



ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ & ΥΛΙΚΟΥ
ΕΡΓΑΣΤΗΡΙΑΚΕΣ ΑΣΚΗΣΕΙΣ ΓΙΑ ΤΟ ΜΑΘΗΜΑ: ΗΡΥ 312
ΟΡΓΑΝΩΣΗ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ 2017-2018

Εργαστήριο 1

**ΣΧΕΔΙΑΣΗ ΚΑΙ ΠΡΟΣΟΜΟΙΩΣΗ ΜΟΝΑΔΑΣ ΑΡΙΘΜΗΤΙΚΩΝ
ΚΑΙ ΛΟΓΙΚΩΝ ΠΡΑΞΕΩΝ (ALU) ΚΑΙ ΑΡΧΕΙΟΥ
ΚΑΤΑΧΩΡΗΤΩΝ (REGISTER FILE)**

Σκοπός του Εργαστηρίου:

Η σχεδίαση σε Γλώσσα Περιγραφής Υλικού (Hardware Description Language) VHDL μιας μονάδας αριθμητικών και λογικών πράξεων και ενός αρχείου καταχωρητών, και η προσομοίωση αυτών (με τα εργαλεία της Xilinx).

Προαπαιτούμενα

Καλή κατανόηση της VHDL στη συμπεριφορική μορφή της (behavioral) και την δομική μορφή της (structural) καθώς και του περιβάλλοντος που προσφέρουν τα εργαλεία της Xilinx (κοινώς γνώσεις που αποκτήθηκαν στο μάθημα Προχωρημένη Λογική Σχεδίαση).

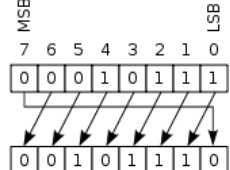
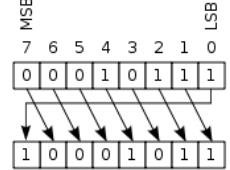
Διεξαγωγή

A) Σχεδίαση της μονάδας αριθμητικών και λογικών πράξεων

Η μονάδα έχει τις εξής εισόδους και εξόδους:

Σήμα	Είδος -Πλάτος	Λειτουργία
A	είσοδος (32 bits)	Πρώτος τελεσταίος σε συμπλήρωμα ως προς 2
B	είσοδος (32 bits)	Δεύτερος τελεσταίος σε συμπλήρωμα ως προς 2
Op	είσοδος (4 bits)	Κωδικός πράξης
Out	έξοδος (32 bits)	Αποτέλεσμα σε συμπλήρωμα ως προς 2
Zero	έξοδος (1 bit)	Ενεργοποιημένη αν το αποτέλεσμα είναι μηδέν
Cout	έξοδος (1 bit)	Ενεργοποιημένη αν υπήρξε κρατούμενο εξόδου (Carry Out)
Onf	έξοδος (1 bit)	Ενεργοποιημένη αν υπήρξε υπερχείλιση

Η συμπεριφορά της ALU είναι η εξής:

Κωδικός	Πράξη	Αποτέλεσμα
Op = 0000	Πρόσθεση	Out = A + B
Op = 0001	Αφαίρεση	Out = A - B
Op = 0010	Λογικό “ΟΧΙ ΚΑΙ”	Out = A NAND B
Op = 0011	Λογικό “Η”	Out = A B
Op = 0100	Αντιστροφή του A	Out = ! A
Op = 1000	Αριθμητική ολίσθηση δεξιά κατά 1 θέση (MSB ← (παλιό MSB))	Out= (int) A >> 1 Αποτέλεσμα = {A[31], A[31], ... A[1]}
Op = 1001	Λογική ολίσθηση δεξιά κατά 1 θέση (MSB ← ‘0’)	Out= (unsigned int) A >> 1 Αποτέλεσμα = {0, A[31], ... A[1]}
Op = 1010	Λογική ολίσθηση αριστερά κατά 1 θέση (LSB ← 0)	Out= A << 1 Αποτέλεσμα = {A[30], A[29],... A[0],0}
Op = 1100	Κυκλικό ολίσθηση (rotate) αριστερά το A κατά 1 θέση	
Op = 1101	Κυκλικό ολίσθηση (rotate) δεξιά το A κατά 1 θέση	

- 1) Γράψτε τον κώδικα που υλοποιεί την ALU σε VHDL. Δώστε προσοχή στις εξόδους Zero, Cout και Onf. Ποίες είναι οι συνθήκες που ορίζουν αυτές τις εξόδους.
- 2) Μεταγλωττίστε και προσομοιώστε την ALU στο περιβάλλον Xilinx.
- 3) Δώστε αρκετές διαφορετικές εισόδους στην προσομοίωση ώστε να ελέγξετε όλες τις (ενδιαφέρουσες) περιπτώσεις των σημάτων εξόδου.

B) Σχεδίαση του Αρχείου Καταχωρητών

Το αρχείο καταχωρητών είναι ένα συνδυασμός από καταχωρητές και συνδυαστική λογική.

B1. Παραγωγή Register

Αρχικά υλοποιήστε έναν καταχωρητή 32 bits σε VHDL. Ο καταχωρητής θα πρέπει να έχει τα εξής σήματα: **ρολόι (CLK - 1 bit)**, **Δεδομένα εισόδου (Data - 32 bits)**, **Δεδομένα εξόδου (Dout - 32 bits)** και ένα σήμα για **Write Enable (WE - 1 bit)**. Η διεπαφή για τον register φαίνεται στην Εικόνα 1.



Εικόνα 1: Καταχωρητής 32 bits

B2. Παραγωγή Register File

Για την παραγωγή ενός αρχείου καταχωρητών (Register File-RF) θα χρησιμοποιήσετε 31 registers, όπως αυτούς που υλοποιήσατε στο βήμα B1. Στην συνέχεια κάνοντας την συνδεσμολογία που παρουσιάζεται στην Εικόνα 2 θα υλοποιήσετε ένα αρχείο καταχωρητών με 32 θέσεις, με τρεις θύρες (δύο ανάγνωσης και μία εγγραφής). Η διεπαφή του αρχείου καταχωρητών έχει τις εξής εισόδους και εξόδους:

Σήμα	Είδος -Πλάτος	Λειτουργία
Ard1	Είσοδος (5 bits)	Διεύθυνση πρώτου καταχωρητή για ανάγνωση
Ard2	Είσοδος (5 bits)	Διεύθυνση δεύτερου καταχωρητή για ανάγνωση
Awr	Είσοδος (5 bits)	Διεύθυνση καταχωρητή για εγγραφή
Dout1	Έξοδος (32 bits)	Δεδομένα πρώτου καταχωρητή
Dout2	Έξοδος (32 bits)	Δεδομένα δεύτερου καταχωρητή
Din	Είσοδος (32 bits)	Δεδομένα για εγγραφή
WrEn	Είσοδος (1 bit)	Ενεργοποίηση Εγγραφής καταχωρητή
Clk	Είσοδος (1 bit)	Ρολόι

Παρατηρήσεις:

1. Στην διεπαφή δεν υπάρχει είσοδος ενεργοποίησης ανάγνωσης, το οποίο σημαίνει ότι το αρχείο καταχωρητών διαβάζει πάντα και από τις δύο θέσεις που υποδεικνύουν οι διευθύνσεις ανάγνωσης.
2. Όλα τα modules που απεικονίζονται στην Εικόνα 2 είναι **ασύγχρονα** (εκτός προφανώς τους registers).
3. Ως γνωστό η τιμή του R0 παραμένει πάντα σταθερά 0.
4. Το **Compare Module** είναι ένα **ασύγχρονο κύκλωμα** που ελέγχει αν οι δύο καταχωρητές (εγγραφής και ανάγνωσης) είναι ίδιοι όταν ενεργοποιηθεί το σήμα WrEn. Αν αυτό ισχύει, τότε στην έξοδο της Register File θα πρέπει να έρχεται η νέα τιμή του καταχωρητή και όχι αυτή που είναι αποθηκευμένη στον αντίστοιχο καταχωρητή.

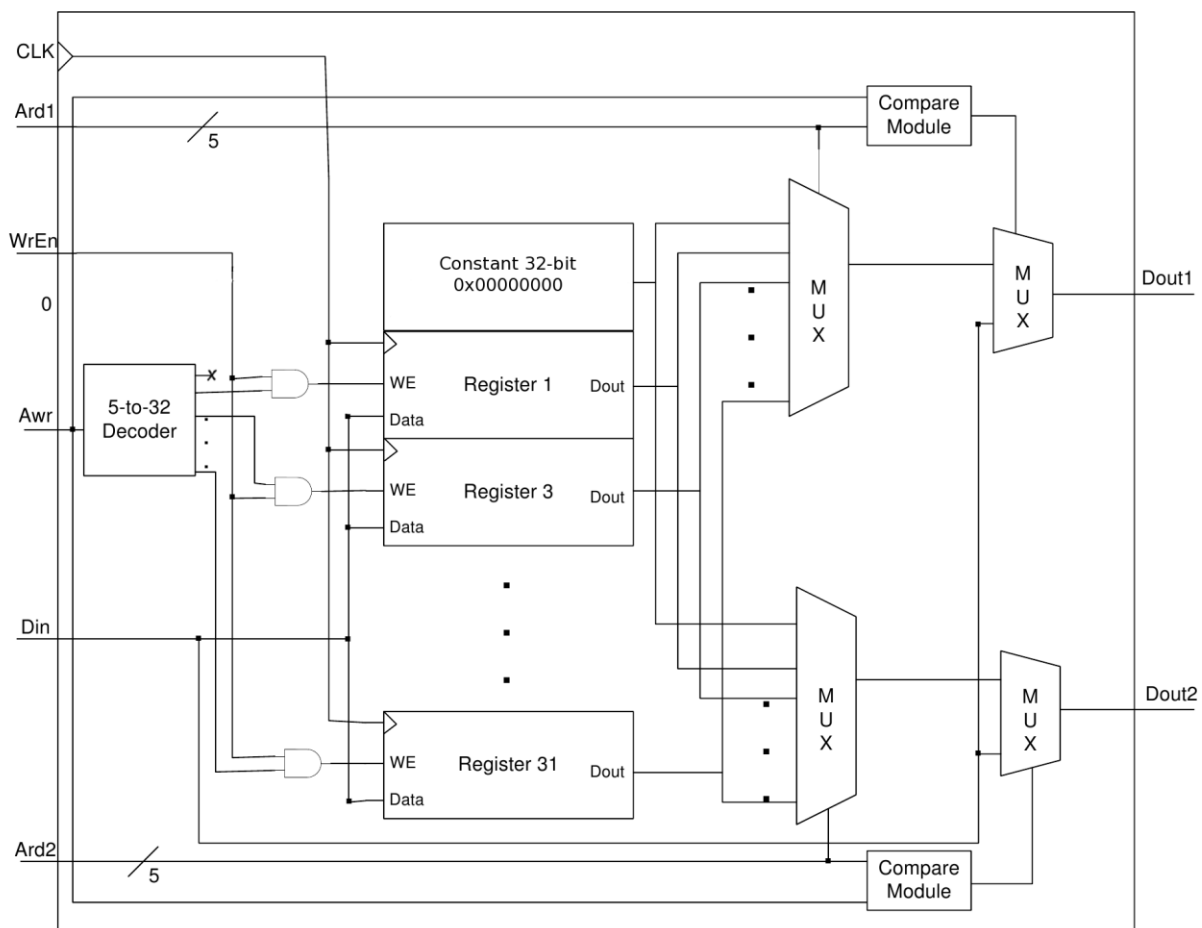
Υλοποίηση:

- 1) Γράψτε τον κώδικα VHDL που υλοποιεί το αρχείο καταχωρητών χρησιμοποιώντας τον καταχωρητή που υλοποιήσατε στο **B1**.
- 2) Μεταγλωττίστε και προσομοιώστε την RF στο περιβάλλον Xilinx.

3) Δώστε αρκετές διαφορετικές εισόδους στην προσομοίωση ώστε να ελέγξετε όλες τις (ενδιαφέρουσες) περιπτώσεις των σημάτων εισόδου-εξόδου.

Παραδοτέα – Βαθμολογία

1. Δώστε ένα σχηματικό διάγραμμα της συνδεσμολογίας που χρησιμοποιήσατε για την υλοποίηση του Αρχείου Καταχωρητών.
2. Κώδικας VHDL (πηγαίος).
3. Κυματομορφές προσομοίωσης.
4. Σύντομη αναφορά στη διαδικασία του εργαστηρίου (μαζί και ενδεχόμενα προβλήματα που παρατηρήθηκαν για μελλοντική του βελτίωση).



Εικόνα 2: Αρχείο καταχωρητών