# Εργασία Hardware I – Αναφορά

## *Άσκηση 1*

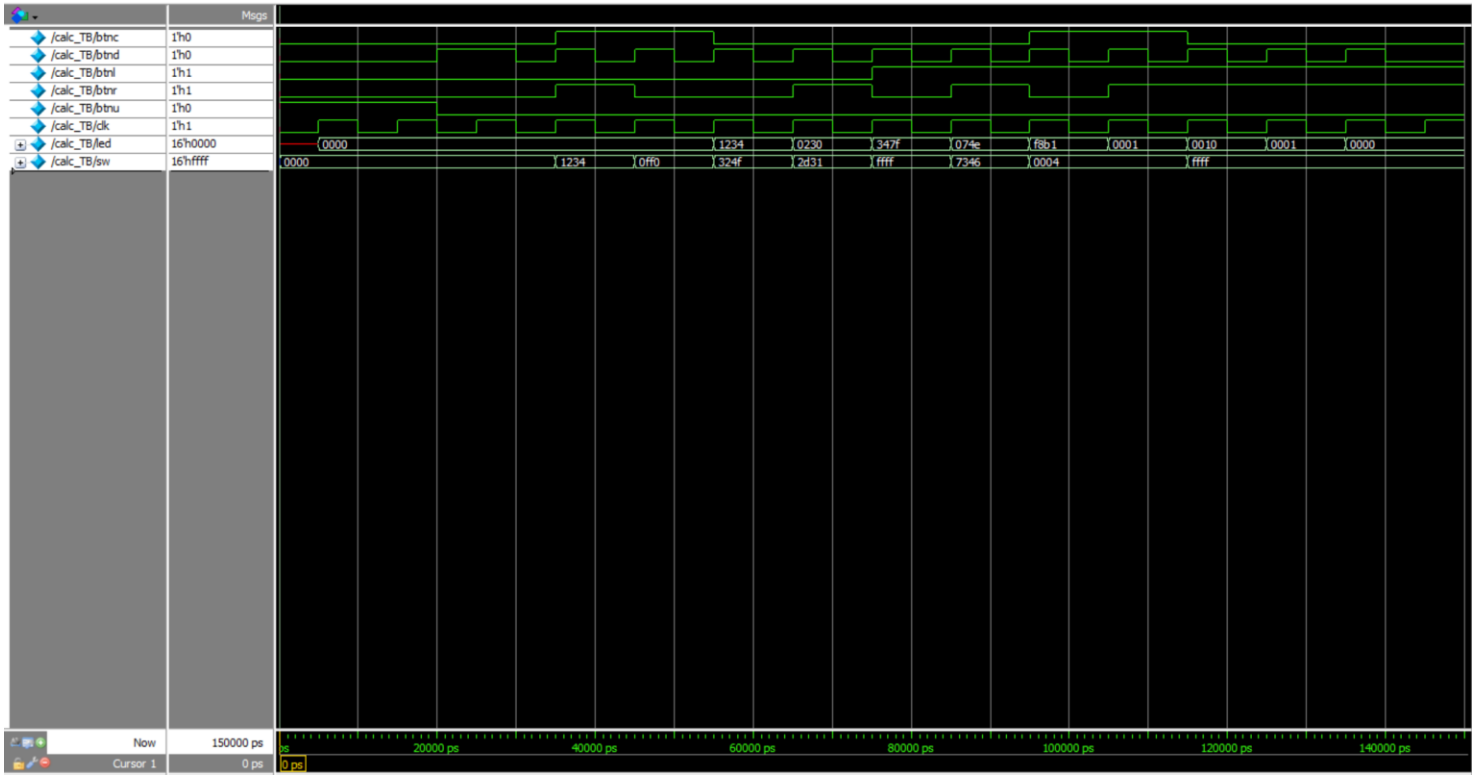
**alu.v**: Χρησιμοποιώ παραμέτρους για τις διάφορες ακολουθίες bit που χαρακτηρίζουν τις εντολές/εργασίες που μπορεί να εκτελέσει η ALU και με ένα case μέσα σε ένα always block αναθέτω στο αποτέλεσμα της ALU το αποτέλεσμα της εκάστοτε πράξης ανάλογα με το ποια εντολή έχω, προσέχοντας πότε κάποια πράξη απαιτεί signed ή unsigned αριθμούς.

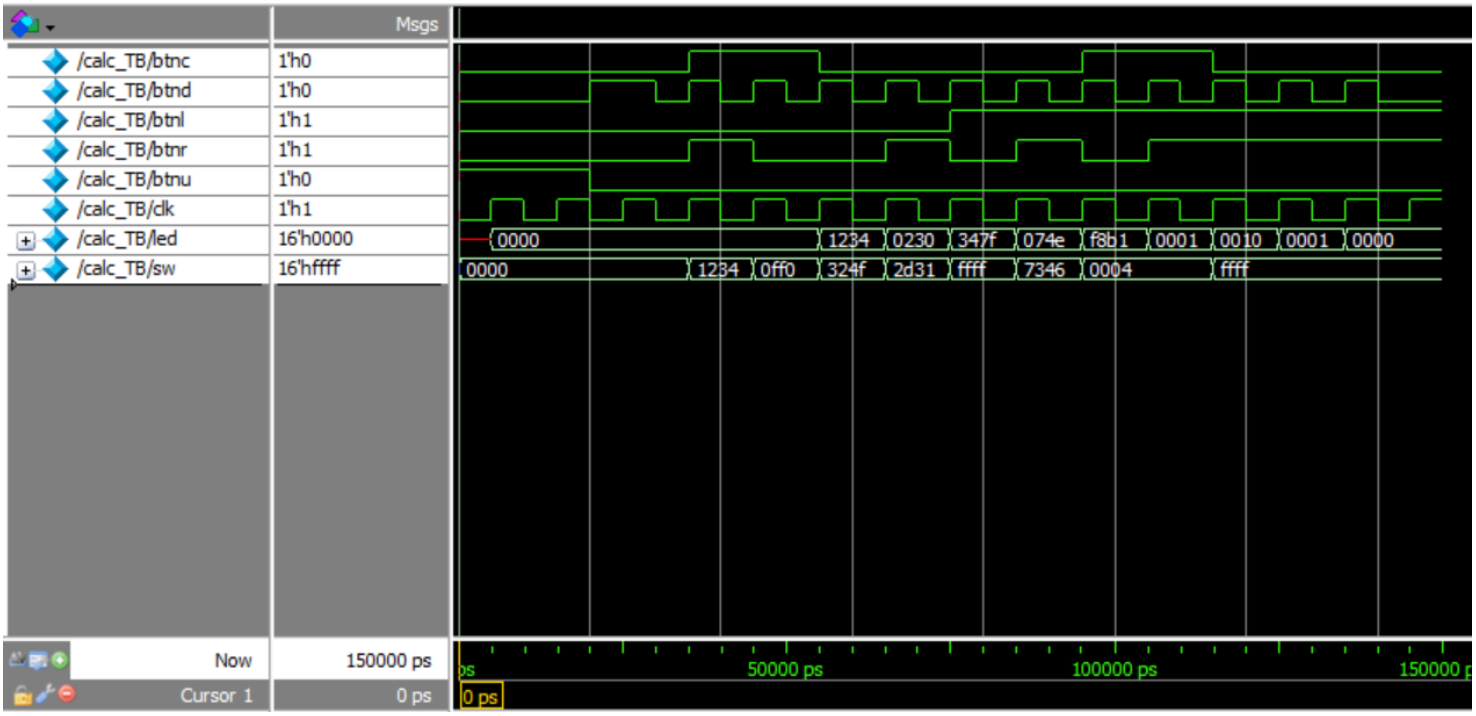
## *Άσκηση 2*

**decoder.v**: Αντικατοπτρίζω τη λογική των 4 συνδυαστικών κυκλωμάτων για τα 4 bit του alu_op χρησιμοποιώντας structural Verilog (με χρήση έτοιμων πυλών).

**calc.v**: Κάνω instantiate τον decoder και την alu και κάνω τις ανάλογες συνδέσεις. Όποτε αλλάζει ο accumulator, ενημερώνω τον op1 και όποτε αλλάζει το sw, ενημερώνω τον op2 και κάνω επέκταση με πρόσημο όπου χρειάζεται εφόσον οι τελεστές της alu είναι 32 bit. Σε ένα always block με τη sensitivity list να περιέχει θετική ακμή του clock και θετική ακμή του btnu (asynchronous reset) ενημερώνω ανάλογα τον accumulator (αν btnd = 1) ή τον μηδενίζω (αν btnu = 1) και τον συνδέω στη έξοδο led, όλα με non-blocking assignments για να μοντελοποιήσω ακολουθιακή λογική.

**calc_TB.v**: Κάνω instantiate τον calc, σχεδιάζω με ένα initial block και ένα always block ένα clock με περίοδο 10 t.u. και μετά με ένα initial block βάζω όλα τα test cases φροντίζοντας να ενημερώνω σωστά (με μια μικρή καθυστέρηση για να περαστούν τα δεδομένα) με το btnd το led.






Όπως φαίνεται και από τις κυματομορφές το αποτέλεσμα της alu και συνεπώς και η έξοδος led παίρνουν τις προσδοκώμενες τιμές δύο ακμές ρολογιού αργότερα. Στην αρχή τα led είναι απροσδιόριστα, γιατί δεν τα αρχικοποιώ.

## *Άσκηση 3*

**regfile.v**: Με έναν πίνακα 32 διανυσμάτων των 32 bits μοντελοποιώ τους registers από τους οποίους μπορώ να διαβάσω ή να γράψω (αν το σήμα εγγραφής είναι 1) προσέχοντας την περίπτωση όπου η διεύθυνση εγγραφής ταυτίζεται με διεύθυνση ανάγνωσης. Σε αυτή την περίπτωση προτεραιότητα έχει η ανάγνωση και όχι η εγγραφή.

## *Άσκηση 4*

**datapath.v**: Κάνω extract το immediate κομμάτι της εντολής και από αυτό μπορώ να κρίνω τι τύπου εντολή έχω (i, s, b type). Κάνω instantiate την alu, το regfile και την INSTRUCTION_MEMORY κάνοντας τις απαραίτητες συνδέσεις. Υλοποιώ τη λογική του PC, του decoding της εκάστοτε εντολής με βάση τα πεδία σε bit, της επιλογής των τελεστών της ALU, της λειτουργίας της ALU, της λειτουργίας της μνήμης ανάλογα με το αν γράφω ή αν διαβάζω και του write back stage (αν το σήμα ελέγχου για write back to register file είναι 1).

## *Άσκηση 5*

**multicycle.v**: Κάνω instantiate το datapath και τη DATA_MEMORY κάνοντας τις απαραίτητες συνδέσεις. Υλοποιώ τα σήματα ελέγχου:

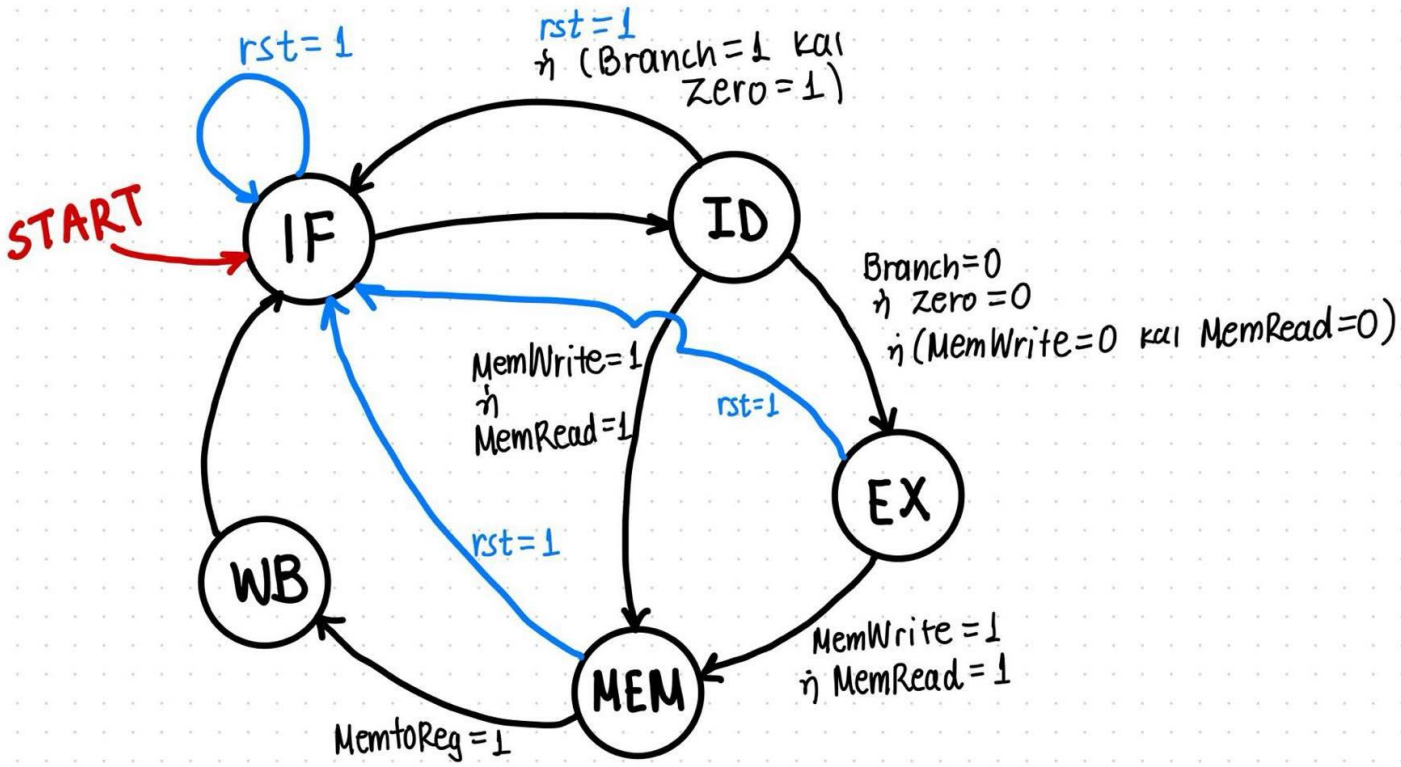
- ALUCtrl: χρησιμοποιώ τον opcode και τα funct3 και funct7 κάθε εντολής για να καταλήξω στο ποιο είναι το ALUCtrl που προσδιορίζει το ποια πράξη θα κάνει η ALU.
- ALUSrc: αποφασίζει ποιος θα είναι ο op2 της ALU.

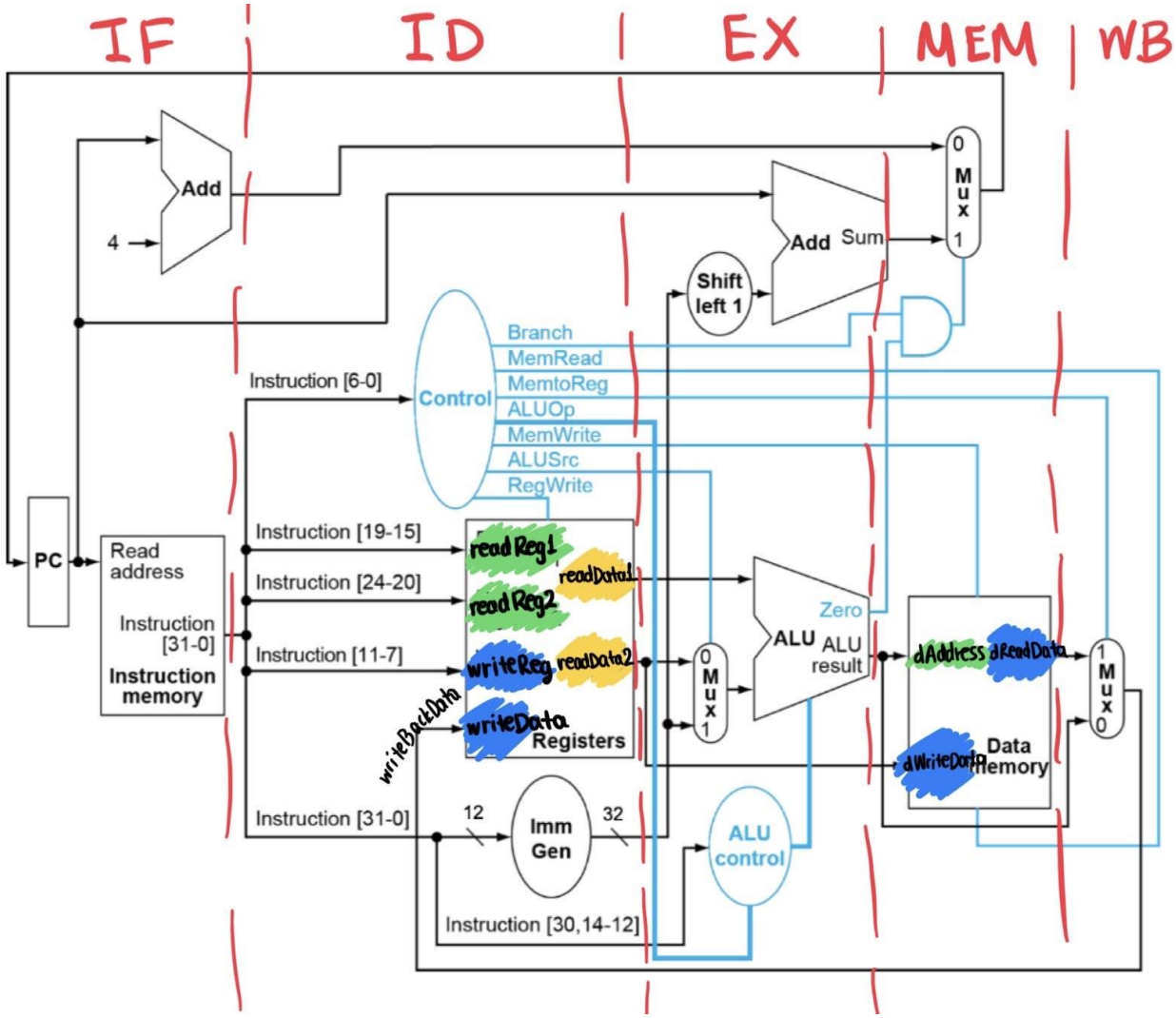
Ύστερα υλοποιώ σε ένα always block τη λογική για το current state του FSM 5 καταστάσεων: IF, ID, EX, MEM, WB.

Σε ένα άλλο always block υλοποιώ τη λογική για το next state χρησιμοποιώντας ένα case ανάλογα σε ποια κατάσταση βρίσκεται το FSM και ανάλογα με το ποια σήματα ελέγχου είναι ενεργά (1) και ποια ανενεργά (0).

Σε ένα τελευταίο always block υλοποιώ τη λογική εξόδου για το FSM βάζοντας τις σχέσεις μετάβασης από κατάσταση σε κατάσταση ανάλογα με τα σήματα ελέγχου και την εκάστοτε εντολή.

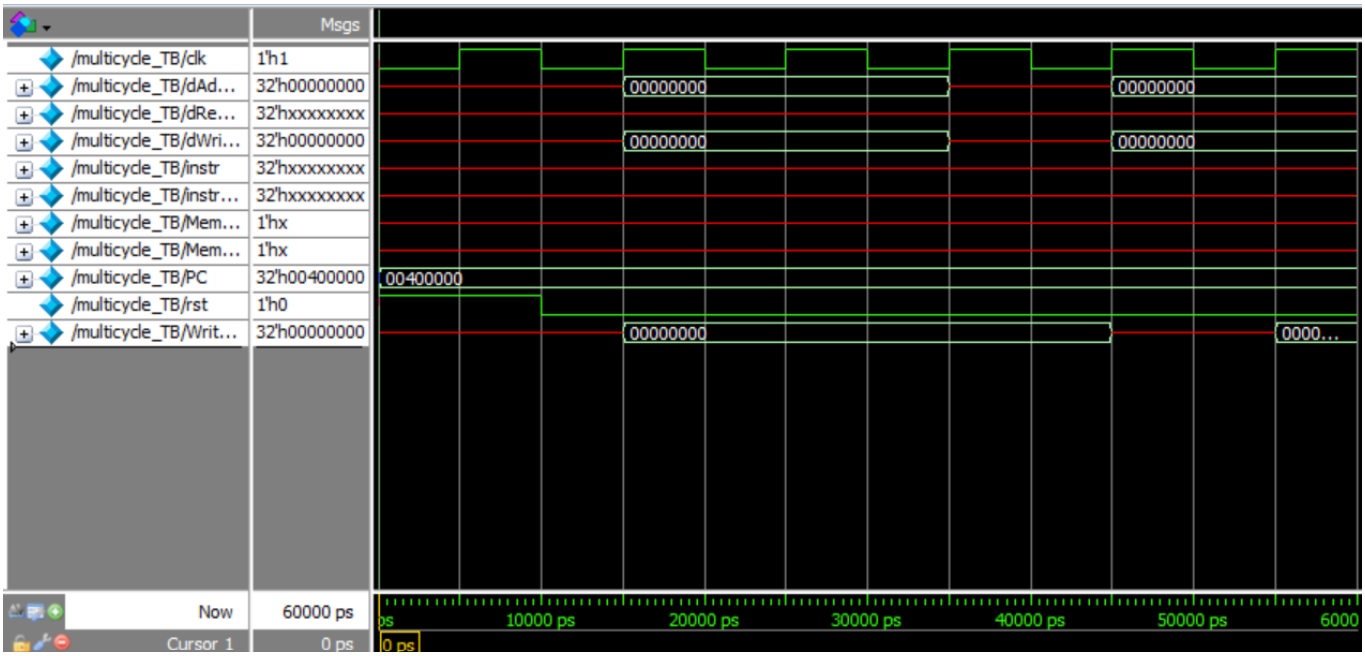
Ακολουθεί το διάγραμμα του FSM 5 καταστάσεων:






**multicycle_TB.v**: Κάνω instantiate το multicycle και φτιάχνω με ένα initial και ένα always block το clock με περίοδο 10t.u. Μετά σε always block κάνω print την κατάσταση του FSM και σε ένα επόμενο initial block κάνω αρχικά reset και μετά διαβάζω από το INSTRUCTION_MEMORY τις εντολές και το επαναλαμβάνω αυτό για 5 κύκλους.

*Παρόλα αυτά, η προσομοίωση βγάζει απροσδιόριστες τιμές, δε λειτουργεί σωστά και δυστυχώς δεν έχω αρκετό χρόνο να κάνω extensive debugging:*




Περισσότερες λεπτομέρειες για την υλοποίηση κάθε module, το σχεδιαστικό σκεπτικό, καθώς και για τη ροή λειτουργίας τους βρίσκονται στο εκάστοτε αρχείο με μορφή σχολίων.

Δήμητρα Γεωργία Κολίτσα

ΑΕΜ 10323