ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
Οργάνωση Υπολογιστών

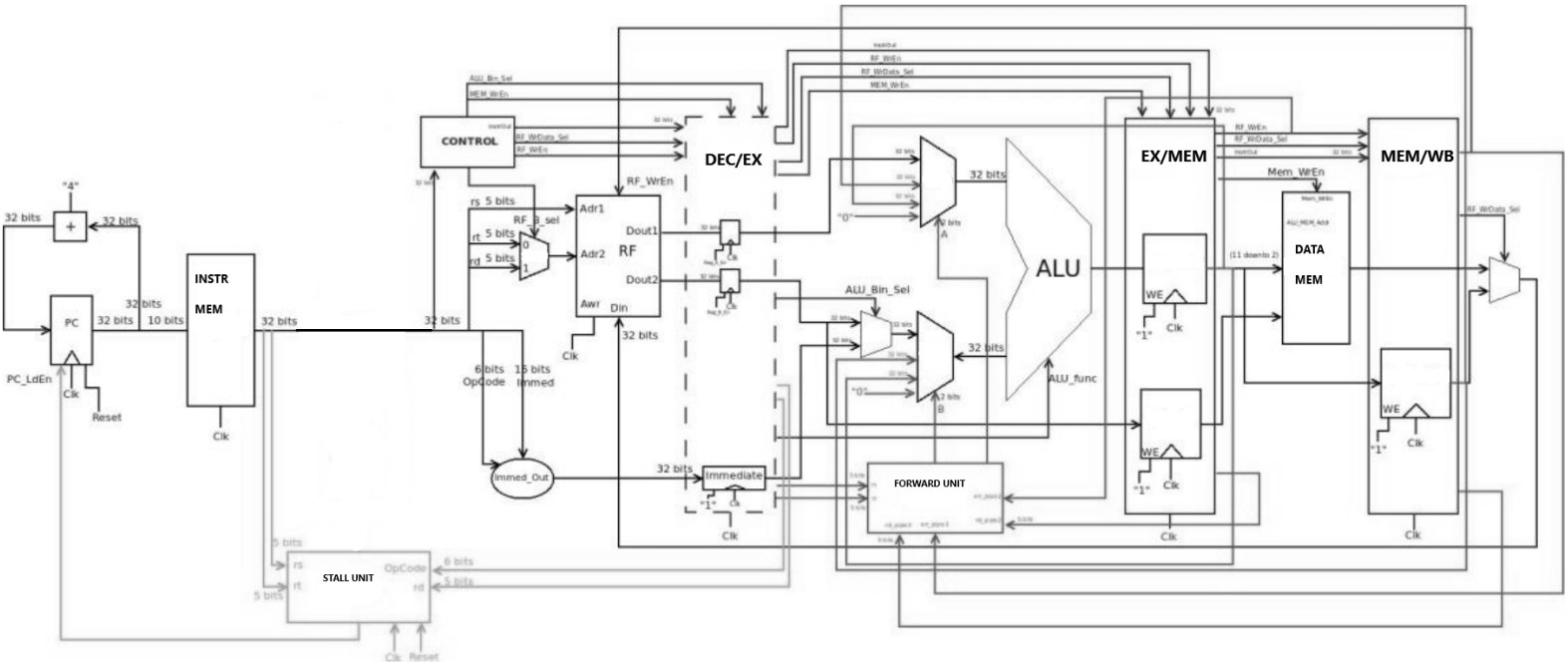
Ομάδα: LAB31235453
Μπαλαμπάνης Ηλίας 2014030127
Μποκαλίδης Αναστάσιος 2014030069

Αναφορά Εργαστηρίου 5

Περιγραφή Άσκησης

Σε αυτό το εργαστήριο μας ζητήθηκε να μετατρέψουμε τον επεξεργαστή μας σε έναν pipelined processor. Στην ουσία προσθέσαμε 3 επιπλέον καταχωρητές οι οποίοι υλοποιούν το pipeline. Τέλος σημαντικό να σημειωθεί πως πλεον το Top Module του Xilinx project είναι το Datapath καθώς το Control έγινε component του.

Παρακάτω παραθέτουμε το datapath σε block diagram, το οποίο βρίσκεται και στο .zip αρχείο.



- ***Control Unit***

Αλλάξαμε το Control Unit, έτσι ώστε να μην είναι μια FSM. Πλέον το Control υλοποιείται μέσα στο Datapath και τα σήματα, που είναι απαραίτητα για την υλοποίηση της εντολής, δίνονται όλα μαζί από αυτό με το γνωστό τρόπο του instruction fetch. Το RF_B_Sel δίνεται κατευθείαν στο Dec Stage και τα υπόλοιπα σήματα που βγαίνουν από το Control πάνε στον DEC/EX καταχωρητή. Από τον DEC/EX βγαίνουν τα σήματα ALU_Bin_Sel και ALU_func, τα οποία ελέγχουν την λειτουργία της ALU, και τα υπόλοιπα σήματα πάνε στον EX/MEM reg. Έπειτα από τον EX/MEM βγαίνει το σήμα MEM_WrEn, το οποίο πάει στην Data Memory, και τα υπόλοιπα σήματα πάνε στον MEM/WB. Τέλος, από το MEM/WB βγαίνουν το σήμα RF_WrData_Sel για την επιλογή του τι θα καταχωρήσει στην RF, το σήμα RF_WrEn και το rd, που πάει στο Dec Stage.

Αποσφαλμάτωση των Data Hazards

- ***Forward***

Δημιουργήσαμε το Forward Unit για να αντιμετωπίσουμε τα data hazards, που μπορεί να δημιουργηθούν κατά την εκτέλεση των εντολών. Το Forward υλοποιείται και σε one-cycle αλλά και σε two-cycle. Για έναν κύκλου ελέγχουμε από τον καταχωρητή MEM/WB το rd με τα rs και rt από τον καταχωρτή DEC/EX. Αν ένα από τα δύο ή και τα δύο είναι ίσα με το rd και το RF_WrEn από τον MEM/WB είναι ενεργό σε αυτή την περίπτωση δίνουμε τα κατάλληλα σήματα στους πολυπλέκτες στις εισόδους της ALU, δηλαδή κάνουμε one-cycle forward. Για να κάνουμε Forward για δύο κύκλους κάνουμε την ίδια ακριβώς διαδικασία με τον ένα κύκλο με τη διαφορά ότι αντί να παίρνουμε τα σήματα από τον MEM/WB τα παίρνουμε από τον EX/MEM. Οι εξόδοι του κουτιού αυτού είναι οι επιλογείς των πολυπλεκτών πριν την ALU.

- ***Stall***

Επίσης προσθέσαμε στο datapath μας το Stall Unit για την περαιτέρω αντιμετώπιση hazards. Σε αυτό το κομμάτι ελέγχουμε την τρέχουσα εντολή με την προηγούμενη. Αν η προηγούμενη είναι εντολή η lw και χρησιμοποιεί τον καταχωρητή, που φορτώνει, στην τρέχουσα (add) δημιουργούμε μία καθυστέρηση δύο κύκλων. Αυτό το κάνουμε για να μπορέσει να πάρει την πληροφορία από τη μνήμη και να τη χρησιμοποιήσει στην add. Για να σιγουρευτούμε ότι δεν θα γίνει παράλληλα με το stall και το forward, γίνεται ο κατάλληλος έλεγχος στο forward unit. Η έξοδος αυτού του κουτιού είναι το PC_LdEn. Επίσης εδώ ελέγχεται και το εξωτερικό σήμα reset, μέσω της μικρής αυτής FSM.

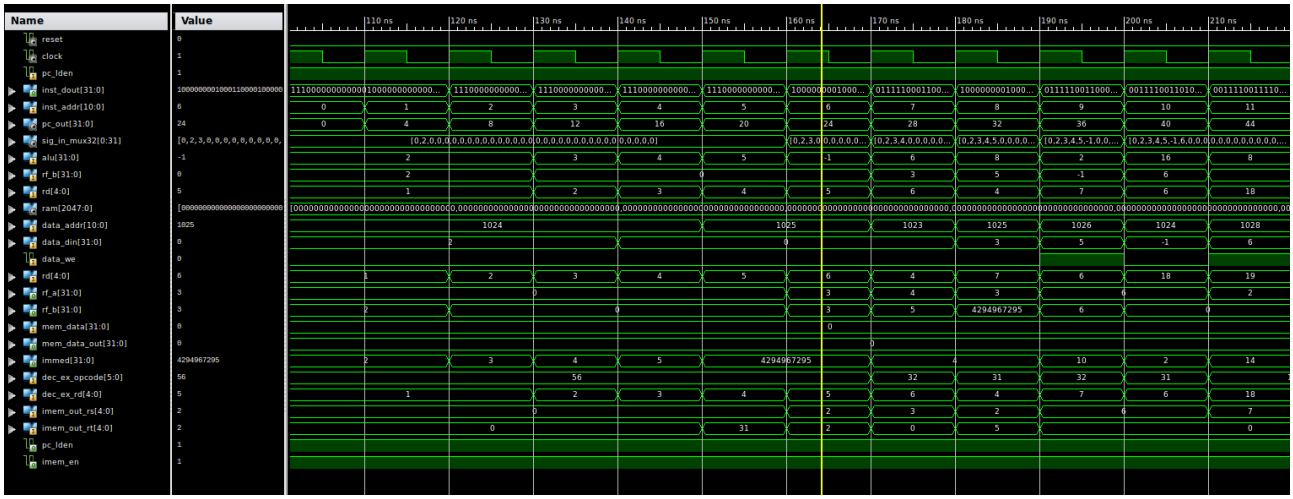
Κυμματομορφές

Παρακάτω βλέπουμε τις εικόνες του simulation για να επιβεβαιώσουμε τη σωστή λειτουργία του pipeline processor.

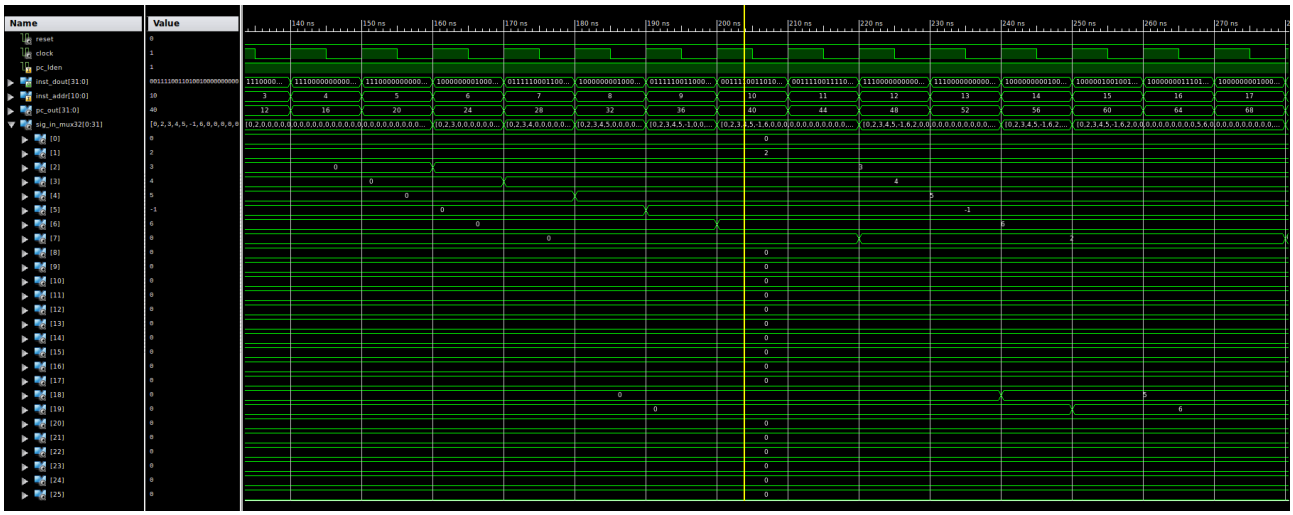
Στη συνέχεια, βλέπουμε το simulation στην αντιμετώπιση των data hazards

- **Εκτέλεση χωρίς data hazards**

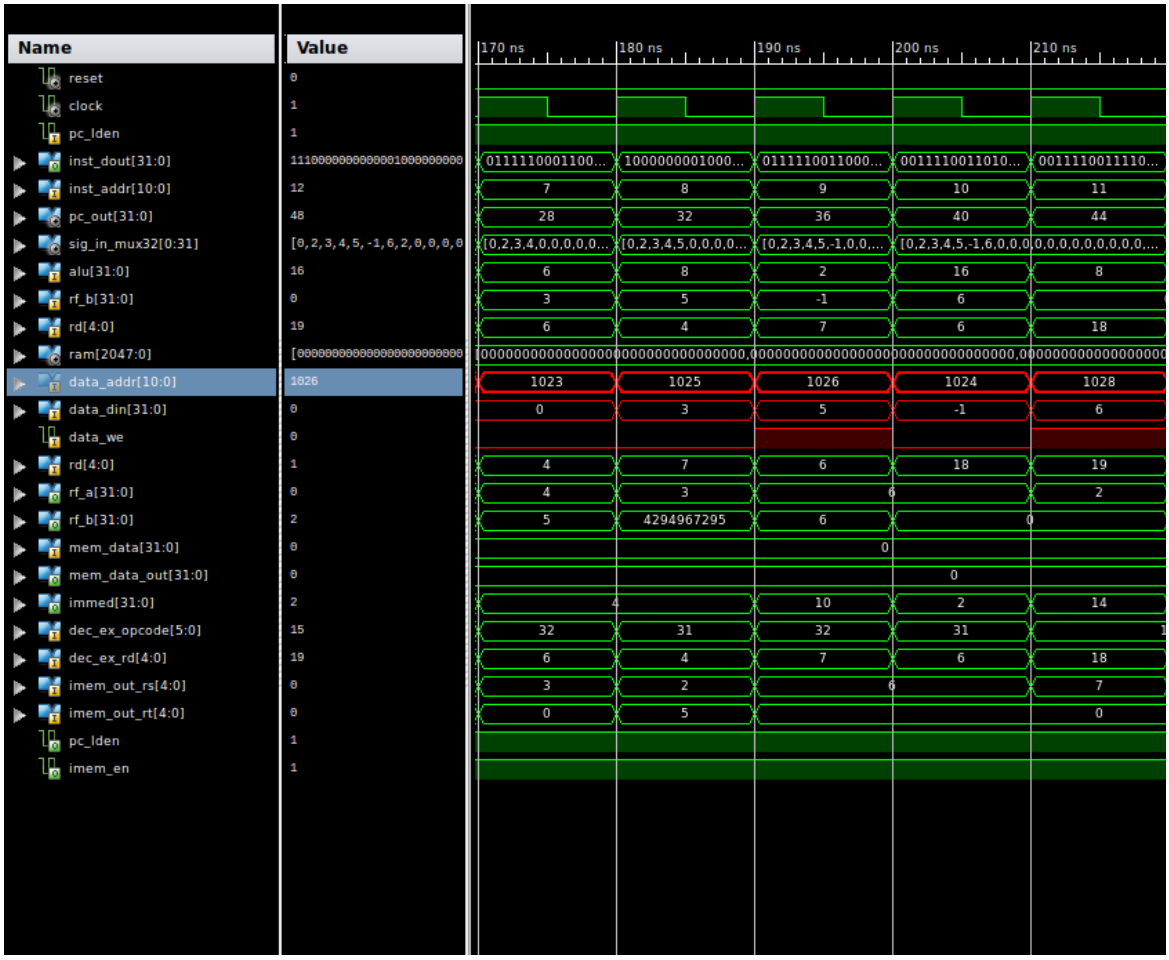
Βλέπουμε δηλαδή πως στον καταχωρητή $1, καταχωρείται το 2, στον $2 το 3 κ.ο.κ , σύμφωνα με τις εντολές που δίνονται στο αρχείο ram.data. Παρατηρούμε το sw, οπου MEM[1026]= 5, ομοίως και για την lw παρακάτω.



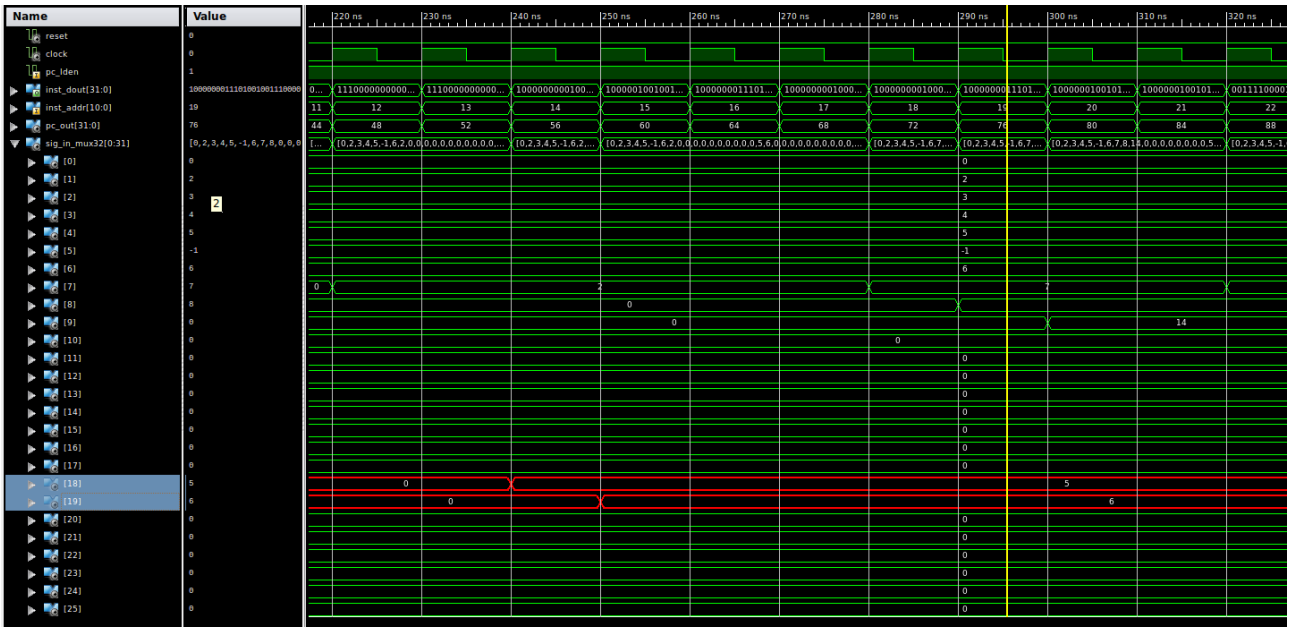
Οι καταχωρητές μετά την εκτέλεση των εντολών



Τα SW


Name
Value
170 ns
180 ns
190 ns
200 ns
210 ns
reset
clock
pc_lden
inst_dout[31:0]
inst_addr[10:0]
pc_out[31:0]
sig_in_mux32[0:31]
alu[31:0]
rf_b[31:0]
rd[4:0]
ram[2047:0]
data_addr[10:0]
data_din[31:0]
data_we
rd[4:0]
rf_a[31:0]
rf_b[31:0]
mem_data[31:0]
mem_data_out[31:0]
immed[31:0]
dec_ex_opcode[5:0]
dec_ex_rd[4:0]
imem_out_rs[4:0]
imem_out_rt[4:0]
pc_lden
imem_en

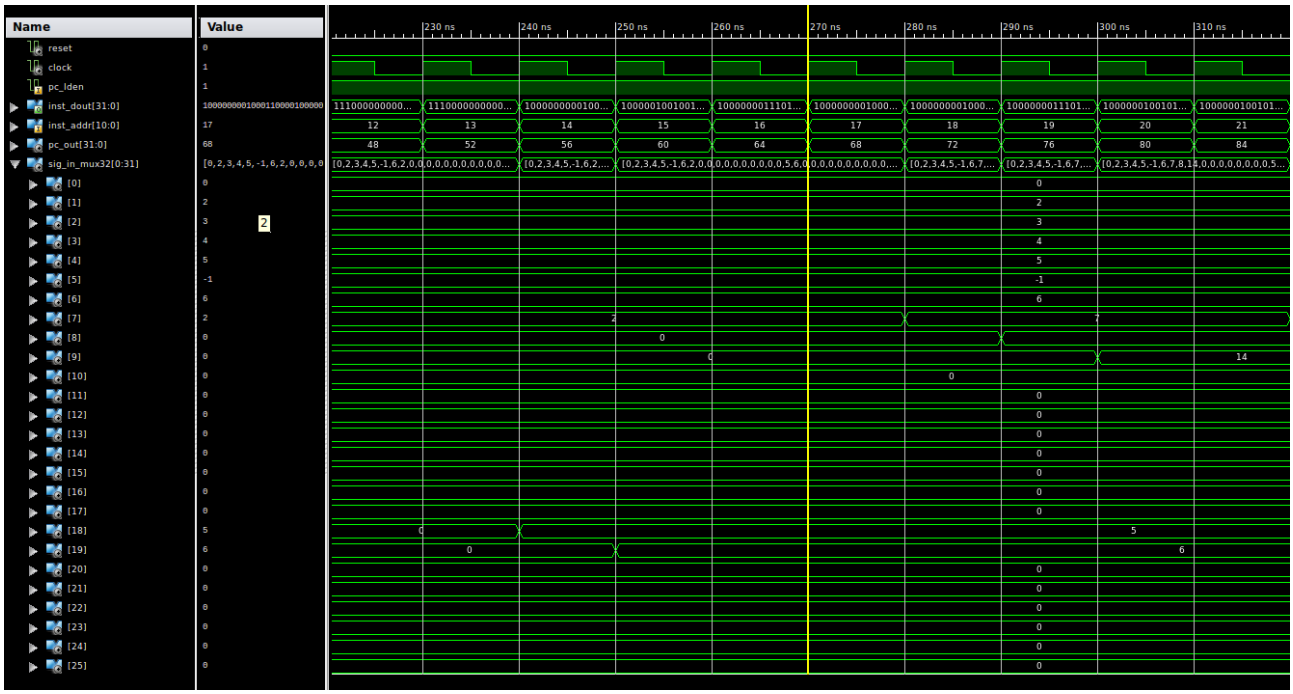
Τα LW


Name
Value
220 ns
230 ns
240 ns
250 ns
260 ns
270 ns
280 ns
290 ns
300 ns
310 ns
320 ns
reset
clock
pc_lden
inst_dout[31:0]
inst_addr[10:0]
pc_out[31:0]
sig_in_mux32[0:31]

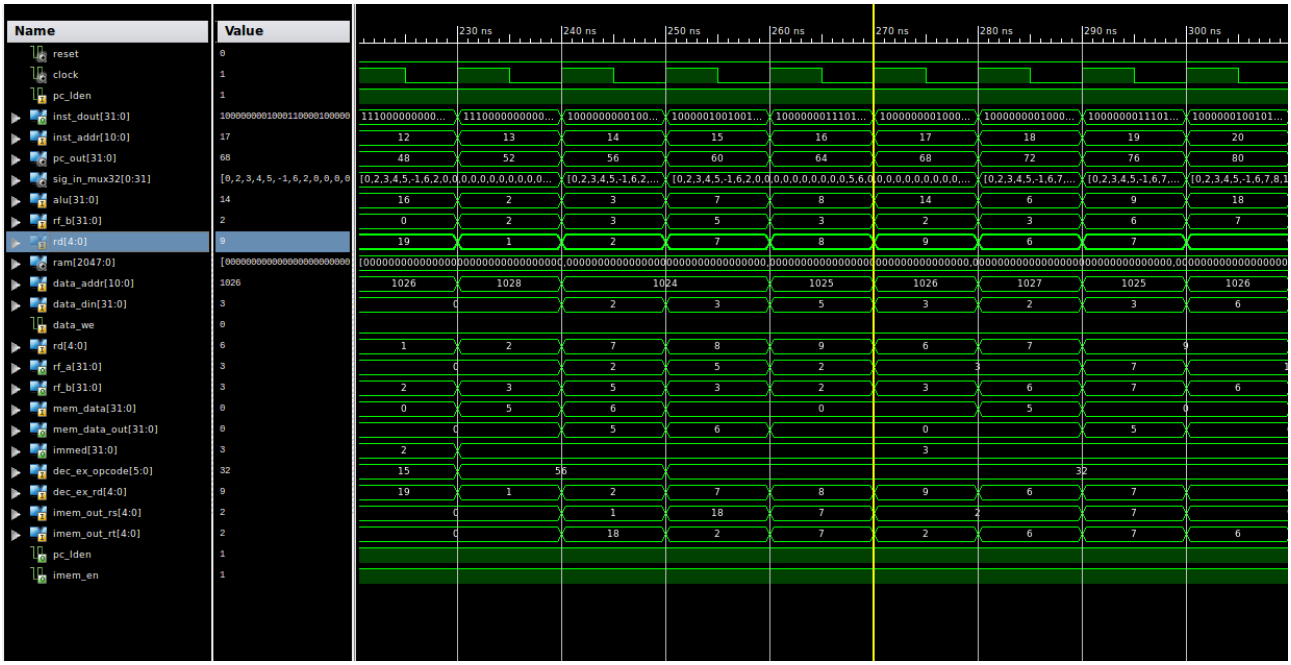
- ***Forward***

Βλέπουμε πως στον καταχωρητή $1 καταχωρείται το 2, στον $2 το 3, έπειτα γίνονται σωστά οι εντολές μέχρι να γίνει η καταχώρηση στον $9 της πρόσθεσης του $7 με τον $7, δηλαδή 14. Στις επόμενες εκτελέσεις της add χρησιμοποιούνται καταχωρητές από δύο κύκλους πριν (forwarding within two cycles). Ενώ στις επόμενες εκτελέσεις της add μέχρι την καταχώρηση στον $10 το 48 χρησιμοποιούνται καταχωρητές από ένα κύκλο πριν (forwarding within one cycle).

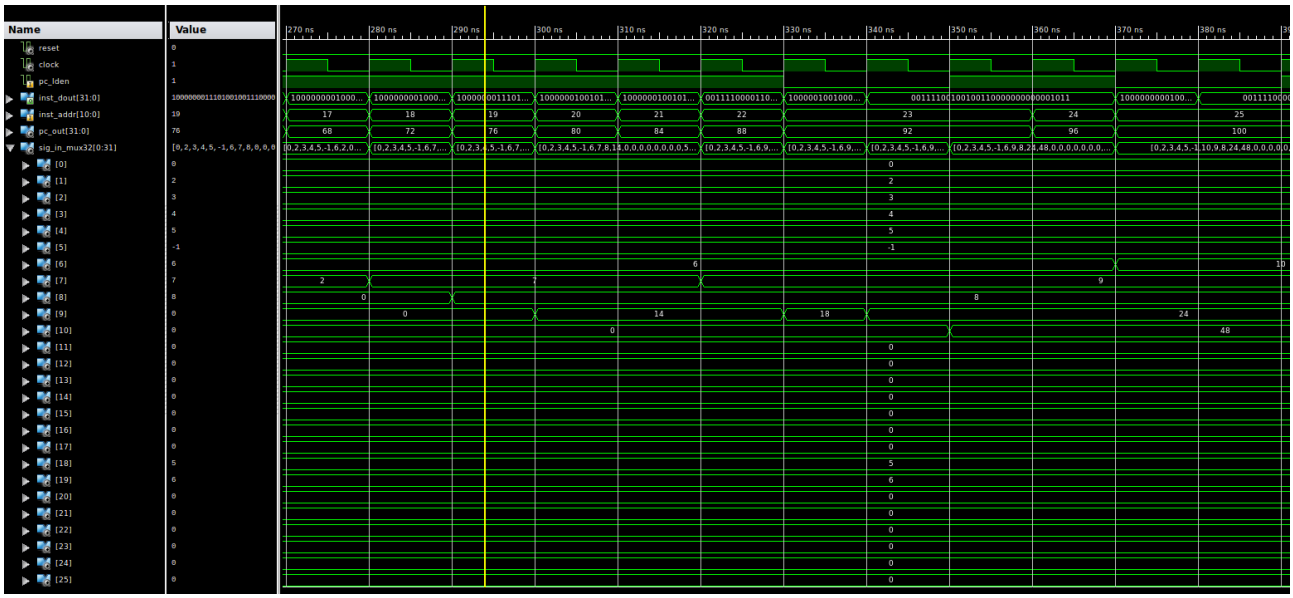
Two cycle regs



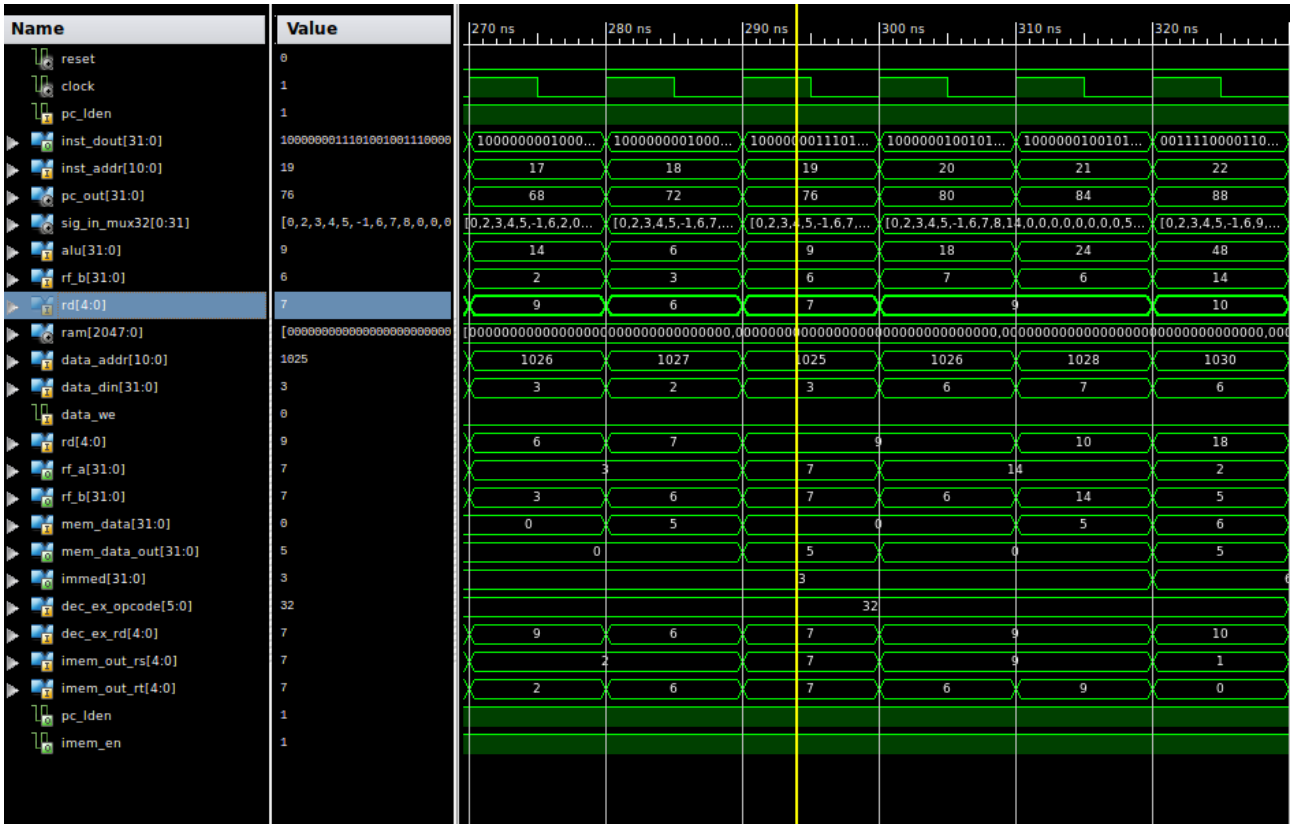
Two cycle run-time



One cycle regs

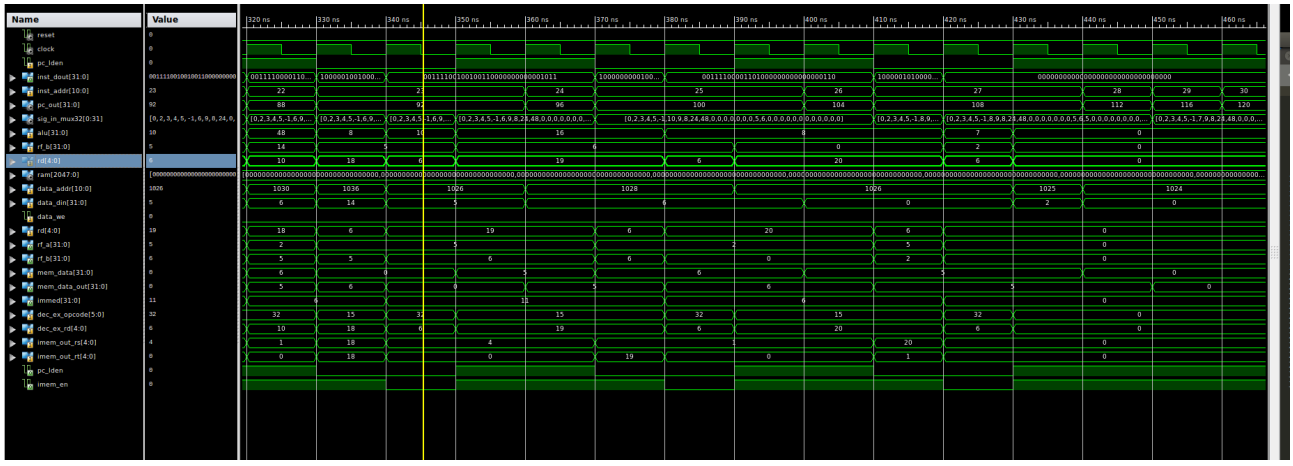


One cycle run-time



- ***Stall***

Στο Simulation αυτό βλέπουμε διαδοχικά την εκτέλεση των εντολών lw και add με stalling. Στον $18(MEM[1026]) καταχωρείται το 5, στον $6 το 10, στον $19 (MEM[1028]) το 6, στον $6 το 8, στον $20(MEM[1026]) το 5 και στον $6 το 7. Στο stall γίνεται καθυστέρηση εντολής για να μπορεί να εκτελεστεί σωστά η επόμενη.



Οι καταχωρητές μετά το Stall