12/3/2017

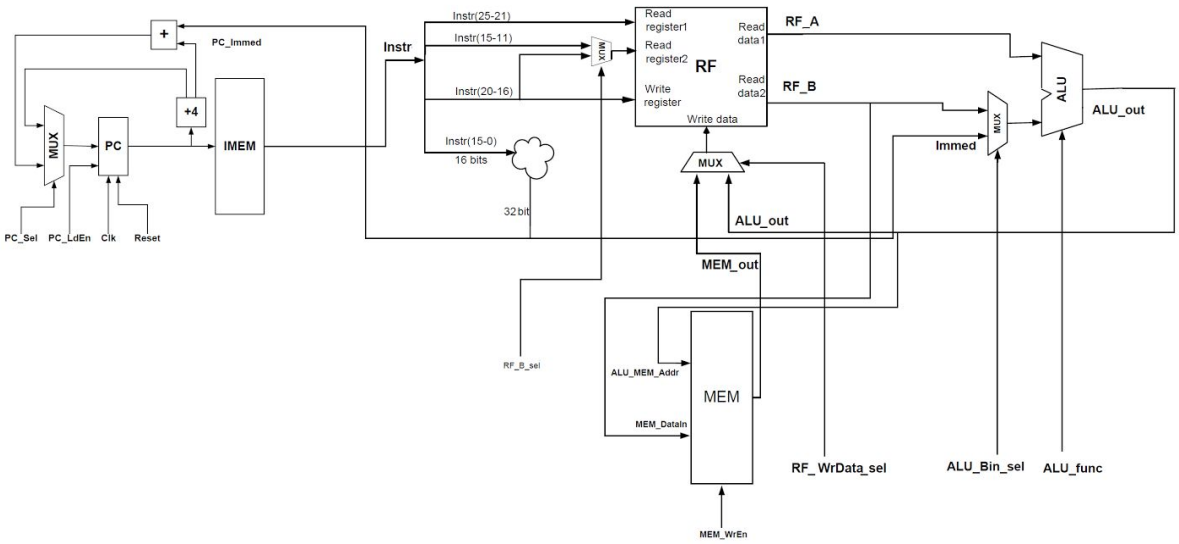
Αναφορά Εργαστηρίου 2

Κωδικός Ομάδας **LAB31231465**

Δι α λ ε κ τά κ η ς Γ ι ώ ρ γ ο ς
Βαϊλάκης Απόστολος Νικόλαος

Προεργασία

Ως προεργασία του 2ου εργαστηρίου μας ζητήθηκε ένα σχηματικό διάγραμμα του ολοκληρωμένου **datapath** (προσχέδιο) όπου φαίνονται οι συνδέσεις μεταξύ των βαθμίδων που υλοποιήσαμε.



Περιγραφή Άσκησης

Σκοπός την 2ης εργαστηριακής άσκησης ήταν η σχεδίαση της βαθμίδας ανάκλησης εντολών (**IFSTAGE**), της βαθμίδας αποκωδικοποίησης εντολών (**DECSTAGE**), της βαθμίδας εκτέλεσης εντολών (**ALUSTAGE**) και της βαθμίδας πρόσβασης μνήμης(**MEMSTAGE**) που αποτελούν το **Datapath** του Επεξεργαστή.

IFSTAGE

Η βαθμίδα ανάκλησης εντολών αποτελείται το σημείο του επεξεργαστή όπου είναι αποθηκευμένο το πρόγραμμα που πρόκειται να εκτελέσει (**ROM**). Αυτή η βαθμίδα περιέχει τον καταχωρητή **PC** ο οποίος και είναι υπεύθυνος για την διευθυνσιοδότηση της μνήμης **ROM**, της οποίας η έξοδος είναι και η εντολή που εισάγεται στον επεξεργαστή. Αυτή η **IFSTAGE** δέχεται ως είσοδο ένα σήμα **Immediate** και ένα σήμα **pc_sel**. Αυτά τα σήματα αφορούν ως σκοπό την επιλογή της επόμενης εντολής από τον προγραμματιστή εκτελώντας εντολές **branch**. Πιο συγκεκριμένα, σε κανονική λειτουργία η **IF_STAGE** περιμένει ένα σήμα ελέγχου (**PC_LdEn**) ώστε να αυξήσει τον **PC** κατά 4 (**PC = PC + 4**). Όταν όμως εντολές **branch** θέλουν να αλλάξουν τον **PC** κατά βούληση, το σήμα **PC_Sel** ενεργοποιείται και το επόμενο **PC** που γράφεται στον καταχωρητή είναι το **PC + 4 + Immed**.

DECSTAGE

Η βαθμίδα **DecStage** είναι υπεύθυνη για την αποκωδικοποίηση της τρέχουσας εντολής, όπως επίσης και για την τροφοδότηση της **Register File** που δημιουργήσαμε στο πρώτο εργαστήριο με τα καταλληλα δεδομένα, έχοντας υπόψιν κάποια εξωτερικά σήματα ελέγχου. Επίσης η βαθμίδα αυτή είναι υπεύθυνη για την παραμετροποίηση του σήματος immediate στις Itype εντολές, και την επέκτασή του από 16 bit σε 32 κάνοντας Sign Extend ή Zero Fill αντίστοιχα.

Οι παραμετροποιήσεις αυτές είναι :

li : SignExtend
lui : Bit Shift αριστερά κατα 16 θέσεις και Zero Fill τις υπολοιπες
addi : SignExtend
andi : Zero Fill
ori : Zero Fill
b : Sign Extend και Bit Shift αριστερά κατά δύο θέσεις
beq : Sign Extend και Bit Shift αριστερά κατά δύο θέσεις
bne : Sign Extend και Bit Shift αριστερά κατά δύο θέσεις
lb : Sign Extend
sb : Sign Extend
lw : Sign Extend
sw : Sign Extend

ALUSTAGE

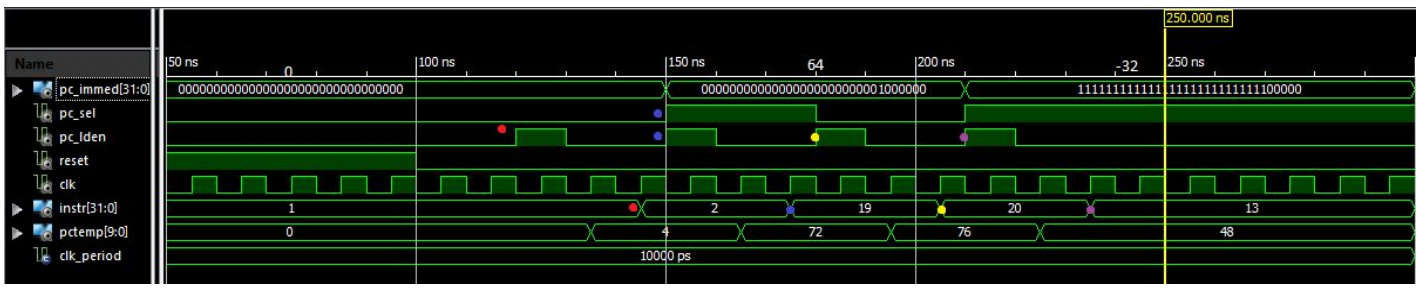
Βασικό κομμάτι αυτού του σταδίου ήταν η χρήση της βαθμίδας εκτέλεσης εντολών ALU που υλοποιήθηκε στο 1ο εργαστήριο η οποία εκτελεί αριθμητικές και λογικές πράξεις. Ο πρώτος τελεσταίος της ALU είναι πάντα ο καταχωρητής **rs** από την Register File. Ο δεύτερος τελεσταίος της ALU επιλέγεται αποένα πολυπλέκτη που δημιουργήθηκε για να εισάγει στην ALU είτε τα δεδομένα ενός καταχωρητή από την Register file, συγκεκριμένα τον **rt** ή τον **rd** είτε έναν **Immediate.** Έτσι πλέον, η ALU εκτελεί πράξεις ανάμεσα σε δύο καταχωρητές ή ανάμεσα σε έναν καταχωρητή και έναν Immediate. Τα παραπάνω αποτελούν την ALUSTAGE.

MEMSTAGE

Τέλος, για την υλοποίηση της βαθμίδας πρόσβασης μνήμης χρησιμοποιήθηκε έτοιμος κώδικας για την παραγωγή μίας **Read First *μνήμης* RAM** 1024 θέσεων η οποία έχει μία θύρα εγγραφής και ανάγνωσης. Η μνήμη έχει τα εξής σήματα : **clk, Mem_WrEn,ALU_MEM_Addr, MEM_DataIn, MEM_DataOut.** Το σήμα Mem_WrEn ενεργοποιείται όταν θέλουμε να γράψουμε στη μνήμη. Το σήμα ALU_MEM_Addr είναι το αποτέλεσμα της ALU που μας λέει σε ποιά διεύθυνση της μνήμης να γράψουμε (εντολές όπως sb,sw) ή από ποιά διεύθυνση να διαβάσουμε (εντολές όπως lb,lw). Το σήμα MEM_DataIn περιέχει τα δεδομένα του καταχωρητή rd που πρέπει να αποθηκευτούν στη μνήμη. Τέλος, το σήμα MEM_DataOut είναι η έξοδος της μνήμης σε περίπτωση load και περιέχει τα δεδομένα που φορτώθηκαν από τη μνήμη και πρέπει να γραφτούν στην Register File.

Κυματομορφές

IFSTAGE



-> Σε αυτήν την περίπτωση έχουμε PC_LdEn ενεργοποιημένο

και βλέπουμε ότι το Instruction γίνεται από 1-> 2 καθώς εκτελέστηκε PC<= PC + 4.

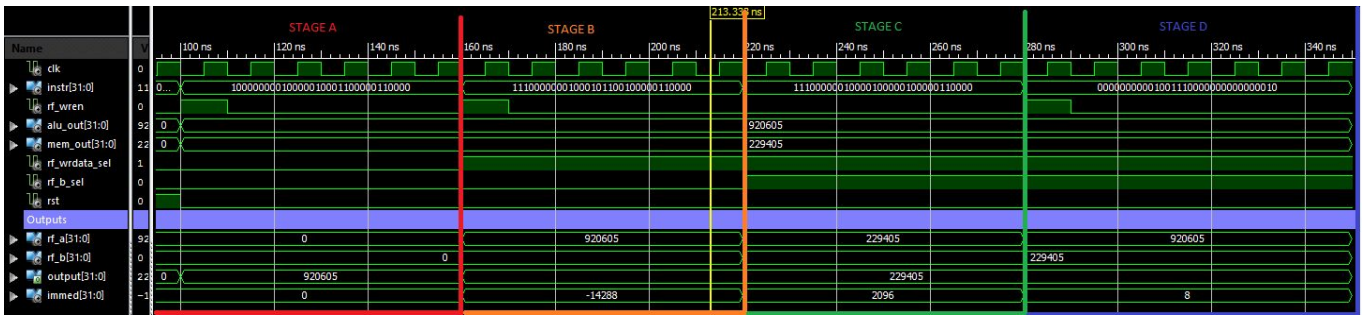
-> Εδώέχουμε PC_LdEn και PC_Sel ενεργοποιημένα και βλέπουμε ότι το Instruction γίνεται από 2-> 19 καθώς εκτελέστηκε PC<= PC + 4 + 64.

-> Σε αυτήτην περίπτωση έχουμε PC_LdEn ενεργοποιημένο και βλέπουμε ότι το Instruction γίνεται από 19-> 20 καθώς εκτελέστηκε PC<= PC + 4.

-> Εδώέχουμε PC_LdEn και PC_Sel ενεργοποιημένα και βλέπουμε ότι το Instruction γίνεται από 20-> 13 καθώς εκτελέστηκε PC<= PC + 4 - 32.

Οι εντολές από το rom.data δείχνουν ουσιαστικά σε ποια γραμμή βρίσκονται μέσα στο αρχείο.

DECSTAGE



Η κυμματομορφήτης decstage χωρίζεται σε 4 στάδια:

STAGE A:

Εδώ πραγματοποιούμε μια πράξη RType στην οποία και γράφουμε στον Rd ($1) την τιμήπου έρχεται από την ALU (920605) και ενεργοποιούμε το write enable του RF

STAGE B:

Το δεύτερο stage πραγματοποιείπράξη με immediate τον οποίο πρέπει και να κάνει Sign Extend. Σε αυτό το stage ο καταχωρητήςrs έχει οριστθεί ως $1 για τον έλεγχο τηςορθής λειτουργίας τηςπροηγούμενηςπράξης. Για άλλη μια φοράτο write enable ενεργοποιείται ωστε να καταγραφείη τιμήτου immediate στον rd ($2).

STAGE C:

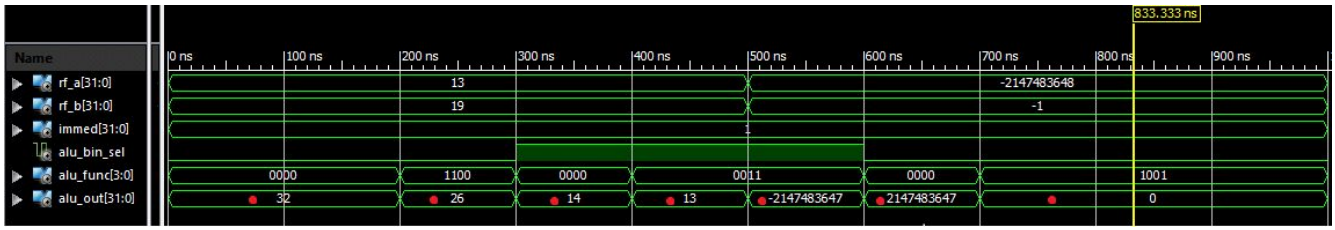
Ακολουθώντας την ίδια λογικήμε το προηγούμενο στάδιο τροφοδοτούμε την DECSTAGE με IType χωρίςόμως να

ενεργοποιήσουμε το write enable. Βλέπουμε όμως οτι ο Immediate έχει κάνει sign extend κατάλληλα (το opcode μας είναι 111000 όπου θέλει Sign Extend). Επίσης ο Rs είναι $2 ωστε να ελεγχθεί η προηγούμενη πράξη.

STAGE D:

Τέλος στο τέταρτο stage πραγματοποιούμε μια πράξη με immediate τύπου branch θέτοντας στον immediate τον αριθμό 2, ο οποίος και πολλαπλασιάζεται επί του 4 (bit shift left κατα 2). Τέλος επιλέγουμε και RF_WrData_Sel = 1 ωστε να υπάρχουν τα δεδομένα απο την Mem_out στο Write Data της RF μας και ξανα ενεργοποιούμε το RF_WrEn.

ALUSTAGE



1η Πράξη: Προσθέτουμε τους καταχωρητές με δεδομένα 13 και 19 καθώς το ALU_Bin_Sel είναι ΄0΄ και παίρνουμε αποτέλεσμα 32.

2η Πράξη: Κάνουμε αριστερή περιστροφή του 1ου καταχωρητή.

3η Πράξη: Πρόσθεση του πρώτου καταχωρητή με τον Immediate καθώς το ALU_Bin_Sel είναι ΄1΄,άρα έχουμε 13+1 = 14.

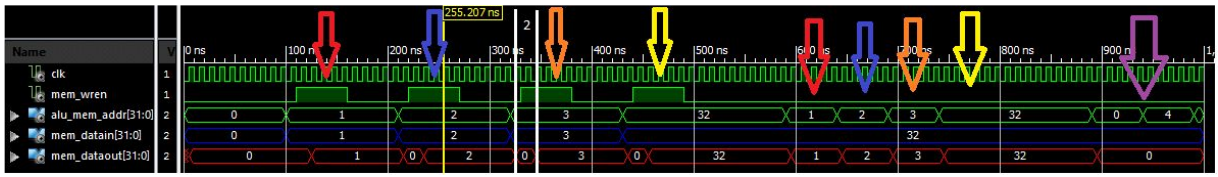
4η Πράξη: Εκτελούμαι OR του πρώτου καταχωρητή με τον Immediate καθώς το ALU_Bin_Sel είναι ΄1΄.

5η Πράξη: Πρόσθεση του πρώτου καταχωρητή με τον Immediate καθώς το ALU_Bin_Sel είναι ΄1΄.

6η Πράξη: Προσθέτουμε τους καταχωρητές με αρνητικά δεδομένα καθώς το ALU_Bin_Sel είναι ΄0΄ και παίρνουμε αποτέλεσμα θετικό πράγμα που σημαίνει ότι έχουμε overflow.

7η Πράξη: Εκτελούμε αριστερή λογική ολίσθηση του 1ου καταχωρητή οπότε και το αποτέλεσμα είναι μηδέν.

MEMSTAGE



Παρατηρούμε,όπως δείχνουν τα βέλη,ότι γίνονται 4 εγγραφές στη μνήμη όταν το σήμα MEM_WrEn είναι ενεργοποιημένο και στη συνέχεια 4 αναγνώσεις από τις ίδιες διευθύνσεις. Οι εγγραφές έχουν αντιστοιχία δεδομένων-διεύθυνσης για την διευκόλυνση του έλεγχου της κυματομορφής. Στο τέλος της κυματομορφής διαβάζουμε και δύο διευθύνσεις στις οποίες και δέν έχουμε γράψει και βλέπουμε οτι είναι αρχικοποιημένες στο 0. Τέλος βλέπουμε κάθε εγγραφή στην μνήμη κοστίζει δύο κύκλους ρολογιού.

Συμπεράσματα

Στα πλαίσια της 2ης εργαστηριακής άσκησης μάθαμε ουσιαστικά από ποιά μέρη αποτελείται το Datapath ενός επεξεργαστή: την IF που φέρνει μία εντολή, την Decode που αποκωδικοποιεί αυτή την εντολή με σκοπό να γνωρίζει ο επεξεργαστής πώς να ενεργήσει, την ALU που εκτελείτις πράξεις και τέλος την MEM που υποστηρίζει εγγραφή και ανάγνωση δεδομένων. Έτσι, φτάσαμε ένα βήμα πριν την υλοποίηση ενός επεξεργαστή πολλαπλών κύκλων.