ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ Η/Υ

ΜΑΘΗΜΑ: ΕΡΓΑΣΤΗΡΙΟ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΝΑΦΟΡΑ

Τίτλος: Παπαναστασίου Στέφανος 1608,31 Οκτωβρίου 2015

Περίληψη: Δημιουργία ενός οδηγού των τεσσάρων ενδείξεων 7-τμημάτων LED της πλακέτας Spartan 3 για την εμφάνιση ενός μηνύματος 16 χαρακτήρων

Εισαγωγή: Οι στόχοι της εργασίας αρχικά ήταν 3:

1. Εμφάνιση Σταθερού Μηνύματος 4 ψηφίων

2. Εμφάνιση Μηνύματος και Περιστροφή του κατά ένα γράμμα με το πάτημα κουμπιού

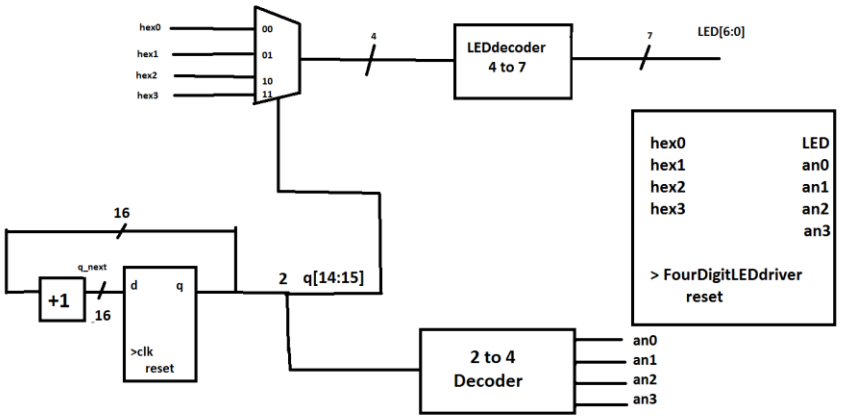
3. Εμφάνιση Μηνύματος και Περιστροφή του κατά ένα γράμμα μετά από συγκεκριμένη καθυστέρηση ενός δευτερολέπτου

Όλοι οι στόχοι υλοποιήθηκαν σταδιακά. Ο πρώτος αποτελούσε βασικό βήμα για τους δύο επόμενους. Πρώτα υλοποιήθηκε το κύκλωμα και επαληθεύτηκε βάση της προσομοίωσης. Η υλοποίηση των 2 τελευταίων στόχων είναι παρόμοια και γι' αυτό το λόγο η επίτευξη του 3ου στόχου έγινε άμεσα μετά την διόρθωση του 2ου

ΜΕΡΟΣ 1 << Υλοποίηση αποκωδικοποιητή 7-τμημάτων και οδήγηση τεσσάρων ψηφίων>>

Υλοποίηση : Σε αυτό το μέρος χρησιμοποιήθηκε αρχικά το LEDdecoder module το οποίο ήταν ο αποκωδικοποιητής, ο οποίος έπαιρνε σαν είσοδο 4bit και επέστρεφε ένα 7bit bus με τις κατάλληλες τιμές των σημάτων (a,b,c,d,e,f,g) των led της fpga τα οποία ρύθμιζε το module FourDigitLEDdriver σε ένα always. Για να εμφανιστεί σωστά το μήνυμα όμως έπρεπε να γίνει η σωστή οδήγηση των ανόδων, δηλαδή να γίνει η κατάλληλη ανάθεση των σημάτων των 4 LED. Συγκεκριμένα χρησιμοποιήθηκε ένας μετρητής 16 bit . Η οδήγηση των ανόδων γινόταν με έλεγχο των δύο σημαντικότερων bit του μετρητή, δίνοντας το περιθώριο στο ανθρώπινο μάτι να δει την αλλαγή των LED. Ο μετρητής ρύθμιζε επίσης και το περιεχόμενο του κάθε LED.Επίσης χρησιμοποιήθηκε μια μονάδα DCM που είχε ως στόχο να προσαρμόσει το κύκλωμα με το ρολόι της πλακέτας .

Επαλήθευση: Για την υλοποίηση σε επίπεδο συμπεριφοράς χρησιμοποιήθηκε ένα testbench που αρχικοποιούσε κατάλληλα τα 4 LED της πλακέτας. Σύμφωνα με την προσομοίωση οι άνοιδοι οδηγούνταν σωστά όπως και τα σήματα των led. Βέβαια χρησιμοποιήθηκε μικρότερος μετρητής , καθώς ο μηδενισμός 16bit μετρητή απαιτεί πολύ μεγάλο χρόνο προσομοίωσης κάτι που δεν είναι πρακτικό στην επαλήθευση του κυκλώματος. Παρακάτω παρατίθεται η σχεδίαση του κυκλώματος στο χαρτί.



Πείραμα/Τελική Υλοποίηση : Για την υλοποίηση του κυκλώματος στην πλακέτα έγινε η αλλαγή του μετρητή σε 16bit , για τον λόγο που αναφέρθηκε παραπάνω. Αφού δημιουργήθηκε το bit file έτρεξα το πρόγραμμα στην πλακέτα. Το μόνο πρόγραμμα που παρουσιάστηκε ήταν ότι στην αρχή τα ψηφία φαινόταν αχνά. Αυτό γινόταν γιατί ο κώδικας περιείχε αρκετά always@* blocks , κάτι που αποδείχθηκε κακή πρακτική. Μετά την τροποποίηση του κώδικα και την παραγωγή του καινούριου bit file το πρόγραμμα δούλεψε σωστά.

ΜΕΡΟΣ 2 << Βηματική Περιστροφή του Μηνύματος με Χρήση Κουμπιού>>

Υλοποίηση : Για την βηματική περιστροφή του μηνύματος απαιτούνταν η χρήση μνήμης. Αρχικά χρησιμοποιήθηκαν 2 32bit bus που τα έκανα 2 πίνακες 8γραμμές X 4 στήλες ,όπου η κάθε γραμμή περιείχε ένα ψηφίο.

Κώδικας Verilog:

```
wire [31:0] message1,message2;

wire [3:0]array1[7:0];

wire [3:0]array2[7:0];

assign message1=32'b00000001001000110100010101100111;

assign message2=32'b10001001101010111100110111101111;

assign {array1[0],array1[1],array1[2],array1[3],array1[4],array1[5],array1[6],array1[7]}=message1;

assign {array2[0],array2[1],array2[2],array2[3],array2[4],array2[5],array2[6],array2[7]}=message2;
```

Το message1 περιείχε τους χαρακτήρες 0,1,2,3,4,5,6,7 κα το message2 τους χαρακτήρες 8,9,a,b,c,d,e,f που χωρίστηκαν κατάλληλα στους δύο πίνακες. Για την περιστροφή χρησιμοποιήθηκε μετρητής 4bit ο οποίος είχε τον ρόλο του δείκτη στην μνήμη. Αρχικοποιούνταν στην τιμή 1111 και κάθε φορά που πατιόνταν το κουμπί αυξανόταν κατά 1. Επίσης υλοποιήθηκε το module Debounce για να αποφευχθούν οι αναπηδήσεις του κουμπιού. Το κύκλωμα του Debounce είναι επίσης ένας μετρητής 20bit ο οποίος ελέγχει πόσες φορές πατήθηκε το κουμπί .

Επαλήθευση: Για την υλοποίηση σε επίπεδο συμπεριφοράς χρησιμοποιήθηκε ένα testbench που αναπαραστούσε το πάτημα του κουμπού δύο φορές. Μετά την προσομοίωση επαληθεύτηκε η περιστροφή του μηνύματος. Το επιθυμητό αποτέλεσμα ήταν η ολική περιστροφή του μηνύματος κάτι που επαληθεύθηκε από τις κυματομορφές.

Το αποτέλεσμα ήταν :

0123 ,1234,2345,3456,4567,5678,6789,789a,89ab,9abc,abcd,bcde,cdef,def0,ef01,f012,0123

Πείραμα/Τελική Υλοποίηση: Το πρόβλημα που παρουσιάστηκε ήταν με τον κύκλωμα του debouncer. Επειδή δεν μπορούσα να το επαληθεύσω μέσω προσομοίωσης παρουσιάστηκαν προβλήματα καθώς με το πάτημα του κουμπιού το μήνυμα περιστρεφόταν κατά 2,3,4 θέσεις. Μετά την λεπτομερή εξέταση και τροποποίηση του κώδικα του συγκεκριμένου module ήρθε το επιθυμητό αποτέλεσμα. Πλέον το μήνυμα περιστρεφόταν κάθε φορά κατά μία θέση.

ΜΕΡΟΣ 3 << Βηματική Περιστροφή του Μηνύματος με σταθερή Καθυστέρηση>>

Υλοποίηση : Για την βηματική περιστροφή του μηνύματος με σταθερή Καθυστέρηση χρησιμοιήθηκε ένα always block που παρίστανε τον 23bit μετρητή. Κάθε φορά που μηδενίζονταν το σήμα m_tick γινόταν 1 και ενεργοποιούσε τον μηχανισμό περιστροφής του μηνύματος κατά μία θέση(σαν να πατιόταν το κουμπί). Για τον μηδενισμό απαιτείται το πέρασμα περίπου 1,4 δευτερολέπτου.

Κώδικας Verilog:

```
always@(posedge clk_dv,posedge reset)
        if(reset)
                q3<=0;
        else
                q3<=q3_next;

assign q3_next=q3+1;
assign m_tick = (q3==0) ? 1'b1 : 1'b0;
```

Επαλήθευση: Για την υλοποίηση σε επίπεδο συμπεριφοράς χρησιμοποιήθηκε ένα testbench που απλώς έτρεχε το κύκλωμα με βάση το ρολόι. Χρησιμοποιήθηκε μικρότερος μετρητής καθώς ο μηδενισμός 23bit μετρητή απαιτεί μεγάλο χρόνο προσομοίωσης. Η ολική περιστροφή του μηνύματος επαληθεύτηκε από τις κυματομορφές , όπως και στο παραπάνω μέρος.

Πείραμα/Τελική Υλοποίηση: Δεν παρουσιάστηκε κάποιο εμπόδιο ,αφού το κύκλωμα δούλεψε αμέσως σωστά. Το στάδιο αυτό ήταν σε μεγάλο βαθμό βασισμένο στο παραπάνω μέρος , επομένως το αποτέλεσμα ήταν αναμενόμενο. Χρησιμοποιήθηκε ο μετρητής 23 bit σε αντίθεση με την προσομοίωση.

Συμπεράσματα : Σε γενικές γραμμές η πορεία που ακολουθήθηκε αποτελούνταν από τα παραπάνω μέρη. Σχεδίαση και επαλήθευση στον προσομοιωτή (διόρθωση οποιουδήποτε προβλήματος) και στη συνέχεια υλοποίηση του πειράματος στην πλακέτα. Το μόνο απρόσμενο πρόβλημα που παρουσιάστηκε ήταν με το κύκλωμα αναπήδησης καθώς δεν μπορούσε να ελεγχθεί χωρίς την πλακέτα

ΙΣΤΟΡΙΚΟ

Ημερομηνία	**ΚΩΔΙΚΑΣ**	**ΣΧΟΛΙΑ**
19/10/15	LEDdecoder.v	Ολοκλήρωση πρώτου μέρους και επαλήθευση
20/10/15	FourDigitLEDdriver.v	Ολοκλήρωση δεύτερου μέρους και επαλήθευση. Εμφάνιση σταθερού μηνύματος.
22/10/15	FourDigitLEDdriver.v	Ολοκλήρωση τρίτου μέρους και επαλήθευση. Εμφάνιση περιστρεφόμενου μηνύματος με χρήση κουμπιού.
23/10/15	FourDigitLEDdriver.v Debounce.v	UPDATE τρίτου μέρους και δημιουργία Deboucer
26/10/15	FourDigitLEDdriver.v	Ολοκλήρωση τέταρτου μέρους και επαλήθευση. Εμφάνιση περιστρεφόμενου μηνύματος με σταθερή καθυστέρηση.