## *Προεργασία 2ου Εργαστηρίου*

**Ομάδα Εργασίας: LAB31239629**

**Κονιδάρη Ηρώ Α.Μ. 2012030049**
**Μάνεσης Αθανάσιος Α.Μ. 2014030061**

### *Σκοπός της Άσκησης*

Σκοπός αυτής της άσκησης ήταν ο ορισμός της αρχιτεκτονικής συνόλου εντολών καθώς επίσης και ο σχεδιασμός και η υλοποίηση τεσσάρων βαθμίδων(ανάκληση, αποκωδικοποίηση, εκτέλεση εντολών και πρόσβαση μνήμης).

### *Περιγραφή/Υλοποίηση της Άσκησης*

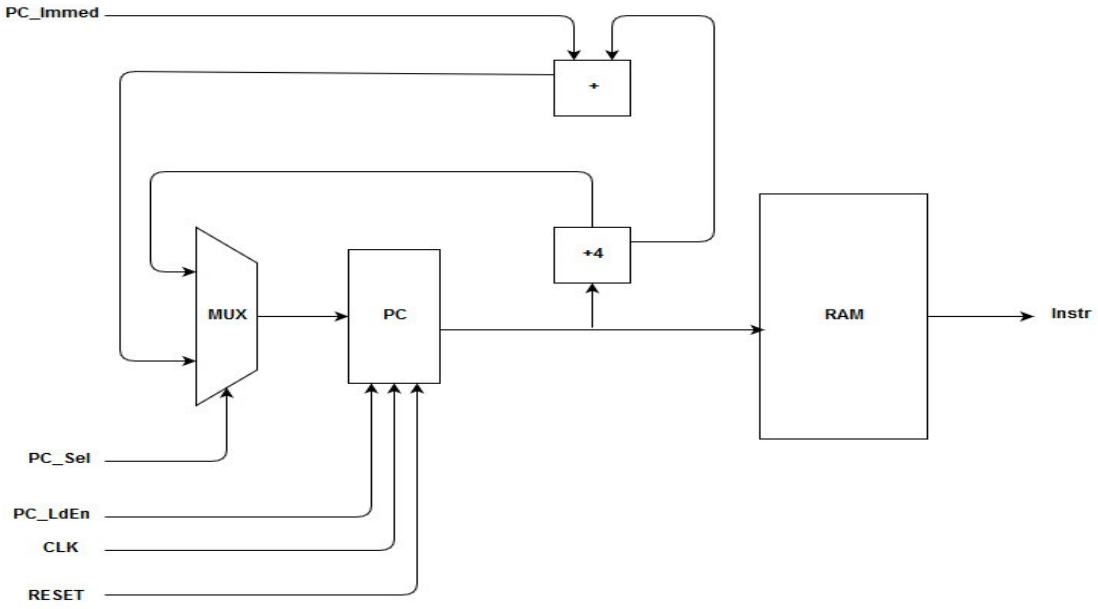
*1ο Μέρος:*

Στο πρώτο μέρος αυτής της άσκησης μελετήσαμε την κωδικοποίηση των εντολών του CHARIS- 4 .

*2ο Μέρος:*

Σε αυτό το μέρος μας ζητήθηκε να σχεδιάσουμε και να υλοποιήσουμε την βαθμίδα ανάκλησης εντολών (**IFSTAGE**). Σκοπός αυτού του module είναι ανάκληση εντολών από μία μνήμη RAM 2048 θέσεων των 32 bits. Προκειμένου να δημιουργήσουμε την μνήμη αυτή χρησιμοποιήσαμε των κώδικα που μας δόθηκε και το αρχείο ram.data έτσι ώστε να γίνει η αρχικοποίηση της. Η μνήμη έχει την διεύθυνση ανάγνωσης εντολής (data_addr- 11 bits) η οποία είναι είσοδος στη μνήμη, το ρολόι (**Clk**) και την έξοδο της μνήμης (data_out- 32 bits) που μας δίνει τα δεδομένα που διαβάστηκαν από αυτή.

### *IfStage*



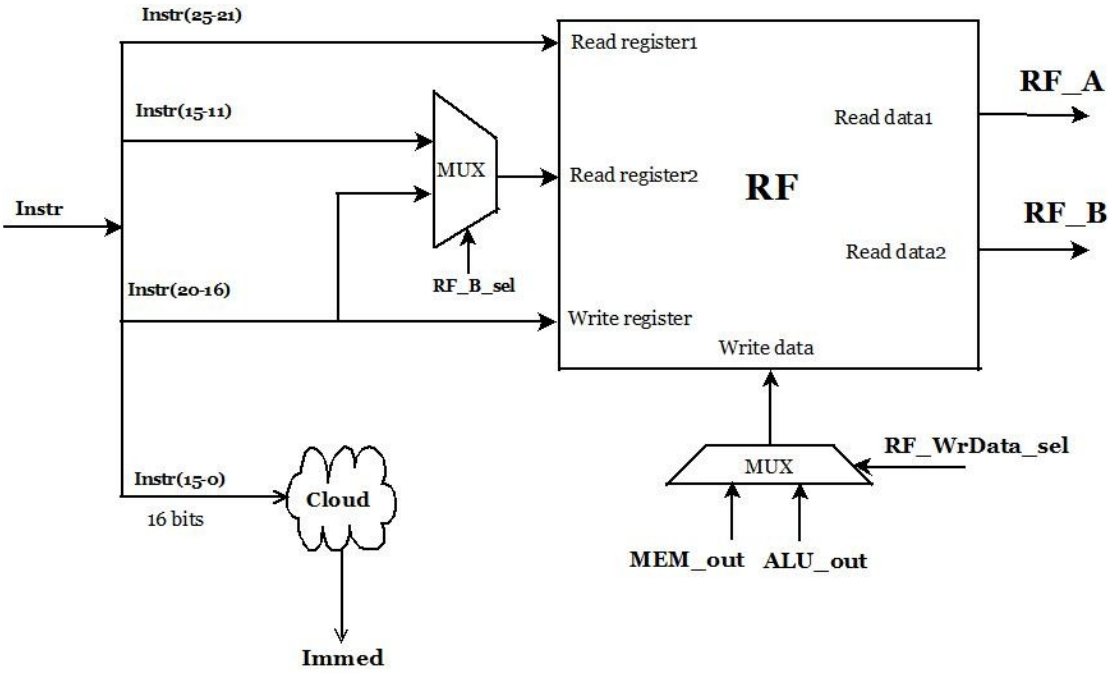

Η IFSTAGE δέχεται ως είσοδο το σήμα **PC_Immed** (32 bits) το οποίο έχει την τιμή Immediate για τις εντολές b, beqz, bnez. Αυτό εισέρχεται μαζί με την έξοδο του αυξητή (PC +4 ), στον αθροιστή, προστίθενται και στη συνέχεια εισέρχεται ως είσοδος

στον πολυπλέκτη. Είσοδος του πολυπλέκτη αποτελεί και η έξοδος του αυξητή ενώ το **PC_sel** (1 bit) είναι το σήμα επιλογής του. Σκοπός αυτού του πολυπλέκτη είναι η επιλογή της εισόδου του καταχωρητή (PC). Αν PC_sel= '1' η τιμή του καταχωρητή αυξάνει κατά 4+ Immediate. Αντίθετα, αν PC_sel= '0' η τιμή του αυξάνει μόνο κατά 4. Για να ενημερωθεί όμως ο καταχωρητής θα πρέπει να είναι ενεργοποιημένο το **PC_LdEn**. Ο λόγος που υπάρχει αύξηση κατά 4 είναι γιατί θέλουμε να περάσουμε στην αμέσως επόμενη εντολή που είναι αποθηκευμένη στη μνήμη αφού ο PC είναι byte addressable.

## *3ο Μέρος:*

Στο μέρος αυτό υλοποιήσαμε το **DECSTAGE**. Αυτό αποτελείται από το **αρχείο καταχωρητών** (Register File - RF) που σχεδιάσαμε στο πρώτο εργαστήριο, από **δύο πολυπλέκτες 2x1**, αλλά και από μία **μονάδα (Cloud)** η οποία μετατρέπει 16 bits σε 32 bits. Το **DECSTAGE** απεικονίζεται παρακάτω:

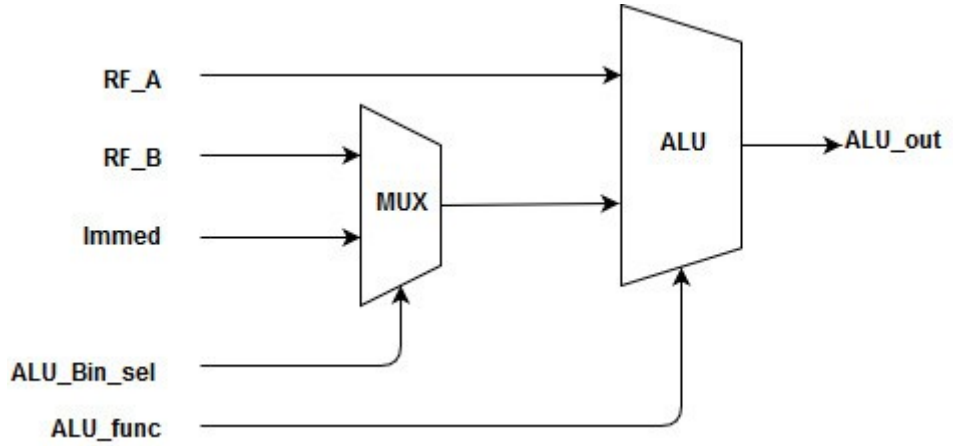
*DecStage*



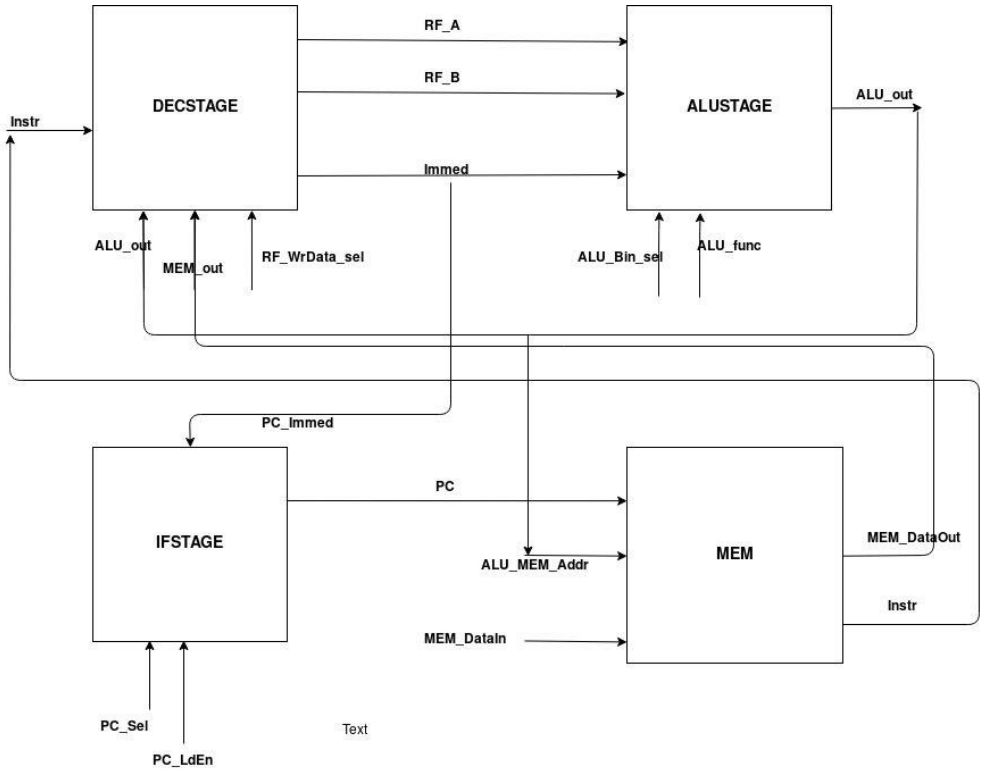

## *4ο Μέρος:*

Σε αυτό το κομμάτι της άσκησης, χρησιμοποιώντας την **ALU** που υλοποιήσαμε στο 1ο εργαστήριο αλλά και έναν **πολυπλέκτη 2x1**, δημιουργήσαμε το **ALUSTAGE** η οποία απεικονίζεται παρακάτω:

*Alustage*

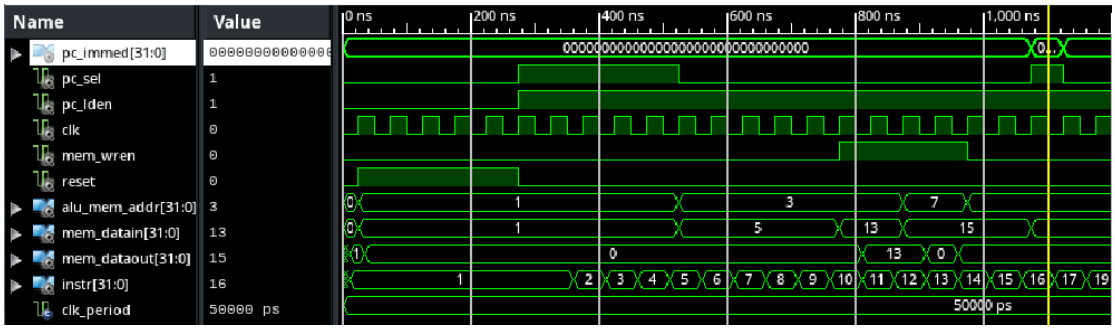


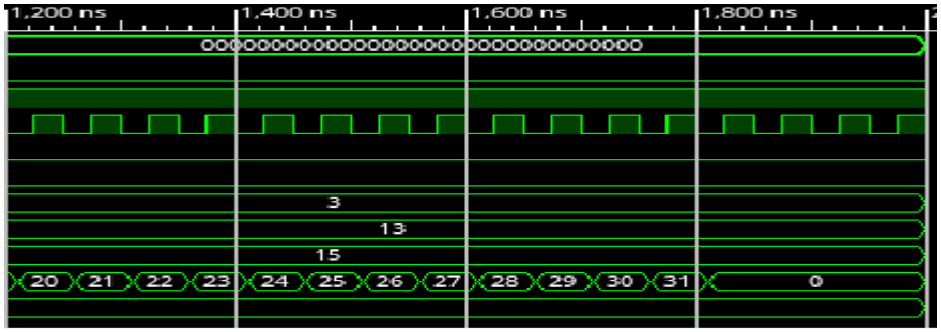

*Datapath Final*



## *Κυματομορφές*

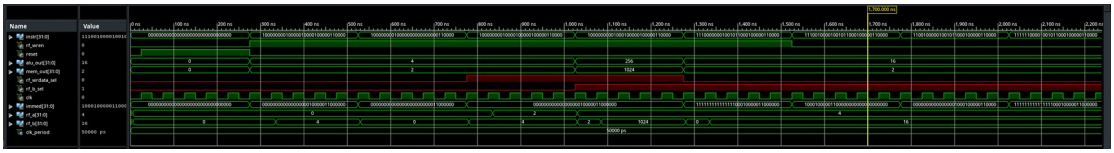
### *Ifmem - stage*




Από τις παραπάνω κυματομορφές μπορούμε να παρατηρήσουμε πως όταν το LdEn ενεργοποιηθεί στον καταχωρητή PC, εμφανίζονται σωστά οι εντολές που διαβάζονται από την μνήμη (ram.data), πηγαίνοντας στην επόμενη εντολή σε κάθε ακμή του ρολογιού. Παρατηρούμε επίσης τα δεδομένα που εισάγονται στο mem_datain να γραφονται σωστα στο mem_addr.

### *Dec – stage*



| **Read Registers** | **Write Register** | **rf_b_sel** | **RF_A** | **RF_B** |
|---|---|---|---|---|
| rs=2 , rt=1 | rd=1 | 0 | 0 (R2) | 4 (R1) |
| rs=2 , rt=0 | rd=0 | 0 | 0 (R2) | 0 (R0) |
| rs=2 , rt=1 | rd=2 | 0 | 2 (R2) | 4 (R1) |
| rs=1 , rt=1 | rd=2 | 1 | 4 (R1) | 1024 (R2) |

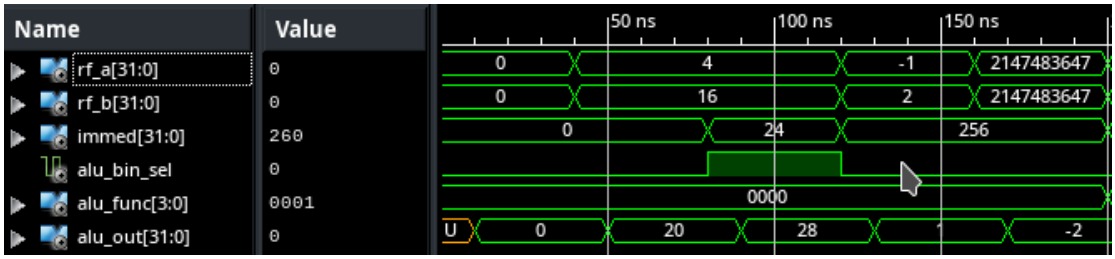
Αρχικά έχουμε reset ενεργό όπότε όλα τα σήματα μας είναι μηδέν.

Αρχικά έχουμε reset οπότε όλες οι έξοδοι είναι 0. Στην συνέχεια κάνουμε Read τους καταχωρητές 2 και 1 και επειδή το **rf_wrdata_sel=0** θα γραφτει στον R1 το 4, το οποίο φαίνεται στην έξοδο rf_b. Κανουμε Read τους καταχωρητές 0 και 2 όπου βλέπουμε σωστά τις εξόδους τους 0. Οταν ενεργοποιήσουμε το σήμα **rf_wrdata_sel**, τότε θα γράφονται οι τιμές του mem_out. Read registers είναι ο 1 ,2 ενώ Write Register είναι ο 2. Παρατηρούμε σωστά στην έξοδο το **rf_a = 2 (R2=2)** και **rf_b=4 (R1=4)**. Στη συνέχεια διαβάζουμε τους καταχωρητές 1 και 1, αλλα επειδή το rf_b_sel είναι 1 , ο δεύτερος καταχωρητής ανάγνωσης θα είναι ο καταχωρητής εγγραφής, άρα ο rf_a που είναι ο καταχωρητής 1 θα έχει την τιμή 4, ενώ ο rf_b που είναι ο καταχωρητής 2 θα έχει την τιμή που

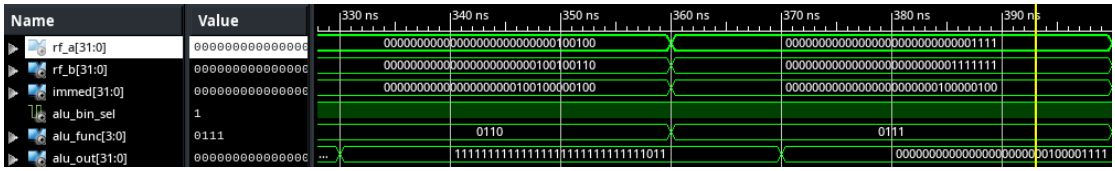
γράφουμε, δηλαδή το 1024. Τέλος δείχνουμε την περίπτωση που το OpCode της εντολής αντιστοιχεί στην εντολες **li, lui, nandi, b** όπου κάνουμε αντίστοιχα **SignExtend(Imm)**, **Imm<<16(zero-fill)**, **ZeroFill(Imm)** και **signextend(Imm)<<2** τον immediate, το οποίο φαίνεται και στην έξοδο immed.

## *ALU stage*

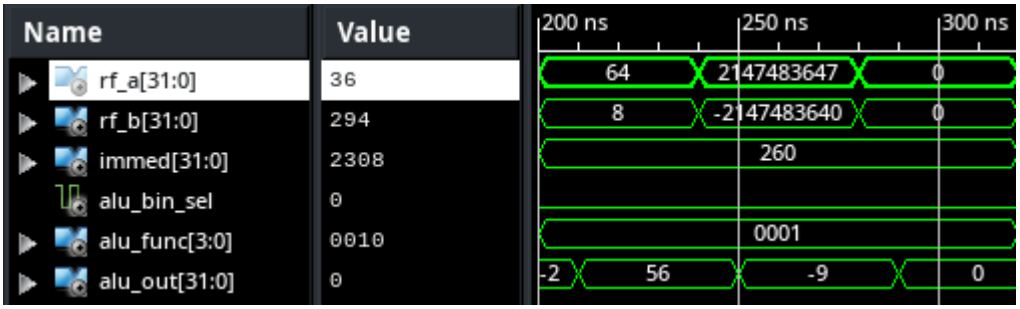
### *Add*




### *Nand – Or - Immediate*




### *Sub*



Ενεργοποιώντας το Alu_bin_sel βλέπουμε να γίνονται οι πράξεις με τον Immed.