ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΗΜΜΥ

ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ & ΥΛΙΚΟΥ

ΕΡΓΑΣΤΗΡΙΑΚΕΣ ΑΣΚΗΣΕΙΣ ΓΙΑ ΤΟ ΜΑΘΗΜΑ:

ΗΡΥ 203 - ΠΡΟΧΩΡΗΜΕΝΗ ΛΟΓΙΚΗ ΣΧΕΔΙΑΣΗ

ΕΑΡΙΝΟΕΞΑΜΗΝΟ 2020

Εργαστήριο 5

ΣΧΕΔΙΑΣΗ ΕΝΟΣ ΑΠΛΟΥ ΣΥΣΤΗΜΑΤΟΣ ΕΛΕΓΧΟΥ ΜΝΗΜΗΣ – ΜΙΑ ΑΠΛΗ ΣΤΟΙΒΑ

ΕΚΠΟΝΗΣΗ: Καθ. Α. Δόλλας,Δρ. Ε. Σωτηριάδης

ΣΥΝΕΡΓΑΤΕΣ: Δρ. Ε.Σωτηριάδης, Μ.Κιμιωνής, Ι.Πολογιώργη

ΕΚΔΟΣΗ : 2.0

Χανιά

Σκοπός

Είναι η μοντελοποίηση και υλοποίηση μιας στοίβας με δυνατότητα εντολών push, pop 4-bit αριθμών.

Ζητούμενα

Να σχεδιάσετε και να υλοποιήσετε ένα κύκλωμα με εισόδους και εξόδους όπως στον Πίνακα 1.

Όνομα	in/out	Πλάτος σε bit	Αντιστοίχιση στο Board
CLK	in	1	Clk
RST	in	1	Btn0
Push	in	1	Btn1
Pop	in	1	Btn2
NumberIN	in	4	SW[3:0]
NumberOUT	Out	4	LD[3:0]
Empty	Out	1	LD[4]
Full	Out	1	LD[5]
AlmostEmpty	Out	1	LD[6]
AlmostFull	Out	1	LD[7]

Πίνακας 1: Είσοδοι - έξοδοι του κυκλώματος

Προδιαγραφές

Το κύκλωμα θα έχει μία μνήμη πλάτους 4-bit και βάθους 12 θέσεων μνήμης. Η στοίβα θα λειτουργεί με τον κατάλληλο έλεγχο ως στοίβα 10 στοιχείων. Κάθε αριθμός που είναι στην είσοδο θα αποθηκεύεται στην στοίβα εφόσον πατηθεί το πλήκτρο Push. Κάθε φορά που θα πατηθεί το πλήκτρο Pop θα διαγράφεται ο αριθμός στην κεφαλή της στοίβας (στην ουσία δεν θα έχουμε διαγραφή αλλά μετακίνηση του Top of Stack - TOS). Η έξοδος της στοίβας NumberOUT θα είναι η κεφαλή της στοίβας (TOS) εκτός αν είναι άδεια οπότε θα είναι η θέση 0 της μνήμης. Όταν η στοίβα είναι άδεια θα ανάβει η έξοδος Empty και κάθε φορά που είναι γεμάτη, δηλαδή έχει 10 στοιχεία, θα ανάβει η έξοδος Full. Όταν η στοίβα είναι άδεια δεν θα μπορεί να αφαιρέσει κανείς στοιχεία (Pop) και όταν είναι γεμάτη (Full) δε θα μπορεί να προσθέσει στοιχεία (Push), αντίστοιχα. Προσοχή: Στις στοίβες με n στοιχεία μνήμης μπορούμε να αποθηκεύσουμε (n-1) δεδομένα γιατί διαφορετικά μας «ξεφεύγουν» οι pointers. Στο παρόν εργαστήριο λύνουμε το πρόβλημα με το να φτιάξουμε μία μνήμη βάθους 12 θέσεων αλλά χρησιμοποιώντας μόνο 11 από αυτές (για 10 χρήσιμα στοιχεία: θέση 0 για άδεια στοίβα, θέσεις 1-10 για αποθήκευση, θέση 11 αχρησιμοποίητη). Επί πλέον από τα παραπάνω σήματα έχουμε και τις εξόδους AlmostEmpty και AlmostFull, και οι οποίες γίνονται 1 εφόσον έχουμε αποθηκευμένα (αντίστοιχα) 1-3 ή 8-10 δεδομένα.

Μας συμφέρει η στοίβα να είναι pre-increment/post-

decrementγιατί ο TOSpointer ανά πάσα στιγμή είναι η διεύθυνση της μνήμης που δείχνουμε στην οθόνη (σε post-increment/pre-decrement θέλουμε δύο pointers, ένα για το TOS και ένα για την θέση που θα δείχνει η οθόνη). Δείτε τις σχετικές σημειώσεις του μαθήματος. Για ευκολία δεν σας ζητάμε να υλοποιήσετε το κύκλωμα λειτουργίας Ovfl/Unfl σε περίπτωση που προσπαθήσουμε να κάνουμε Push σε γεμάτη στοίβα ή Pop από άδεια στοίβα.

Θέματα Υλοποίησης

Στα εργαστήρια 1-4 κάναμε το σχεδιαστικό μέρος που αντιστοιχεί στο FrontEnd, δηλαδή την υψηλού επιπέδου σχεδίαση. Μέρος του 5ου εργαστηρίου είναι και το BackEnd, δηλαδή η σύνθεση (Synthesis), η «τοποθέτηση και διασύνδεση» (PlaceandRoute) που δημιουργεί το αρχείο .bit, και το κατέβασμα του αρχείου .bit στο φυσικό ολοκληρωμένο κύκλωμα (FPGA – Field Programmable Gate Array).

Τα βήματα αυτά περιλαμβάνουν, εκτός από την χρήση επί πλέον εργαλείων, και κάποια επί πλέον βήματα. Για παράδειγμα, η σχεδίαση, όπως την κάναμε μέχρι τώρα, μπορεί να απεικονιστεί σε διαφορετικά ολοκληρωμένα κυκλώματα (διαφορετικούς τύπους ή και από διαφορετικές οικογένειες). Αυτό θέτει το ερώτημα του πως απεικονίζουμε καθένα από τα σήματά μας (π.χ. το δικό μας σήμα Empty) σε κάποιο ακροδέκτη της FPGA; Η απάντηση είναι ότι δημιουργούμε το αρχείο .ucf, και το οποίο περιλαμβάνει την πληροφορία για τον τύπο του ολοκληρωμένου κυκλώματος και την αντιστοίχηση των σημάτων. Θα σας δοθούν οδηγίες και υπόδειγμα.

Το «κατέβασμα» του αρχείου .bit γίνεται μέσω του καλωδίου JTAG που συνδέει τον σταθμό εργασίας μας με την πλακέτα που έχει την FPGA.

Προεργασία 30% - Προσομοίωση 50% - Υλοποίηση 20%

Όταν έρθετε στο εργαστήριο πρέπει να έχετε μοντελοποιήσει το κύκλωμα σε VHDL και να έχετε ελέγξει τη λειτουργία του με testbench με πλήρη προσομοίωση. Η σχεδίαση πρέπει να είναι πλήρης με PlaceandRoute με σωστό αρχείο .ucf και πρέπει να έχετε και το .bitfile. Η έλλειψη προεργασίας οδηγεί σε απόρριψη στη συγκεκριμένη άσκηση.

Παρατηρήσεις/Σημειώσεις

(1) Η μνήμη θα δημιουργηθεί αυτόματα από τα εργαλεία της Xilinx
(2) Ο έλεγχος της μνήμης θα γίνει από μία FSM.
(3) Η FSM μαζί με τη μνήμη θα ενωθούν με ένα toplevel αρχείο.

(4) Η υλοποίηση του εργαστηρίου στο αναπτυξιακό basys2 απαιτεί την ύπαρξη αρχείου .ucf που θα σας δοθεί για αυτό είναι πολύ σημαντικό να κρατήσετε τα ονόματα εισόδων εξόδων όπως σας έχουν δοθεί.

(5) Λεπτομέρειες για το αναπτυξιακό Basys2 υπάρχουν στο link:**https://reference.digilentinc.com/_media/basys2:basys2_rm.pdf**

Παραδοτέα:

Πηγαίος κώδικας VHDL, κυματομορφές προσομοίωσης, παρουσίαση κυκλώματος, επίσης να δείξετε τη διαδικασία λύσης, τι χρησιμοποιήσατε από τη θεωρία και πως, π.χ. πίνακας καταστάσεων.

Βαθμολογία:

Διεξαγωγή εργαστηρίου	Προετοιμασία:30%
	Προσομοίωση: 50%
	Υλοποίηση: 20%

ΠΡΟΣΟΧΗ!

1) Η προεργασία να είναι σε ηλεκτρονική μορφή, σε αρχείο pdf.

2) Η έλλειψη προετοιμασίας οδηγεί σε απόρριψη.

3)Η διαπίστωση αντιγραφής σε οποιοδήποτε σκέλος της άσκησης οδηγεί στην απόρριψη από το σύνολο των εργαστηριακών ασκήσεων. Αυτό γίνεται οποιαδήποτε στιγμή στη διάρκεια του εξαμήνου.

ΚΑΛΗ ΕΠΙΤΥΧΙΑ!☺