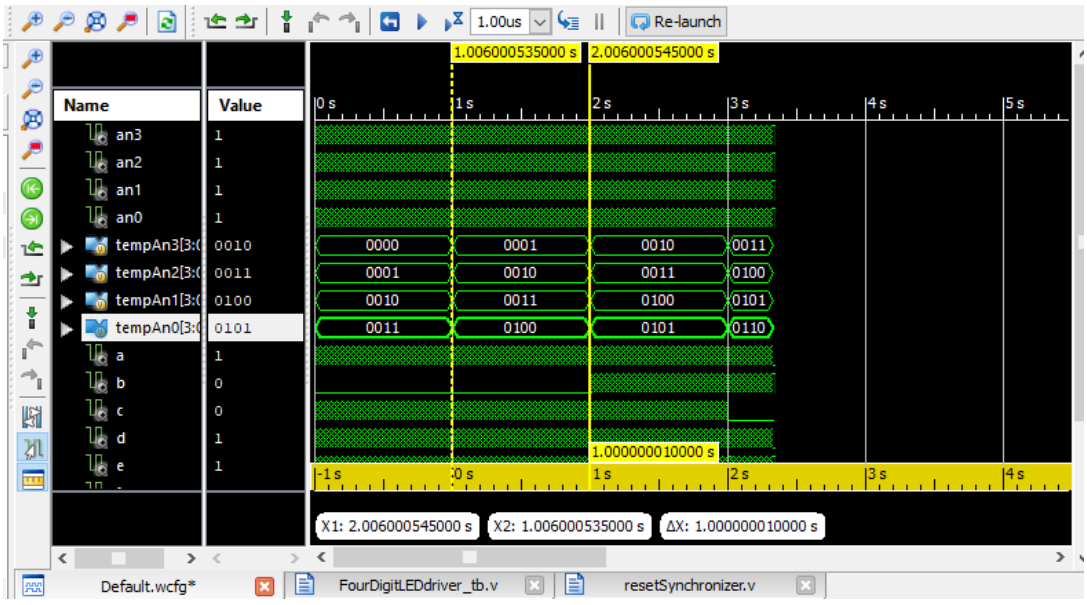
***Για να αποφύγουμε την προβολή αριθμών που δεν βρίσκονται σε διαδοχικές θέσεις , συμπεριφορά που οφείλεται στην αλλαγή των index values σε κάποια τυχαία τιμή του state counter χρησιμοποιούμε προσωρινές μεταβλητές οι οποίες μεταβάλλονται με το πάτημα του κουμπιού αλλά αναθέτονται στα index όταν ο state counter πάρει την τιμή 0(initial state).

- LEDdecoder

  Όπως στο μέρος Α.



Dataflow diagram partD



Behavioral simulation partD