Πανεπιστήμιο Πατρών

Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών

Εργαστήριο Σχεδίασης Ολοκληρωμένων Κυκλωμάτων

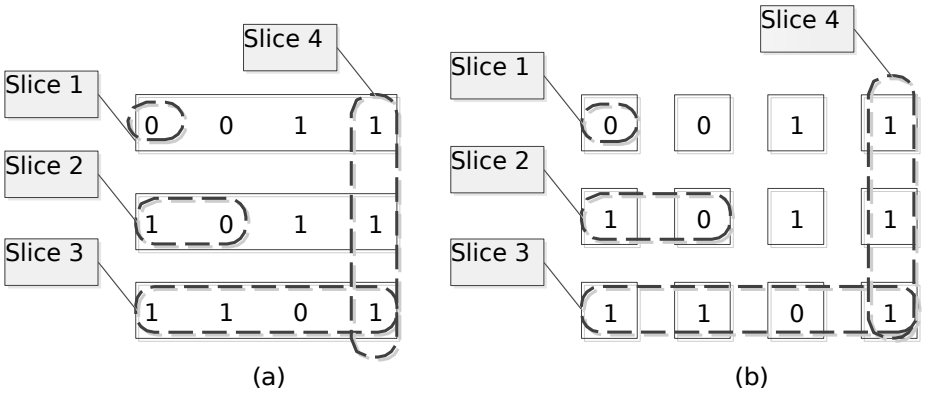
Σχεδιασμός Ολοκληρωμένων Συστημάτων

Χειμερινό Εξάμηνο 2025-2026

Εργαστήριο 1

Άσκηση 1: Δεικτοδότηση σε ARRAYS

Θεωρείστε τα παρακάτω x1D (Σχήμα α) και 2D (Σχήμα β) ARRAYS.



Α. Για κάθε ARRAY, γράψτε κώδικα VHDL που να αρχικοποιεί τις τιμές του (όχι απαραίτητα σε αυτές του σχήματος) και να επιστρέφει τις τιμές των στοιχείων των ARRAYS που καθορίζονται από τα slices που φαίνονται στο σχήμα.

Συγκεκριμένα, για το σχήμα (α)

- το slice 1 αντιστοιχεί σε <u>ένα οποιοδήποτε ψηφίο οποιασδήποτε γραμμής</u>,
- το slice 2 αντιστοιχεί στα <u>δύο αριστερά ψηφία οποιασδήποτε γραμμής</u>,
- το slice 3 αντιστοιχεί στα <u>4 ψηφία οποιασδήποτε γραμμής</u>,
- το slice 4 αντιστοιχεί <u>σε μία στήλη με 4 ψηφία</u>

Ομοίως για το Σχήμα β.

<u>Καθορίστε με σαφήνεια τη διεπαφή (είσοδοι /έξοδοι) του κυκλώματος.</u>

Άσκηση 2: Προγραμματιζόμενος κωδικοποιητής προτεραιότητας

Σχεδιάστε ένα κύκλωμα κωδικοποιητή με προγραμματιζόμενες προτεραιότητες. Το κύκλωμα πρέπει να κωδικοποιεί τη διεύθυνση του υψηλότερης τάξης ψηφίου της εισόδου που είναι σε υψηλή στάθμη (δηλαδή, ‘1’). Το κύκλωμα έχει μία είσοδο r (8-bit), μία είσοδο c (3-bit) και μια έξοδο code (3-bit).

Η προτεραιότητα καθορίζεται ανάλογα με την τιμή του σήματος c. Δηλαδή αν c = “111” τότε οι προτεραιότητες έχουν ως εξής r(7), r(6), r(5). Δηλαδή, το ψηφίο (r7) έχει τη μεγαλύτερη προτεραιότητα, το r(6) την αμέσως μεγαλύτερη προτεραιότητα κοκ.. Αν c = “011” τότε οι προτεραιότητες έχουν ως εξής r(3), r(2), r(1) r(0), r(7), r(6)...., δηλαδή το ψηφίο r(3) έχει τη μεγαλύτερη προτεραιότητα, το r(2) την αμέσως μεγαλύτερη προτεραιότητα κοκ..

Παραδείγματα:

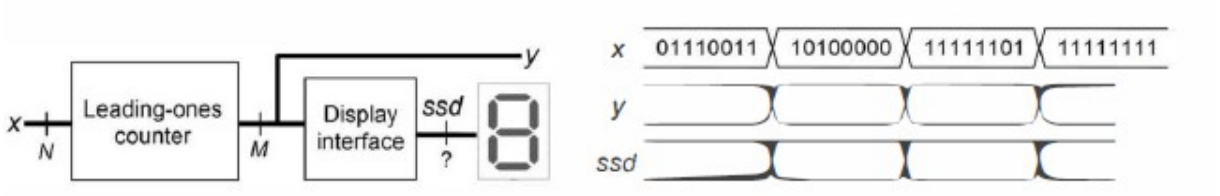
r = “11001000”, c = “111”, τότε code = “111”
r = “01001000” , c = “111”, τότε code = “110”
r = “01001000” , c = “100”, τότε code = “011”
r = “01001000” , c = “010”, τότε code = “110”
r = “01001001” , c = “000”, τότε code = “000”

Χρησιμοποιήστε επίσης, ένα σήμα εξόδου (active) το οποίο θα ελέγχει αν η είσοδος r έχει κάποιο bit ενεργό. Στην περίπτωση που r= “00000000” τότε active = ‘0’ και η έξοδος y μπορεί να πάρει οποιαδήποτε τιμή.

<u>Δώστε αν μπορείτε περισσότερες από μία υλοποιήσεις.</u>

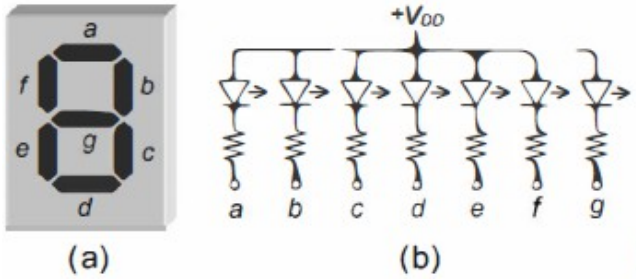
Άσκηση 3: Μετρητής συνεχόμενων 1s και οδήγηση οθόνης επτά τμημάτων

Το κύκλωμα πρέπει να μετρά το πλήθος των συνεχόμενων άσσων που υπάρχουν στην είσοδο x <u>πριν την εμφάνιση του πρώτου ψηφίου με μηδενική τιμή</u>, ξεκινώντας από το MSB. Το κύκλωμα έχει δύο εξόδους. Η πρώτη έξοδος *y* είναι η δυαδική αναπαράσταση του πλήθους των συνεχόμενων 1 που μετρήθηκαν, ενώ η δεύτερη έξοδος *ssd* οδηγεί μία οθόνη επτά τμημάτων.



(a) (b)

Σχ1: (a) Κυκλωματικό διάγραμμα (b) Κυμματομορφές προσομοίωσης Η οδήγηση της οθόνης επτά τμημάτων εξηγείται στο παρακάτω σχήμα.



(c)

char	*abcdefg*	char	*abcdefg*
0	0000001	8	0000000
1	1001111	9	0000100
2	0010010	A	0001000
3	0000110	b	1100000
4	1001100	C	0110001
5	0100100	d	1000010
6	0100000	E	0110000
7	0001111	F	0111000

Σχ2: Οδήγηση οθόνης επτά τμημάτων

Α. Όταν η είσοδος x είναι των 8 ψηφίων (N=8), ποιο είναι το πλήθος των ψηφίων M της εξόδου y και ποιο είναι το πλήθος των ψηφίων της εξόδου ssd;

Β. Γράψτε κώδικα VHDL που να μοντελοποιεί τη λειτουργία του κυκλώματος όπου όλα τα σήματα εισόδου και εξόδου είναι τύπου std_logic. Ο κώδικας πρέπει να είναι παραμετρικός ως προς το πλήθος των ψηφίων N της εισόδου x.

Γ. Προσομοιώστε τον κώδικα σας για N = 8 χρησιμοποιώντας τις εισόδους του Σχ 1(β).

Άσκηση 4 Μέγιστο πλήθος συνεχόμενων 1s και οδήγηση οθόνης επτά τμημάτων

Θεωρήστε το κύκλωμα της προηγούμενης άσκησης. Όμως αυτή τη φορά η έξοδος y πρέπει να επιστρέφει το μέγεθος του μεγαλύτερου συνόλου από συνεχόμενους 1 σε οποιοδήποτε τμήμα της λέξης εισόδου.

<u>Παράδειγμα</u>: <u>αν x = 0011010111100 τότε y = 4</u>.

Γράψτε κώδικα VHDL που να μοντελοποιεί τη λειτουργία του κυκλώματος όπου όλα τα σήματα εισόδου και εξόδου είναι τύπου std_logic. Ο κώδικας πρέπει να είναι παραμετρικός ως προς το πλήθος των ψηφίων N της εισόδου x.

Το κύκλωμα εξακολουθεί να οδηγεί μία οθόνη επτά τμημάτων.

Άσκηση 5: Μετατροπή Binary-σε-BCD

Γράψτε κώδικα VHDL που θα μοντελοποιεί κύκλωμα μετατροπής από Binary-σε-BCD. Το κύκλωμα έχει μία είσοδο των 12 bit τύπου std_logic.

Οδηγίες

1. Όλες οι ασκήσεις να υλοποιηθούν με concurrent VHDL κώδικα. Δεν επιτρέπεται η χρήση port maps, process κλπ.

2. Η αναφορά σας θα πρέπει να περιλαμβάνει:
   - Σχολιασμό VHDL κώδικα σε συνδυασμό με το παραγόμενο RTL Analysis Elaborated Schematic στο Vivado
   - Σχολιασμό των αποτελεσμάτων των Synthesis και Implementation Schematics
   - Καταγραφή του Implementation Project Summary (Utilization, Power)

3. Όλες τις ασκήσεις να συνοδεύονται από επαρκείς προσομοιώσεις μέσω του Vivado οι οποίες θα πρέπει να σχολιάζονται ώστε να αποδεικνύεται ΠΛΗΡΩΣ η ορθότητα του κάθε κώδικα.

4. Μέ χρήση του Implementation Schematic και του Post-Implementation Timing Simulation να γίνει διερεύνηση του critical path και της καθυστέρησής του.

Χρήση του device part XC7Z020clg400-1 σε όλες τις ασκήσεις. Θα πρέπει να παράγετε το κατάλληλο constraints file (.xdc) για κάθε άσκηση, πριν παράξετε τα αποτελέσματα, κάνοντας το I/O Planning, σύμφωνα με το tutorial στο path

`Tutorials/Basic/Adam_Taylor/introduction_to_vivado-master/Introduction to Vivado_lab.pdf`, steps 33-40