# Modules

## Instruction Fetch

Inputs:

1. [31:0] PC\_Immed: Τιμή Immediate για εντολές b, beq, bne.
2. PC\_Sel: Επιλογή εισόδου για ενημέρωση του PC (PC+4 ή PC+4+Immed).
3. PC\_LdEn: Ενεργοποίηση εγγραφής στον PC.
4. Reset: Είσοδος Reset για αρχικοποίηση του καταχωρητή PC.
5. Clk: Ρολόι.

Outputs:

1. wire [31:0] Instr: Η εντολή που ανακλήθηκε.

Internal signals:

1. reg [31:0] PC: Register που αποθηκεύει την τιμή του Program Counter.
2. wire [2 \* 32 - 1:0] mux\_input: Bus 64-bits (2 x 32) για την είσοδο δεδομένων στον multiplexer.
3. wire [31:0] mux\_to\_pc: Η έξοδος του multiplexer για την ανανέωση της τιμής του Program Counter.
4. wire [31:0] pc\_to\_memory: Bus για την σύνδεση του program counter με την μνήμη.

Η βαθμίδα Instruction Fetch διαβάζει εντολές από την μνήμη που υλοποιείται μέσα στο module IMEM, ανάλογα με την τιμή του program counter (PC). Η τιμή του PC καθορίζεται από το προηγούμενο instruction που εκτελέστηκε, το οποίο καθορίζει το PC\_Sel του multiplexer. Αν η προηγούμενη εντολή είναι branch τότε PC\_Sel = 1 και o PC γίνεται PC+4 + Immediate διαφορετικά PC\_Sel = 0 και o PC γίνεται PC+4.

## Decode Instruction

Inputs:

1. [31:0] Instr
2. RF\_WrEn: Ενεργοποίηση εγγραφής καταχωρητή.
3. [31:0] ALU\_out: Δεδομένα εγγραφής καταχωρητή προερχόμενα από την ALU.
4. [31:0] MEM\_out: Δεδομένα εγγραφής καταχωρητή προερχόμενα από τη μνήμη.
5. RF\_WrData\_sel: Επιλογή πεδίου που καθορίζει την προέλευση δεδομένων προς εγγραφή.
6. RF\_Bsel: Επιλογή πεδίου που καθορίζει τον δεύτερο καταχωρητή ανάγνωσης.
7. Clk

Outputs:

1. reg [31:0] Immed: Immediate προς τις επόμενες βαθμίδες.
2. wire [31:0] RF\_A: H τιμή του 1ου καταχωρητή.
3. wire [31:0] RF\_B : H τιμή του 2ου καταχωρητή.

Internal signals:

1. wire [2 \* 32 - 1:0] write\_select\_bus: Συνδέει την έξοδο της ALU και της μνήμης με την είσοδο του πολυπλέκτη για την εγγραφή δεδομένων στην Register File.
2. wire [2 \* 5 - 1:0] read\_addr\_select\_bus: Συνδέει τα κομμάτια της εντολής Instr[15:11] και Instr[20:16] στον πολυπλέκτη για την ανάγνωση δεδομένων από την Register File.
3. wire [31:0] write\_data: Συνδέει την έξοδο του πολυπλέκτη για την εγγραφή δεδομένων στην Register File.
4. wire [4:0] read\_addr\_2: Συνδέει την έξοδο του πολυπλέκτη για την ανάγνωση διεύθυνσης στην Register File.

Στην βαθμίδα αυτή υλοποιείται το module Register File και η μετατροπή του Immed από 16-bits σε 32. Στην περίπτωση που το Instruction που δεν είναι τύπου immediate τότε Immed = 0.

## ΑLU Stage

Inputs:

1. [31:0] RF\_A: RF[rs].
2. [31:0] RF\_B: RF[rt] ή RF[rd].
3. [31:0] Immed: Immediate.
4. ALU\_Bin\_sel: Eπιλογή Εισόδου B της ALU από RF\_B ή Immediate.
5. [3:0] ALU\_func: Πράξη ALU.

Outputs:

1. wire [31:0] ALU\_out: Αποτέλεσμα ALU.
2. wire zero

Internal signals:

1. wire [2\*32 - 1:0] inputs: Είσοδοι του πολυπλέκτη.
2. wire [31:0] mux\_to\_ALU: Η έξοδος του πολυπλέκτη που πέρναει ως είσοδος στο ALU Module.

Στην βαθμίδα αυτή υλοποιείται το ALU module για την υλοποίηση αριθμητικών και λογικών πράξεων.

## Memory stage

Inputs:

1. clk: Ρολόι
2. Mem\_WrEn: Σημαία ενεργοποίησης εγγραφής στη μνήμη.
3. Mem\_In\_Out\_Sel: Σήμα επιλογής για την προσθήκη πολυπλκτών στο module.
4. [31:0] ALU\_MEM\_Addr: Αποτέλεσμα ALU (βλέπε εντολές lb, sb, lw, sw).
5. [31:0] MEM\_DataIn: Αποτέλεσμα RF[rd] για αποθήκευση στη μνήμη για εντολές swap και sb, sw.

Outputs:

1. [31:0] MEM\_DataOut: Δεδομένα που φορτώθηκαν από τη μνήμη προς εγγραφή σε καταχωρητή για εντολές lb, lw.

Internal signals:

1. wire [32 \* 2 - 1:0] input\_mux\_data: Είσοδος στον πολυπλέκτη στην είσοδο της μνήμης.
2. wire [31:0] input\_mux\_to\_mem: Συνδέει την έξοδο του πολυπλέκτη με την είσοδο στην μνήμη.
3. wire [32 \* 2 - 1:0] output\_mux\_data: Είσοδος στον πολυπλέκτη που βρίσκεται στην έξοδο της μνήμης.
4. wire [31:0] mem\_to\_output\_mux: Συνδέει την έξοδο της μνήμης με τις είσοδους του πολυπλέκτη που βρίσκεται στην έξοδο της μνήμης.

Στην βαθμίδα αυτή υλοποιείται η μνήμη και προστέθηκαν δύο multiplexers για να πραγματοποιηθούν οι εντολές sb και lb. Τα δεδόμενα που πρέπει να αποθηκευτούν έρχονται σε 32-bit μορφή, ωστόσο το τελευταίο byte περιέχει την επιθυμητή πληροφορία και όλα τα υπόλοιπα στοιχεία πρέπει να είναι 0. Για να επιτευχθεί αυτό γίνεται ένα bitwise and μεταξύ του δεδομένου αριθμού και του 00000000\_00000000\_00000000\_11111111 αφού x and 1 = x, δηλαδή υλοποιείται masking του αριθμού.

# Helper modules:

## Multiplexer

Parameters:

1. BUS\_WIDTH: Τα bits από μια είσοδο.
2. SEL: Ο αριθμός των select bits.

Inputs:

1. [(BUS\_WIDTH\*(2\*\*SEL))-1:0] Din: Οι είσοδοι στον multiplexer.
2. [SEL-1:0] Sel: Η τιμή του SEL που καθορίζει την έξοδο του πολυπλέκτη.

Outputs:

1. reg [BUS\_WIDTH-1:0] Dout: Η έξοδος του multiplexer.

Internal signals:

1. reg [BUS\_WIDTH - 1:0] inputs [2 \*\* SEL - 1:0]: Ένα array από 2^SEL registers μήκους BUS\_WIDTH bits που κρατάει την κάθε είσοδο των 32 bits.

H είσοδος Din που δίνουμε στον multiplexer είναι ένα bus που είναι 2^SEL x BUS\_WIDTH bits, σε τμήματα του οποίου βρίσκονται όλες οι είσοδοι που δίνονται στον multiplexer.

# Datapath

Στο Instruction Fetch διάβαζεται η επόμενη εντολή από την μνήμη που πρέπει να εκτελεστεί και περνιέται στο στάδιο του decode. Από τμήματα του instruction εξάγονται οι διευθύνσεις μνήμης για την Register File , ώστε να βγουν τα περιεχόμενα των καταχωρητών ως σήματα RF\_A και RF\_B. Ανάλογα το instruction, καθορίζεται και το Immediate. Στη συνέχεια, το RF\_A περνάει στην πρώτη είσοδο της ALU και το RF\_B με το Immediate περνάνε ως είσοδοι στον πολυπλέκτη, η έξοδος του οποίου καθορίζει ποια θα είναι η δεύτερη είσοδος στην ALU. Έχοντας ολοκληρώσει την λειτουργία της, η έξοδός της, όταν πρόκειται για memory instructions περνάει ως είσοδος στην μνήμη και καθορίζει σε ποια διεύθυνση θα πραγματοποιηθεί εγγραφή ή ανάγνωση διαφορετικά, περνάει στο decode stage. Επίσης, ως είσοδος στο memory stage περνάει και το περιεχόμενο του καταχωρητή RF\_B, που χρησιμοποιείται ως δεδομένα εισόδου στην περίπτωση εντολής αποθήκευσης. Αυτό μαζί με το σήμα (RF\_B & 00000000\_00000000\_00000000\_11111111) εισάγονται στην μνήμη μέσω ενός πολυπλέκτη. Η έξοδος της μνήμης MEM\_Out και (MEM\_Out & 00000000\_00000000\_00000000\_11111111) περνάνε σ’ έναν πολυπλέκτη και η έξοδός του δίνεται ως είσοδος σ’ έναν πολυπλέκτη που καθορίζει τα δεδομένα εγγραφής στην Register File του decode stage μαζί με την έξοδο που είχε βγάλει η ALU.

# Control