Οδηγός Ένδειξης 7-τμημάτων

Εργαστήριο Ψηφιακών Κυκλωμάτων (2023-24)

Ιωάννης Αθανασιάδης 03491

09/11/2023

Περίληψη

Αναφορά για την εργασία του μαθήματος του Εργαστηρίου Ψηφιακών Κυκλωμάτων (ECE333), μέσω της οποία γίνεται ανάλυση των μεθόδων ανάπτυξης και *debugging* ενός *RTL design* στα πλαίσια του προγράμματος ***Xilinx Vivado*** και της ***Digilent Nexys A7-100T*** *FPGA*. Για να το κάνουμε αυτό αναλύουμε με διάφορους μεθόδους (όπως σχήματα ροής δεδομένων) τις κυκλωματική υλοποίηση της ***Verilog*** που αποτελούνε την εργασίας.

Εισαγωγή

Ο στόχος της εργασίας ήταν η οδήγηση μιας τετραψήφιας οθόνης 7-τμημάτων που είναι ενσωματωμένη στην *Nexys A7-100T*. Πιο αναλυτικά η περιστροφική παρουσίαση ενός μηνύματος ακριβώς 16 χαρακτήρων. Η περιστροφή θα λειτουργεί είτε με το πάτημα ενός κουμπιού είτε μετά από ένα χρονικό διάστημα, κάνοντας ολίσθηση προς τα δεξιά σε κάθε περίπτωση. Η εργασία θεωρήθηκε επιτυχημένη αφού όλα τα μέρη που την αποτελούν όπως και οι στόχοι που αναφέραμε παραπάνω ολοκληρώθηκαν με επιτυχία.

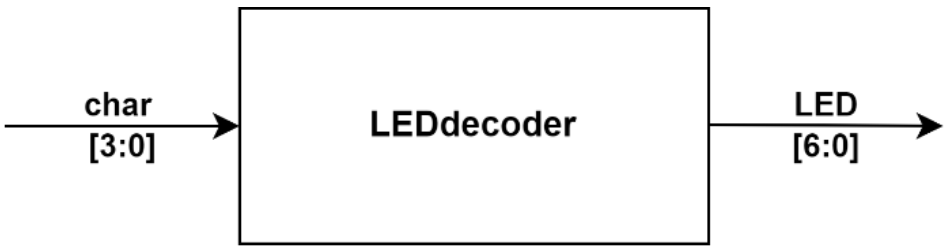
Μέρος Α – Υλοποίηση Αποκωδικοποιητή 7-τμημάτων

Το πρώτο μέρος της εργασίας είναι αρκετά απλό μιας και είναι η υλοποίηση ενός απλού αποκωδικοποιητή μέσα στο module LEDdecoder.

Για την αντιστοιχία των τιμών εισόδων/εξόδων χρησιμοποιήθηκε ο παρακάτω πίνακας στον οποίο αντιστοιχήζεται κάθε τιμή ενός μονοψήφιου δεκαεξαδικού αριθμού με την αντίστοιχη εμφάνηση του στην οθόνη, δηλαδή όταν:

$$char = 0x9 \ \rightarrow \tau o \ \varepsilon v \ \lambda \acute{o} \gamma \omega \ \psi \eta \varphi \acute{\iota} o \ \tau \eta \varsigma \ o \theta \acute{o} v \eta \varsigma \ \delta \varepsilon \acute{\iota} \chi v \varepsilon \iota \ 9$$

char	LED
0x0	0000001
0x1	1001111
0x2	0010010
0x3	0000110
0x4	1001100
0x5	0100100
0x6	0100000
0x7	0001111
0x8	0000000
0x9	0000100
0xA	0001000
0xB	1100000
0xC	0110001
0xD	1000010
0xE	0110000
0xF	0111000



Σημείωση: παρόλο που η εκφώνηση μιλάει για οδήγηση του κάθε τμήματος της οθόνης στο 1 ισχύει το αντίθετο, έτσι χρειάστηκε να βρω τους αντιστρόφους του τιμών του LED σε κάθε περίπτωση.

```
initial begin
        char = 4'h0;
    #10 char = 4'h1;
    #10 char = 4'h2;
    #10 char = 4'h3;
    #10 char = 4'h4;
    #10 char = 4'h5;
    #10 char = 4'h6;
    #10 char = 4'h7;
    #10 char = 4'h8;
    #10 char = 4'h9;
    #10 char = 4'ha;
    #10 char = 4'hb;
    #10 char = 4'hc;
    #10 char = 4'hd;
    #10 char = 4'he;
    #10 char = 4'hf;
    #10 $finish;
end
```

Επαλήθευση

Το πλαίσιο δοκιμών του πρώτου μέρους είναι αρκετά απλό γιατί το κύκλωμα μας είναι συνδυαστικό οπότε δεν χρειάζονται σήματα όπως *ρολόι* και *reset* .

Τα διανύσματα που χρησιμοποιήθηκαν είναι πρακτικά όλοι οι συνδυασμοί ενός *4-bit* αριθμού, δηλαδή *από 0 έως 16 (δεκαδικό).* Το testbench θα μπορούσαμε να χρησιμοποιήσουμε μία *for-loop* αλλά τα διανύσματα προς δοκιμή θεωρήθηκαν λίγα ώστε να είναι εύκολη η γραφή με τον τρόπο που φαίνεται στο πλαίσιο στα δεξιά (*χειροκίνητα*).

Μέρος Β – Οδήγηση Τεσσάρων Ψηφίων

Το ζητούμενο στο δεύτερο μέρος της εργασίας είναι η οδήγηση και των τεσσάρων ψηφίων που αποτελούν την οθόνη μας.

Για την οδήγηση του κάθε ψηφίου χρησιμοποιούνται 4 είσοδοι (*AN3, AN2, AN1, AN0*) μέσω των οποίων επιλέγεται ένα ψηφίο την φορά (*που οδηγείται στο 0*). Δηλαδή όταν:

$$\{AN3, AN2, AN1, AN0\} = 4'b0100 \rightarrow οδηγείται\ 2^o\ πιο\ σημαντικό\ ψηφίο$$

Για top-level μονάδα έχουμε το FourDigitLEDdriver: