ΗΡΥ 301 - ΟΡΓΑΝΩΣΗ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΣΚΗΣΗ 2: Σχεδίαση Επεξεργαστή πολλαπλών Κύκλων

Ομάδα 69

Καράλης Αστερινός 2020030107

Σταμούλης Πολυχρόνης 2021030006

ΣΚΟΠΟΣ ΤΗΣ ΑΣΚΗΣΗΣ

Σκοπός της άσκησης είναι η μετατροπή του επεξεργαστή ενός κύκλου που σχεδιάσαμε στην

φάση 1 σε επεξεργαστή πολλαπλών κύκλων.

ΑΛΛΑΓΕΣ ΣΤΗΝ ΑΡΧΙΚΗ ΥΛΟΠΟΙΗΣΗ

Για την μετατροπή κάναμε αλλαγές στα αρχεία **DATAPATH.vhd, CONTROL.vhd**, και **DECODE.vhd** ενώ επίσης δημιουργήσαμε και ένα καινούργιο αρχείο, το **equal_comparator.vhd**.

- **CONTROL**: Αφαιρέσαμε την συνάρτηση compareVectors που χρησιμοποιούταν για τις εντολές beq, bne. Προσθέσαμε μια FSM όπως ζητήθηκε από την εκφώνηση μαζί με ένα ακόμη process για συγχρονισμό της FSM.
  Συγκεκριμένα τα states της FSM είναι: **RESET, FETCH, I_DEC, BRANCH_COMPL, R_EXEC, MEM_EXEC, R_WB, S_MEM, L_MEM, L_WB** και λειτουργούν ως εξής
  Το reset μας πάει στο αρχικό state που είναι το FETCH για να πάρουμε το instruction από την μνήμη. Έπειτα από το FETCH πάμε στο I_DEC που γίνεται decoded το instruction έπειτα ανάλογα την εντολή: αν έχουμε memory access (LW, SW, LB ,SB) πάμε στο MEM_EXEC και από εκεί ανάλογα αν κάνουμε read ή write πάμε είτε στο L_MEM και έπειτα στο L_WB είτε στο S_MEM αντίστοιχα. Άν απο την άλλη έχουμε instruction για την ALU τότε από το I_DEC πάμε στο R_EXEC. Αν έχουμε branch instruction πάμε στο BRANCH_COMPL όπου απο εκεί θα πάμε πάλι στο FETCH για το επόμενο συγκεκριμένο instruction. Το state R_WB χρησιμοποιήται στις περιπτώσεις που θέλουμε να γράψουμε κάτι πίσω σε κάποιο register του RF.

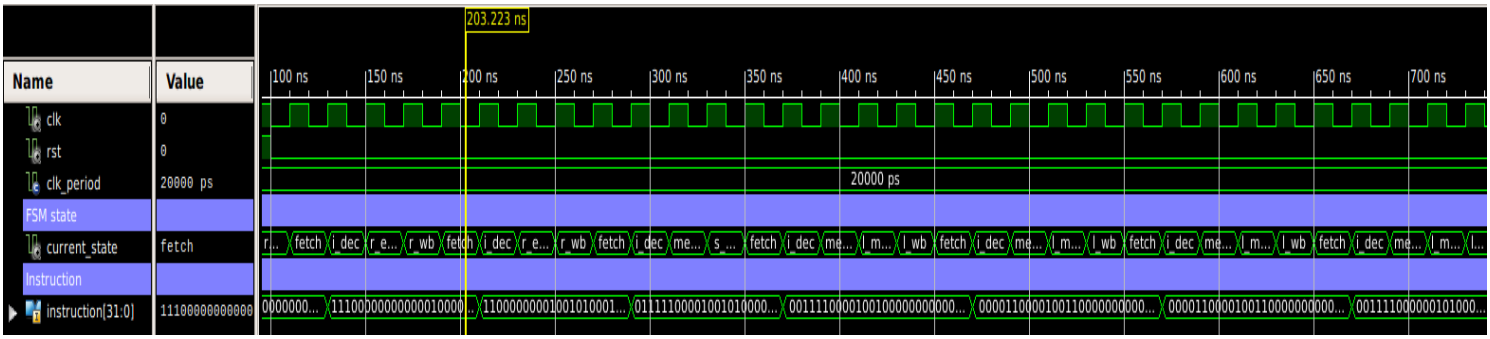
- **DECODE**: Στο decode προσθέσαμε ένα νέο module το equal_comperator, το οποίο συγκρίνει τα περιεχόμενα δύο καταχωρητών και αν είναι ίσα τότε επιστρέφει 1 αλλιώς 0. Ο σκοπός του είναι να αντικαταστήση της συνάρτηση compareVectors για τις BEQ, BNE.

- **DATAPATH**: Στο dapath προσθέσαμε καταχωρητές ώστε να κρατάμε τις τιμές που “ενώνουν” τα διάφορα stages. Συγκεκριμένα προσθέσαμε: register IR που ενώνει το IFSTAGE με το DECODE,

registers A_reg και B_reg που ενώνουν τα output του decode με το EXSTAGE, register E_reg που κρατάει την τιμή 1 όταν έχουμε ίσα outputs στο decode (ο συγκεκριμένος απλά πηγαίνει σαν output του datapath και ενώνεται με το control), register ALU_out_reg που κρατάει το αποτέλεσμα της ALU που χρησιμοποιήται στο MEMSTAGE και τέλος ο register MEM_out_reg που κρατάει το data που είναι για να γραφτεί σε κάποιο register του RF.

- **EQUAL_COMPERATOR:** Ένα απλό module που συγκρίνει δύο vectors ως ints . Αν είναι ίσα επιστρέφει 1 αλλιώς 0.

ΑΠΟΤΕΛΕΣΜΑΤΑ SIMULATION



Στο παραπάνω simuation τρέξαμε τις εξής εντολές. Στις πρώτες δύο εντολές επιβεβαιώνεται ότι για εντολές τις ALU έχουμε FETCH->I_DEC->R_EXEC->R_WB όπως περιμέναμε ενώ για τις εντολές 3 εώς 7 έχουμε FETCH->DEC->MEM_EXEC και μετά ανάλογα άν έχουμε read ή write στην μνήμη πάμε στο αντίστοιχο state. (Οι τιμές των καταχωρητών είναι ίδιες όπως και στην προηγούμενη αναφορά)

```
1. li $1, 4
2. addi $5, $1, 0
3. sw $5, 4($1)
4. lw $4, 4($1)
5. lb $6, 4($1)
6. lb $6, 5($1)
7. lw $10, 4($0)
```