ΟΡΓΑΝΩΣΗ ΥΠΟΛΟΓΙΣΤΩΝ
ΗΡΥ 312

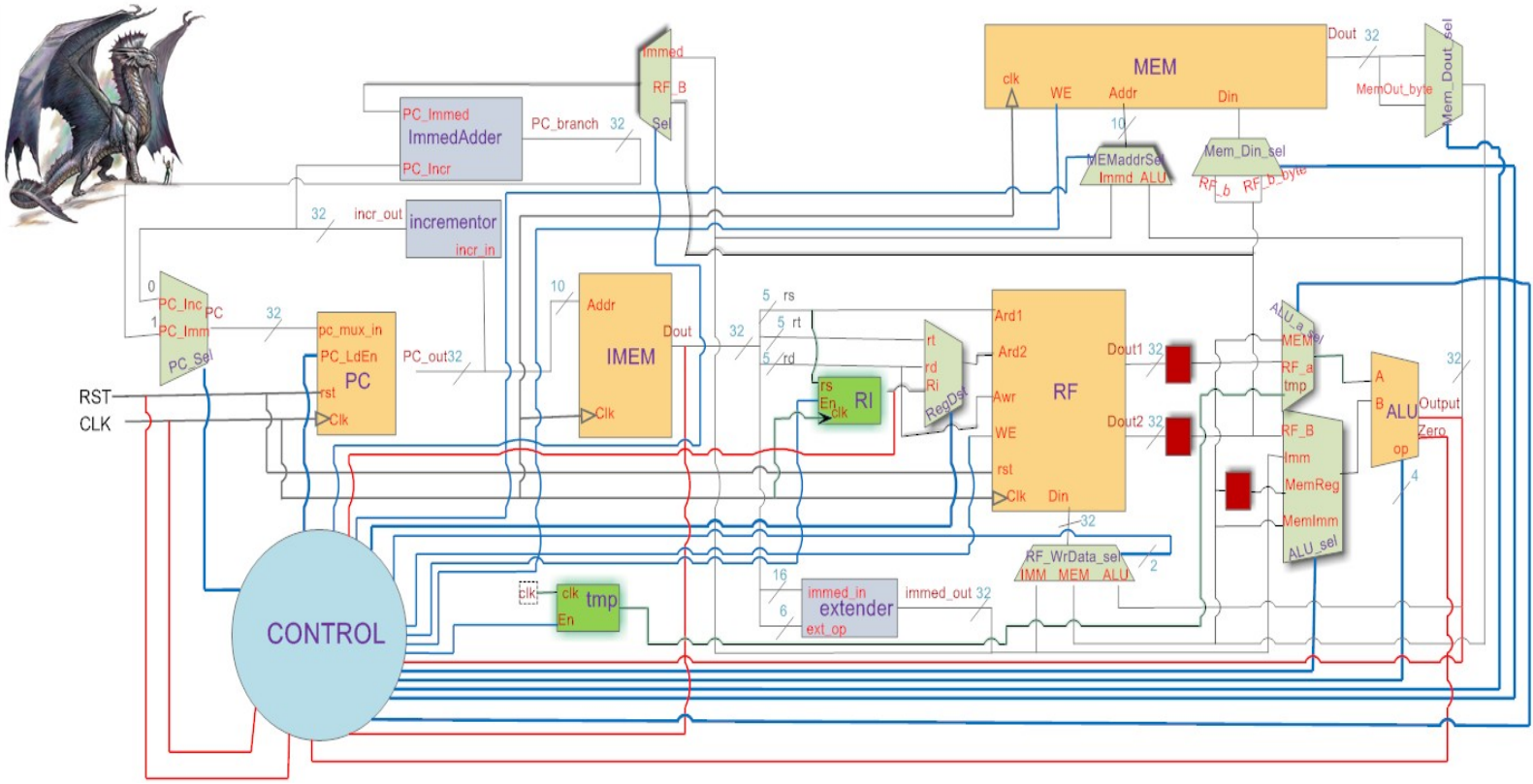
ΕΡΓΑΣΤΗΡΙΟ 4
Αναφορά Εργαστηριακής Άσκησης

Ομάδα Εργασίας **LAB31220188**

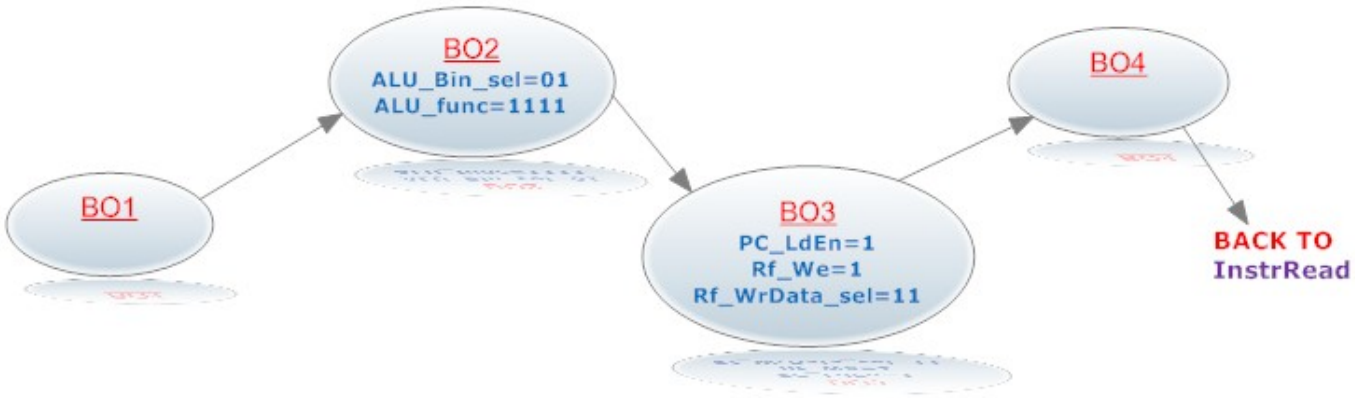
α/α	Α.Μ.	Ονοματεπώνυμο
1	2011030010	Χριστοδουλου Θεοφιλος
2	2011030009	ΚΑΡΙΜΠΙΔΗΣ ΔΙΟΝΥΣΗΣ

Στο 4ο εργαστήριο επεκτείναμε την λειτουργία του 3ου ώστε να υποστηρίζει ακόμη κάποιες πιο σύνθετες εντολές. Οι εντολές που προσθέσαμε αλλά και τα format τους είναι οι εξής:
Load_byte_BO (I-format), **AddLE** (R-format), **Add_Mem** (I-format), **Branch_Equal_Reg-Mem** (R-format), **Store_registers** (I-format)

Και ακολουθεί το DATAPATH της υλοποίησης..
(στο σχήμα αυτό δεν εμφανίζεται ο register μετά την έξοδο της alu αλλά και το ldBO_module που περιγράφουμε παρακάτω)



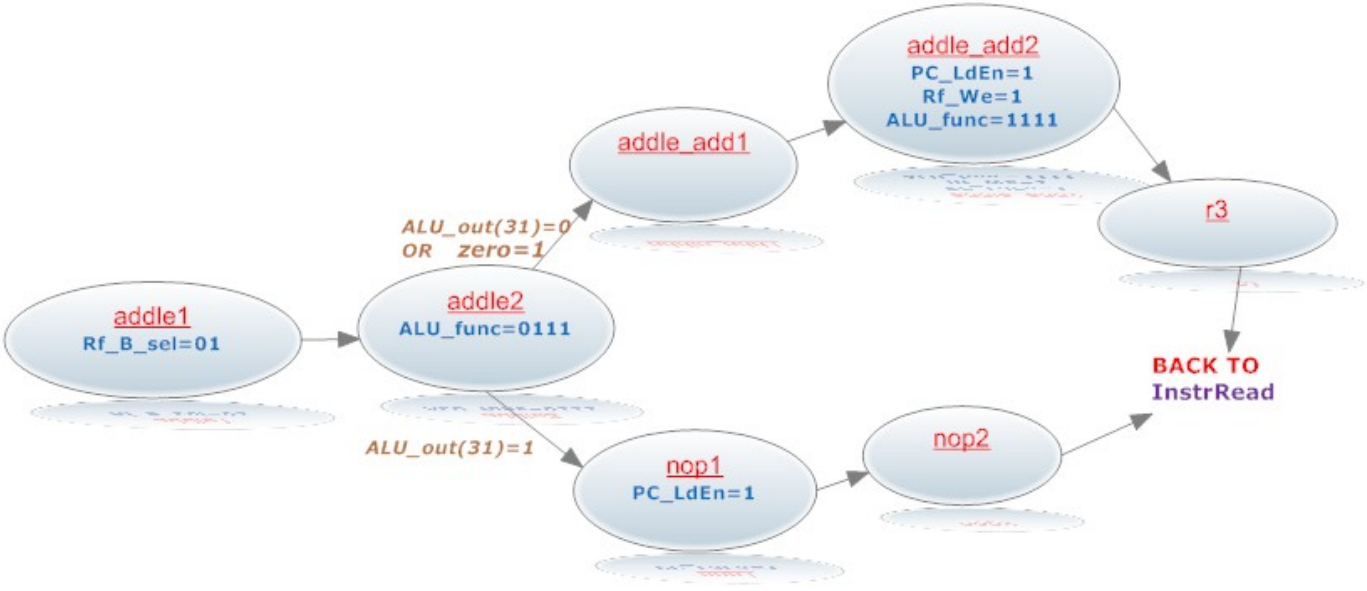
1)LOAD_BYTE_BO



*Τα σήματα που δεν εμφανίζονται βρίσκονται στη default κατάστασή τους.

Η εντολή αυτή εκτελείται σε 5 κύκλους και για να την υλοποιήσουμε προσθέσαμε έναν καταχωρητή ο οποίος “κρατάει” την τιμή του byte offset που είναι τα 2lsb της διεύθυνσης της μνήμης. Η πληροφορία αυτή παράγεται στην alu οπότε συνδέσαμε την έξοδό της στον καταχωρητή που δημιουργήσαμε,διότι το περιεχόμενο αυτής της θέσης μνήμης θα παραχθεί στον επόμενο κύκλο. Η έξοδος του καταχωρητή αυτού όπως και η έξοδος της μνήμης γίνονται οι είσοδοι σε ένα νέο module το οποίο λειτουργεί ασύγχωνα και ουσιαστικά διαλέγει ανάλογα το byte offset ποιο byte θα γράψει πίσω στην RF. Τέλος προσθέσαμε στον πολυπλέκτη ο οποίος επιλέγει τι θα γράψει στην RF ακόμη μια είσοδο που προέρχεται απο το module που περιγράψαμε.
(Το εν λόγω module δε φαίνεται στην εικόνα του datapath παρακάτω,αλλά υπάρχει στον κώδικα της υλοποίησης.)

2)ADDLE



*Τα σήματα που δεν εμφανίζονται βρίσκονται στη default κατάστασή τους.

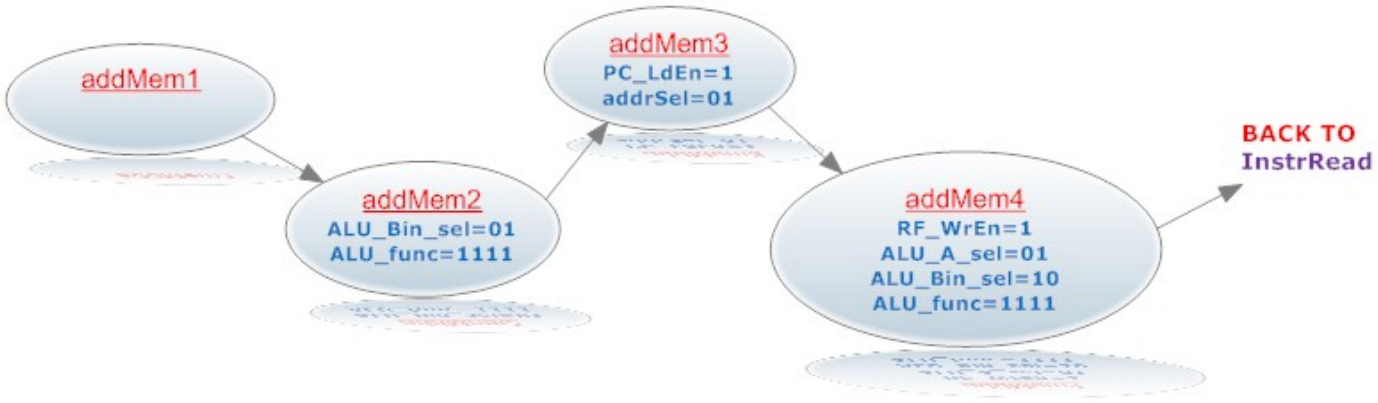
Χριστοδούλου Θεόφιλος **A.M.: 2011030010**

Καριμπίδης Διονύσης **A.M.: 2011030009**

Μάιος 2014

Αρχικά επιλέγουμε στον πολυπλέκτη που διαλέγει την 2η διεύθυνση ανάγνωσης της RF το σήμα RegDst (rf_b_sel στον κώδικά μας) να είναι '1' ώστε να διαβάσουμε το περιεχόμενο της διεύθυνσης RF[rd]. Στην συνέχεια δίνουμε τα κατάλληλα σήματα στην alu ώστε να εκτελέσει μια αφαίρεση ανάμεσα στο RF[rs]-RF[rd]. Αν zero = '1' τότε οι τιμές είναι ίσες άρα εκτελούμε την εντολή add στους επόμενους κύκλους, αλλιώς αν το msb(alu_output) = '1' τότε το RF[rd] είναι μεγαλύτερο του RF[rs] άρα πάμε στην κατάσταση nop1 ώστε να εκτελέσουμε την nop, αλλιώς το RF[rd] είναι μικρότερο του RF[rs] οπότε πάλι εκτελούμε μια add.
Για την εκτέλεση της εντολής add *προσθέσαμε απλώς νέες καταστάσεις αντί να μεταβούμε σε ήδη υπάρχουσες και να χρειαστεί να τις αλλάζουμε.*

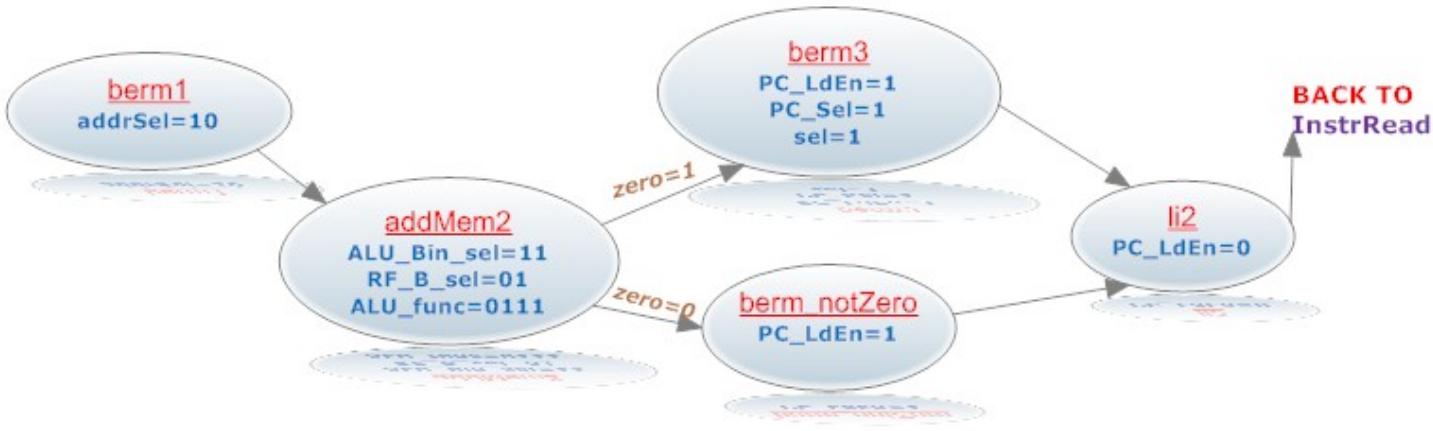
3) ADD_MEM



**Τα σήματα που δεν εμφανίζονται βρίσκονται στη default κατάστασή τους.*

Για την υλοποίηση αυτής της εντολής προσθέσαμε έναν καταχωρητή ο οποίος κρατάει την έξοδο της μνήμης MEM[RF[rs]. Στον επόμενο ακριβώς κύκλο έχει παραχθεί και το MEM[RF[rs] + SignExtend(Imm) οπότε διαλέγουμε τα κατάλληλα σήματα ώστε να γίνει η πρόσθεση μεταξύ αυτών των 2 τιμών και να γραφτεί πίσω στην RF.

4)BRANCH_EQUAL_REG-MEM



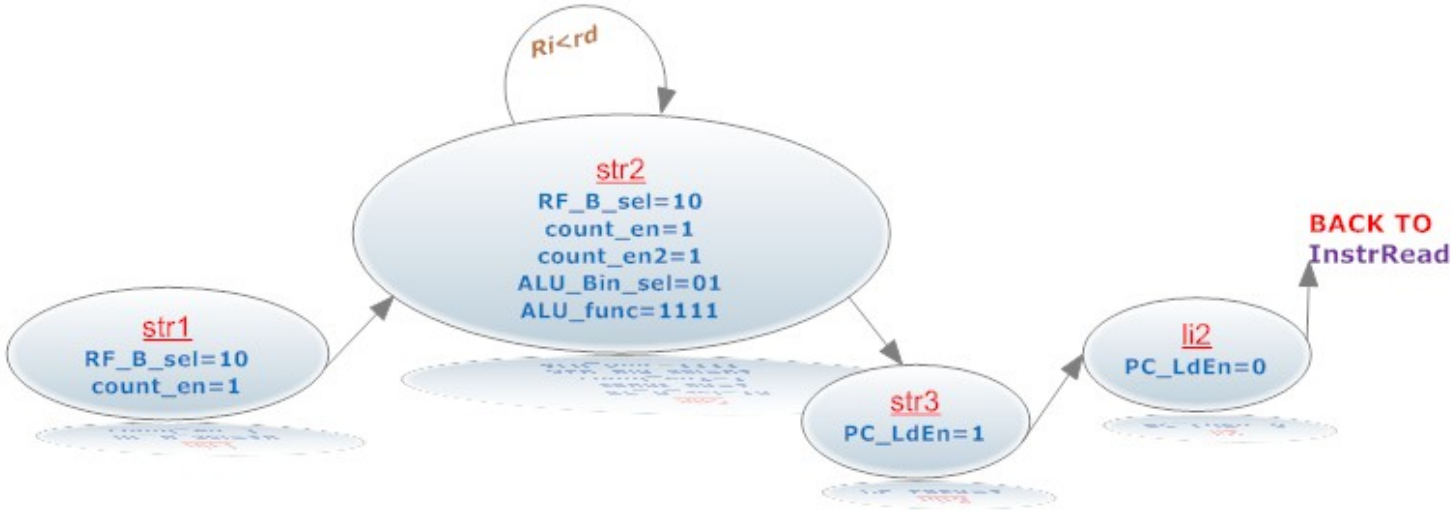
**Τα σήματα που δεν εμφανίζονται βρίσκονται στη default κατάστασή τους.*

Χριστοδούλου Θεόφιλος **A.M.: 2011030010**

Καριμπίδης Διονύσης **A.M.: 2011030009**

Προσθέσαμε πριν την είσοδο Addr της μνήμης έναν πολυπλέκτη ο οποίος επιλέγει εάν η διεύθυνση που θα διαβάσει η μνήμη θα προέρχεται από την έξοδο της alu ή κατευθείαν απο το immediate. Έτσι στην περίπτωση αυτής της εντολής περιμένουμε ένα κύκλο ώστε να παραχθεί το MEM[Imm << 2] αλλά και το RF[rs]. Διαλέγουμε πάλι τα κατάλληλα σήματα ώστε να κάνουμε μια αφαίρεση μεταξύ αυτών των 2 τιμών και ανάλογα το σήμα zero επιλέγουμε αν θα πάμε στην επόμενη εντολή ή όσες εντολές ορίζονται απο το αποτέλεσμα PC + 4 + RF[rt]. Για να υπολογίσουμε αυτό το αποτέλεσμα έχουμε προσθέσει έναν πολυπλέκτη ο οποίος μόνο στις περιπτώσεις που είναι επιτυχημένη η **Branch_Equal_Reg-Mem** επιλέγει το RF[rt] ως είσοδο του module ImmedAdder και αντίστοιχα ενεργοποιούμε το κατάλληλο pc_sel.

5) STORE_REGISTERS



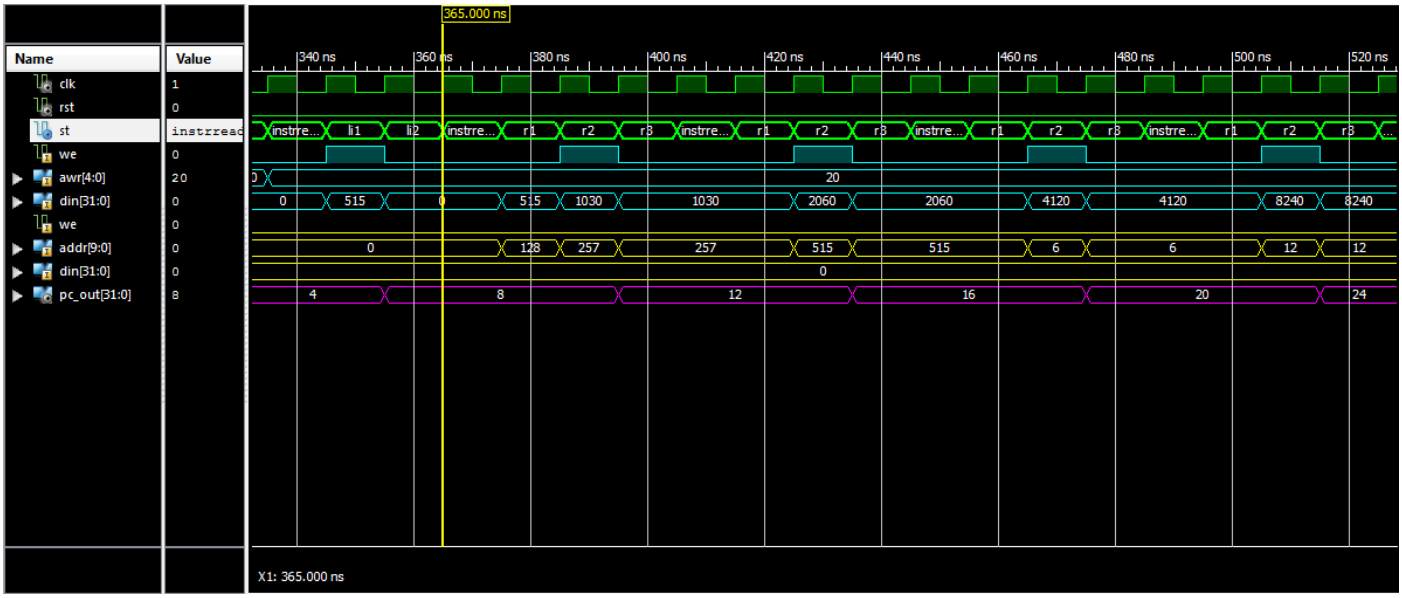
*Τα σήματα που δεν εμφανίζονται βρίσκονται στη default κατάστασή τους.

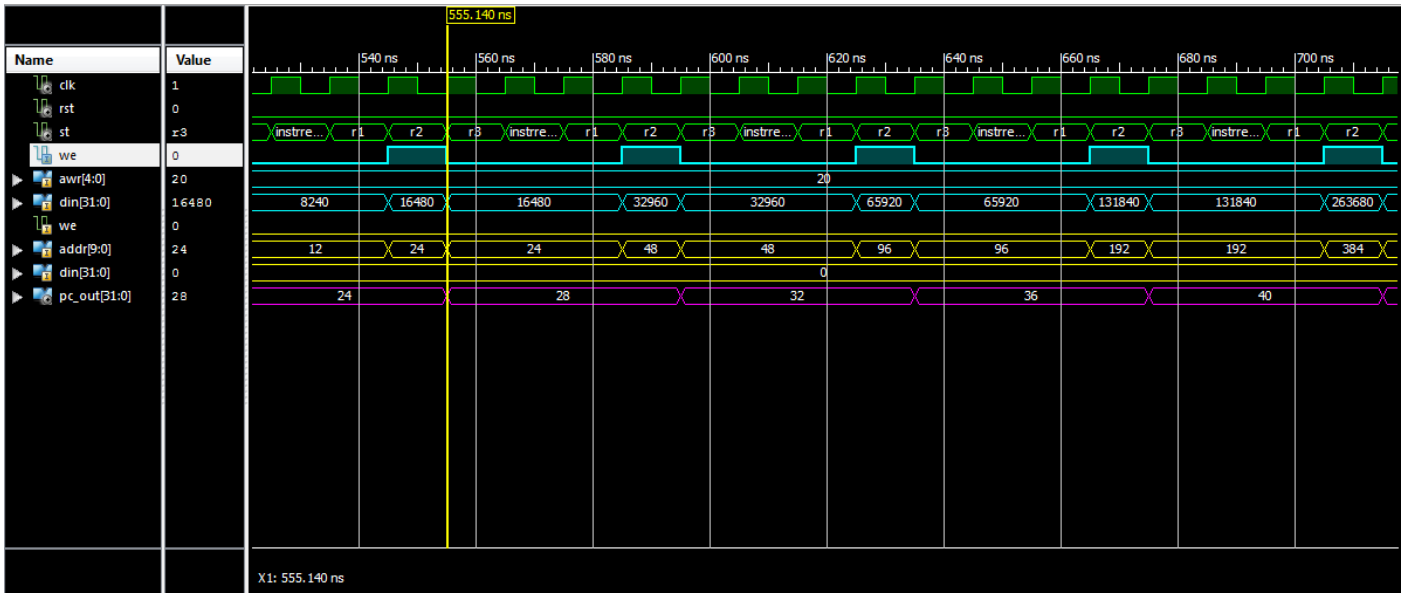
Σε αυτήν την εντολή χρειαστήκαμε 2 modules που ουσιαστικά λειτουργούν σαν counters. Ο ένας ξεκινάει απο το 0 και αυξάνεται βγάζοντας ώς αποτέλεσμα σε κάθε κύκλο ρολογιού το temp+1 και ο άλλος ξεκινάει απο την τιμή rs και αυξάνεται αντίστοιχα κάθε φορά υπολογίζοντας το καινούριο i. Και οι 2 αυτοί counters διαθέτουν ένα σήμα enable. Έτσι ενεργοποιούμε ένα χρόνο πιο πριν τον counter Ri διότι για να βγει το αποτέλεσμα που επιθυμούμε πρέπει να περάσει ένας κύκλος, μιας και περνάει μέσα απο την RF, και στον ακριβώς επόμενο κύκλο ενεργοποιούμε και το enable του temp. Η έξοδος του tmp γίνεται είσοδος στον πολυπλέκτη της A εισόδου της alu όπου και πραγματοποιούμε την πρόσθεση μεταξύ Immed + ($temp << 2) που μας δίνει την θέση διεύθυνσης της μνήμης που θα γράψουμε την τιμή RF[i].

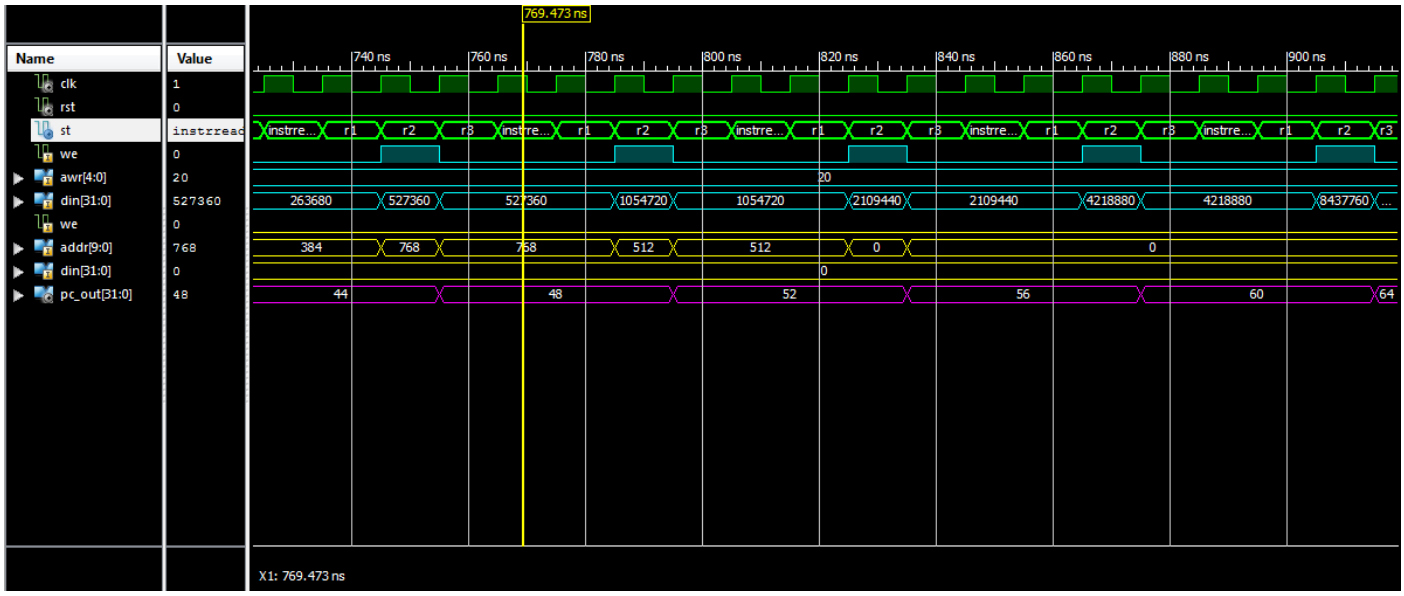
Simulation:

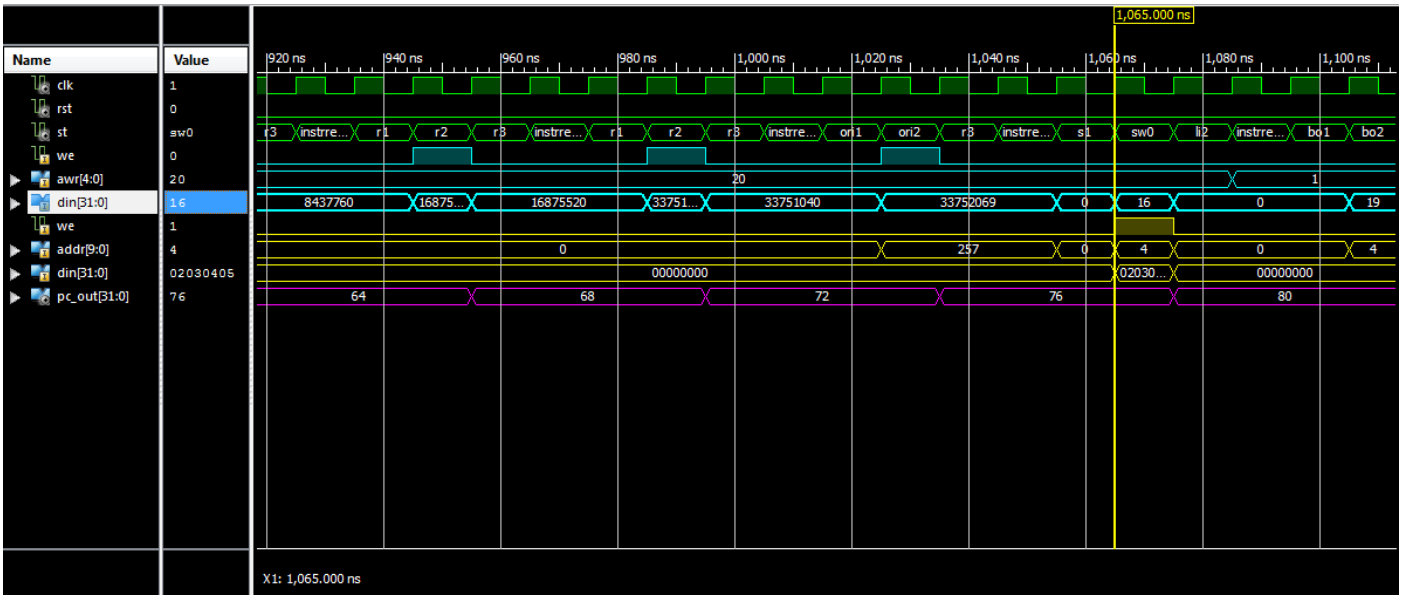
(με γαλάζιο εμφανίζονται τα σήματα της RF, με κίτρινο της μνήμης και με μοβ του pc_counter)

Οι πρώτες 4 εμφανίζουν την πρώτη li και όλες τις επόμενες shl όπου ουσιαστικά διπλασιάζουμε την τιμή του καταχωρητή 20





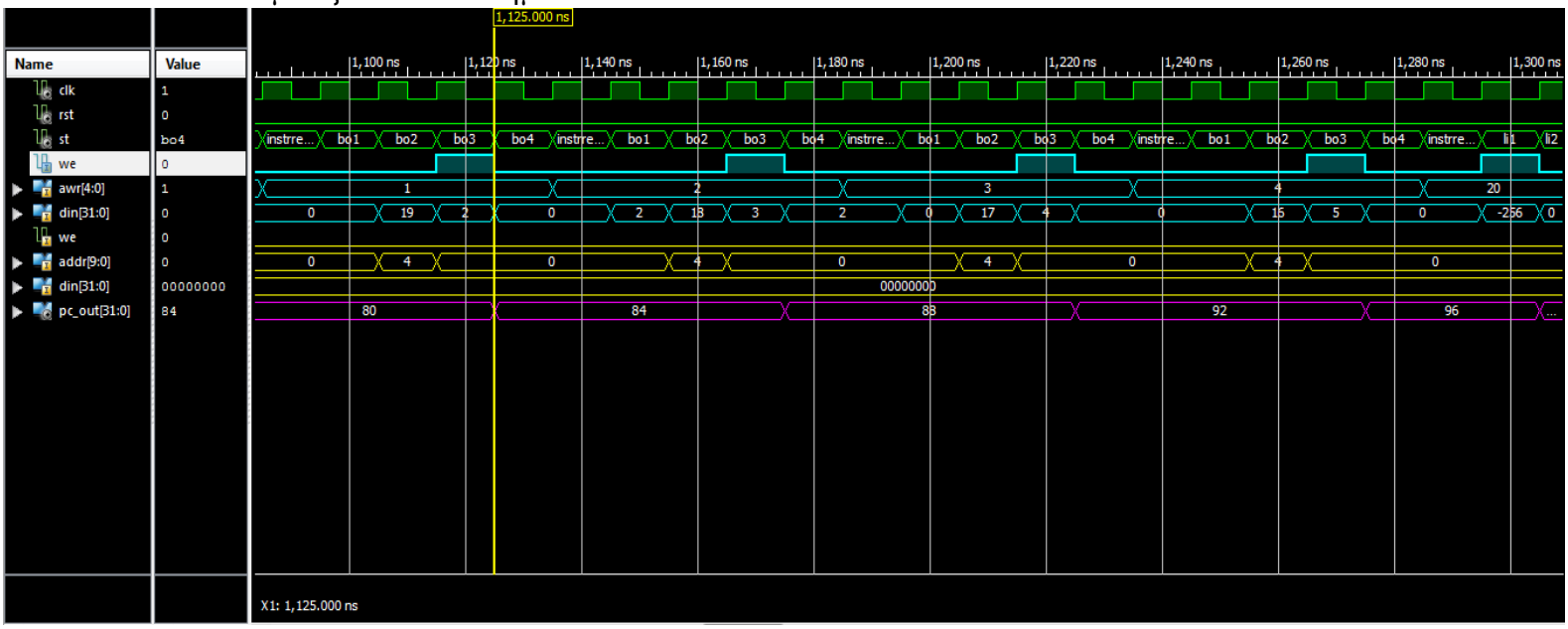




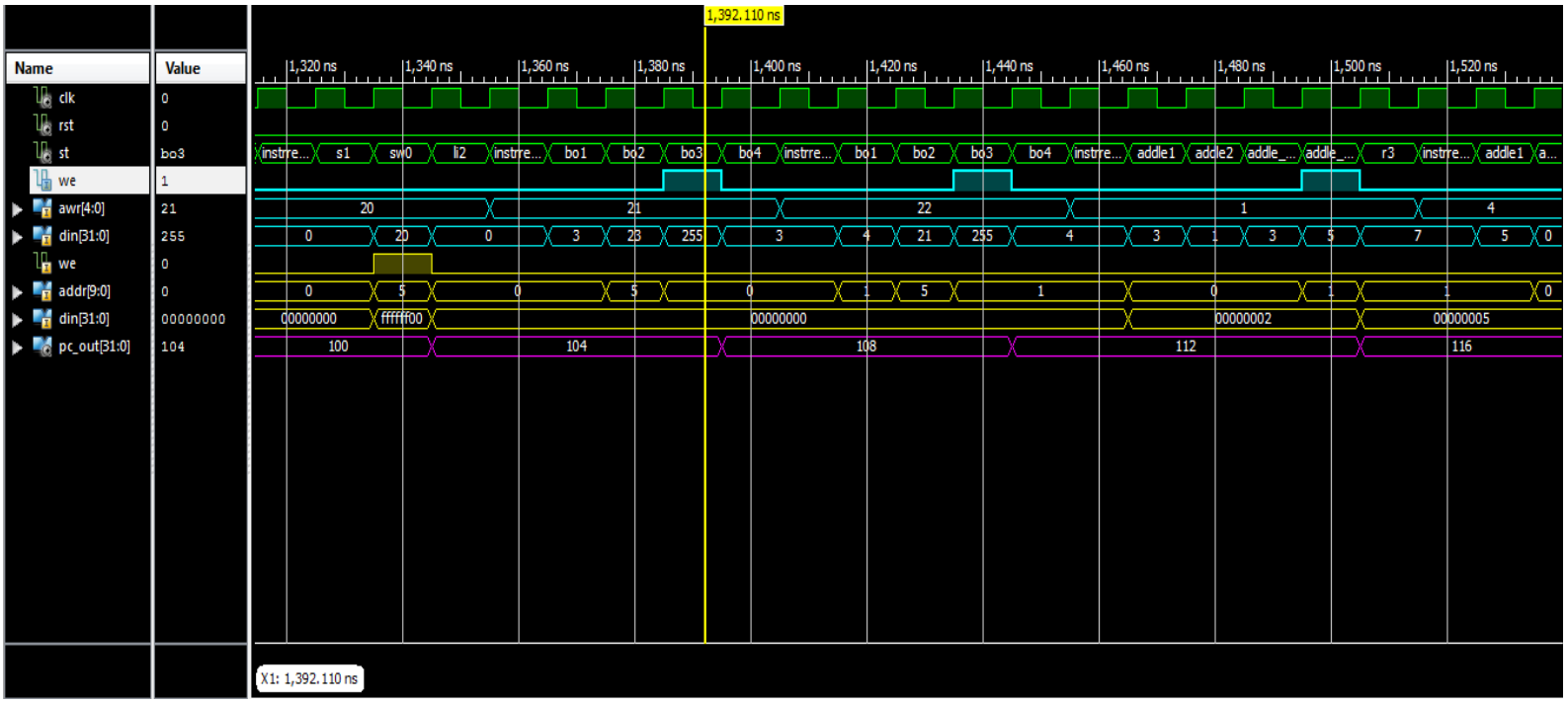
Χριστοδούλου Θεόφιλος **A.M.: 2011030010**

Καριμπίδης Διονύσης **A.M.: 2011030009**

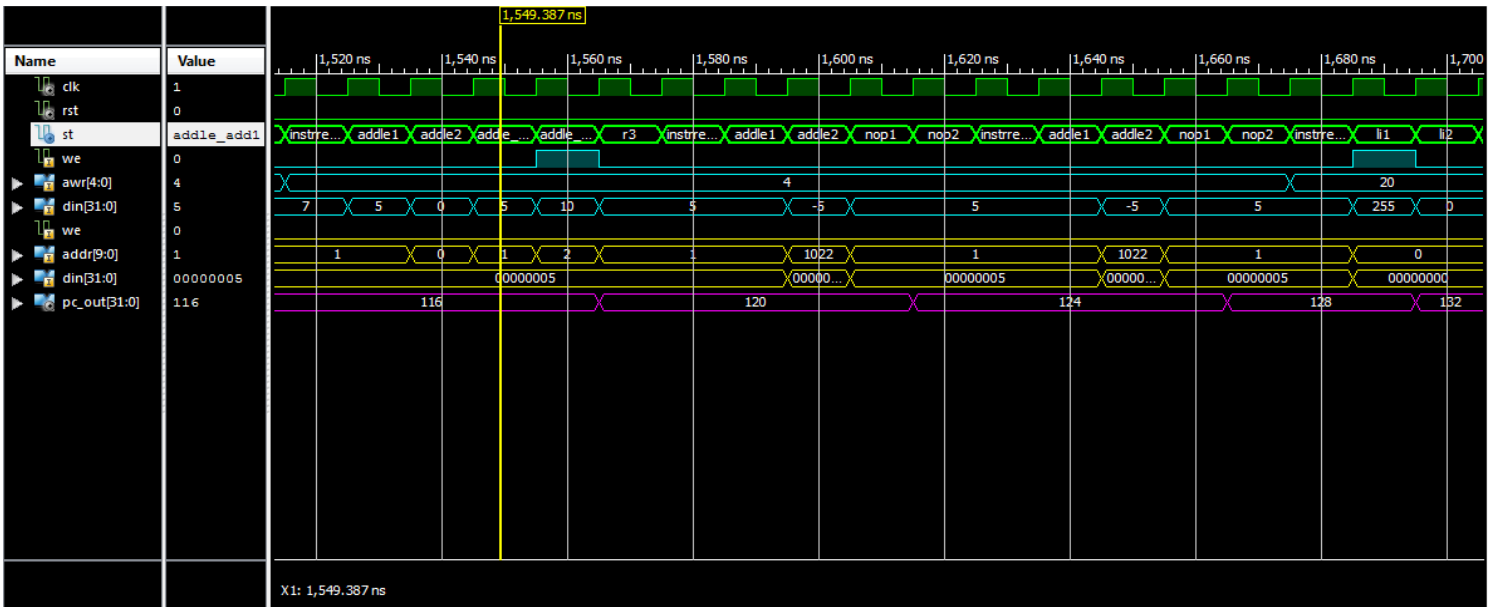
παρατηρούμε ότι για τις εντολές ldBO στον καταχωρητή 1 εχουμε την τιμή 2, στον 2 την 3,στον 3 την 4 και στον 4 την 5. Επίσης η τιμη -256 που εμφανίζεται να γράφεται στον 20 είναι η ffffff00 σε δεμαεξαδικό σύστημα.



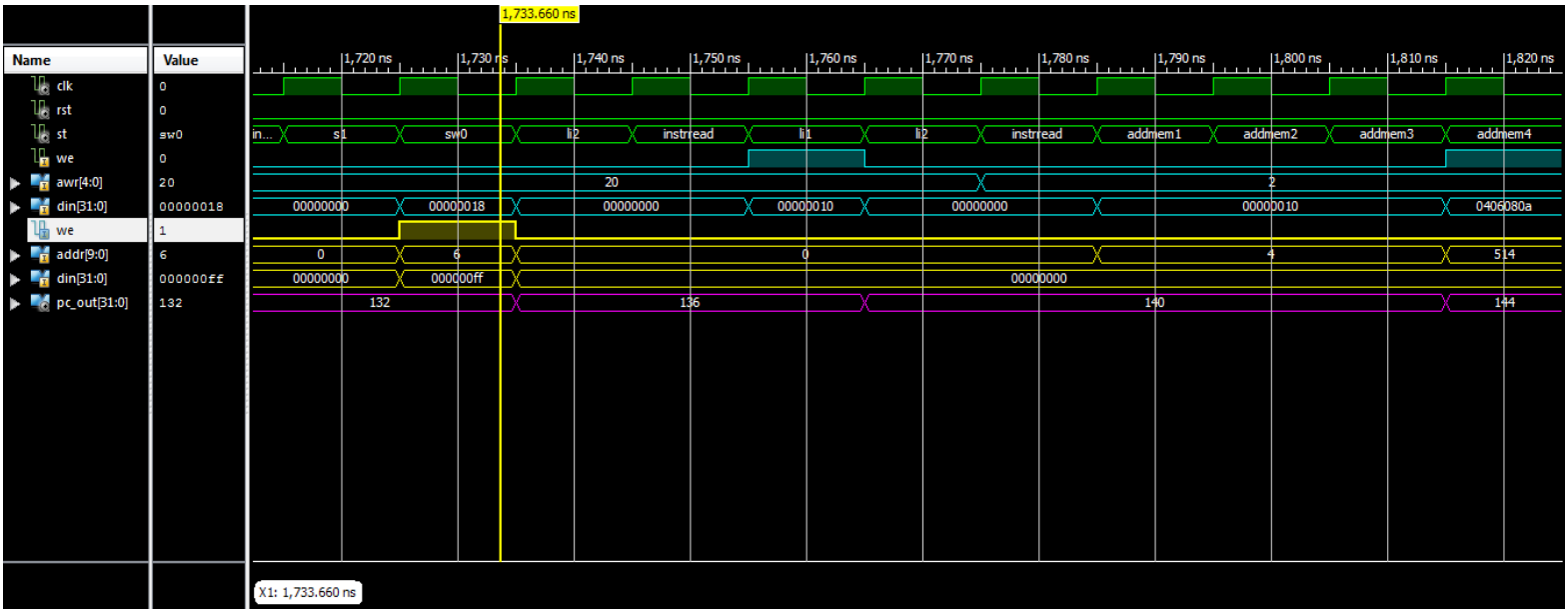
Έχουμε στην διεύθυνση μνήμης 5 την επιθυμητή τιμή και τις αντίστοιχες τιμές στους καταχωρητές 21, 22 και 1



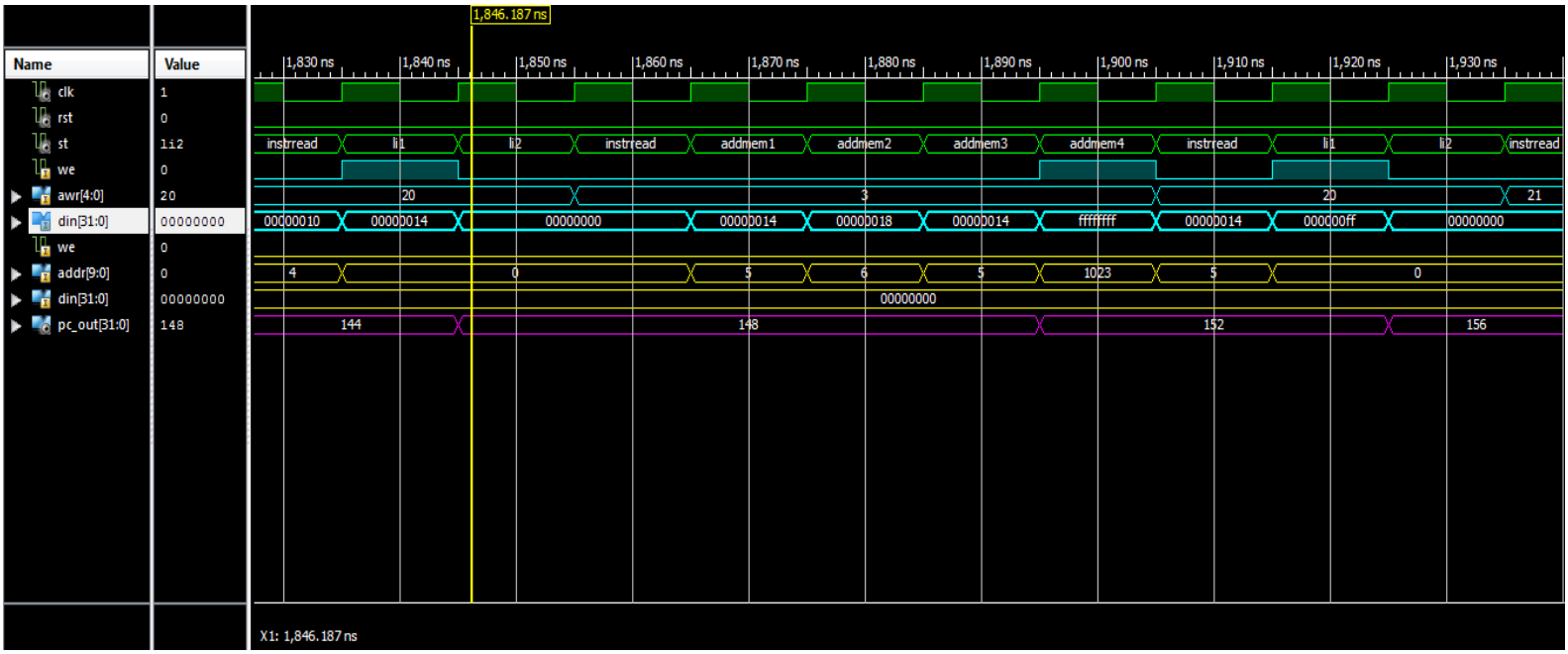
Στην περίπτωση αυτή φαίνεται πως η εντολή addle μεταβαίνει σε μια add ή σε μια nop
(το 255 που γράφεται στον reg20 είναι η τιμή 000000ff σε δεκαεξαδικό σύστημα οπότε και είναι σωστή)



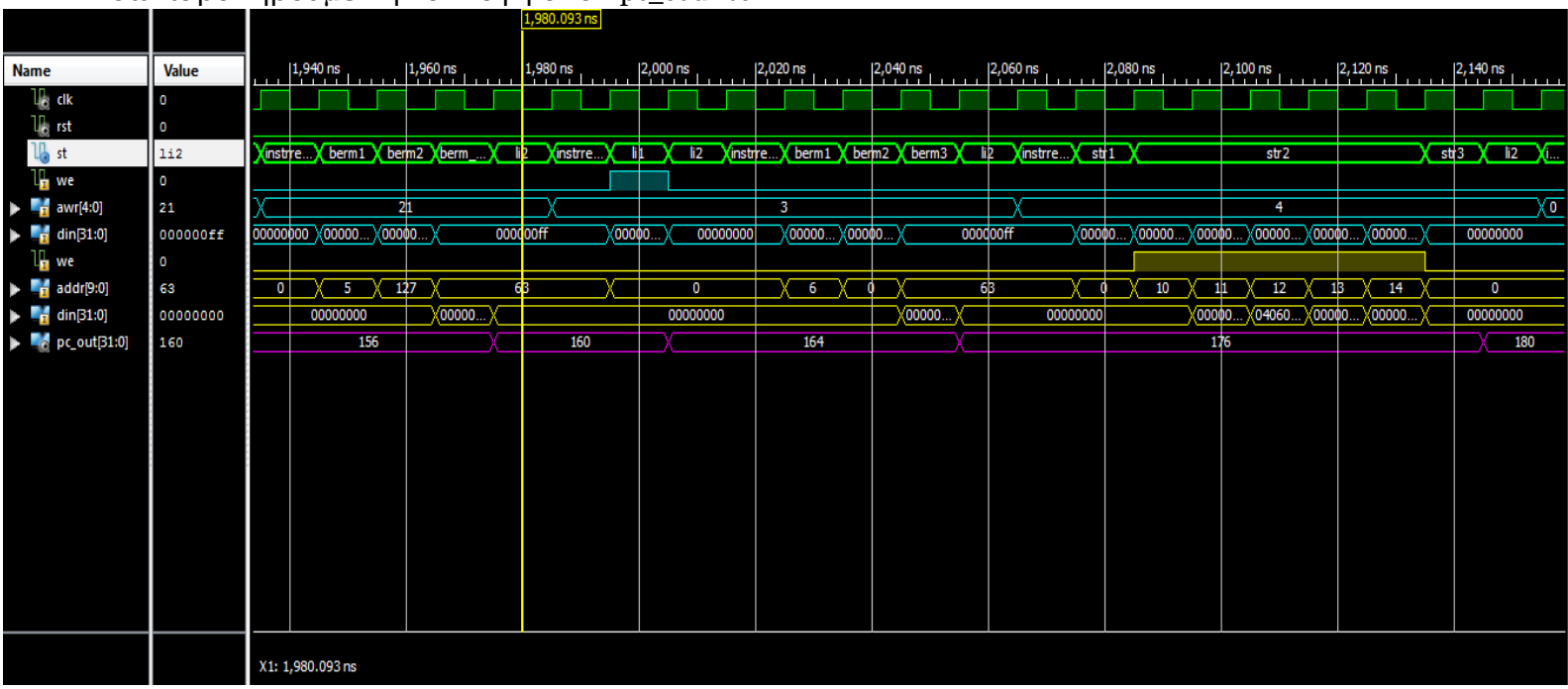
addmem
(το 00000010 που γράφεται στον reg20 είναι η τιμή 16 σε δεκαδικό σύστημα οπότε και είναι σωστή)



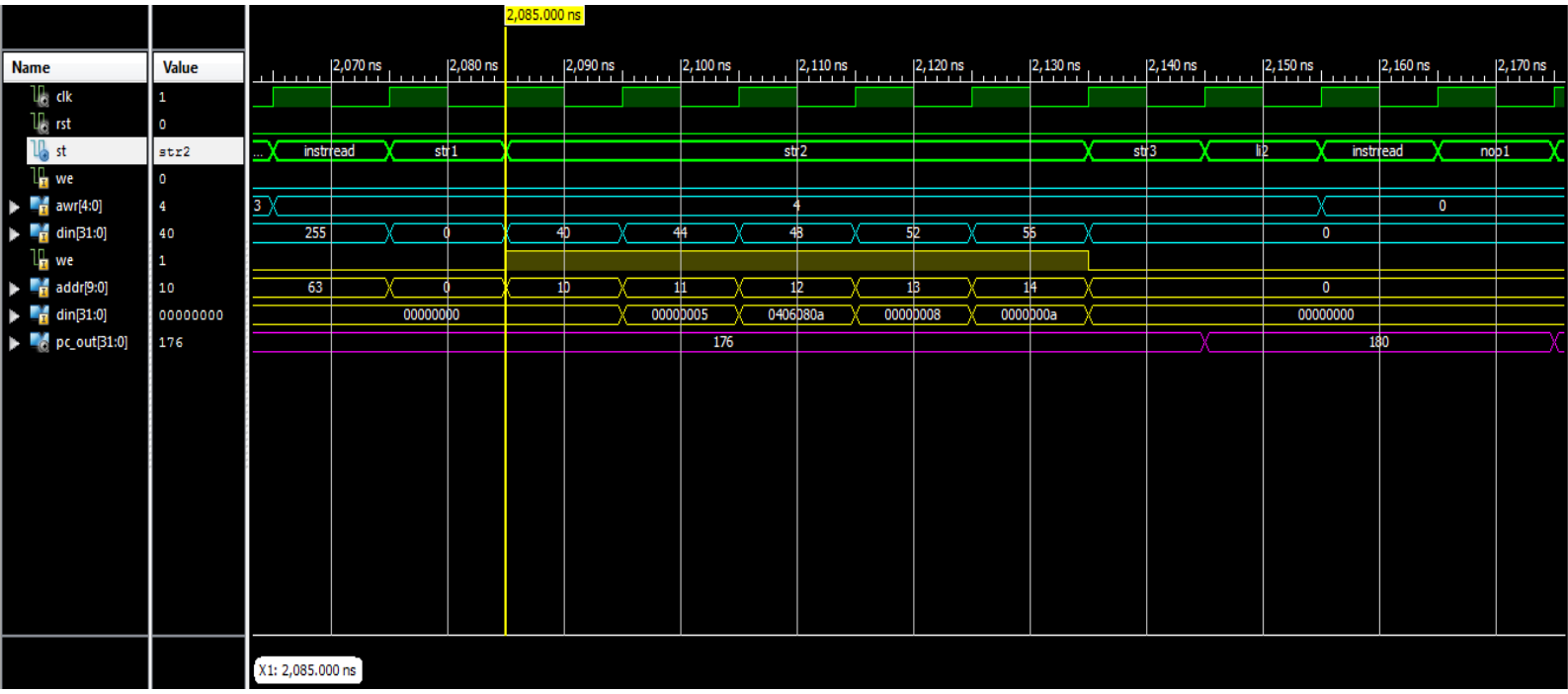
Οι υπόλοιπες addmem
(το 00000014 που γράφεται στον reg20 είναι η τιμή 20 σε δεκαεξαδικό σύστημα οπότε και είναι σωστή)



Εδώ παρατηρούμε την αλλάγη στον pc_counter



Εδώ παρατηρούμε τις εγγραφές στην μνήμη που εκτελεί η εντολή store registers
(οι τιμές εμφανίζονται σε hexadecimal μορφή)



Χριστοδούλου Θεόφιλος **A.M.: 2011030010**

Καριμπίδης Διονύσης **A.M.: 2011030009**