ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ Τ.Ε.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΣΧΕΔΙΑΣΗ ΨΗΦΙΑΚΟΥ ΚΥΚΛΩΜΑΤΟΣ

Βασίλειος Α. Ζορμπάς

Επιβλέπων: Φώτιος Βαρτζιώτης

Επίκουρος Καθηγητής

Άρτα, Νοέμβριος, 2020

ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ Τ.Ε.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΣΧΕΔΙΑΣΗ ΨΗΦΙΑΚΟΥ ΚΥΚΛΩΜΑΤΟΣ

Βασίλειος Α. Ζορμπάς

Επιβλέπων: Φώτιος Βαρτζιώτης

Επίκουρος Καθηγητής

Άρτα, Νοέμβριος, 2020

DESIGN OF A DIGITAL CIRCUIT

Εγκρίθηκε από τριμελή εξεταστική επιτροπή

Τόπος, Ημερομηνία

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Επιβλέπων καθηγητής

   Όνομα Επίθετο,

   τίτλος, βαθμίδα

2. Μέλος επιτροπής

   Όνομα Επίθετο,

   τίτλος, βαθμίδα

3. Μέλος επιτροπής

   Όνομα Επίθετο,

   τίτλος, βαθμίδα

Ο/Η Προϊστάμενος/η του

Τμήματος Όνομα Επίθετο,

τίτλος, βαθμίδα

Υπογραφή

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ

Δηλώνω υπεύθυνα και γνωρίζοντας τις κυρώσεις του Ν. 2121/1993 περί Πνευματικής Ιδιοκτησίας, ότι η παρούσα πτυχιακή εργασία είναι εξ ολοκλήρου αποτέλεσμα δικής μου ερευνητικής εργασίας, δεν αποτελεί προϊόν αντιγραφής ούτε προέρχεται από ανάθεση σε τρίτους. Όλες οι πηγές που χρησιμοποιήθηκαν (κάθε είδους, μορφής και προέλευσης) για τη συγγραφή της περιλαμβάνονται στη βιβλιογραφία.

Βασίλειος Α. Ζορμπάς

Υπογραφή

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία με θέμα «Σχεδίαση Ψηφιακού Κυκλώματος» πραγματοποιήθηκε στο πλαίσιο της πτυχιακής εργασίας του τμήματος Μηχανικών Πληροφορικής του Τεχνολογικού Εκπαιδευτικού Ιδρύματος Άρτας το έτος 2020.

Θα ήθελα να εκφράσω τις θερμές ευχαριστίες μου στον επιβλέποντα καθηγητή κύριο Φώτη Βαρτζιώτη, ο οποίος μου εμπιστεύτηκε την παρούσα πτυχιακή εργασία και είχε την απόλυτη υπομονή να μου απαντάει όλα τα ερωτήματα και να μου παρέχει βοήθεια και συμβουλές όποτε τις χρειαζόμουν.

Περαιτέρω, θα ήθελα να ευχαριστήσω την οικογένειά μου για την αμέριστη κατανόηση, συμπαράσταση και υποστήριξη που έδειξε όλα αυτά τα χρόνια που με βοηθούσαν κατά τη διάρκεια της φοιτητικής μου ζωής στην Άρτα καθώς χωρίς αυτούς δε θα μπορούσα να καταφέρω τίποτα.

Τέλος, θα ήθελα να ευχαριστήσω τους φίλους μου και τους συμφοιτητές μου για όλα αυτά τα χρόνια που περάσαμε μαζί αναφέροντας ονομαστικά τον καθένα από αυτούς. Είναι οι Άγγελος, Αλέξανδρος, Αντώνης, Άρης, Δανάη, Διονύσης, Διονύσης, Θεοδόσης και Ξένια.

Βασίλειος Α. Ζορμπάς

Άρτα 2020

ΠΕΡΙΛΗΨΗ

Ένα σημαντικό κομμάτι της τεχνολογίας που χρησιμοποιούμε ακόμα και σήμερα σε αμέτρητο αριθμό συσκευών, είναι η τεχνολογία των ψηφιακών κυκλωμάτων, διότι υπάρχουν σε κάθε τι ηλεκτρονικό, από έναν ηλεκτρικό φούρνο μέχρι και έναν υπερυπολογιστή.

Σκοπός αυτής της πτυχιακής εργασίας είναι, όπως λέει και ο τίτλος της, η σχεδίαση ενός ψηφιακού κυκλώματος. Πιο συγκεκριμένα, η σχεδίαση αυτή θα γίνει μέσω τεσσάρων εργαστηριακών ενοτήτων. Ξεκινώντας από απλά πράγματα, όπως η σχεδίαση μίας απλής πύλης και καταλήγοντας σε πιο εξειδικευμένα, όπως η προσομοίωση του επεξεργαστή, με αποτέλεσμα την πλήρη κατανόηση σχεδίασης ψηφιακών κυκλωμάτων.

Για τη δημιουργία αυτών των κυκλωμάτων χρησιμοποιήσαμε ορισμένα εργαλεία και γλώσσες περιγραφής υλικού. Αναλυτικότερα, χρησιμοποιήσαμε το πρόγραμμα Electric για σχεδίαση και επαλήθευση των κυκλωμάτων, καθώς και το πρόγραμμα ModelSim με το οποίο προσομοιώσαμε τα σχέδιά μας για να δούμε την σωστή λειτουργία τους. Η προσομοίωση αυτή έγινε, όπως αναφέραμε, με μία γλώσσα περιγραφής υλικού, την Verilog, η οποία περιγράφει τα σχέδια που δημιουργήσαμε.

Τέλος, μετά το πέρας των εργαστηριακών ενοτήτων, είμαστε σε θέση να κατανοούμε την λειτουργία των κυκλωμάτων καθώς και να τα σχεδιάζουμε, όπως επίσης και που μπορούμε να βελτιωθούμε.

Λέξεις-κλειδιά: Σχεδίαση, Ψηφιακά Κυκλώματα, Επαλήθευση, Προσομοίωση, Electric, ModelSim

ABSTRACT

An important part of the technology that we still use today in countless devices, is the technology of digital circuits, because they exist in everything electronic, from an electric oven to a supercomputer.

The purpose of this thesis is, as its title suggests, to design a digital circuit. More specifically, this design will be done through four laboratory modules. Starting with simple things, such as designing a simple gateway and ending with more specialized ones, such as processor simulation, resulting in a complete understanding of digital circuit design.

We used some hardware description tools and languages to create these circuits. More specifically, we used the Electric program to design and verify the circuits, as well as the ModelSim program in which we simulated our designs to see if they work properly. This simulation was done, as we mentioned, with a hardware description language, Verilog, which describes the designs we created.

Finally, after the end of the laboratory modules, we are able to understand the operation of the circuits as well as how to design them, and eventually we see where we can improve.

Keywords: Design, Digital Circuits, Verification, Simulation, Electric, ModelSim

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ **3**

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ **5**

ΕΥΧΑΡΙΣΤΙΕΣ **6**

ΠΕΡΙΛΗΨΗ **7**

ABSTRACT **8**

ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ/ΕΙΚΟΝΩΝ **11**

ΚΕΦΑΛΑΙΟ 1: ΕΙΣΑΓΩΓΗ **13**

1.1 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ 13

1.2 ΣΤΟΧΟΙ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ 17

1.3 ΠΕΡΙΓΡΑΦΗ ΚΕΦΑΛΑΙΩΝ 17

ΚΕΦΑΛΑΙΟ 2: ΕΡΓΑΛΕΙΑ ΚΑΙ ΤΕΧΝΟΛΟΓΙΕΣ **19**

2.1 ELECTRIC 19

2.2 MODELSIM 21

2.3 ΤΕΧΝΟΛΟΓΙΕΣ 22

2.4 ΚΑΝΟΝΕΣ ΣΧΕΔΙΑΣΗΣ 23

2.5 Ο ΕΠΕΞΕΡΓΑΣΤΗΣ MIPS-8BIT 26

ΚΕΦΑΛΑΙΟ 3: ΠΡΩΤΗ ΕΡΓΑΣΤΗΡΙΑΚΗ ΕΝΟΤΗΤΑ **28**

3.1 ΕΙΣΑΓΩΓΗ 28

3.2 ΥΛΟΠΟΙΗΣΗ 1ΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΑΣΚΗΣΗΣ 28

3.2.1 ΔΗΜΙΟΥΡΓΙΑ ΣΧΗΜΑΤΙΚΟΥ (SCHEMATIC) 28

3.2.2 ΔΗΜΙΟΥΡΓΙΑ LAYOUT 30

3.2.3 ΔΗΜΙΟΥΡΓΙΑ ΕΙΚΟΝΙΔΙΟΥ 32

3.2.4 LOGIC VERIFICATION 33

3.3 ΑΠΟΤΕΛΕΣΜΑΤΑ 1ΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ 35

3.3.1 INVERTER 35

3.3.2 AND 2 ΕΙΣΟΔΩΝ 37

3.3.4 NOR ΚΑΙ OR 2 ΕΙΣΟΔΩΝ 39

3.4 ΣΥΜΠΕΡΑΣΜΑΤΑ 42

ΚΕΦΑΛΑΙΟ 4: ΔΕΥΤΕΡΗ ΕΡΓΑΣΤΗΡΙΑΚΗ ΕΝΟΤΗΤΑ **43**

4.1 ΕΙΣΑΓΩΓΗ 43

4.2 ΥΛΟΠΟΙΗΣΗ 2ΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ 43

4.2.1 RTL SIMULATION ΚΑΙ WORDSLICES 43

4.2.2 ΔΗΜΙΟΥΡΓΙΑ ΣΧΗΜΑΤΙΚΟΥ ΓΙΑ 8-BIT AND ΚΑΙ 8-BIT OR 44

4.2.3 ΔΗΜΙΟΥΡΓΙΑ LAYOUT ΓΙΑ 8-BIT AND ΚΑΙ 8-BIT OR 45

4.2.4 ΕΠΑΛΗΘΕΣΗ WORDSLICES 47
4.2.5 ΣΧΗΜΑΤΙΚΟ, LAYOUT ΚΑΙ ΕΠΑΛΗΘΕΥΣΗ ΤΗΣ ALU 48
4.2.6 ΣΧΗΜΑΤΙΚΟ ΚΑΙ LAYOUT ΤΟΥ DATAPATH 50
4.3 ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ 2ης ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ 51
ΚΕΦΑΛΑΙΟ 5: ΤΡΙΤΗ ΕΡΓΑΣΤΗΡΙΑΚΗ ΕΝΟΤΗΤΑ **54**
5.1 ΕΙΣΑΓΩΓΗ 54
5.2 ΥΛΟΠΟΙΗΣΗ ΤΡΙΤΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ 54
5.2.1 ALUDEC 54
5.2.2 CONTROLLER PLA 56
5.2.3 CONTROLLER 60
5.3 ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ 3ης ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ 61
ΚΕΦΑΛΑΙΟ 6: ΤΕΤΑΡΤΗ ΕΡΓΑΣΤΗΡΙΑΚΗ ΕΝΟΤΗΤΑ **63**
6.1 ΕΙΣΑΓΩΓΗ 63
6.2 ΥΛΟΠΟΙΗΣΗ ΤΕΤΑΡΤΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ 63
6.2.1 ΣΧΗΜΑΤΙΚΟ ΚΑΙ LAYOUT MIPS 63
6.2.2 ΣΧΗΜΑΤΙΚΟ ΚΑΙ LAYOUT CHIP 66
6.3 ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ 4ΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ 68
ΚΕΦΑΛΑΙΟ 7: ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΠΡΟΤΑΣΕΙΣ ΚΑΤΑ ΤΗ ΣΧΕΔΙΑΣΗ ΚΥΚΛΩΜΑΤΩΝ **69**
7.1 ΣΥΜΠΕΡΑΣΜΑΤΑ 69
7.2 ΠΡΟΤΑΣΕΙΣ ΚΑΤΑ ΤΗ ΣΧΕΔΙΑΣΗ ΚΥΚΛΩΜΑΤΩΝ 70
ΒΙΒΛΙΟΓΡΑΦΙΑ **71**
ΕΛΛΗΝΙΚΗ ΒΙΒΛΙΟΓΡΑΦΙΑ 71
ΞΕΝΟΓΛΩΣΣΗ ΒΙΒΛΙΟΓΡΑΦΙΑ 71

ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ/ΕΙΚΟΝΩΝ

Εικόνα 1 Οι δημιουργοί του πρώτου τρανζιστορ: Joe Bardeen, William Shockley, Walter Brattain ..13
Εικόνα 2 Το πρώτο τρανζίστορ ..13
Εικόνα 3 Jack Kilby: Ιδρυτής του πρώτου ολοκληρωμένου κυκλώματος (Νομπελ Φυσικής, 2000) ..13
Εικόνα 4 Το πρώτο ολοκληρωμένο κύκλωμα που δημιουργήθηκε στην Texas Instruments ..14
Εικόνα 5 Robert Noyce: ιδρυτής των εταιρειών Fairchild Semiconductor και της Intel ..14
Εικόνα 6 Ρέπλικα του πρώτου τρανζίστορ ..15
Εικόνα 7 Απεικόνηση σύγχρονων ολοκληρωμένων κυκλωμάτων ..15
Εικόνα 8 Απεικόνηση του νόμου του Moore μέχρι τη χρονιά 2018 ..16
Εικόνα 9 Λογότυπο της Static Free Softaware ..19
Εικόνα 10 Γενική παρουσίαση δυνατοτήτων του Electric ..20
Εικόνα 11 Παράδειγμα Σχηματικού ενός ψηφιακού κυκλώματος με PMOS και NMOS ..20
Εικόνα 12 Λογότυπο ModelSim ..21
Εικόνα 13 Αρχική σελίδα προγράμματος ModelSim ..21
Εικόνα 14 Παράδειγμα προσωμοίωσης στο ModelSim ..22
Εικόνα 15 Ελάχιστες αποστάσεις που μπορούν να υπάρξουν κατά τη φυσική σχεδίαση ..25
Εικόνα 16 Διάγραμμα απεικόνησης MIPS επεξεργαστή ..27
Εικόνα 17 Παράθυρο εισαγωγής ονόματος βιβλιοθήκης κατά τη δημιουργία της στο Electric ..28
Εικόνα 18 Παράθυρο δημιουργίας Cell στην Βιβλιοθήκη του Electric ..29
Εικόνα 19 Απεικόνηση σχηματικού πύλης NAND2 ..29
Εικόνα 20 Παράθυρο εισαγωγής Ports στο κύκλωμά μας ..30
Εικόνα 21 Αποτελέσματα DRC για την NAND2 πύλη ..30
Εικόνα 22 Απεικόνηση P-Well και N-well ..31
Εικόνα 23 Απεικόνηση Layout της πύλης NAND 2 ..31
Εικόνα 24 Αποτελέσματα DRC, ERC, NCC για την πύλη NAND2 ..32
Εικόνα 25 Απεικόνηση icon της πύλης NAND2 στο Electric ..33
Εικόνα 26 Παράθυρο προσωμοίωσης στο ModelSim ..34
Εικόνα 27 Αποτελέσματα προσωμοίωσης NAND2 ..34
Εικόνα 28 Απεικόνηση σχηματικού αντστροφέα ..35
Εικόνα 29 Απεικόνηση layout INVERTER ..36
Εικόνα 30 Αποτελέσματα DRC, ERC, NCC για τον INVERTER ..36
Εικόνα 31 Αποτελέσματα προσωμοίωσης INVERTER ..37
Εικόνα 32 Απεικόνηση σχηματικού AND2 χρησιμοποιώντας τα icons NAND2 και INVERTER ..37
Εικόνα 33 Απεικόνηση layout AND2 ..38
Εικόνα 34 Αποτελέσματα DRC, ERC, NCC για AND2 ..38
Εικόνα 35 Αποτελέσματα προσωμοίωσης AND2 ..39
Εικόνα 36 Απεικόνηση σχηματικού NOR2 ..39
Εικόνα 37 Απεικόνηση σχηματικού OR2 χρησιμοποιώντας τα icons NOR2 και INVERTER ..39
Εικόνα 38 Απεικόνηση layout NOR2 ..40
Εικόνα 39 Αποτελέσματα DRC, ERC, NCC για NOR2 ..40
Εικόνα 40 Απεικόνηση layout OR2 ..41
Εικόνα 41 Αποτελέσματα DRC, ERC, NCC για OR2 ..41
Εικόνα 42 Αποτελέσματα προσωμοίωσης NOR2 ..42
Εικόνα 43 Αποτελέσματα προσωμοίωσης OR2 ..42
Εικόνα 44 Απεικόνηση σχηματικού 8-BIT AND ..44
Εικόνα 45 Απεικόνηση σχηματικού 8-BIT OR ..44

Εικόνα 46 Απεικόνηση layout 8-BIT AND. Αριστερά ολόκληρο το κύκλωμα και δεξιά το ένα κελί 45
Εικόνα 47 Απεικόνηση layout 8-BIT OR. Αριστερά ολόκληρο το κύκλωμα και δεξιά το ένα κελί....46
Εικόνα 48 Αποτελέσματα DRC, ERC, NCC για 8-BIT AND....47
Εικόνα 49 Αποτελέσματα DRC, ERC, NCC για 8-BIT OR....47
Εικόνα 50 Απεικόνηση σχηματικού της ALU....48
Εικόνα 51 Απεικόνηση layout της ALU....49
Εικόνα 52 Αποτελέσματα DRC, ERC, NCC για ALU....50
Εικόνα 53 Απεικόνηση σχηματικού του DataPath....50
Εικόνα 54 Απεικόνηση layout του DataPath....51
Εικόνα 55 Αποτελέσματα DRC, ERC, NCC για DataPath....52
Εικόνα 56 Αποτελέσματα προσωμοίωσης DataPath....53
Εικόνα 57 Μήνυμα παραθύρου Transcript κατά την επιτυχή ολοκλήρωση της προσομοίωσης....53
Εικόνα 58 Απεικόνηση σχηματικού της ALUdec....55
Εικόνα 59 Απεικόνηση layout της ALUdec....55
Εικόνα 60 Αποτελέσματα DRC, ERC, NCC για ALUdec....56
Εικόνα 61 Διάγραμμα FSM προσπέλασης μιας εντολής....57
Εικόνα 62 Απεικόνηση σχηματικού του Controller_PLA....58
Εικόνα 63 Απεικόνηση layout του Controller_PLA....58
Εικόνα 64 Αποτελέσματα DRC, ERC, NCC για Controller_PLA....59
Εικόνα 65 Αποτελέσματα προσωμοίωσης Controller_PLA....59
Εικόνα 66 Απεικόνηση σχηματικού του Controller....60
Εικόνα 67 Απεικόνηση layout του Controller....60
Εικόνα 68 Αποτελέσματα DRC, ERC, NCC για Controller....61
Εικόνα 69 Αποτελέσματα προσωμοίωσης Controller....62
Εικόνα 70 Μήνυμα παραθύρου Transcript κατά την επιτυχή ολοκλήρωση της προσομοίωσης....62
Εικόνα 71 Απεικόνηση σχηματικού του Mips....63
Εικόνα 72 Απεικόνηση layout του Mips....64
Εικόνα 73 Αποτελέσματα DRC, ERC, NCC για Mips....64
Εικόνα 74 Αποτελέσματα προσωμοίωσης Mips....65
Εικόνα 75 Μήνυμα παραθύρου Transcript κατά την επιτυχή ολοκλήρωση της προσομοίωσης....65
Εικόνα 76 Απεικόνηση σχηματικού του Chip....67
Εικόνα 77 Απεικόνηση layout του Chip....67
Εικόνα 78 Αποτελέσματα DRC, ERC, NCC για Chip....68

ΚΕΦΑΛΑΙΟ 1: ΕΙΣΑΓΩΓΗ

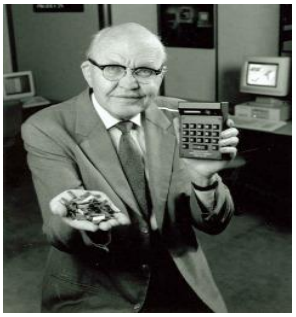
1.1 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Το έναυσμα για την δημιουργία των τρανζίστορ δόθηκε στις 23 Δεκεμβρίου του 1947 όταν οι John Bardeen και Walter Brattain στα Bell Laboratories δημιούργησαν το πρώτο λειτουργικό τρανζίστορ επαφής σημείου με κρυσταλλικό Γερμάνιο και επαφές χρυσού. Μέχρι και τότε στα ηλεκτρονικά κυκλώματα χρησιμοποιούνταν μεγάλες, ακριβές και δαπανηρές ως προς την ενέργεια τους λυχνίες κενού. Οι κρυσταλλοτρίοδοι, όπως ονομάζονται στην ελληνική γλώσσα, ανάλογα με την τάση της πόλωσης ρυθμίζουν την ροή του ηλεκτρικού ρεύματος που απορροφούν από την πηγή τάσης που συνδέονται. Ακόμη, ποικίλλουν στην λειτουργικότητά τους καθώς ενεργούν ως ενισχυτές και ως διακόπτες. Για αυτή τους την ανακάλυψη τιμήθηκαν με Νόμπελ το 2000. [3]

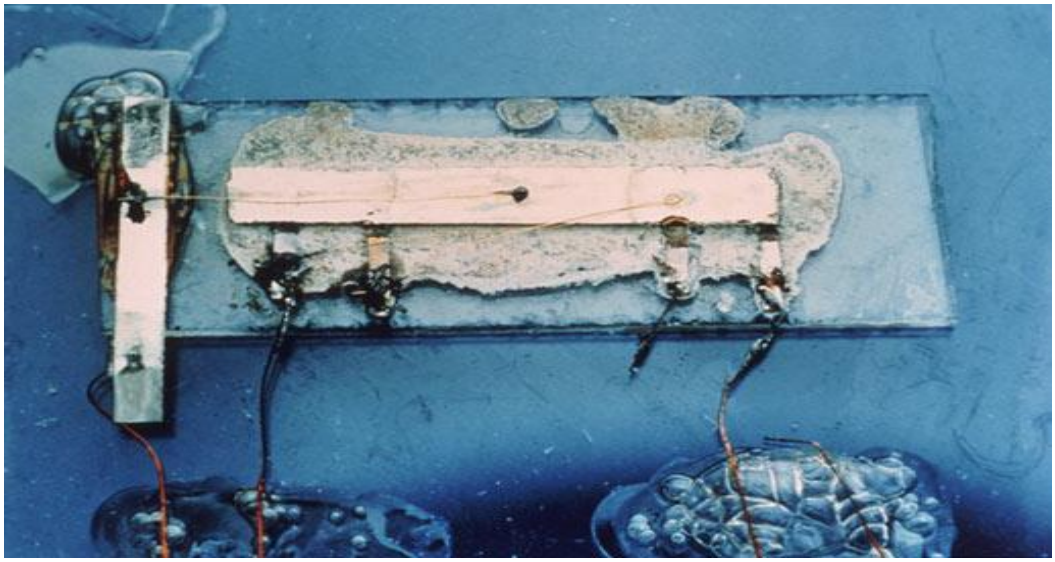
Εικόνα 1

Εικόνα 2

Εν συνεχεία, στις 12 Σεπτεμβρίου του 1958 ο Jack Kilby στην Texas Instruments, παρουσιάζει το πρώτο ολοκληρωμένο κύκλωμα, IC-Integrated Circuit, όπου αποδείκνυε την λειτουργική και αποδοτική συνύπαρξη αντιστάσεων και πυκνωτών στο ίδιο κομμάτι ημιαγωγών. Χρησιμοποιώντας αγωγούς χρυσού συνδεδεμένους με σύρμα και προσαρμόζοντας πέντε παθητικά στοιχεία με την πλάκα γερμανίου υλοποίησε την καινοτομία του. Ο νεοφερμένος τότε, Jack Kilby, μην έχοντας δικαίωμα για άδεια διακοπών έκατσε όλο το καλοκαίρι σε ένα από τα εγκαταλελειμμένα εργαστήρια της Texas Instruments για να δουλέψει πάνω στην ιδέα του. [3][6][8]

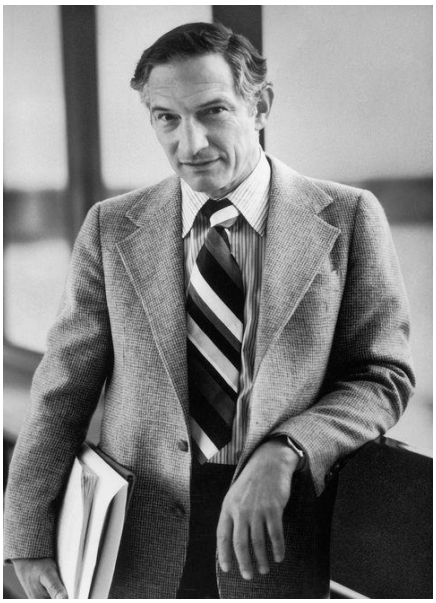


Εικόνα 3



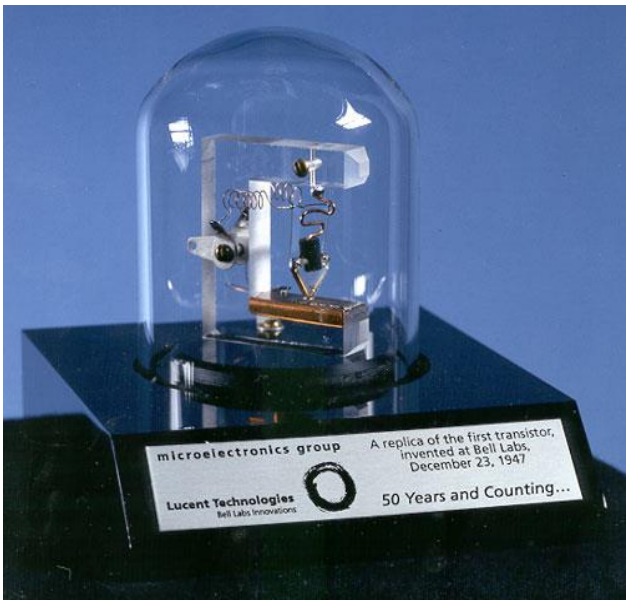
Εικόνα 4

Στην εικόνα 4 παρουσιάζεται η επαναστατική εφεύρεση του Jack Kimbly. Το ολοκληρωμένο κύκλωμα είχε διαστάσεις 10×1,6 χιλιοστά. Συγκριτικά με τις προγενέστερες τεχνολογικές δημιουργίες όπως αυτή των John Bardeen και Walter Brattain υπερτερούσε σε ισχύ, συνέφερε περισσότερο από οικονομικής άποψης και ήταν ταχύτερο. Το ενδιαφέρον αυτής της πρωτοπορίας προκλήθηκε από τον Robert Noyce , συνιδρυτή της intel και γνωστό ως "Δήμαρχος της Silicon Valley", που στις 23 Ιανουαρίου του 1959 εξετάζοντας το κύκλωμα έμελλε να αναβαθμίσει την δεύτερη γενιά υπολογιστών και να εμπλουτίσει το κύκλωμα με την χρήση μεταλλικών συνδέσμων ως επίστρωση. Με το γεγονός αυτό ο Robert Noyce βιομηχανοποίησε την μέθοδο παραγωγής των ολοκληρωμένων κυκλωμάτων και θεωρήθηκε, με την συμβολή του συνεφευρέτης, καθώς του αποδόθηκε το 2000 Νόμπελ Φυσικής. [3][4][7]

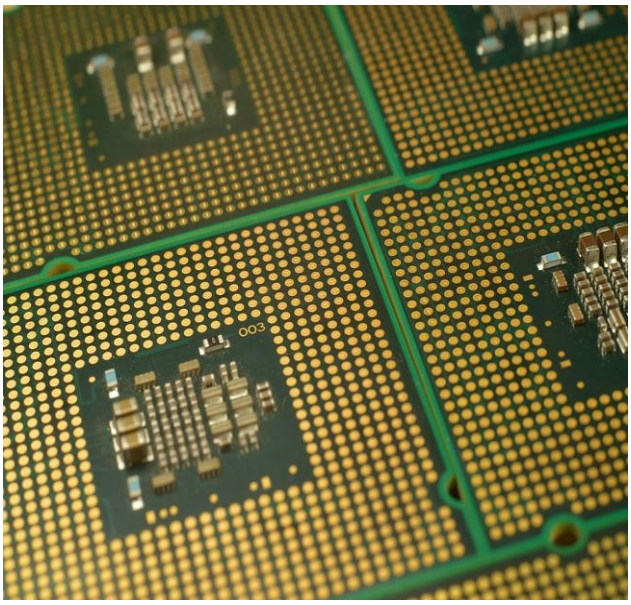


Εικόνα 5

Η συνεχής και αδιάκοπη ανάπτυξη στον συγκεκριμένο τομέα έχει επιφέρει πολλές αλλαγές μέχρι σήμερα. Ο κύριος προβληματισμός ήταν ο μεγάλος όγκος των τρανζίστορ αφού τα καθιστούσε πιο δαπανηρά σε ισχύ και κόστος. Με την πάροδο του χρόνου και με την επερχόμενη πρόοδο τα τρανζίστορ έγιναν μικρότερα με αποτέλεσμα να είναι ταχύτερα, αποδοτικότερα και με λιγότερη ισχύ. [3]



Εικόνα 6



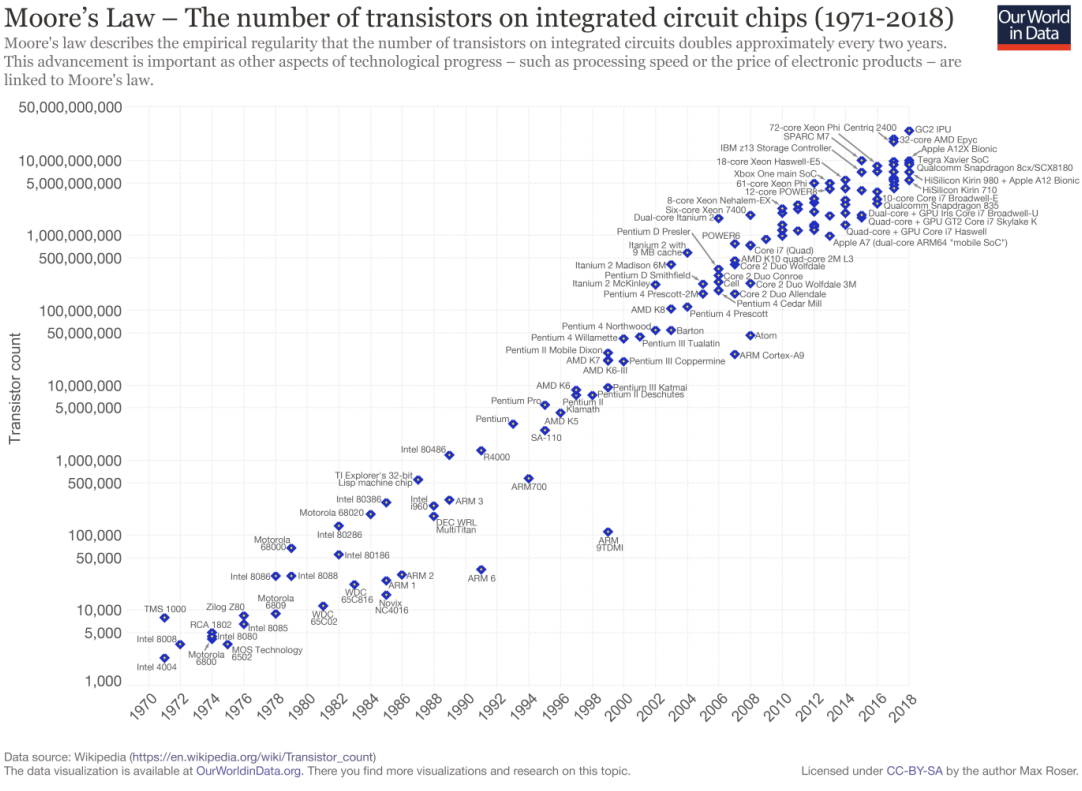
Εικόνα 7

Αξίζει να αναφερθεί, πως εδώ και 50 χρόνια διατηρεί τον μεγαλύτερο ρυθμό ανάπτυξης με 53% σε όλο το τεχνολογικό φάσμα .Η προσθήκη των πρώτων επεξεργαστών έγινε σε αριθμομηχανές και έπειτα σε εκτυπωτές και τηλεχειριστήρια. Σταθμός βέβαια για την πορεία αυτή ήταν ο πρώτος επεξεργαστής της Intel 4004. Με την δημιουργία του αυξήθηκαν και οι

απαιτήσεις για γλώσσες προγραμματισμού. Επιπροσθέτως, με την τεχνολογική άνοδο των μικροεπεξεργαστών και ειδικότερα με την κατασκευή ενός οκτάμπιτου (8-bit) μικροεπεξεργαστή παράχθηκε ο πρώτος μικροϋπολογιστής στα μέσα του 1970. [3]

Εντύπωση προκαλεί το γεγονός πως οι υπερυπολογιστές και η επεξεργαστική δύναμη που είχαν, σήμερα είναι διαθέσιμα σε καταναλωτικές συσκευές, όπως κινητά τηλέφωνα και φορητούς υπολογιστές. Για παράδειγμα όταν άρχισε η ψηφιοποίηση των αρχείων μιας μεγάλης εταιρίας η και ενός δήμου η μνήμη που χρειάζονταν πριν 30 χρόνια είναι σήμερα διαθέσιμη στο κινητό τηλέφωνο που κρατάμε. [3]

Το 1965, ο Gordon Moore συνιδρυτής της Intel και της Fairchild πρόβλεψε ότι ο αριθμός των τρανζίστορ που υπάρχουν μέσα σε μία τετραγωνική ίντσα στα ολοκληρωμένα κυκλώματα θα διπλασιαζόταν κάθε δεκαετία. Ακόμη, προχώρησε στο συμπέρασμα πως η ισχύς των επεξεργαστών θα διπλασιαζόταν κάθε δύο χρόνια και πως θα υπάρχει επερχόμενη μείωση στο κόστος. Μελλοντικά αναθεώρησε ως προς το χρονικό διάστημα σε 18 μήνες. Στο πόρισμα αυτό βοήθησε ο David House. Η παρατήρηση αυτή ονομάστηκε *"Νόμος του Moore"* και έγινε σημείο αναφοράς για πολλές εταιρίες και για το πως θα εξελιχθεί η τεχνολογία αυτή θέτοντάς τον ως 'στόχο'. Πράγματι έτσι και έγινε έως και το 2012, όπου η Intel επιβεβαίωσε ότι θα αρχίσει να επιβραδύνει με την εισαγωγή ημιαγωγών 22 νανόμετρων και το 2014 με την εισαγωγή των 14 νανόμετρων. Σύμφωνα με τον νόμο, το 2016 θα ήταν η χρονιά του 10 νανόμετρων αλλά 'πήρε' παράταση μέχρι το 2017 όπου και πλέον αυξήθηκε ο χρόνος εξέλιξης των τρανζίστορ στα δυόμιση χρόνια. Το 2018 βγήκαν στην αγορά επεξεργαστές 7 νανόμετρων, αριθμό απίστευτο λόγω του ότι έχει αρχίσει να εμφανίζεται το φαινόμενο 'quantum tunnelling'. Βέβαια ούτε αυτό σταμάτησε τους επιστήμονες με το γεγονός ότι σήμερα, το 2020 υπάρχουν επεξεργαστές 5 nm και στόχο δημιουργίας επεξεργαστή 2 nm το 2022 όπου πιστεύεται ότι θα είναι και οι τελευταίοι. [3][5]



Εικόνα 8

Όλα τα παραπάνω ήταν μία αναγκαία ιστορική αναδρομή, για να μπούμε στο θέμα της πτυχιακής αυτής εργασίας όπου δεν είναι άλλο από τη σχεδίαση ενός ψηφιακού κυκλώματος με την ολοκλήρωση της οποίας θα έχουμε σχεδιάσει με επιτυχία έναν μικροεπεξεργαστή MIPS. Στο επόμενο κεφάλαιο θα συνεχιστεί η εισαγωγή με στόχους της πτυχιακής εργασίας.

1.2 ΣΤΟΧΟΙ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Όπως προαναφέρθηκε και στο τέλος του προηγούμενου κεφαλαίου ο στόχος της πτυχιακής εργασίας είναι η σχεδίαση ενός ψηφιακού κυκλώματος και συγκεκριμένα ενός μικροεπεξεργαστή MIPS μέσω εργαλείων συγκεκριμένων για τα οποία θα μιλήσουμε και παρακάτω.

Αναλυτικότερα, η σχεδίαση αυτή θα γίνει μέσω τεσσάρων εργαστηριακών ενοτήτων ξεκινώντας από απλά πράγματα, όπως η σχεδίαση μίας απλής πύλης και καταλήγοντας σε πιο εξειδικευμένα, όπως η προσομοίωση του επεξεργαστή, με αποτέλεσμα την πλήρη κατανόηση σχεδίασης ψηφιακών κυκλωμάτων. Αρχικά είναι αναγκαίο να γνωρίσουμε τα εργαλεία που χρησιμοποιήθηκαν για την εκτέλεση και βέβαια, την ολοκλήρωση της εργασίας. Τα προγράμματα αυτά είναι το Electric και το ModelSim, δύο βασικά εργαλεία για την σχεδίαση και επαλήθευση ψηφιακών κυκλωμάτων που θα μιλήσουμε για αυτά αναλυτικότερα σε επόμενο κεφάλαιο.

Είναι σημαντικό να σημειωθεί επίσης, ότι κατά την ολοκλήρωση αυτής της πτυχιακής θα έχουμε εξοικειωθεί με τις τεχνολογίες που χρησιμοποιήθηκαν όπως την RTL σχεδίαση, το Testbench (προγράμματα δοκιμής και επαλήθευσης) και τις γλώσσες περιγραφής υλικού.

Εν κατακλείδι, οι στόχοι αυτής της εργασίας είναι ξεκάθαροι ως προς το περιεχόμενο τους οπότε μας είναι και εύκολο να καταλάβουμε τα εργαστηριακά κεφάλαια, τα προγράμματα και τις τεχνολογίες που θα χρησιμοποιήσουμε, και θα μας γίνεται πιο εύκολο ακόμα καθώς συνεχίζουμε. Στο κεφάλαιο που ακολουθεί, θα γίνει περιγραφή των επόμενων κεφαλαίων για να ξεκαθαρίσει το τοπίο ως προς το περιεχόμενο της πτυχιακής.

1.3 ΠΕΡΙΓΡΑΦΗ ΚΕΦΑΛΑΙΩΝ

Στην ενότητα αυτή, θα αναλύσουμε περιληπτικά το περιεχόμενο του κάθε κεφαλαίου που θα ακολουθήσει για να μπορέσουμε να καταλάβουμε με τι πραγματεύεται η πτυχιακή εργασία, η οποία είναι, η σχεδίαση ενός ψηφιακού κυκλώματος.

Κλείνοντας την ενότητα αυτή, θα μιλήσουμε για τα προγράμματα και τις τεχνολογίες που θα χρησιμοποιήσουμε αναλυτικά, που δεν είναι άλλες από το Electric, το ModelSim, την RTL σχεδίαση, το Testbench και τις γλώσσες περιγραφής υλικού. Θα ήταν εύλογο εδώ κάποιος να αναρωτηθεί γιατί, αλλά όπως θα δούμε και παρακάτω, είναι σημαντικές πληροφορίες για την κατανόηση της σχεδίασης και των προγραμμάτων.

Στη συνέχεια, θα ξεκινήσουμε να λύνουμε τις τέσσερεις εργαστηριακές ενότητες, ξεκινώντας, προφανώς από την πρώτη, η οποία είναι και η πιο βασική. Συγκεκριμένα, θα

διδαχθούμε τις βασικές λειτουργίες του εργαλείου Electric, πως να σχεδιάζουμε το σχηματικό και το layout λογικών πυλών, καθώς και το πως να τα προσομοιώνουμε αργότερα στο ModelSim. Ξεκινάμε με απλές πύλες NAND, NOR, OR και AND, όλες δύο εισόδων, και με έναν αντιστροφέα. Τελειωνούμε με την επαλήθευση του καθενός ξεχωριστά και αφού βεβαιωθούμε για τη σωστή λειτουργία τους, θα συνεχίσουμε στο επόμενο.

Η επόμενη εργαστηριακή ενότητα καθώς και οι δύο τελευταίες ολοκληρώνουν έναν μικροεπεξεργαστή 8-bits MIPS. Το μεγαλύτερο κομμάτι του επεξεργαστή δίνεται από την εκφώνηση των εργαστηρίων αλλά σε κάθε μία από τις ενότητες λείπει ένα ενδιαφέρον κομμάτι που θα πρέπει να ολοκληρώσουμε. Στη δεύτερη ενότητα θα ξεκινήσουμε τον επεξεργαστή μαθαίνοντας για τον σχεδιασμό του Datapath, κατασκευάζοντας και συνδέοντας wordslices στο κομμάτι της ALU (Arithmetic Logic Unit).

Στην επόμενη ενότητα, θα ασχοληθούμε με τον controller του επεξεργαστή, που είναι υπεύθυνος για τη δημιουργία σημάτων προς το Datapath για να γίνει προσπέλαση και εκτέλεση οποιασδήποτε εντολής. Στο πρώτο σκέλος του δημιουργήσαμε ένα ALU decoder και τέλος θα χρησιμοποιήσαμε ένα εργαλείο, το PLA (Programmable Logic Array) Generator για τη δημιουργία ενός Controller. Πρόκειται για ένα εργαλείο που παράγει ένα PLA από μία συνθήκη γραμμένη σε Verilog και αυτό μας γλιτώνει από παραπάνω προσπάθεια σχεδίασης.

Αφού έχουμε επαληθεύσει ότι όλες προηγούμενες ενότητες λειτουργούν σωστά χωρίς λάθη, καταλήγουμε στην τελευταία ενότητα, η οποία είναι και η καθοριστική γιατί θα βάλουμε όλα τα κομμάτια του επεξεργαστή μαζί κι έτσι μπορούμε να δούμε ότι λειτουργεί σωστά.

Κλείνοντας το κεφάλαιο αυτό, βλέπουμε πως υπάρχουν αρκετοί άγνωστοι ορισμοί, για τους οποίους θα δοθεί εξήγηση στα επόμενα κεφαλαία. Σημασία αυτού του κεφαλαίου ήταν η περιγραφή των ενοτήτων που θα ακολουθήσουν παρακάτω ώστε να υπάρχει μια αρχική κατανόηση για το που αρχίζει και που καταλήγει αυτή η εργασία.

ΚΕΦΑΛΑΙΟ 2: ΕΡΓΑΛΕΙΑ ΚΑΙ ΤΕΧΝΟΛΟΓΙΕΣ

2.1 ELECTRIC

Το Electric (Electric VLSI Design System) είναι ένα πρόγραμμα σχεδίασης ολοκληρωμένων κυκλωμάτων ανοιχτού κώδικα. Έχει αναπτυχθεί από τον Steve Rubin και υποστηρίζονταν για αρκετά χρόνια από την Sun Microsystems. Γράφτηκε σε Java και ως εκ τούτου τρέχει σε οποιοδήποτε λειτουργικό σύστημα, όπως Windows, Linux και MacOS. Το λογισμικό αυτό διατίθεται δωρεάν μέσω της Static Free Software και πλέον δεν δέχεται αναβαθμίσεις, παρά μόνο μικρές διορθώσεις. [10]

Όπως μόλις ειπώθηκε, είναι ένα σύστημα σχεδιασμού με τη βοήθεια υπολογιστή για ηλεκτρικά ολοκληρωμένα κυκλώματα. Προορίζεται κυρίως για διάταξη ολοκληρωμένου κυκλώματος, αλλά χειρίζεται επίσης σχήματα, ακόμη και γλώσσες περιγραφής υλικού. Κατανοεί μια σειρά διαφορετικών τεχνολογιών (περιβάλλοντα σχεδιασμού), όπως MOS (nMOS και αρκετές CMOS). Εκτός από τις τεχνολογίες διάταξης IC, είναι ακόμα σε θέση να συνεργαστεί με πολλές άλλες μορφές γραφικών, όπως σχηματικά (schematics), αρχιτεκτονικές FPGA (Field - programmable gate array) και πολλά άλλα. Ένας ενσωματωμένος 'editor' τεχνολογίας επιτρέπει τη δημιουργία και την τροποποίηση νέων περιβαλλόντων σχεδίασης. [10][16]

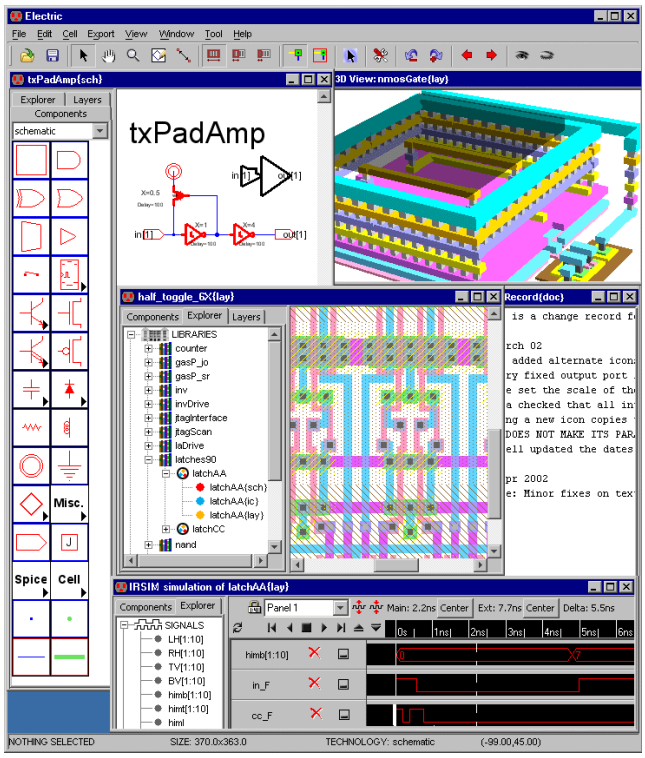
Εικόνα 9

Επίσης, ενσωματώνει πολλά διαφορετικά εργαλεία για την ανάλυση και τη σύνθεση κυκλωμάτων. Το σύστημα προσφέρεται μαζί με ελεγκτές κανόνων σχεδίασης, προσομοιωτές, δρομολογητές κ.α.. Επιπλέον, διαθέτει ένα εύχρηστο μοντέλο για την ενσωμάτωση εργαλείων που καθιστά σχετικά εύκολη την προσθήκη νέων τεχνολογιών. Εκτός από το ότι μπορεί να χειριστεί αυθαίρετες τεχνολογίες και εργαλεία, η Electric διαθέτει ένα ισχυρό front-end που παρέχει περιορισμούς διάταξης και φορητότητα πλατφόρμας. Το σύστημα περιορισμού επιτρέπει στα συνδεδεμένα εξαρτήματα να παραμένουν σωστά συνδεδεμένα, ακόμα και όταν ο σχεδιασμός έχει τροποποιηθεί. [10][16]

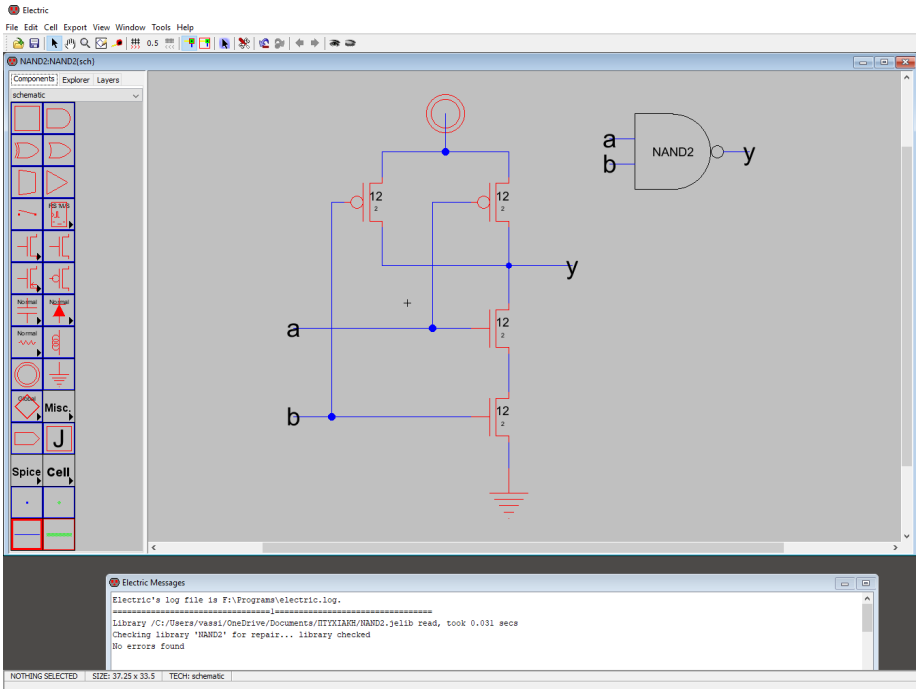
Παρότι το πρόγραμμα έχει να κάνει με ένα δύσκολο κομμάτι της τεχνολογίας, είναι αρκετά εύχρηστο και φιλικό προς το χρήστη και αυτό φαίνεται από την πρώτη κιόλας εκκίνηση του προγράμματος. Η εικόνα 10 θα μας δώσει μια γενική ιδέα του προγράμματος με τις περισσότερες δυνατότητές του. Δηλαδή, εργαλεία σχεδίασης σχηματικού και 'layout', προσομοιώσεις και την βασική γραμμή εργαλείων.

Στην εικόνα 11, βλέπουμε το παράθυρο που εμφανίζεται όταν ανοίγουμε το πρόγραμμα. Αμέσως παρατηρούμε την γραμμή εργαλείων. Πάνω βλέπουμε ο παράθυρο που βάση σχεδίασης με τις καρτέλες δεξιά των 'Components', 'Explorer' και 'Layers' , και κάτω το παράθυρο 'Electric Messages' το όποιο μας ενημερώνει για τυχόν αλλαγές ή λάθη στην σχεδίασή μας.

Συνοψίζοντας, βλέπουμε ότι το ‘Electric’ είναι ένα σύστημα σχεδιασμού ολοκληρωμένων με πολλά εργαλεία στην διάθεση μας. Είναι εύχρηστο, πράγμα το οποίο είναι πολύ βασικό για την κατανόηση των σχεδίων στα επόμενα κεφάλαια. Στην συνέχεια, θα ακολουθήσει εισαγωγή για άλλο πρόγραμμα που χρησιμοποιήσαμε για να σχεδιάσουμε τον μικροεπεξεργαστή, το ModelSim το οποίο είναι βασικό για την επαλήθευση της λειτουργίας των σχεδιασμών.



Εικόνα 10



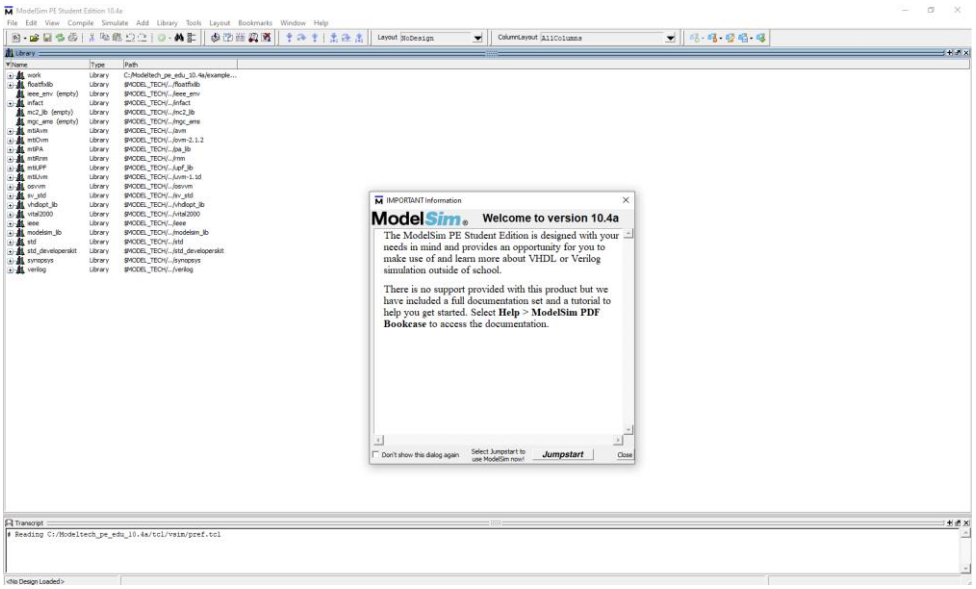
Εικόνα 11

2.2 MODELSIM

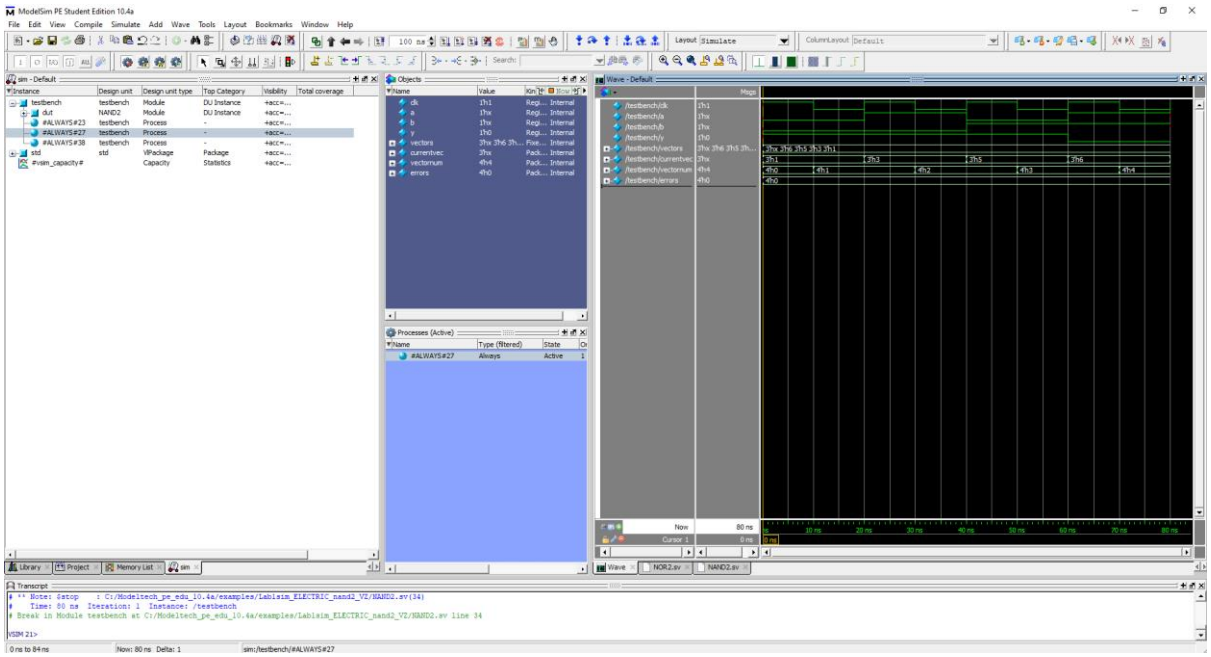
Το ModelSim είναι ένα περιβάλλον προσομοίωσης υλικού και εντοπισμού σφαλμάτων που στοχεύει κυρίως σε μικρότερους σχεδιασμούς ASIC, FPGA καθώς και HDL, VHDL, Verilog κλπ. Έχει δημιουργηθεί από την Mentor Graphics η οποία είναι μία πολυεθνική εταιρεία ηλεκτρονικών αυτοματισμών (EDA) με βάση τις ΗΠΑ για ηλεκτρολόγους μηχανικούς και ηλεκτρονικά. Η εταιρεία ιδρύθηκε το 1981 και πωλήθηκε στη Siemens το 2017. [11]

Εικόνα 12

Στην παρούσα εργασία θα χρησιμοποιήσουμε τη ‘Student’ έκδοση του προγράμματος που παρέχεται δωρεάν από την Mentor Graphics. Συγκεκριμένα, αυτή η έκδοση Το προορίζεται για χρήση από τους μαθητές στην επιδίωξη των ακαδημαϊκών μαθημάτων τους και των βασικών εκπαιδευτικών προγραμμάτων. Όπως και στην κανονική έκδοση, έτσι και σε αυτή, υποστηρίζονται σχέδια VHDL και ‘Verilog’. Το περιβάλλον χρήσης του, αν και εύχρηστο δεν είναι τόσο ξεκάθαρο όσο στο Electric, λόγω και τον περισσότερων δυνατοτήτων που διαθέτει. Στην επόμενη εικόνα βλέπουμε το περιβάλλον χρήσης καθώς ανοίγουμε το πρόγραμμα στον υπολογιστή. Παρόλα αυτά το ModelSim θα αποτελέσει αναπόσπαστο εργαλείο για την επαλήθευση των project που θα δημιουργήσουμε για τον μικροεπεξεργαστή. [11]



Εικόνα 13



Εικόνα 14

Όπως βλέπουμε ανοίγοντας το πρόγραμμα, φαίνονται οι επιλογές πάνω πάνω, η γραμμή εργαλείων από κάτω, το παράθυρο με τις διαθέσιμες βιβλιοθήκες και τέλος το παράθυρο ‘Transcript’ στο οποίο εμφανίζονται μηνύματα λάθους. Χωρίς περαιτέρω ανάλυση, στην εργασία θα χρησιμοποιήσουμε το ModelSim για να επεξεργαστούμε τον κώδικα που θα χρησιμοποιούμε, θα γίνεται compile για να τον επαληθεύσουμε και στην συνέχεια θα προσομοιώνουμε τον κώδικα, ο οποίος θα περιγράφει τον μικροεπεξεργαστή, για να δούμε αν τα αποτελέσματα και συγκεκριμένα τα waveforms δείχνουν την σωστή λειτουργία του.

Το ModelSim, όπως αναφέρθηκε είναι ένα περιβάλλον προσομοίωσης υλικού. Είναι εύκολο στη χρήση και θα είναι, όπως και το Electric τα βασικά εργαλεία για να ολοκληρώσουμε την εργασία αυτή δημιουργώντας τον μικροεπεξεργαστή. Στο επόμενο κεφάλαιο ακολουθεί μία εισαγωγή στις τεχνολογίες που θα χρησιμοποιήσουμε στην εργασία, όπως το Testbench, Verilog, RTL σχεδίαση και τί είναι οι γλώσσες περιγραφής υλικού.

2.3 ΤΕΧΝΟΛΟΓΙΕΣ

Με τον όρο τεχνολογίες σε αυτή την εργασία εννοούμε τις γλώσσες περιγραφής υλικού, την RTL σχεδίαση, το Testbench και την Verilog. Όλα αυτά όπως θα δούμε ξεκινώντας τις εργαστηριακές ασκήσεις είναι βασικά για να κατανοήσουμε καλύτερα την λειτουργία του μικροεπεξεργαστή. Τέλος είναι σημαντικό να αναφέρουμε επίσης και τα σχηματικά και τα layout που θα δημιουργήσουμε με το πρόγραμμα Electric.

Η Verilog, ή αλλιώς IEEE 1364, είναι μια γλώσσα περιγραφής υλικού (HDL) που χρησιμοποιείται για τη μοντελοποίηση ηλεκτρονικών συστημάτων. Χρησιμοποιείται συνηθέστερα στο σχεδιασμό και την επαλήθευση ψηφιακών κυκλωμάτων στο επίπεδο μεταφοράς αφαίρεσης. Χρησιμοποιείται επίσης στην επαλήθευση αναλογικών κυκλωμάτων και κυκλωμάτων μικτού σήματος, καθώς και στο σχεδιασμό γενετικών κυκλωμάτων. Το 2009,

το πρότυπο Verilog (IEEE 1364-2005) συγχωνεύτηκε στο πρότυπο SystemVerilog, δημιουργώντας το πρότυπο IEEE 1800-2009. Από τότε, η Verilog είναι επίσημα μέρος της γλώσσας SystemVerilog. Η τρέχουσα έκδοση είναι το πρότυπο IEEE 1800-2017. [22]

Το Testbench τώρα, είναι κώδικας HDL που μας επιτρέπει να παρέχουμε ένα τεκμηριωμένο, επαναλαμβανόμενο σύνολο ερεθισμάτων (stimuli) που είναι φορητό ανάμεσα στους προσομοιωτές. Επίσης, το testbench μπορεί να είναι τόσο απλό όσο ένα αρχείο με ρολόι και δεδομένα εισόδου ή ένα πιο περίπλοκο αρχείο που περιλαμβάνει έλεγχο σφαλμάτων, είσοδο και έξοδο αρχείου και περιστασιακή δοκιμή. Μπορεί κάποιος να δημιουργήσει ένα Testbench με έναν απλό text editor, όπως για παράδειγμα το σημειωματάριο. Σε γενικές γραμμές θα μας χρησιμεύσει στο να επαληθεύουμε λογικά τα κυκλώματά μας. Η λογική επαλήθευση περιλαμβάνει την απόδειξη ότι τα κελιά εκτελούν σωστά την προκαθορισμένη από εμάς λειτουργία τους. Ένας τρόπος για να το κάνουμε αυτό είναι να προσομοιώσουμε το κελί και να εφαρμόσουμε ένα σύνολο διανυσματικών δοκιμών 1 και 0 στις εισόδους τους και στη συνέχεια να ελέγξουμε ότι τα αποτελέσματα αντιστοιχούν σε αυτό που περιμέναμε. Βέβαια, όλο αυτό είναι αυτοματοποιημένο ή ακόμα καλύτερα, ενσωματωμένο στο Testbench, οπότε κάνει την διαδικασία πιο εύκολη. [14]

Τα σχέδια που θα δημιουργήσουμε περιγράφονται συνήθως σε τρία επίπεδα αφαίρεσης (levels of abstraction). Η περιγραφή του επιπέδου μεταφοράς καταχωρητή (RTL) είναι ένα αρχείο Verilog ή VHDL που καθορίζει τη συμπεριφορά του κελιού σε όρους καταχωρητών και συνδυαστικής λογικής (registers and combinational logic). Συχνά χρησιμεύει ως η προϋπόθεση του τι πρέπει να κάνει ο επεξεργαστής ή οι πύλες ή ό,τι σχεδιάζουμε. Το σχηματικό είναι μία αναπαράσταση των στοιχείων ενός συστήματος, στη συγκεκριμένη περίπτωση των τρανζίστορ, που χρησιμοποιούν γραφικά σύμβολα και όχι ρεαλιστικές εικόνες. Η φυσική διάταξη, ή καλύτερα ‘layout’ δείχνει πώς τα τρανζίστορ ή τα κελιά είναι φυσικά διατεταγμένα. [23]

Τέλος, στις τεχνολογίες που χρησιμοποιήσαμε, ανήκει και το Pad Frame. Τα μικροσκοπικά τρανζίστορ του σχεδίου που δημιουργήσαμε πρέπει να συνδεθούν με τον εξωτερικό κόσμο. Αυτό γίνεται με το Pad Frame. Ένα Pad Frame αποτελείται από μεταλλικά ‘pads’ τα οποία, είναι αρκετά μεγάλα για να προσαρτηθούν στον επεξεργαστή κατά την κατασκευή του με μικρά λεπτά καλώδια από χρυσό. Αν και τελευταίο βήμα, είναι ένα πολύ σημαντικό κομμάτι της σχεδίασης. Στο πρόγραμμα Electric υπάρχει το Pad Frame Generator, το οποίο μας βοηθάει στο να δημιουργήσουμε το pad frame για τον επεξεργαστή μας. [19]

2.4 ΚΑΝΟΝΕΣ ΣΧΕΔΙΑΣΗΣ

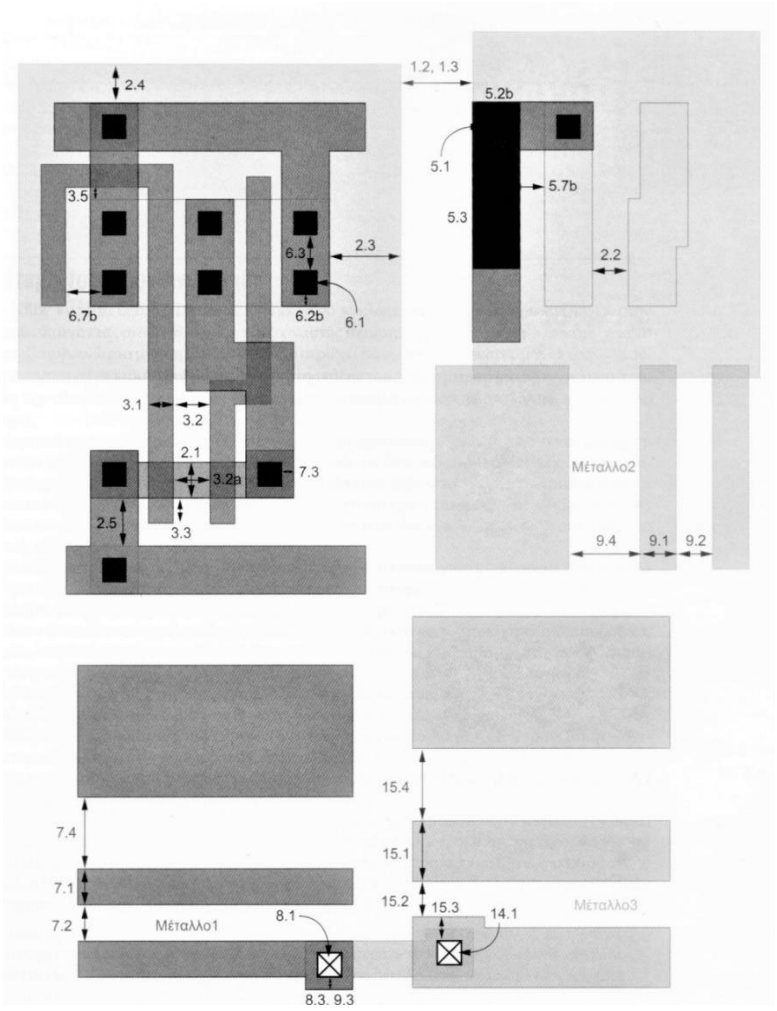
Οι κανόνες φυσικής σχεδίασης (layout) περιγράφουν, το πόσο μικρά μπορούν να είναι τα χαρακτηριστικά μεγέθη ενός στοιχείου, στη συγκεκριμένη περίπτωση, ενός κυκλώματος, και πόσο κοντά μπορούν να κατασκευάζονται τα τρανζίστορ σε μια ορισμένη τεχνολογία. Ο πιο διαδεδομένος τρόπος γίνεται με βάση ενός κανόνα σχεδιασμού που βασίζεται στους Mead και Conway και κάνει χρήση της παραμέτρου λ , η οποία και χαρακτηρίζει την ανάλυση της διαδικασίας. Γενικά η παράμετρος λ είναι το ήμισυ του ελάχιστου δυνατού μήκους του καναλιού (channel) του τρανζίστορ, δηλαδή, της απόστασης

μεταξύ πηγής (source) και υπόδοχής (drain) και αντιστοιχεί στο ελάχιστο πλάτος του αγωγού πολυπυριτίου. [2]

Ο παρακάτω πίνακας, μαζί με την εικόνα, μας δίνει τις ελάχιστες αποστάσεις που μπορούν να έχουν τα τρανζίστορ μεταξύ τους, πράγμα που μας είναι πολύ χρήσιμο, διότι στην συγκεκριμένη εργασία ασχολούμαστε πάρα πολύ με το να σχεδιάζουμε. Στα επόμενα κεφάλαια, στα οποία δημιουργήσαμε τη φυσική σχεδίαση των κυκλωμάτων, βοήθησαν πάρα πολύ στην κατανόηση της σχεδίασης. [3]

Κανόνες Σχεδίασης SUBM της MOSlS (3 στρώσεις μετάλλου, 1 στρώση πολυπυρίτιου με προβλέψεις για περάσματα & επαφές			
Στρώση	**Κανόνας**	**Περιγραφή**	**Κανόνας (λ)**
N-πηγάδι	1.1	Πλάτος	12
	1.2	Απόσταση έως πηγάδι σε διαφορετικό δυναμικό	18
	1.3	Απόσταση έως πηγάδι σε ίδιο δυναμικό	6
Ενεργή (διάχυση)	2.1	Πλάτος	3
	2.2	Απόσταση έως ενεργή	3
	2.3	Πηγή/υποδοχή που περιβάλλεται από πηγάδι	6
	14	Επαφή υποστρώματος/πηγαδιού που περιβάλλεται από πηγάδι	3
	2.5	Απόσταση έως ενεργή αντίθετου τύπου	4
Πολυπυρίτιο	3.1	Πλάτος	2
	3.2	Απόσταση έως πολυπυρίτιο πάνω από οξείδιο πεδίου	3
	3.2a	Απόσταση έως πολυπυρίτιο πάνω από ενεργή	3
	3.3	Επέκταση πύλης έξω από ενεργή	2
	3.4	Επέκταση ενεργής έξω από πολυπυρίτιο	3
	3.5	Απόσταση πολυπυρίτιου έως ενεργή	1
Επιλογής (n ή p)	4.1	Απόσταση από επαφή υποστρώματος/ πηγαδιού έως πύλη	3
	4.2	Επικάλυψη ενεργής	2
	4.3	Επικάλυψη επαφής υποστρώματος/πηγαδιού	1
	4.4	Απόσταση έως επιλογής	2
Επαφών (με πολυπυρίτιο ή ενεργή)	5.116.1	Πλάτος (ακριβές)	2x2
	5.2b, 6.2b	Επικάλυψη από πολυπυρίτιο ή ενεργή	1
	5.3,	Απόσταση έως επαφή	3
	5.4, 6.4	Απόσταση έως πύλη	2
	5.5b	Απόσταση επαφής πολυιτυριτίου έως άλλο πολυπυρίTI0	5
	5.7b, 6.7b	Απόσταση έως ενεργή/πολυπυρίτιο για πολλαπλές επαφές πολυπυριτίου/ ενεργής	3
	6.8b	Απόσταση επαφής ενεργής περιοχής έως επαφής πολυπυριτίου	4
Μέταλλο1, Μέταλλο2	7.1, 9.1	Πλάτος	3
	7,2, 9.2	Απόσταση έως ίδια στρώση μετάλλου	3
	7,3, 8.3, 9.3	Επικάλυψη επαφής ή περάσματος	1

	9.4	Απόσταση έως μέταλλο για γραμμές πλάτους μεγαλύτερου από 10 λ	6
Πέρασμα1, Πέρασμα2	8.1, 14.1	Πλάτος (ακριβές)	2x2
	82 14.2	Απόσταση έως πέρασμα στην ίδια στρώση	3
Μέταλλο3	15.1	Πλάτος	5
	15.2	Απόσταση έως μέταλλο	3
	15.3	Επικάλυψη περάσματος	2
	15.4	Απόσταση έως μέταλλο για γραμμές πλάτους μεγαλύτερου από 10 λ	6
Τομή επύαλου (overglass)	10.1	Πλάτος ανοίγματος ακροδέκτη δεσμού	60 μm
	10.2	Πλάτος ανοίγματος ακροδέκτη δοκιμής	20 μm
	10.3	Επικάλυψη Μετάλλου3 σε τομή overglass	6 μm
	10.4	Απόσταση μετάλλου ακροδέκτη με μη-σχετιζόμενο μέταλλο	30 μm
	10.5	Απόσταση μετάλλου ακροδέκτη με ενεργή ή πολυπυρίτιο	15 μm



Εικόνα 15

2.5 Ο ΕΠΕΞΕΡΓΑΣΤΗΣ MIPS-8BIT

Είναι σημαντικό σε αυτή την ενότητα να μιλήσουμε για τον επεξεργαστή MIPS, διότι είναι και το αποτέλεσμα αυτής της εργασίας, δηλαδή, η σχεδίαση, η επαλήθευση και η προσομοίωση αυτού του επεξεργαστή. Ακόμα, θα δούμε τα κύρια κομμάτια του και τη λειτουργία του προκειμένου να έχουμε τη βασική ιδέα γιατί θα το ξαναδούμε παρακάτω.

Ένας επεξεργαστής MIPS είναι μία έκδοση ενός υπολογιστή με μειωμένο αριθμό εντολών (RISC). Πρόκειται για ένα μοντέλο που μελετάται συχνά σε μαθήματα αρχιτεκτονικής υπολογιστών πανεπιστημίου επειδή έχει αρκετή πολυπλοκότητα για να συμπεριλάβει όλες τις κύριες πτυχές των σύγχρονων επεξεργαστών, αλλά δεν είναι υπερβολικά περίπλοκο για να εμποδίσει τους μαθητές στις λεπτομέρειες. Οι επεξεργαστές MIPS υπάρχουν από τις αρχές της δεκαετίας του 1980 και απέκτησαν μεγάλη δημοτικότητα τη δεκαετία του 1990. Εκτιμήθηκε ότι το ένα τρίτο όλων των επεξεργαστών RISC χρησιμοποίησαν την αρχιτεκτονική MIPS, συμπεριλαμβανομένων ενσωματωμένων συστημάτων όπως το Sony PlayStation 2 και το PlayStation Portable. [12]

Ο MIPS έχει τρία κύρια στοιχεία:

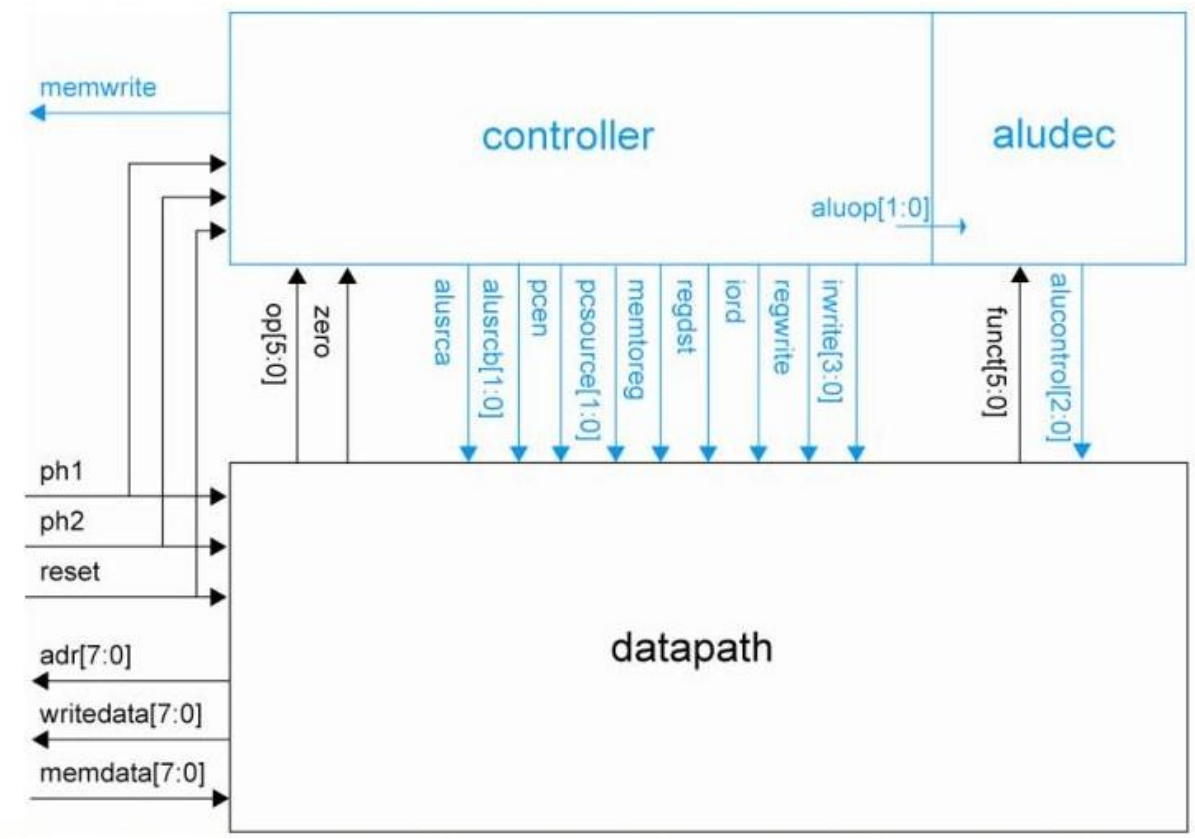
- Datapath
- Controller
- ALU Control

Datapath: Η λογική του Datapath είναι η λογική (όπως υποδηλώνει το όνομα) για την επεξεργασία των δεδομένων εισόδου και τη δημιουργία των σωστών δεδομένων εξόδου. Η λειτουργικότητα της λογικής διαδρομής δεδομένων συνήθως χωρίζεται σε μπλοκ και αυτό προσφέρει τη δυνατότητα βελτιστοποίησης για ταχύτητα ή ‘area’. Είναι σαφές ότι υπάρχουν επιλογές βελτιστοποίησης της ροής δεδομένων λαμβάνοντας υπόψη τον καλύτερο τρόπο μετακίνησης των δεδομένων μεταξύ των καταχωρητών. [12][24]

Controller: Ο ελεγκτής MIPS είναι υπεύθυνος για την αποκωδικοποίηση των ανακτημένων εντολών από τη μνήμη και την αποστολή του σήματος ελέγχου στη διαδρομή δεδομένων, όπου θα οδηγούσαν από το ‘zipper’ και το σήμα ελέγχου ALU στο ALU control. Ο ελεγκτής είναι επίσης υπεύθυνος για τις εγγραφές μνήμης και τον απαριθμητή προγράμματος. Αξίζει να σημειωθεί ότι ο ελεγκτής λειτουργεί ως μηχανή πεπερασμένης κατάστασης. [12]

ALU Control: Όπως αναφέρθηκε παραπάνω, η πραγματική ALU βρίσκεται στο τέλος του bitslice, μία ALU ανά bitslice. Ο ALU Control λαμβάνει σήματα ALUOp, δύο bits, που καθορίζουν τη λειτουργία που πρέπει να εκτελέσει η ALU. Στη συνέχεια, ο ALU Control στέλνει τα σήματα ελέγχου στην ALU προκειμένου να πραγματοποιηθεί η κάθε λειτουργία. Η ALU λαμβάνει τρία σήματα ελέγχου για να προσδιορίσει τη λειτουργία που πρέπει να εκτελέσει η ALU. [12]

Έχοντας τα παραπάνω στοιχεία καταλαβαίνει κανείς πως ένας επεξεργαστής MIPS λειτουργεί και είναι σημαντικό γιατί έχουμε την γενική ιδέα της λειτουργίας του, που θα μας χρησιμεύσει στα παρακάτω κεφάλαια στα οποία έχουμε τα βήματα δημιουργίας ενός επεξεργαστή MIPS 8-bit.


memwrite
controller
aludec
aluop[1:0]
op[5:0]
zero
alusrca
alusrcb[1:0]
pcen
pcsource[1:0]
memtoreg
regdst
iord
regwrite
irwrite[3:0]
funct[5:0]
alucontrol[2:0]
ph1
ph2
reset
adr[7:0]
writedata[7:0]
memdata[7:0]
datapath

Εικόνα 16

ΚΕΦΑΛΑΙΟ 3: ΠΡΩΤΗ ΕΡΓΑΣΤΗΡΙΑΚΗ ΕΝΟΤΗΤΑ

3.1 ΕΙΣΑΓΩΓΗ

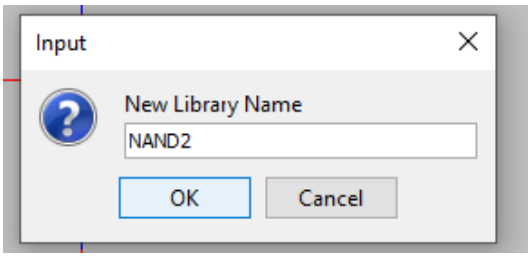
Στο κεφάλαιο που ακολουθεί θα ξεκινήσουμε τη δημιουργία του μικροεπεξεργαστή. Είναι η πρώτη από τις τέσσερεις ενότητες και είναι και η πιο θεμελιώδης, διότι μας εισάγει στα βασικά μέρη λειτουργίας του προγράμματος Electric. Μάθαμε να σχεδιάζουμε, να προσομοιώνουμε και να επαληθεύουμε σχηματικά και 'layouts' των πυλών που θα δημιουργήσουμε. Τέλος, μάθαμε ότι το ModelSim είναι εργαλείο ικανό να παράγει αποτελέσματα μέσω του testbench, που είναι πρόγραμμα επαλήθευσης.

Πιο συγκεκριμένα, αυτό το εργαστήριο μας καθοδήγησε στο σχεδιασμό μιας πύλης NAND δύο εισόδων. Στη συνέχεια, σχεδιάσαμε τη διάταξη και επαληθεύσαμε ότι πληροί τους κανόνες σχεδίασης και ταιριάζει με τη διάταξη. Χρησιμοποιώντας την πύλη NAND και έναν μετατροπέα, σχεδιάσαμε μια πύλη AND δύο εισόδων. Τέλος, σχεδιάσαμε τις δικές μας πύλες NOR και OR δύο εισόδων. [16]

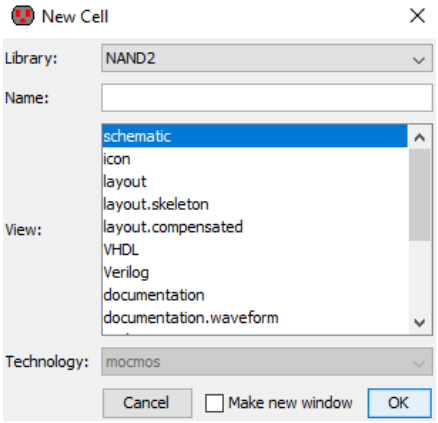
3.2 ΥΛΟΠΟΙΗΣΗ 1ΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΑΣΚΗΣΗΣ

3.2.1 ΔΗΜΙΟΥΡΓΙΑ ΣΧΗΜΑΤΙΚΟΥ (SCHEMATIC)

Ας ξεκινήσουμε λοιπόν να δημιουργήσουμε μία πύλη NAND 2 εισόδων. Αρχικά, ανοίγουμε το electric και δημιουργούμε μία βιβλιοθήκη στο πρόγραμμα ονομάζοντας την NAND2 πατώντας File > New Library > Πληκτρολογούμε το όνομα NAND2 (Εικόνα 14) > OK. Για τη δημιουργία σχηματικού στη συγκεκριμένη βιβλιοθήκη πατάμε δεξί κλικ στο όνομα της βιβλιοθήκης και στη συνέχεια 'Create New Cell' και εμφανίζεται το παράθυρο της εικόνας 15, δίνουμε το όνομα που θέλουμε, θα ήταν καλό να συμπίπτει με αυτό που θα σχεδιάσουμε, τέλος επιλέγουμε τι τύπος είναι, που στη συγκεκριμένη περίπτωση είναι 'Schematic' και έπειτα 'OK'. Πλέον, μπορούμε να ξεκινήσουμε να σχεδιάζουμε την πύλη NAND 2 εισόδων. [16]

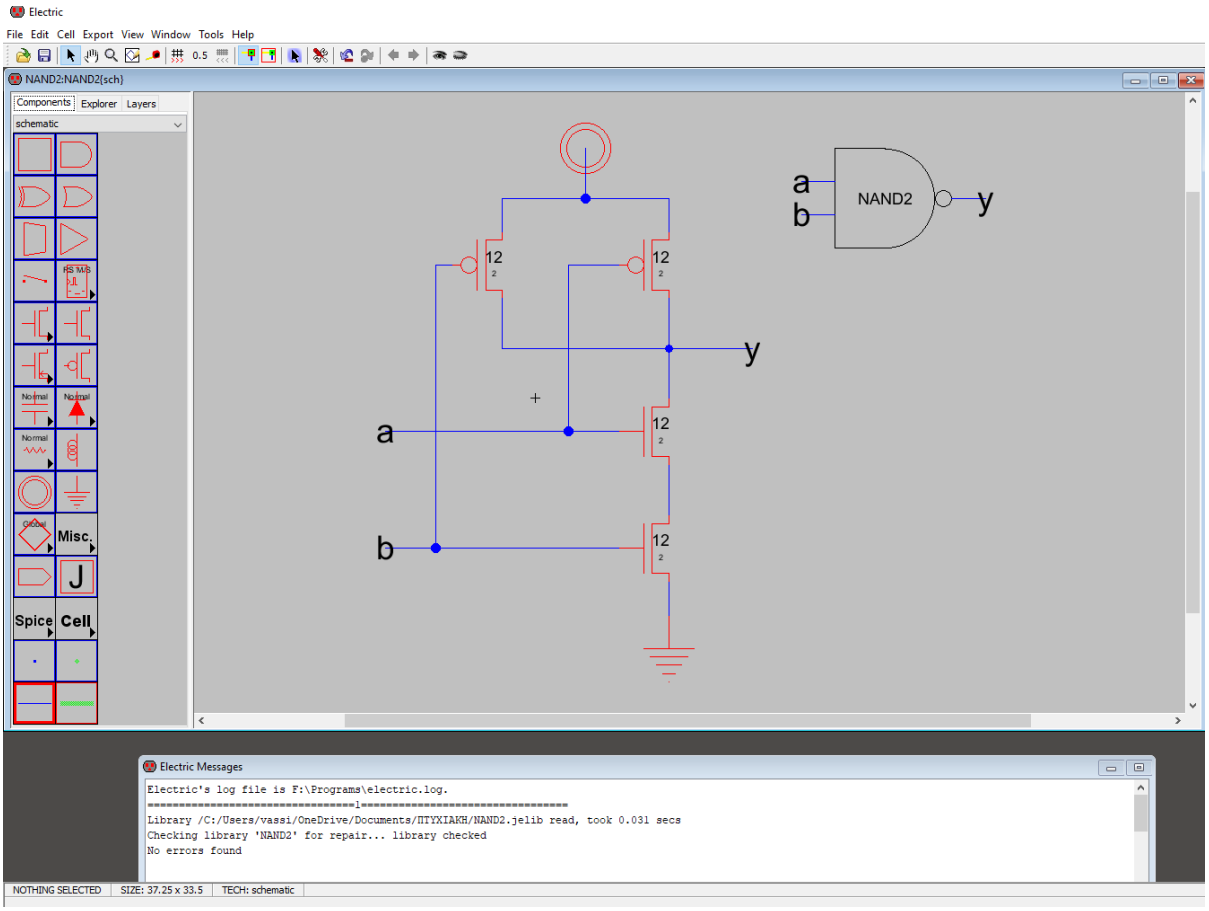


Εικόνα 17



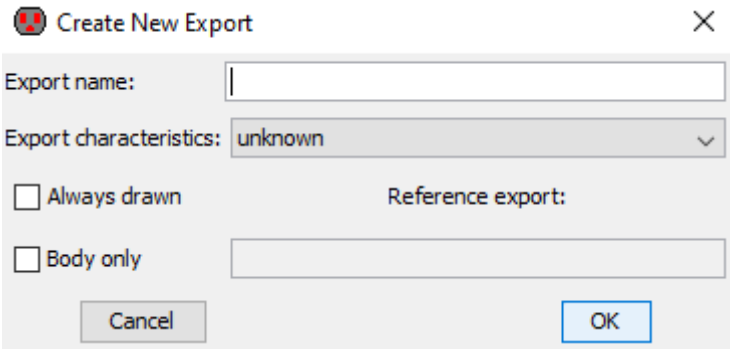
Εικόνα 18

Το να τοποθετήσουμε τα σύμβολα προς το παρόν είναι απλό, αριστερά στο παράθυρο πατάμε Components και θα εμφανιστούν τα εργαλεία που θέλουμε για το κύκλωμα. Όπως θα δούμε στην παρακάτω εικόνα, από τα εργαλεία θα χρειαστούμε μία πηγή τροφοδοσίας, μία γείωση και τέλος PMOS και NMOS. Στην Εικόνα 18 βλέπουμε 2 NMOS σε σειρά και 2 PMOS παράλληλα. Τα πρώτα τοποθετήθηκαν έτσι, ώστε να οδηγούν την έξοδο χαμηλά όταν οποιαδήποτε είσοδος είναι υψηλή. Τα PMOS είναι σε σειρά ώστε να οδηγούν την έξοδο ψηλά όταν και οι 2 είσοδοι είναι χαμηλά. Έτσι, η έξοδος δεν μένει σε καμία περίπτωση αιωρούμενη ή σε κατάσταση υπερφόρτισης.



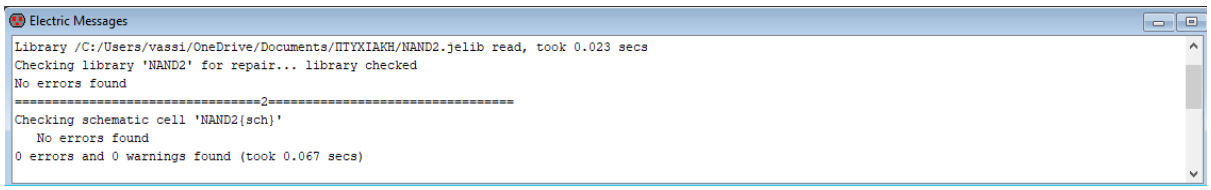
Εικόνα 19

Για να ολοκληρώσουμε το σχηματικό χρειάζεται να ορίσουμε τις εισόδους και τις εξόδους του. Πηγαίνοντας στο Export > Create Export… εμφανίζεται το παράθυρο της εικόνας 18. Δίνουμε όνομα και στην συνέχεια επιλέγουμε τον τύπο και τέλος πατάμε οκ. Στην συγκεκριμένη περίπτωση θέλουμε 2 εισόδους και 1 έξοδο.



Εικόνα 20

Αφού βάλαμε και τις εισόδους, πλέον μπορούμε να επαληθεύσουμε αν είναι σωστό το κύκλωμά μας κάνοντας έλεγχο μέσω του DRC (Design Rule Check). DRC είναι ο έλεγχος κανόνων σχεδίασης για να δούμε εάν η διάταξη μας έχει σφάλματα. Για να το χρησιμοποιήσουμε πατάμε Tools > DRC > Check Hierarchically. Τα αποτελέσματα αναφέρονται στο παράθυρο Electric Messages. Εάν υπάρχουν σφάλματα, χρησιμοποιούμε το πλήκτρο > για να δούμε κάθε ένα σφάλμα ξεχωριστά το οποίο επισημαίνεται και στη διάταξη. Διορθώνουμε τυχόν σφάλματα και επαναλαμβάνουμε μέχρι να διορθωθούν όλα τα σφάλματα. Στο σχήμα στην εικόνα 18 δεν υπάρχουν σφάλματα και το βλέπουμε στην παρακάτω εικόνα με το εξής μήνυμα που φαίνεται. [16]



Εικόνα 21

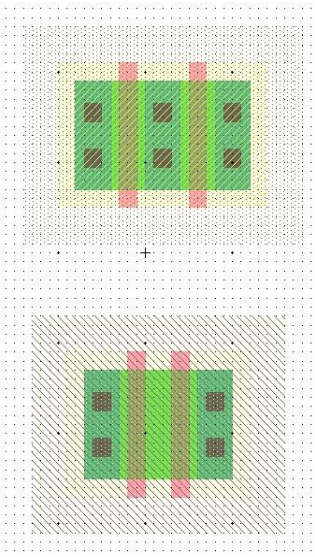
Πλέον αφού δεν έχουμε σφάλματα μπορούμε να ξεκινήσουμε να σχεδιάζουμε το layout του κυκλώματος.

3.2.2 ΔΗΜΙΟΥΡΓΙΑ LAYOUT

Το layout όπως είπαμε και στο προηγούμενο κεφάλαιο δείχνει πώς τα τρανζίστορ ή τα κελιά είναι φυσικά διατεταγμένα. Η καθαρότητα και η ακρίβεια είναι επιτακτική ανάγκη να έχουμε ένα καλό layout. Τα κενά που θα υπάρχουν μεταξύ των Components πρέπει να είναι συγκεκριμένα για να βγει σωστά. Παρόλα αυτά όμως, έχουμε το Electric το οποίο από μόνο του μας βοηθάει να μην κάνουμε τέτοια λάθη.

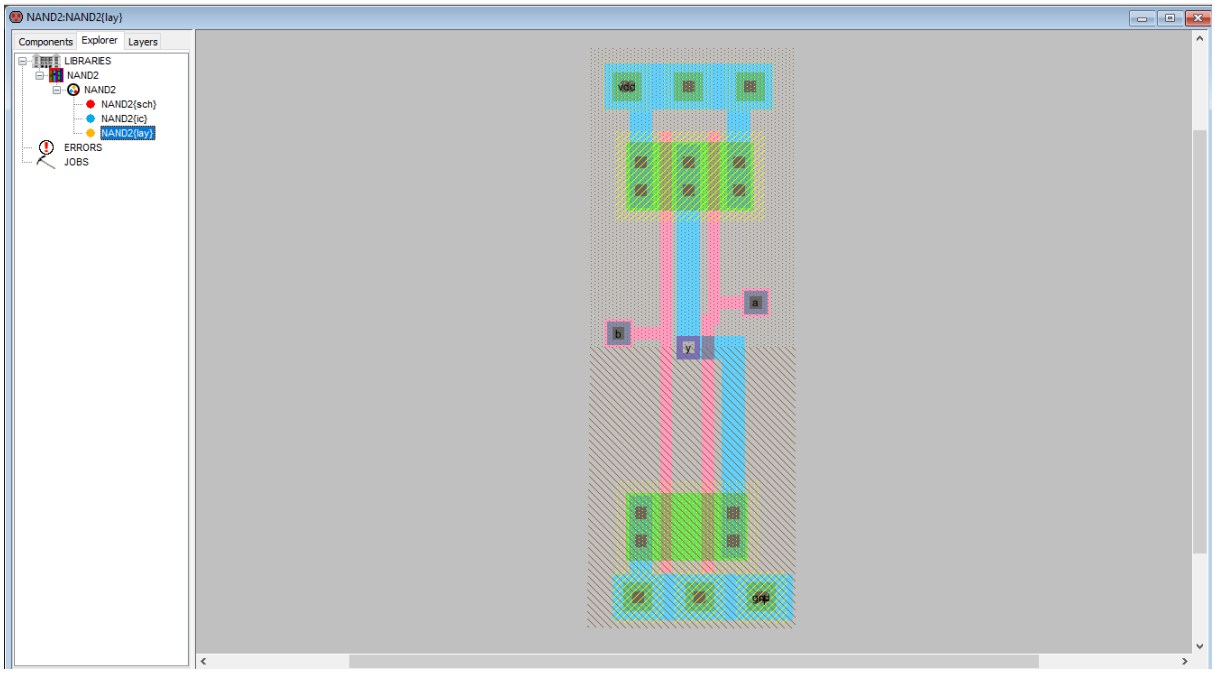
Προκειμένου να μην χαθούμε ανάμεσα στα σχέδιά μας , δημιουργούμε ένα καινούργιο ‘Cell’ όπως είπαμε στην προηγούμενη ενότητα αλλά αυτή τη φορά επιλέγουμε στο παράθυρο που μας δείχνει και η εικόνα 15 layout και πατάμε ‘OK’. Αμέσως καταλαβαίνουμε τη διαφορά

αφού η καρτέλα Components πλέον δείχνει τα Components που θα χρησιμοποιήσουμε για να φτιάξουμε το layout . Ένα layout αποτελείται από transistor NMOS και PMOS. Ένα NMOS transistor σχηματίζεται όταν το πολυπυρίτιο διασχίζει την διάχυση-N (N-diffusion), η οποία αντιπροσωπεύεται στο Electric με χρώμα πράσινο και περιβάλλεται από ένα διακεκομμένο κίτρινο στρώμα N-Select, όλα μέσα σε ένα κατακερματισμένο μαύρο P-well. Εύκολα μπορεί να μπερδευτεί κάποιος εδώ, αλλά όλα ξεκαθαρίζονται κοιτώντας την εικόνα 19 όπου έχει δημιουργηθεί η παραπάνω επεξήγηση και πιο συγκεκριμένα έχει τοποθετηθεί κατάλληλα ώστε τα PMOS να είναι παράλληλα και τα NMOS σε σειρά. Τώρα μας μένει να βάλουμε τη



Εικόνα 22

τροφοδοσία, τη γείωση, τις εισόδους και την έξοδο. Ακολουθώντας τις οδηγίες της εργαστηριακής ενότητας για το layout καταλήγουμε στην εικόνα 20 όπου πλέον έχουμε το κύκλωμα ολοκληρωμένο. [16]

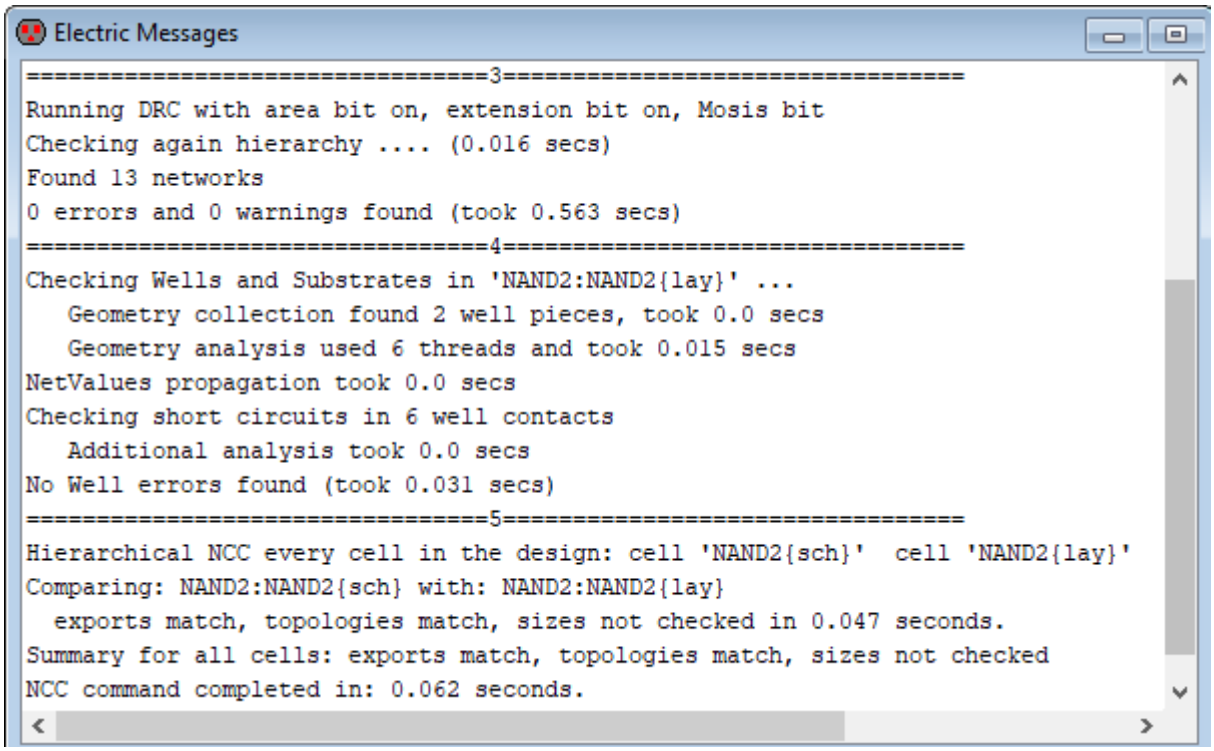


Εικόνα 23

Αφού λοιπόν ολοκληρώσουμε το σχέδιο μένει μόνο να γίνουν οι επαληθεύσεις DRC, οι οποίες έγιναν για το σχηματικό στην προηγούμενη ενότητα αλλά χρειάζεται και το layout, ERC (Electrical Rule Checking), το οποίο είναι ένας ελεγκτής κανόνων που διασφαλίζει ότι κάθε περιοχή Well και κάθε περιοχή υποστρώματος (substrate area) έχουν τουλάχιστον μία επαφή. Τέλος, επαληθεύουμε με NCC (Network Consistency Checking), με το οποίο γίνεται σύγκριση του σχηματικού και του layout, συγκρίνοντας εισόδους, εξόδους, τοπολογία των τρανζίστορ και τα μεγέθη τους. Όλα γίνονται μέσω των Tools ως εξής :

- Tools > DRC > Check Hieratically.
- Tools > ERC > Check Wells.
- Tools > NCC > Schematic and Layout Views of Cell in Current Window

Στην παρακάτω εικόνα βλέπουμε τα αποτελέσματα για την NAND2 που μόλις φτιάξαμε και των τριών επαληθεύσεων.



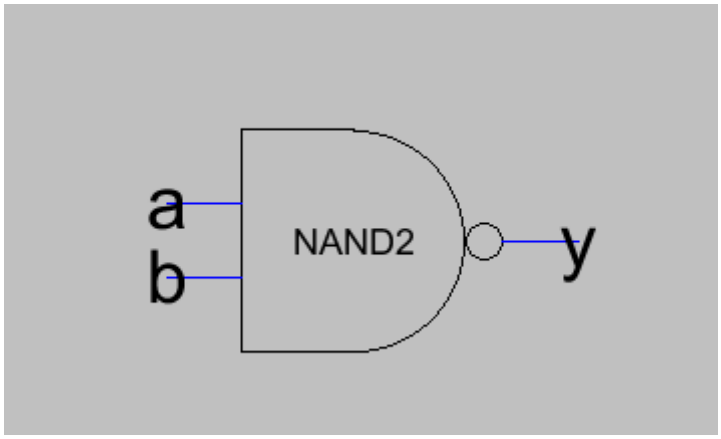
Εικόνα 24

Έχοντας ολοκληρώσει και τις τρεις αυτές επαληθεύσεις χωρίς κάποιο μήνυμα λάθους, είναι έτοιμο για λογική επαλήθευση, αλλά ας δούμε πρώτα εν συντομία την δημιουργία εικονιδίου παρακάτω.

3.2.3 ΔΗΜΙΟΥΡΓΙΑ ΕΙΚΟΝΙΔΙΟΥ

Η δημιουργία εικονιδίου είναι εύκολη υπόθεση και κυρίως γίνεται για τη δική μας διευκόλυνση όταν είναι να φτιάξουμε άλλα σχηματικά χρησιμοποιώντας μόνο το εικονίδιο, και κυρίως για ιεραρχική σχεδίαση. Όπως είμαστε στο σχηματικό του κυκλώματός μας πατάμε View > Make Cell Icon και θα δούμε στο σχηματικό αλλά και στην βιβλιοθήκη μας να υπάρχει

εκτός από το σχηματικό και το layout, το Icon του κυκλώματος μας. Πατώντας το, ανοίγει και ταυτόχρονα αλλάζει η καρτέλα με τα components με εργαλεία για τον σχηματισμό του Icon. Αφού λοιπόν δημιουργούμε την NAND2, δημιουργούμε το αντίστοιχο εικονίδιο ακολουθώντας πάντα τις οδηγίες των εργαστηριακών ασκήσεων το οποίο το βλέπουμε στην παρακάτω εικόνα. [16]



Εικόνα 25

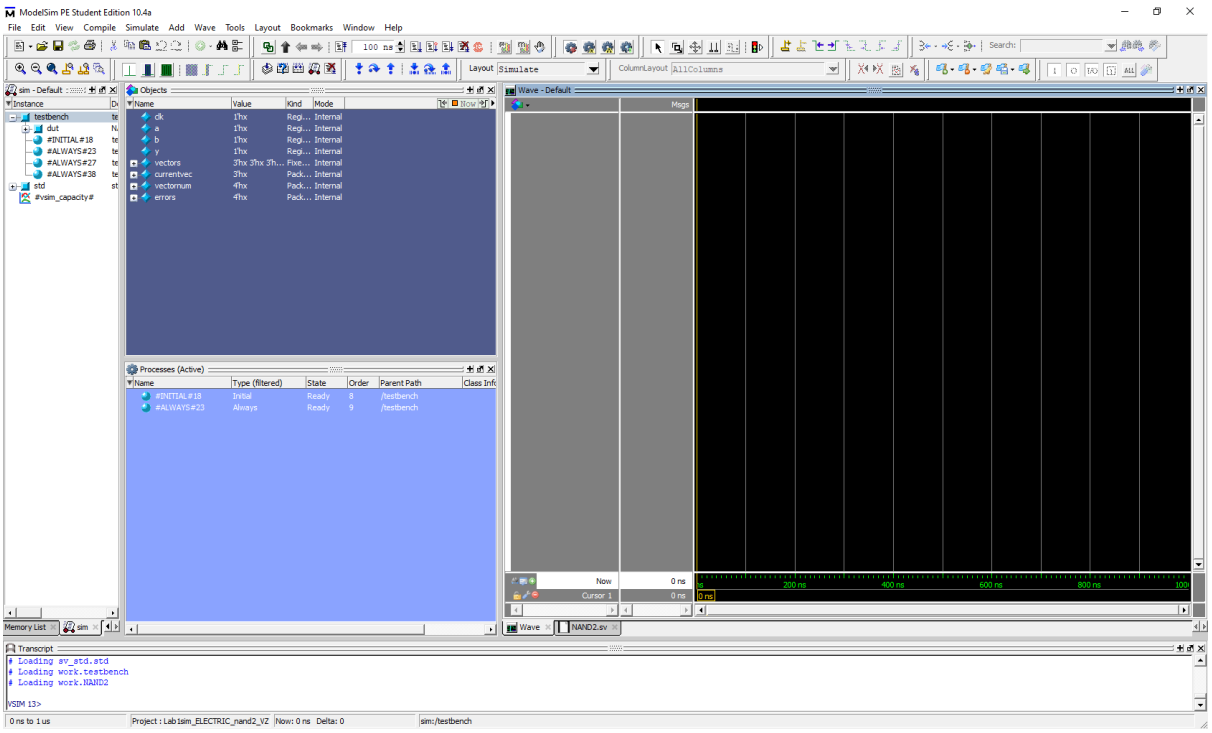
Έχοντας ολοκληρώσει και τις τρεις αυτές επαληθεύσεις χωρίς κάποιο μήνυμα λάθους και το κελί της NAND2, είναι έτοιμο για λογική επαλήθευση (Logic Verification). Θα δούμε τον τρόπο στην επόμενη υπό ενότητα.

3.2.4 LOGIC VERIFICATION

Για αυτή την επαλήθευση αναφερθήκαμε αναλυτικά στο προηγούμενο κεφάλαιο και τώρα θα δούμε και στην πράξη μέσω του προγράμματος ModelSim πως θα γίνει πλήρη στοίχιση για το κύκλωμα που δημιουργήσαμε, την πύλη NAND 2 εισόδων. Στην ουσία δηλαδή το ModelSim θα μας δείξει τον πίνακα αληθείας των κυκλωμάτων.

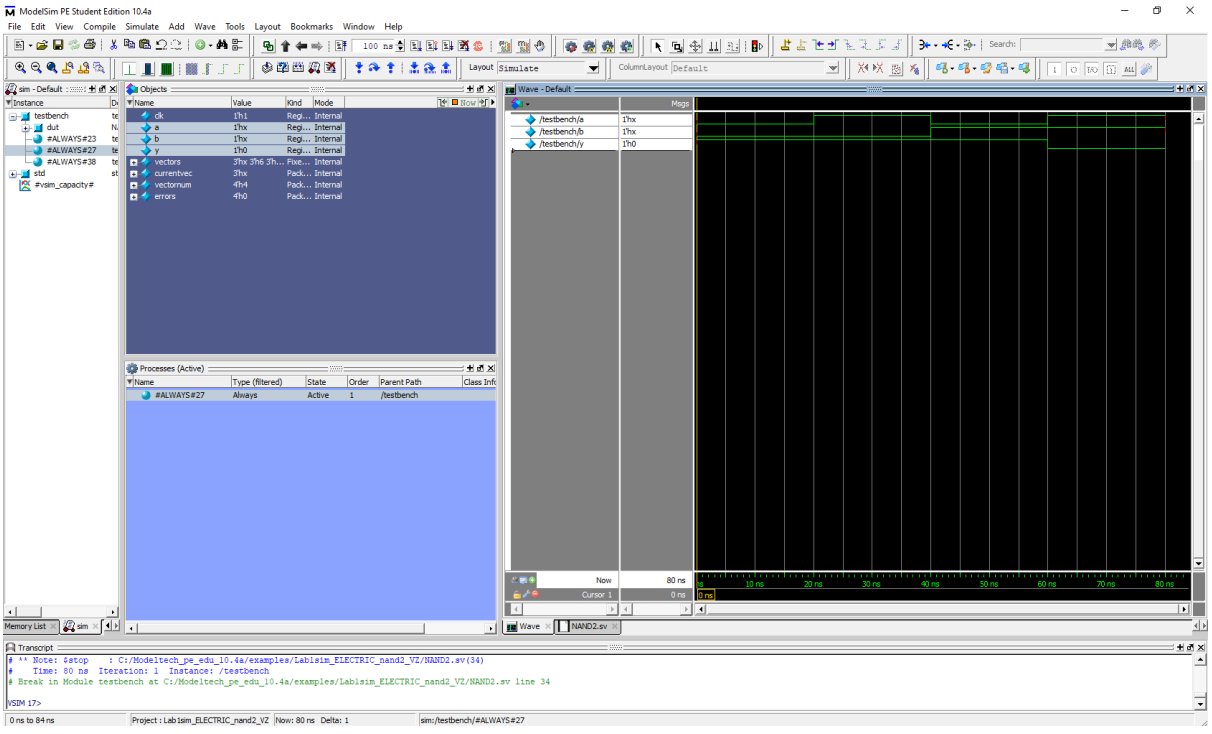
Ανοίγοντας το ModelSim, δημιουργούμε ένα καινούργιο project ακολουθώντας τις οδηγίες της εκφώνησης της 1ης εργαστηριακής άσκησης. Για να επαληθεύσουμε όμως το κύκλωμα που δημιουργήσαμε στο Electric, ανοίγουμε το Electric στη βιβλιοθήκη NAND2, πατάμε όσο είμαστε στο layout Tools > Simulation (Verilog) > Write Verilog Deck και το electric "μεταφράζει" με λίγα λόγια το κύκλωμά μας σε κώδικα Verilog. Αφού σώσουμε το αρχείο, το μεταφέρουμε μέσα στο project μαζί με το αρχείο NAND2-vectors που θα δημιουργήσουμε σε .txt, που είναι τα vectors της NAND και θα τα διαβάσει μέσω του testbench για να γίνει η επαλήθευση. [16]

Αφού, λοιπόν, έχουμε τα αρχεία μέσα στο project μας, μπορούμε να κάνουμε compile για να μπορέσουμε να ξεκινήσουμε την προσομοίωση του κυκλώματος μέσω των waveforms. Πατώντας στο ModelSim πάνω από την γραμμή εργαλείων Compile > Compile All γίνεται compile χωρίς προβλήματα. Στη συνέχεια πατάμε Simulate > Start Simulate και ανοίγει ένα παράθυρο όπου θα επιλέξουμε τι να κάνει simulate και επιλέγουμε work > testbench και θα ανοίξουν τα παράθυρα της προσομοίωσης.



Εικόνα 26

Για να ξεκινήσει η προσομοίωση και να δούμε τα αποτελέσματα στο wave επιλέγουμε από τα objects αυτά που θέλουμε, στην συγκεκριμένη περίπτωση a, b (Είσοδοι) και c (Έξοδος) και τα σέρνουμε στο παράθυρο του wave και τέλος πατάμε από την γραμμή εργαλείων run -all και έχουμε τα αποτελέσματα της Εικόνας 24 όπου μπορεί να δει κανείς ότι είναι σωστά σύμφωνα και με τον πίνακα αληθείας της NAND αλλά και με τα δεδομένα του αρχείου test-vectors.



Εικόνα 27

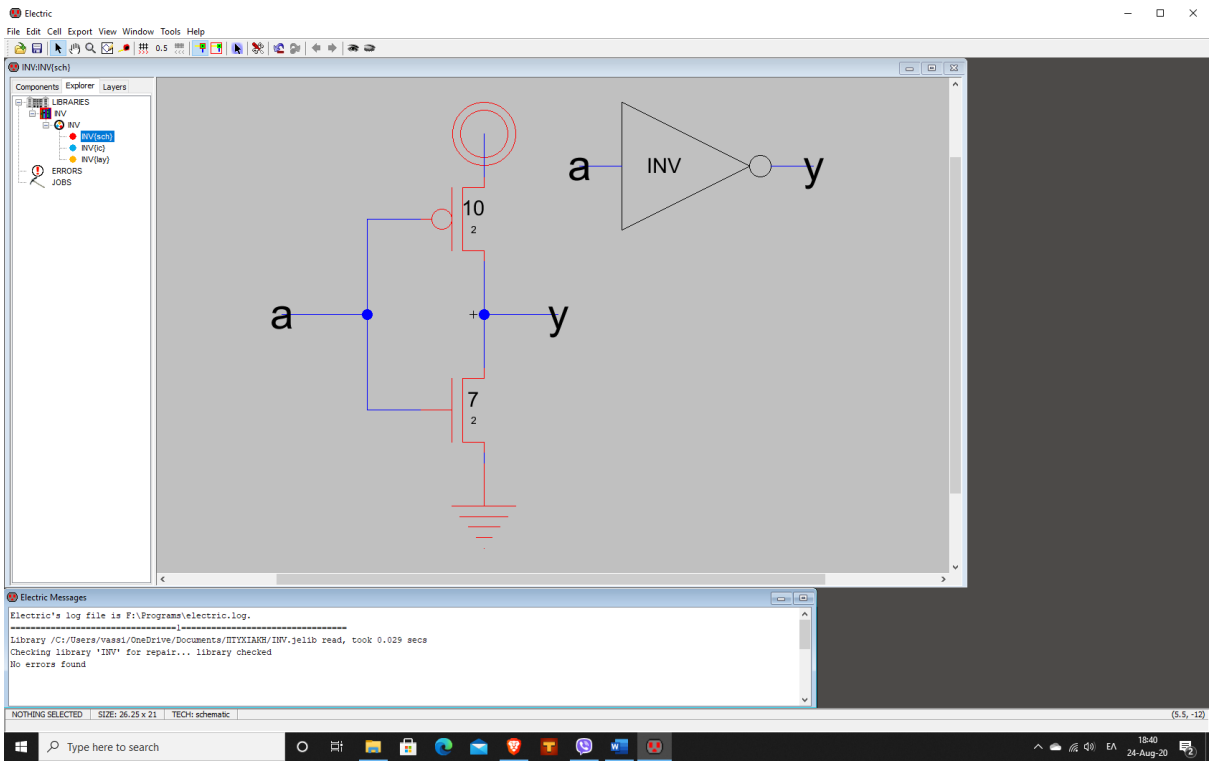
Σύμφωνα με τα παραπάνω καταλήγουμε σε ένα σωστό και λειτουργικό κύκλωμα. Ας δούμε λοιπόν και τα αποτελέσματα των υπόλοιπων ζητούμενων της εργαστηριακής άσκησης, βασισμένα πάντα στα παραπάνω λεγόμενα.

3.3 ΑΠΟΤΕΛΕΣΜΑΤΑ 1ΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ

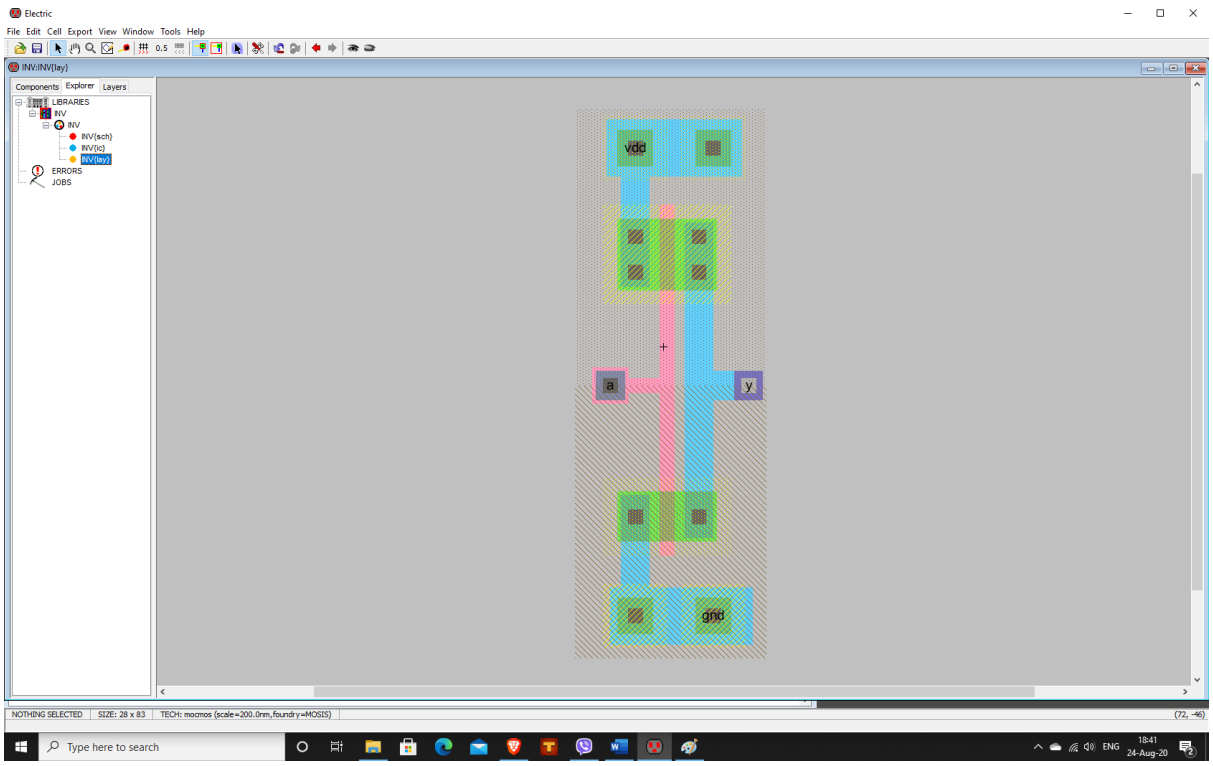
Στην αρχή του κεφαλαίου μιλήσαμε για τα ζητούμενα της πρώτης εργαστηριακής ενότητας, των οποίων τα αποτελέσματά θα εξετάσουμε σε αυτό το υποκεφάλαιο. Όπως είδαμε για την πύλη NAND δύο εισόδων υπήρχε η διαδικασία που πραγματοποιήσαμε στις προηγούμενες ενότητες και ακολουθήσαμε την ίδια ακριβώς διαδικασία και για τα υπόλοιπα AND, NOR και OR δύο εισόδων καθώς και του inverter που χρειάζεται.

3.3.1 INVERTER

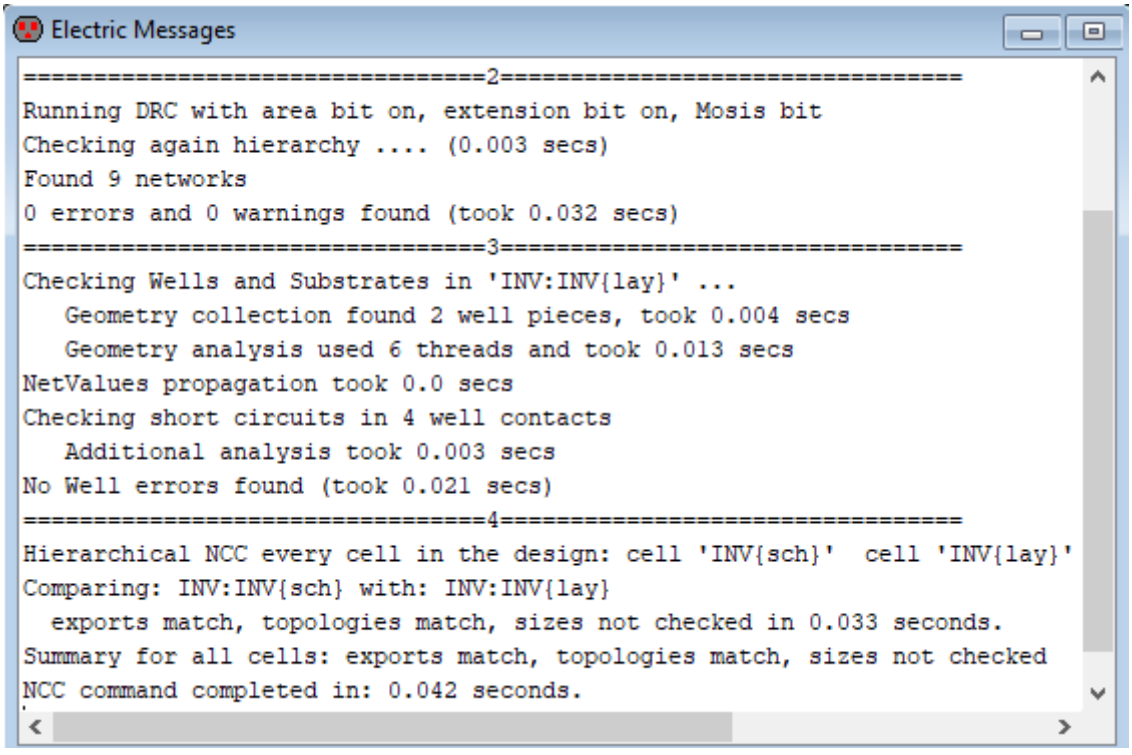
Ξεκινώντας, δημιουργήσαμε τον inverter διότι είναι σημαντικό κομμάτι προκειμένου να φτιάξουμε τις πύλες OR και AND δύο εισόδων. Στις παρακάτω εικόνες βλέπουμε τα τελικά σχέδια καθώς και τις επαληθεύσεις και τις προσομοιώσεις που έγιναν σε Electric και ModelSim.



Εικόνα 28



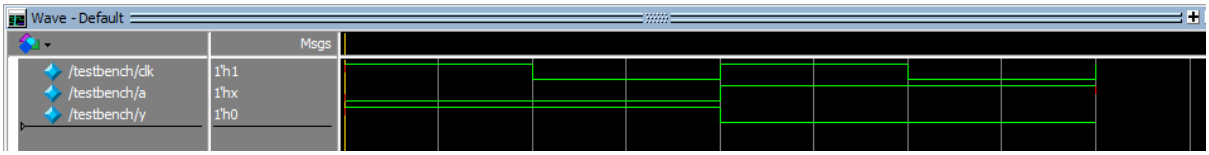
Εικόνα 29



Εικόνα 30

Προκειμένου να γίνει επαλήθευση μέσω του ModelSim χρησιμοποιούμε το testbench που έχουμε με κάποιες μικρές αλλαγές στον κώδικα ώστε να γίνει σωστά η προσομοίωση. Πιο συγκεκριμένα, έγιναν αλλαγές στο πρόγραμμα στο όνομα της πύλης αλλά και στο πόσες

εισόδους βάλαμε, που εδώ είναι μόνο μία και στην παρακάτω εικόνα 29 μπορούμε να δούμε ότι έγινε κανονικά προσομοίωση χωρίς κάποιο πρόβλημα.

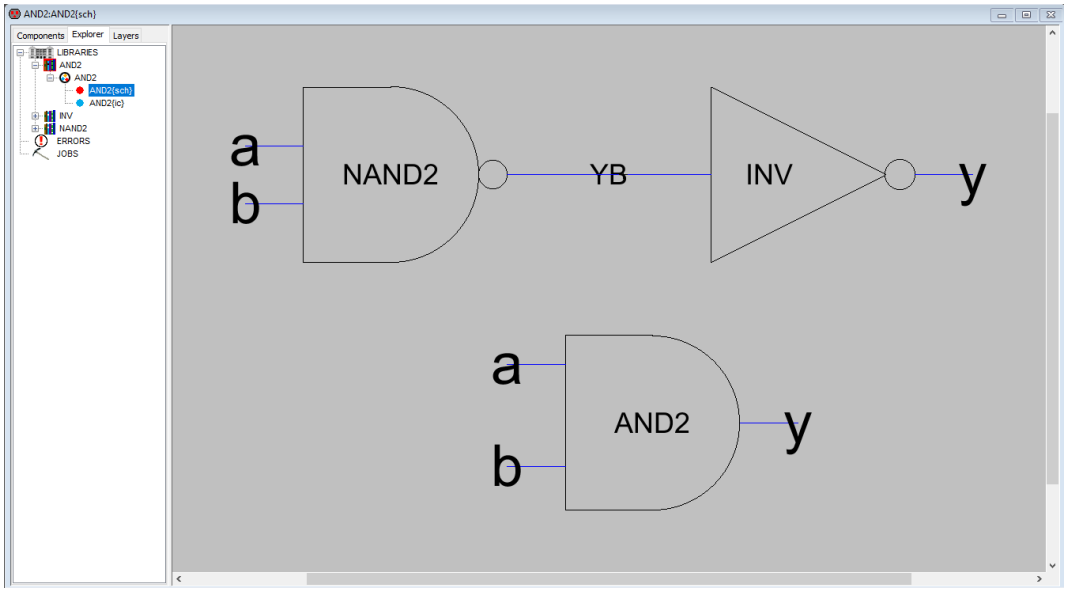


Εικόνα 31

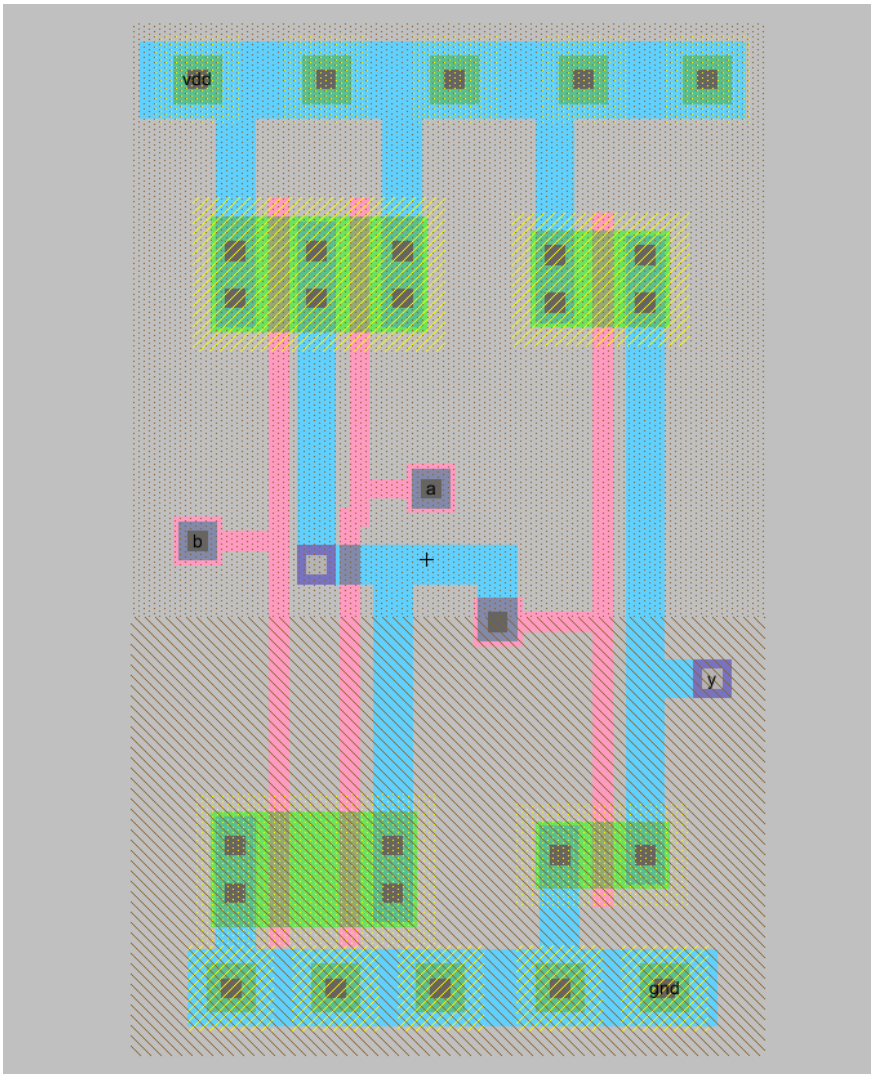
Επομένως, βλέπουμε πως το κύκλωμά μας δουλεύει κανονικά χωρίς προβλήματα, οπότε συνεχίζουμε παρακάτω στην AND 2 εισόδων να δούμε τα αποτελέσματα.

3.3.2 AND 2 ΕΙΣΟΔΩΝ

Η δουλειά μας εδώ έγινε πιο απλή διότι, αφού έχουμε ήδη δημιουργήσει την NAND και τον inverter, μπορούμε να χρησιμοποιήσουμε συνδυαστικά τα σχηματικά και τα layouts των προηγούμενων κυκλωμάτων. Ειδικά για το σχηματικό, λόγω του electric, μπορούμε να χρησιμοποιήσουμε τα icons των NAND2 και INV. Στις παρακάτω εικόνες βλέπουμε τα αποτελέσματα των σχηματικών και των layouts καθώς και το Logic Verification μέσω του ModelSim. [16]



Εικόνα 32

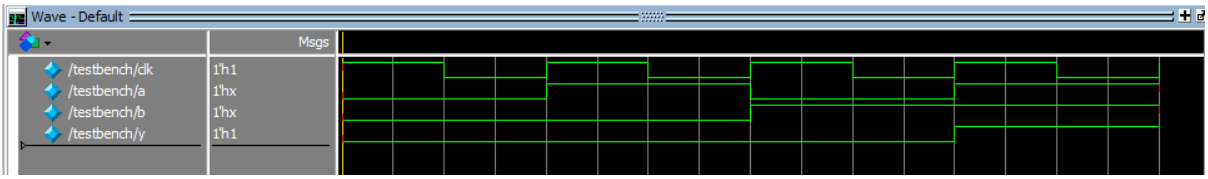


Εικόνα 33

Electric Messages

```
================================3================================
Running DRC with area bit on, extension bit on, Mosis bit
Checking again hierarchy .... (0.003 secs)
Found 16 networks
0 errors and 0 warnings found (took 0.031 secs)
================================4================================
Checking Wells and Substrates in 'LAB1_VZ:AND2{lay}' ...
   Geometry collection found 2 well pieces, took 0.006 secs
   Geometry analysis used 6 threads and took 0.011 secs
NetValues propagation took 0.001 secs
Checking short circuits in 10 well contacts
   Additional analysis took 0.003 secs
No Well errors found (took 0.022 secs)
================================5================================
Hierarchical NCC every cell in the design: cell 'AND2{sch}'  cell 'AND2{lay}'
Comparing: LAB1_VZ:AND2{sch} with: LAB1_VZ:AND2{lay}
  exports match, topologies match, sizes not checked in 0.041 seconds.
Summary for all cells: exports match, topologies match, sizes not checked
NCC command completed in: 0.052 seconds.
```

Εικόνα 34

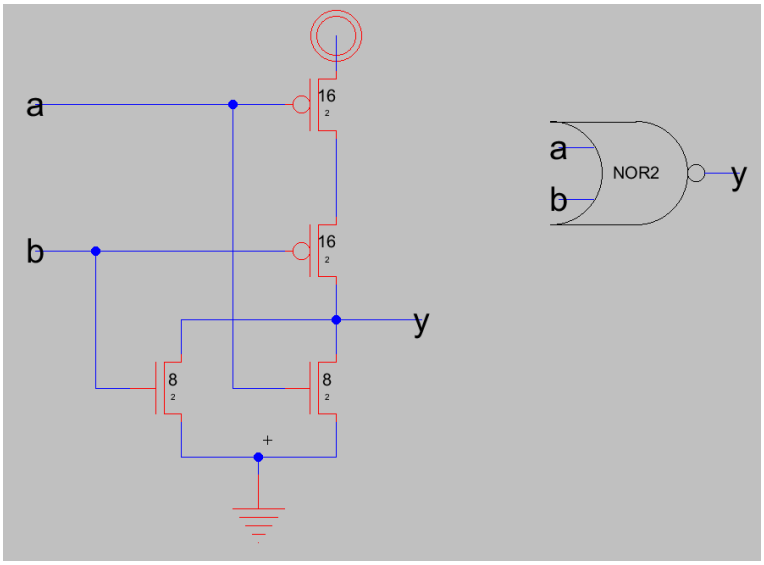


Εικόνα 35

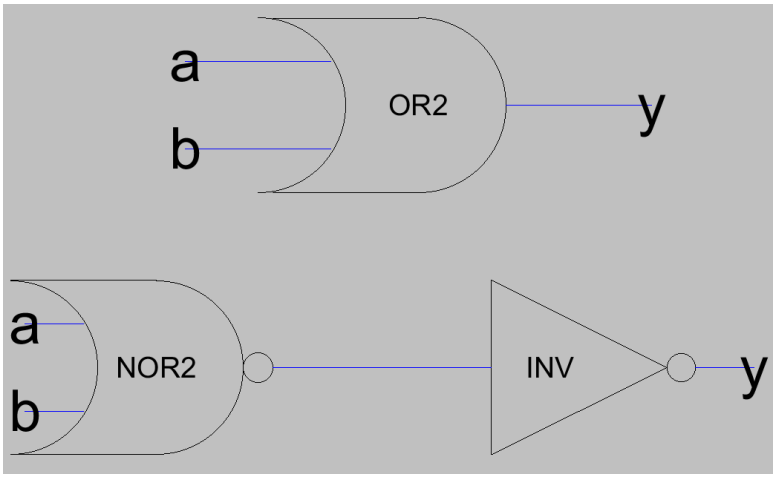
Έχοντας ακολουθήσει τις προηγούμενες διαδικασίες, είδαμε ότι και αυτό το κύκλωμα είναι σωστό, σύμφωνα πάντα με τις επαληθεύσεις και τις προσομοιώσεις των προγραμμάτων που χρησιμοποιήσαμε.

3.3.4 NOR ΚΑΙ OR 2 ΕΙΣΟΔΩΝ

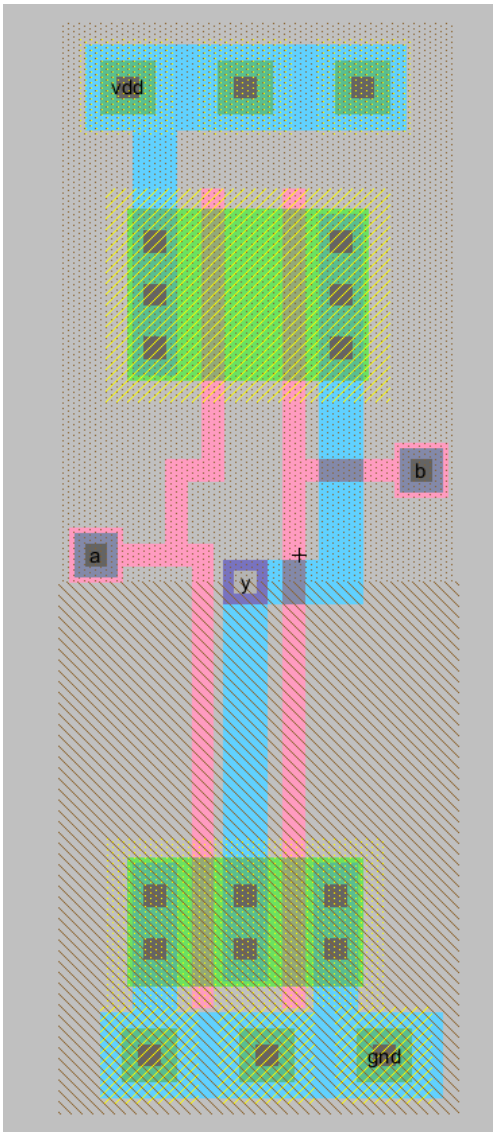
Σε αυτήν την ενότητα συνεχίζουμε με τα αποτελέσματα των ζητούμενων, που όπως παρατηρείται από τις επόμενες εικόνες είναι σωστά χωρίς να έχουμε αντιμετωπίσει κάποιο πρόβλημα κατά την εκτέλεση.



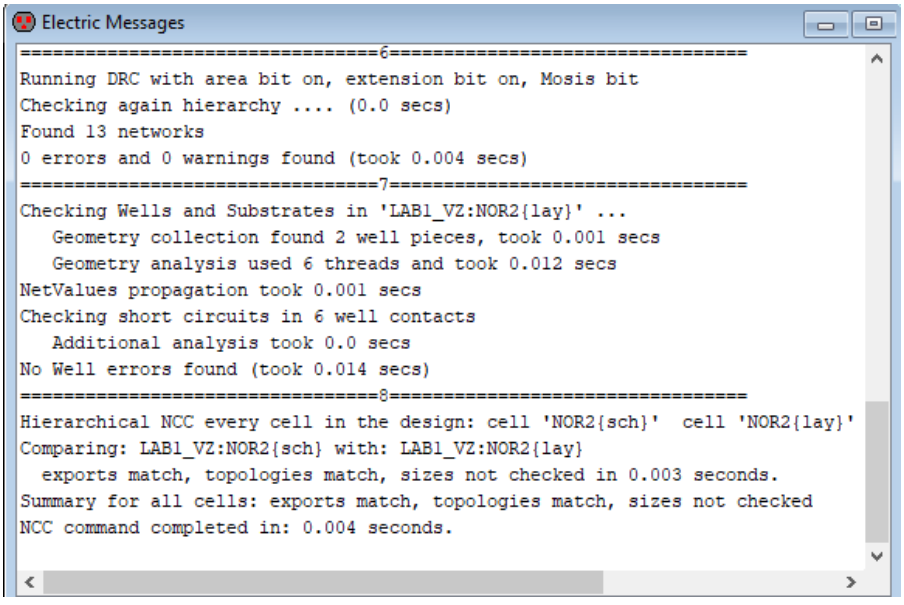
Εικόνα 36



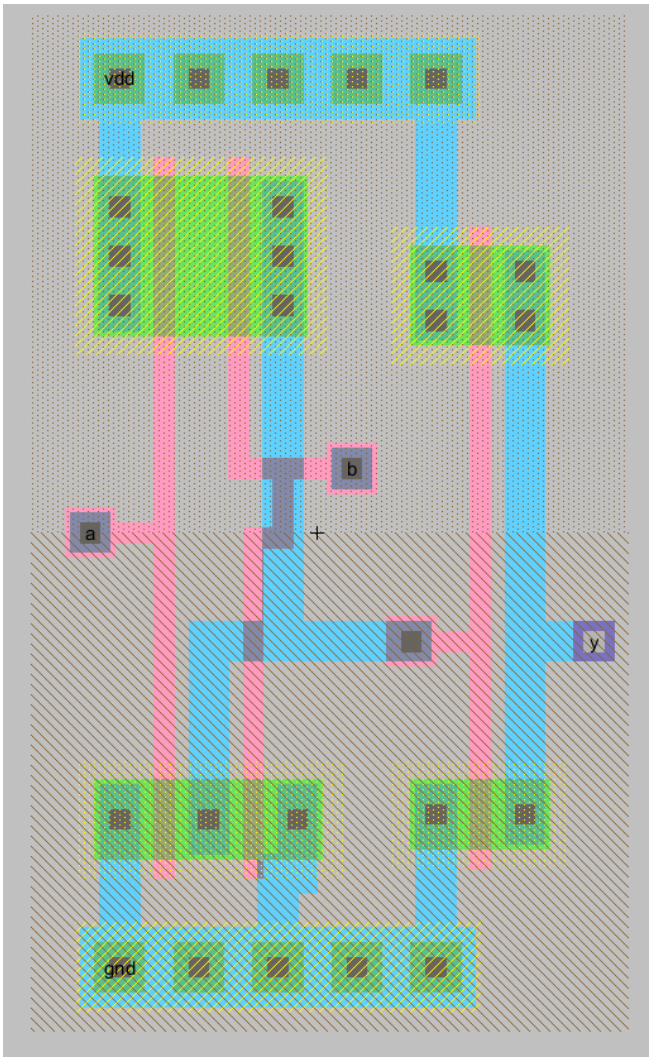
Εικόνα 37



Εικόνα 38



Εικόνα 39

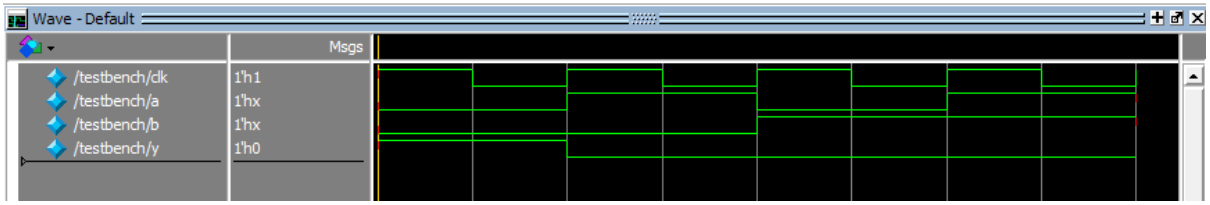


Εικόνα 40

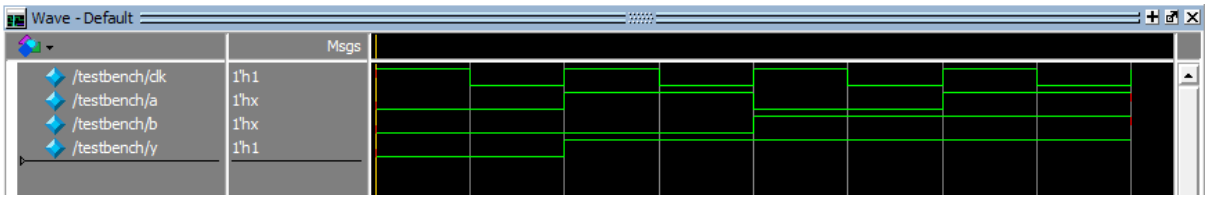
Electric Messages

```
================================9================================
Running DRC with area bit on, extension bit on, Mosis bit
Checking again hierarchy .... (0.0 secs)
Found 16 networks
0 errors and 0 warnings found (took 0.003 secs)
================================10================================
Checking Wells and Substrates in 'LAB1_VZ:OR2{lay}' ...
   Geometry collection found 2 well pieces, took 0.002 secs
   Geometry analysis used 6 threads and took 0.011 secs
NetValues propagation took 0.0 secs
Checking short circuits in 10 well contacts
   Additional analysis took 0.0 secs
No Well errors found (took 0.013 secs)
================================11================================
Hierarchical NCC every cell in the design: cell 'OR2{sch}'  cell 'OR2{lay}'
Comparing: LAB1_VZ:OR2{sch} with: LAB1_VZ:OR2{lay}
  exports match, topologies match, sizes not checked in 0.004 seconds.
Summary for all cells: exports match, topologies match, sizes not checked
NCC command completed in: 0.005 seconds.
```

Εικόνα 41



Εικόνα 42



Εικόνα 43

Ολοκληρώσαμε με τα ζητούμενα της εργαστηριακής άσκησης και προχωράμε στα συμπεράσματα που έχουμε μετά την ολοκλήρωση και των τελευταίων δύο κυκλωμάτων.

3.4 ΣΥΜΠΕΡΑΣΜΑΤΑ

Χωρίς να μακρηγορούμε, τελειώνοντας την 1η εργαστηριακή ενότητα καταλαβαίνει κανείς πώς η σχεδίαση ψηφιακών κυκλωμάτων μπορεί να γίνει εύκολη ακολουθώντας τους κανόνες σχεδίασης. Ωστόσο, είναι αξιοσημείωτο πως πολλές φορές οι μικρές λεπτομέρειες κάνουν την διαφορά και κυρίως στα layouts, όπου χρειάζεται να κάνουμε λεπτούς χειρισμούς για να πετύχουμε τα σωστά αποτελέσματα χωρίς να υπάρξουν προβλήματα.

Επομένως, σύμφωνα με τις παραπάνω ενότητες βλέπουμε πως τα συγκεκριμένα κυκλώματα ήταν σχετικά εύκολα, αλλά ήταν σημαντική η υλοποίηση τους γιατί μαθαίνουμε πως να σχεδιάζουμε, να επαληθεύουμε και να προσομοιώνουμε ένα κύκλωμα που μόλις δημιουργήσαμε. Είναι τελικά, η βάση για την 2η εργαστηριακή ενότητα που θα ακολουθήσει παρακάτω, όπου θα ξεκινήσουμε να σχεδιάζουμε τον 8-bit MIPS επεξεργαστή, που είναι και το αποτέλεσμα αυτής της εργασίας.

ΚΕΦΑΛΑΙΟ 4: ΔΕΥΤΕΡΗ ΕΡΓΑΣΤΗΡΙΑΚΗ ΕΝΟΤΗΤΑ

4.1 ΕΙΣΑΓΩΓΗ

Σε αυτή την εργαστηριακή ενότητα, αφού μάθαμε να δουλεύουμε τα προγράμματα που χρησιμοποιήσαμε με τη δημιουργία βασικών και απλών κυκλωμάτων, αρχίζουμε να σχεδιάζουμε τον 8-bit MIPS processor. Όπως προαναφέρθηκε, τα περισσότερα μέρη του επεξεργαστή δίνονται, αλλά κάποια βασικά και ενδιαφέροντα μέρη έχουν αφαιρεθεί. Τέλος, είναι σημαντικό να αναφέρουμε ότι προκειμένου να προσομοιώσουμε τα σχέδια μας στο ModelSim δεν χρησιμοποιήσαμε τη μέθοδο του προηγούμενου κεφαλαίου. Χρησιμοποιήσαμε ένα πρόγραμμα που καλείται μέσω του testbench, το οποίο προσομοιώνει όλο το σύστημα. [17]

Καλούμαστε, λοιπόν, να μάθουμε το Datapath και τη λειτουργία του, ώστε να το ολοκληρώσουμε, καθώς και να σχεδιάσουμε και να συνδέσουμε τα λεγόμενα wordslices μέσα στην ALU του επεξεργαστή μας.

4.2 ΥΛΟΠΟΙΗΣΗ 2ΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ

4.2.1 RTL SIMULATION ΚΑΙ WORDSLICES

Το RTL Simulation από αυτό το κεφάλαιο θα διαφέρει σε σχέση με τα προηγούμενα. Τώρα θα προσομοιώνουμε το σύστημα εξ’ ολοκλήρου, δηλαδή, όλο τον μικροεπεξεργαστή MIPS, και όχι όπως στις προηγούμενες ενότητες.

Η ALU για να ολοκληρωθεί χρειάζεται ένα 8-bit AND και ένα 8-bit OR wordslice. Τα wordslices είναι μία τεχνική που χρησιμοποιείται για την κατασκευή ενός επεξεργαστή από ενότητες επεξεργαστών μικρότερου πλάτους bit, με σκοπό την αύξηση του μήκους της λέξης. Πιο απλά χρησιμοποιείται για να φτιάξουμε έναν επεξεργαστή n-bit. Κάθε μία από αυτές τις λειτουργικές μονάδες επεξεργάζεται ένα πεδίο bit ή "slice" ενός τελεστή. Τα ομαδοποιημένα στοιχεία επεξεργασίας θα έχουν τότε τη δυνατότητα να επεξεργάζονται το επιλεγμένο πλήρες μήκος μιας συγκεκριμένης σχεδίασης λογισμικού. Ενδιαφέρον αποτελεί το γεγονός πως ο καταμερισμός αυτός λίγο πολύ έχει εξαφανιστεί λόγω της εμφάνισης των μικροεπεξεργαστών. Πρόσφατα όμως, έχει χρησιμοποιηθεί σε ALU για κβαντικούς υπολογιστές, όπως επίσης και ως τεχνική λογισμικού. [17]

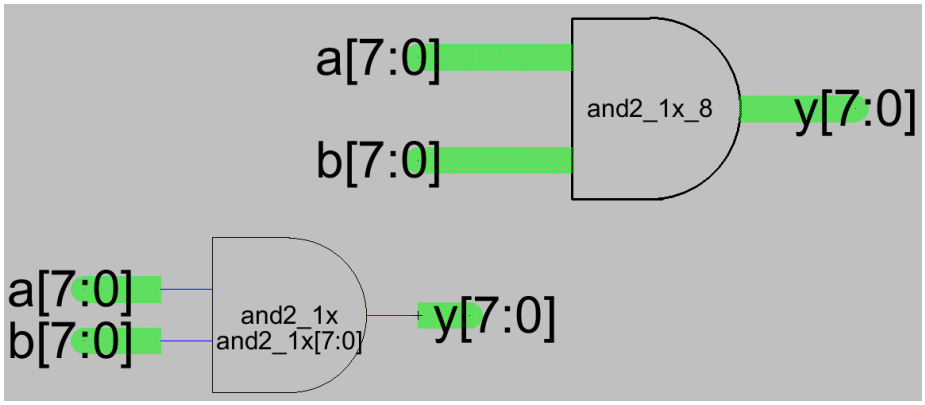
Όπως αναφέραμε, αντί του testbench που εφαρμόζει και υποστηρίζει διανύσματα, θα επαληθεύουμε μέσω μίας εξωτερικής μονάδας μνήμης που θα φορτώνει ένα πρόγραμμα δοκιμής αποθηκευμένο στο αρχείο memfile.dat. Το πρόγραμμα δοκιμάζει τη βασική λειτουργικότητα του επεξεργαστή και, εάν είναι επιτυχής, γράφει το 7 στη διεύθυνση μνήμης 0x4C. Το testbench ελέγχει ότι το ο επεξεργαστής έγραψε την τιμή με επιτυχία. Τα προγράμματα αυτά μπορούμε να τα δούμε και στο αρχείο της συγκεκριμένης πτυχιακής. [17]

Στην επόμενη υποενότητα ξεκινάμε την υλοποίηση 8-bit AND και ένα 8-bit OR wordslice. Οι παραπάνω πληροφορίες ήταν αναγκαίες να γίνουν γνωστές, διότι μας βοηθούν στην υλοποίηση της συγκεκριμένης εργαστηριακής ενότητας, καθώς και των επομένων.

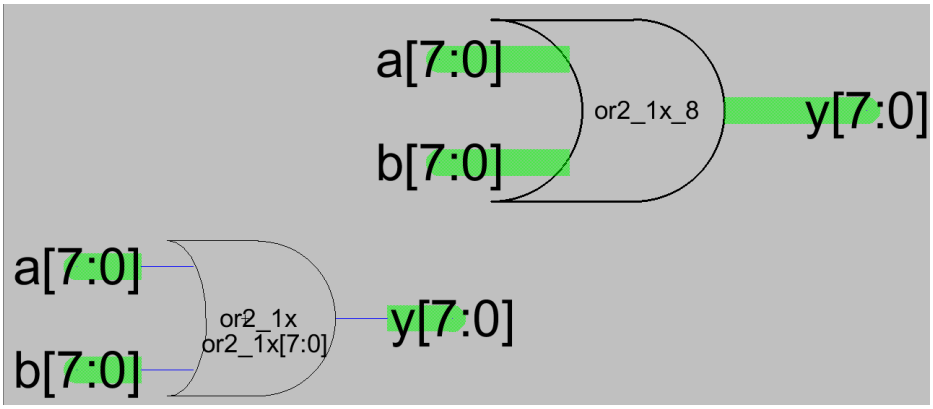
4.2.2 ΔΗΜΙΟΥΡΓΙΑ ΣΧΗΜΑΤΙΚΟΥ ΓΙΑ 8-BIT AND ΚΑΙ 8-BIT OR

Η δημιουργία ενός wordslice, ακολουθώντας πάντα τις οδηγίες της εργαστηριακής άσκησης, θα δει κανείς ότι είναι εύκολη σχετικά. Κυρίως για το σχηματικό απλώς χρησιμοποιούμε την and_1x και or_1x, η οποία δίνεται στην βιβλιοθήκη muddlib που υπάρχει στα αρχεία των ασκήσεων και διαθέτει πάρα πολλά έτοιμα μοντέλα κυκλωμάτων που μας γλυτώνουν χρόνο. [17]

Στις δύο παρακάτω εικόνες παρατηρεί κανείς πως στα exports των κυκλωμάτων έχουμε δώσει τα ονόματα, για παράδειγμα, a[7:0], που σημαίνει ότι το πρόγραμμα θα δημιουργήσει 8 αντίγραφα, γιατί, όπως προαναφέραμε, θα αυξήσουμε το μήκος της λέξης. Επιπλέον, για να γίνει εφικτή η σύνδεση ενώνουμε με bus τα exports κι όχι με wire. Το Bus arcs των Schematics technology είναι ένα ειδικό arc που μπορεί να μεταφέρει πολλαπλά σήματα. Όταν δίνουμε ένα όνομα δικτύου στα busses, είναι δυνατό να καθορίσουμε σύνθετα ονόματα busses, όπως ειπώθηκε παραπάνω, a[7:0]. [17]



Εικόνα 44

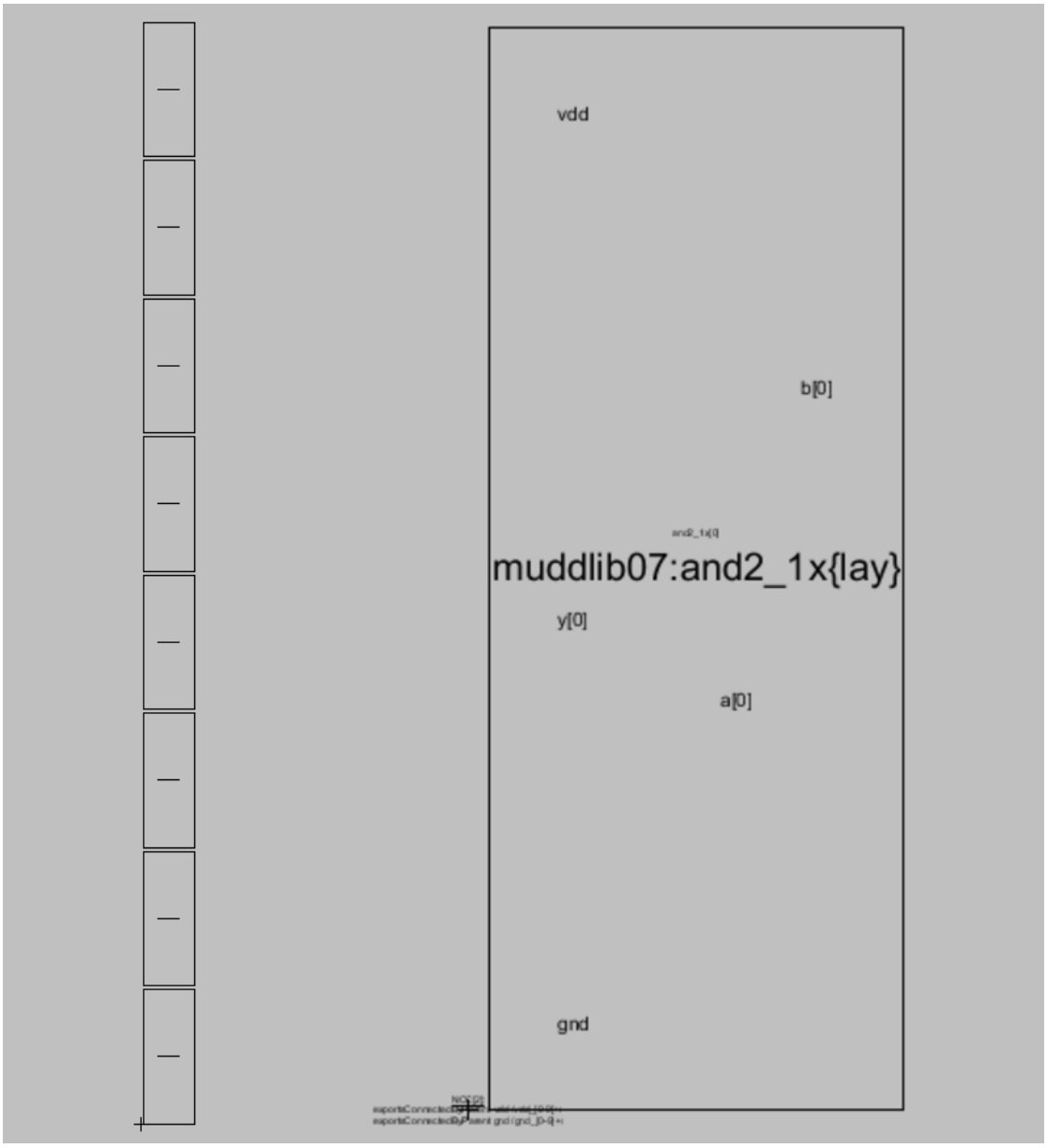


Εικόνα 45

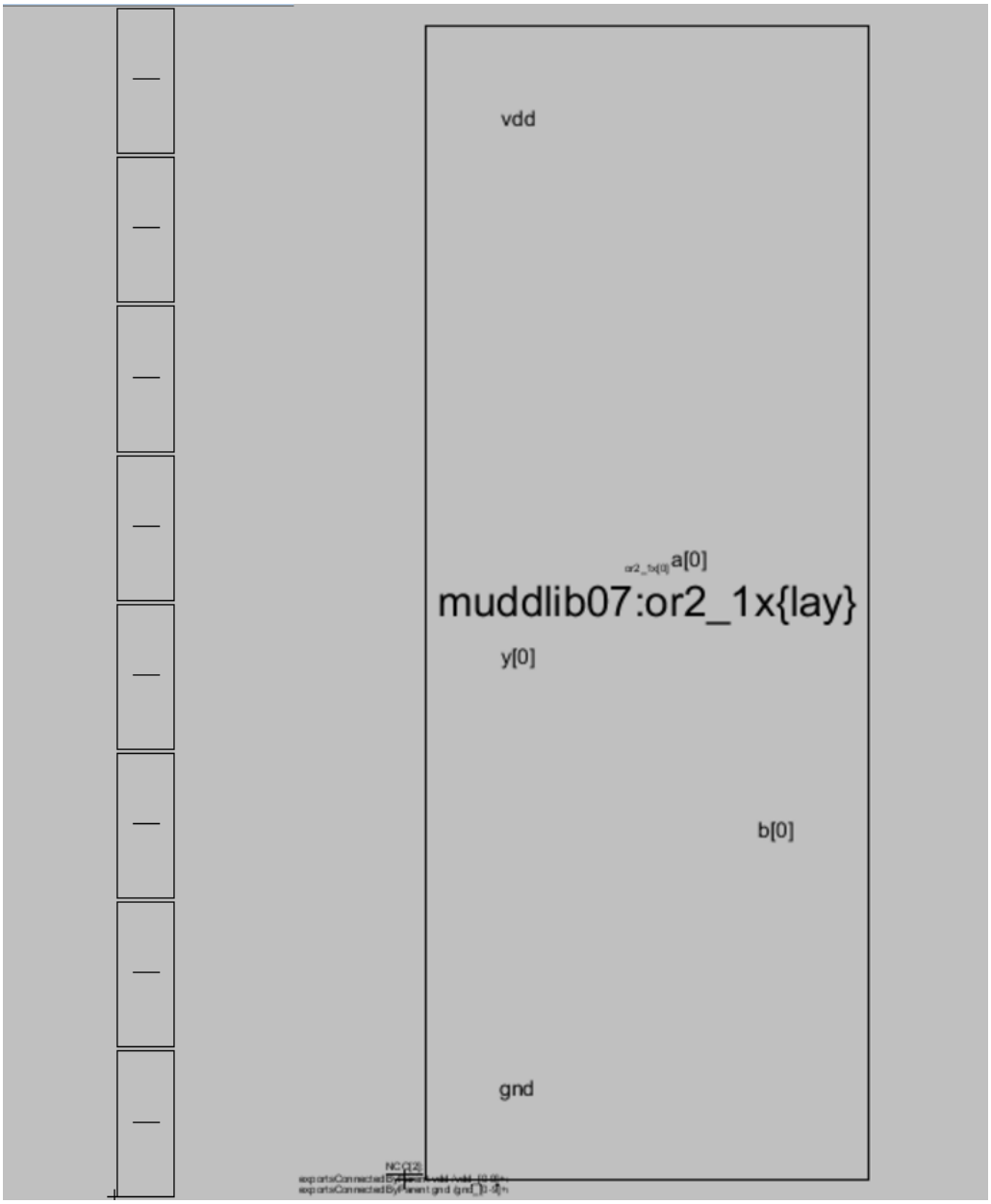
4.2.3 ΔΗΜΙΟΥΡΓΙΑ LAYOUT ΓΙΑ 8-BIT AND ΚΑΙ 8-BIT OR

Στο κεφάλαιο 3 δημιουργήσαμε ένα μόνο Layout που είχε αρκετές δυσκολίες, αλλά με πολύ προσοχή στην λεπτομέρεια γίνεται αμέσως η δουλειά. Στην υποενότητα αυτή, λόγω του ότι το ζητούμενο είναι η δημιουργία wordslice, για την δημιουργία των κυκλωμάτων 8-bit AND και 8-bit OR χρησιμοποιήσαμε εργαλεία του Electric, καθώς επίσης και τα δεδομένα των κυκλωμάτων από τη βιβλιοθήκη muddlib7.

Για τη δημιουργία του worldslice χρησιμοποιήσαμε και στα δύο κυκλώματα τη λειτουργία Array και mimic-stich του Electric για να κάνουμε πιο γρήγορη τη δημιουργία του layout. Δημιουργήσαμε άμεσα επτά αντίγραφα του κυκλώματος σε μια σειρά, όπως φαίνεται στην παρακάτω εικόνα για την κάθε είσοδο και έξοδο. [17]



Εικόνα 46



Εικόνα 47

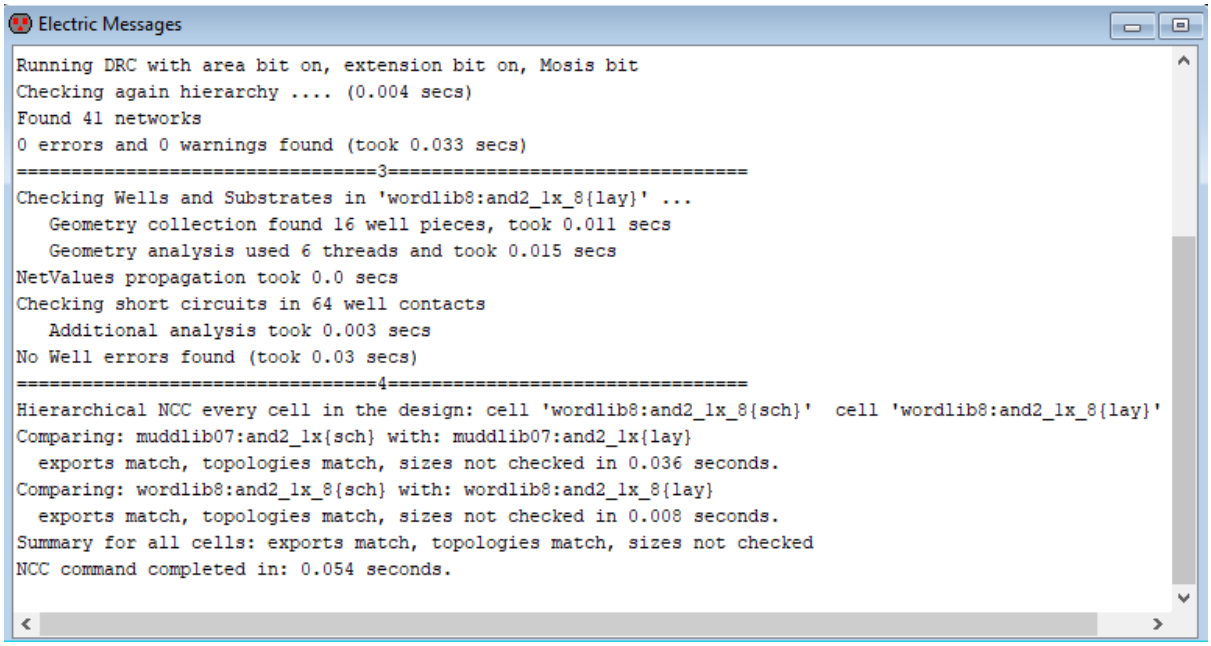
Προκειμένου το Electric να επαληθεύσει σωστά τόσο σε ERC και NCC πρέπει να συνδέσουμε το vdd και ground. Αυτό θα γίνει σε μεγαλύτερο επίπεδο σχεδίασης, αλλά προς στιγμήν, για να γίνει η επαλήθευση χωρίς κάποιο μήνυμα Error προχωρήσαμε στις εξής ρυθμίσεις του Electric:

- TOOL > NCC > ADD NCC ANNOTATION TO CELL > EXPORTS CONNECTED BY PARENT VDD
- TOOL > NCC > ADD NCC ANNOTATION TO CELL > EXPORTS CONNECTED BY PARENT GND

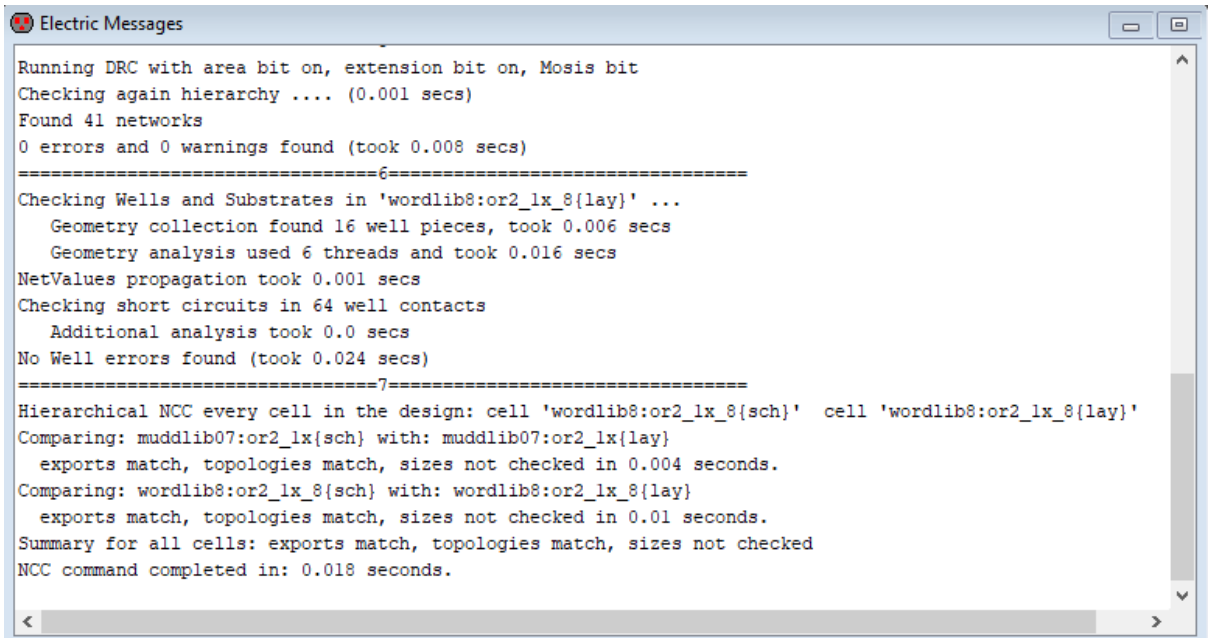
Παρατηρούμε όπως και στις παραπάνω εικόνες πως έχει εμφανιστεί πάνω στο σχέδιο μας η αντίστοιχη εντολή και πλέον είμαστε έτοιμοι να επαληθεύσουμε. Αφού δούμε τις επιτυχείς επαληθεύσεις στην επόμενη υποενότητα θα προχωρήσουμε στην ολοκλήρωση της ALU.

4.2.4 ΕΠΑΛΗΘΕΣΗ WORDSLICES

Στην αρχή αυτού του κεφαλαίου αναφερθήκαμε στην ολοκλήρωση ενός σημαντικού στοιχείου του επεξεργαστή, την ALU. Για να γίνει όμως αυτό, πρέπει πρώτα να έχει γίνει η επαλήθευση μέσω του Electric σε DRC, ERC και NCC. Στις παρακάτω εικόνες βλέπουμε την επαλήθευση αυτή.



Εικόνα 48

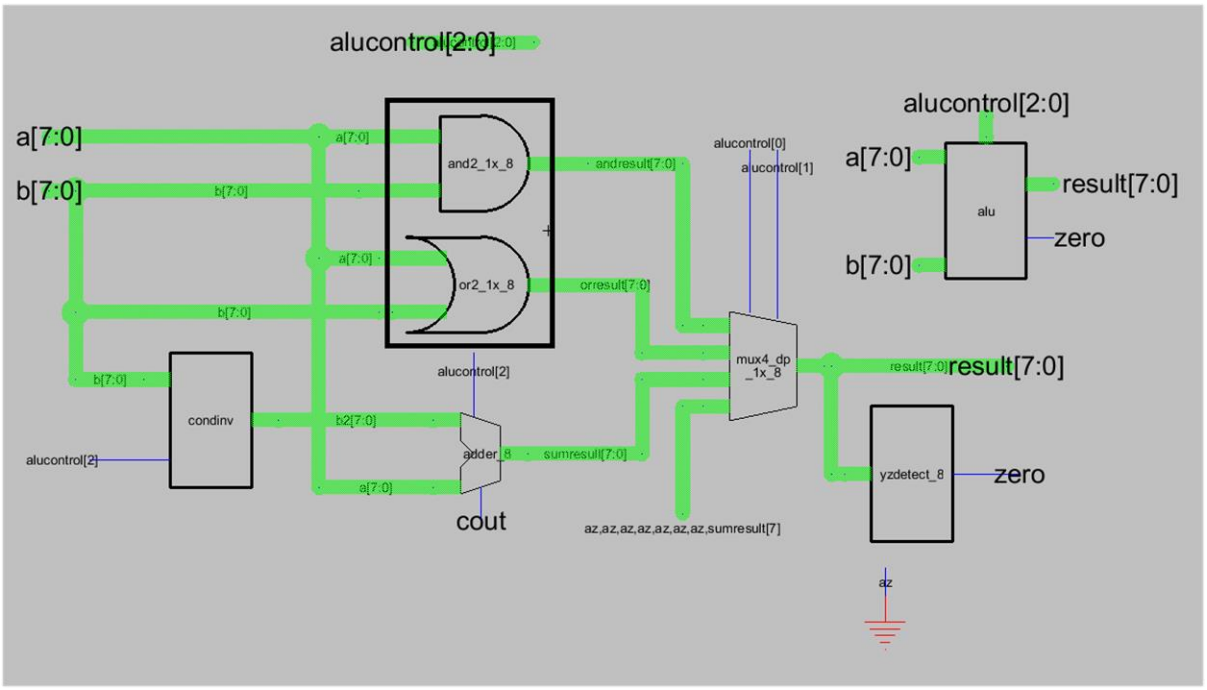


Εικόνα 49

Είδαμε λοιπόν πως τα σχέδια μας δεν εμφάνισαν κάποιο λάθος, οπότε συνεχίζουμε στο επόμενο βήμα που είναι η ολοκλήρωση της ALU και κατ' επέκταση η ολοκλήρωση του Datapath.

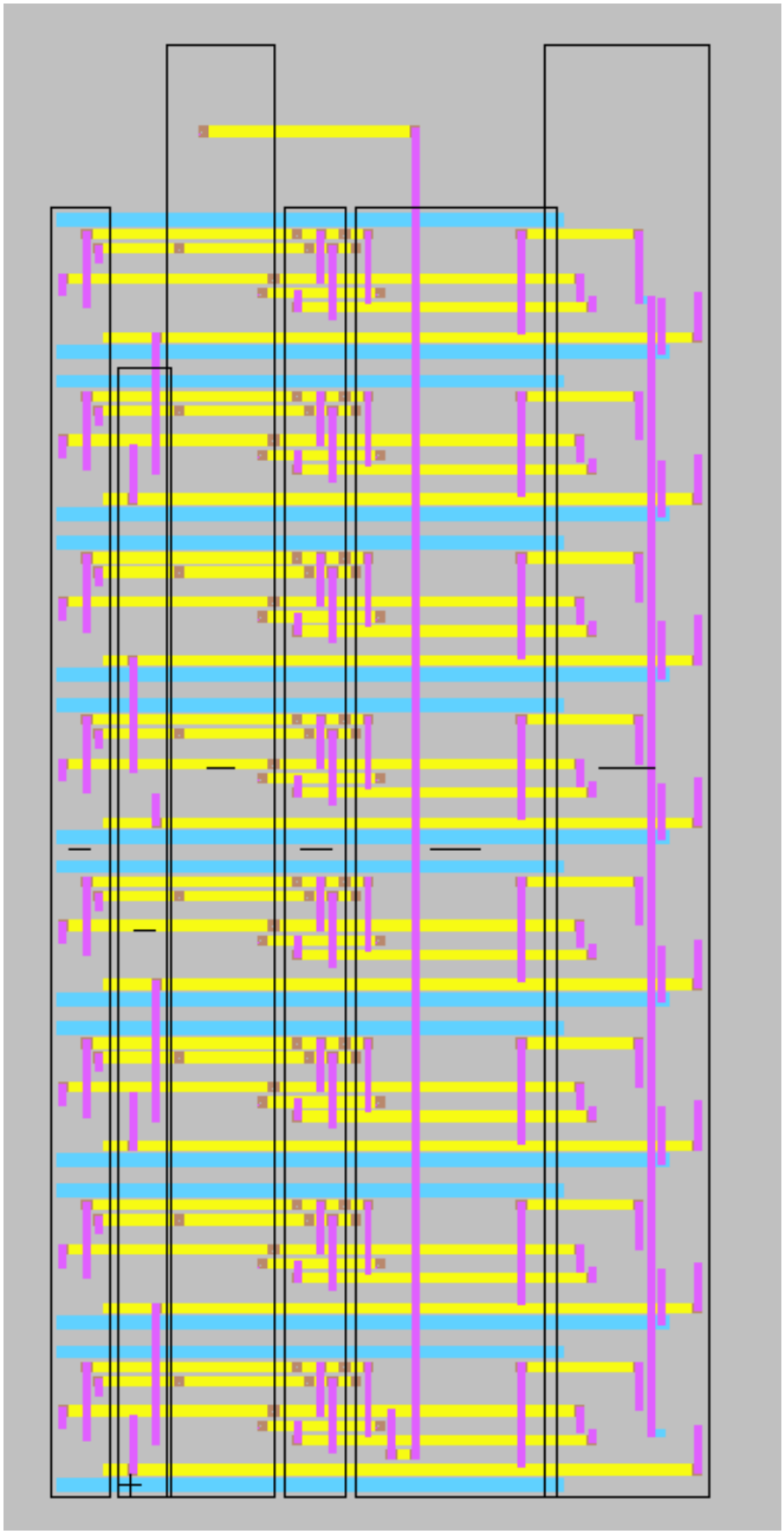
4.2.5 ΣΧΗΜΑΤΙΚΟ, LAYOUT ΚΑΙ ΕΠΑΛΗΘΕΥΣΗ ΤΗΣ ALU

Για την ολοκλήρωση και τη συνδεσμολογία της ALU θα χρησιμοποιήσουμε τα κυκλώματα που τελειοποιήσαμε και επαληθεύσαμε στις προηγούμενες υποενότητες. Ανοίγοντας την βιβλιοθήκη mips, θα δούμε όλα τα μέρη του επεξεργαστή. Αφού βρήκαμε την ALU, προσθέτουμε στο σχηματικό με τα busses την or2_1x_8 και and2_1x_8 στις κενές θέσεις του σχηματικού όπως ακριβώς απεικονίζεται στην παρακάτω εικόνα. Αξίζει να σημειωθεί πως η ALU εκτελεί όλες τις απαραίτητες λειτουργίες που απαιτούνται για την εφαρμογή του ζητούμενου σετ εντολών. [17]



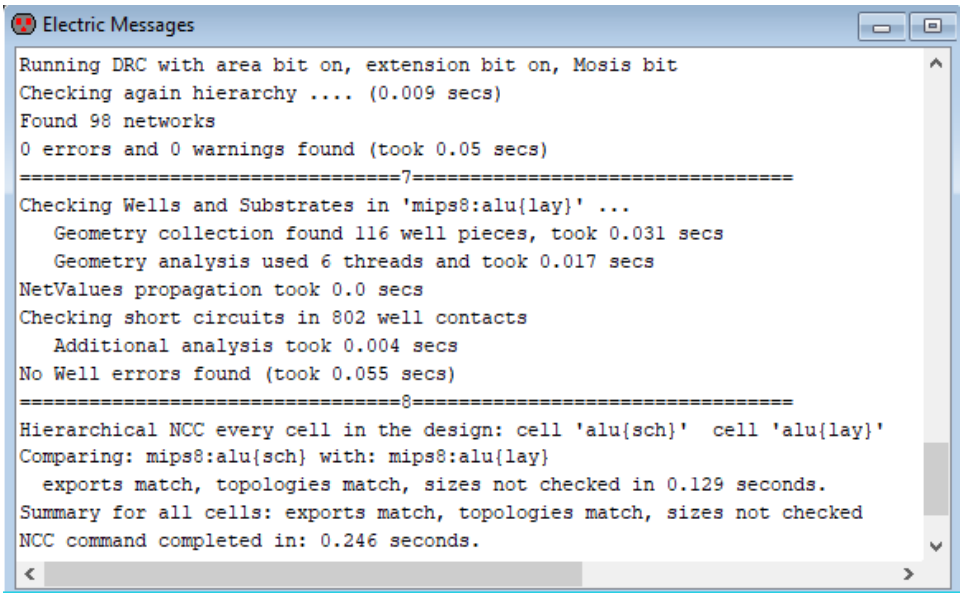
Εικόνα 50

Συνεχίζοντας, το layout είναι επίσης μία εύκολη διαδικασία που θέλει προσοχή για το πως θα συνδέσουμε την or2_1x_8 και and2_1x_8 στο layout, ώστε να μην υπάρχουν απρόσμενα λάθη. Στην παρακάτω εικόνα φαίνεται το layout που ολοκληρώσαμε βάζοντας τα στοιχεία που θέλαμε. Επειδή στην εικόνα δε διακρίνονται τα ονόματα των components της ALU, ξεκινώντας από τα αριστερά είναι τα εξής : and2_1x_8, yzdetect_8, condinv, or2_1x_8, adder_8 και mux4_1x_8.



Εικόνα 51

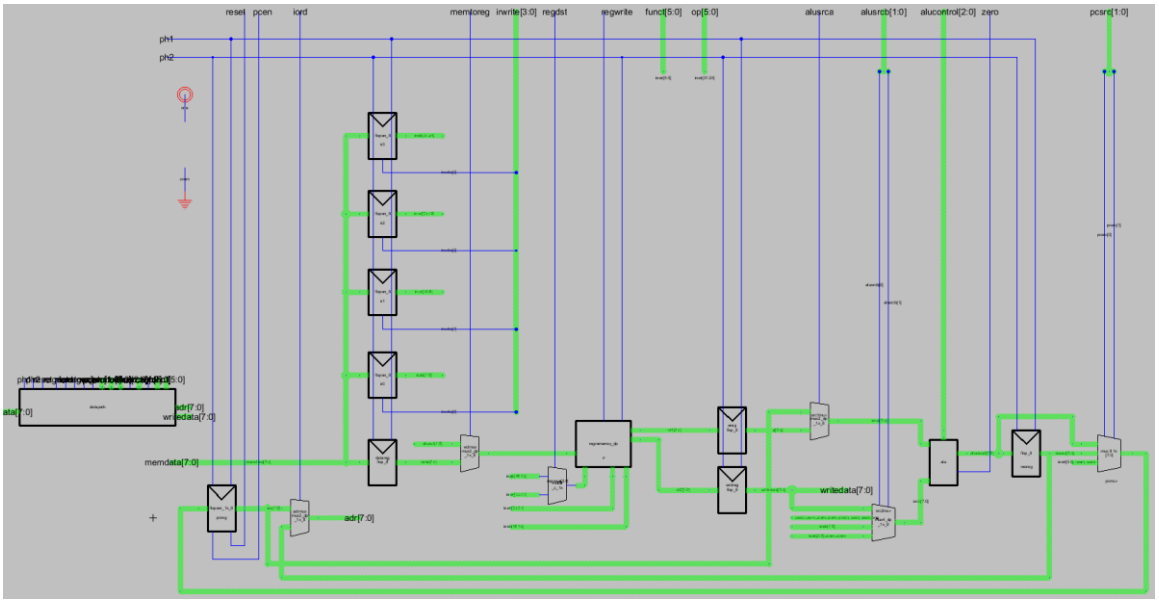
Τέλος, έχοντας ολοκληρώσει το σχέδιο μας, επαληθεύουμε για τυχόν λάθη, που όπως βλέπουμε στην παρακάτω εικόνα, δεν υπήρχαν, άρα μπορούμε να συνεχίσουμε βάζοντας την ALU στο Datapath για να ολοκληρώσουμε την 2η εργαστηριακή ενότητα.



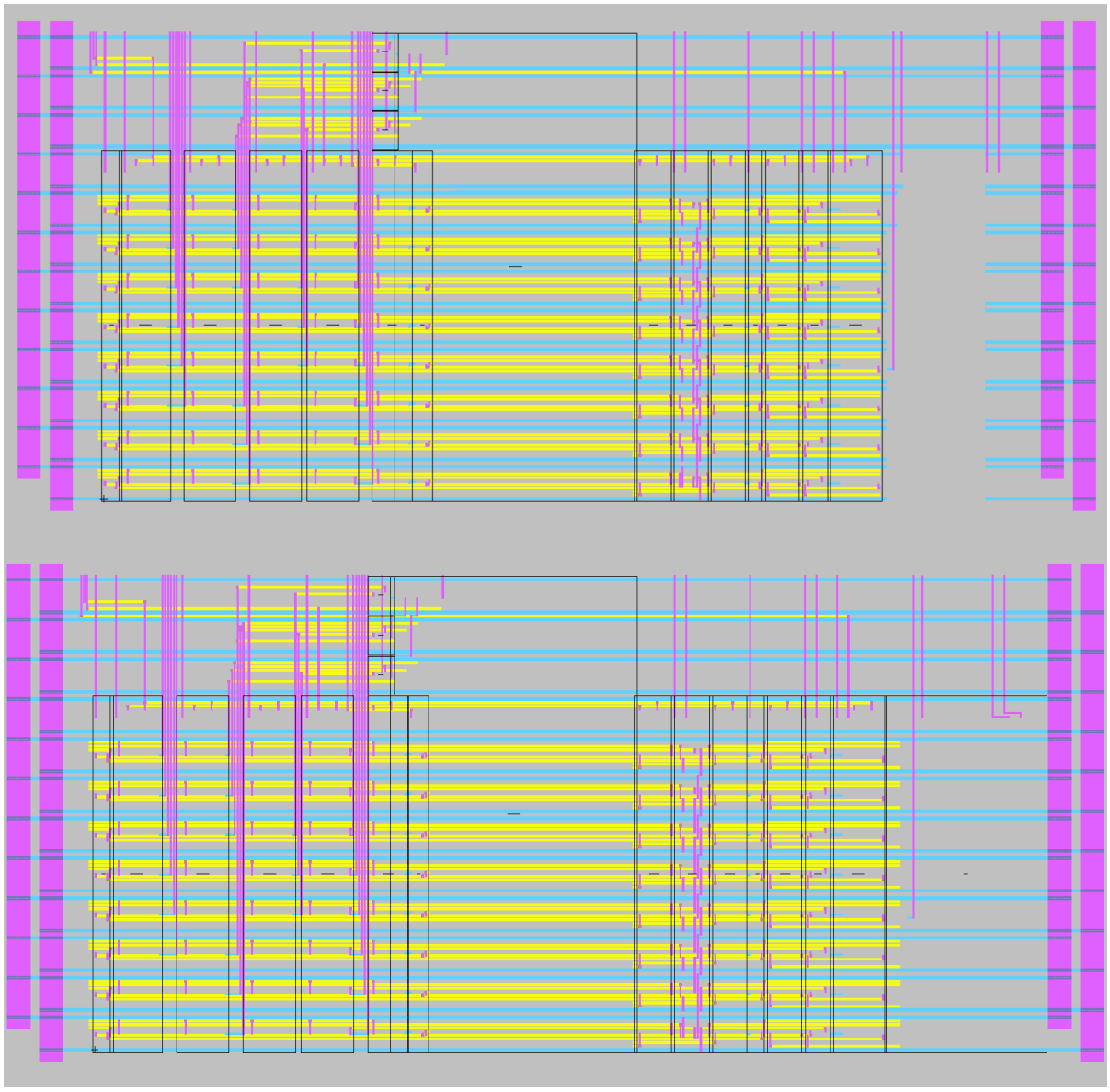
Εικόνα 52

4.2.6 ΣΧΗΜΑΤΙΚΟ ΚΑΙ LAYOUT ΤΟΥ DATAPATH

Έχοντας ολοκληρώσει και επαληθεύσει τους παραπάνω σχεδιασμούς, συνεχίσαμε παρακάτω ολοκληρώνοντας το Datapath. Το Datapath είναι ένα σύνολο λειτουργικών μονάδων που πραγματοποιούν εργασίες επεξεργασίας δεδομένων. Στις παρακάτω εικόνες βλέπουμε το σχέδιο που ολοκληρώσαμε τόσο σε σχηματικό όσο και σε Layout.



Εικόνα 53



Εικόνα 54

Λόγω του μεγάλου σχεδίου δεν διακρίνονται τα ονόματα των components. Επισημαίνεται, λοιπόν, πως η ALU στο σχηματικό μπήκε στο κενό που υπήρχε, ενώνοντας τους εισόδους και του εξόδους του. Στο Layout η ALU είναι το τελευταίο component δεξιά στο οποίο χρειάστηκε λίγη προσοχή γιατί όπως έχουμε προαναφέρει το Layout θέλει λεπτούς χειρισμούς για να μην υπάρχουν λάθη. Στην επόμενη ενότητα θα δούμε τα αποτελέσματα καθώς και την επαλήθευση του σχεδίου μας μέσω μίας εξωτερικής μονάδας μνήμης που θα φορτώνει ένα πρόγραμμα δοκιμής.

4.3 ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ 2ης ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ

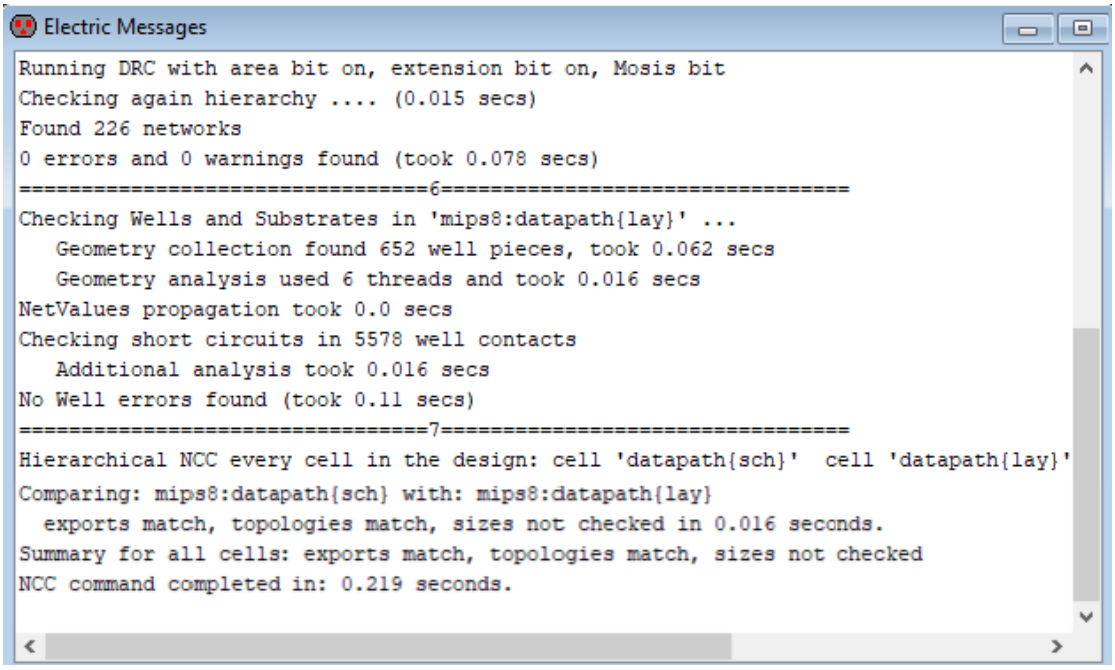
Στην αρχή αυτού του κεφαλαίου μιλήσαμε για τα ζητούμενα της δεύτερης εργαστηριακής ενότητας, των οποίων τα αποτελέσματά θα δούμε εδώ. Η ουσία αυτού του

κεφαλαίου είναι να καταλήξουμε στο component του επεξεργαστή, το Datapath. Όπως είπαμε, κάποια από τα σημαντικά μέρη του επεξεργαστή έχουν αφαιρεθεί για να μπορέσουμε να καταλάβουμε καλύτερα την διαδικασία της σχεδίασης.

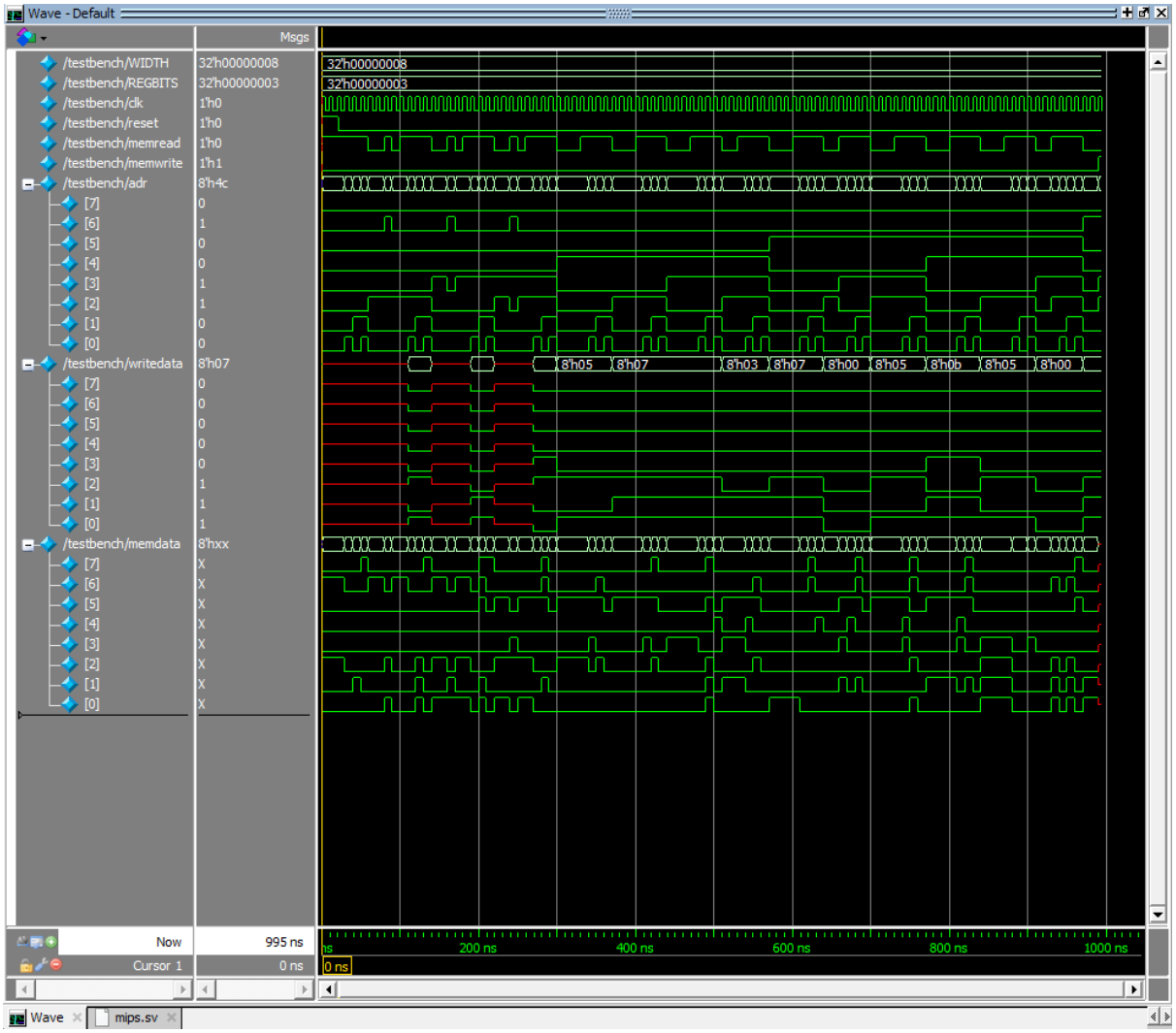
Τα ζητούμενα ήταν η δημιουργία 2 wordslices AND και OR, η τοποθέτησή τους μέσα στην ALU, η τοποθέτησή της ALU μέσα στο Datapath και τέλος η επαλήθευση του Datapath σε DRC, ERC και NCC καθώς και η προσομοίωσή του μέσω RTL μέσω του ModelSim. Στις παρακάτω εικόνες παρουσιάζονται όλα τα παραπάνω εκτός των AND, OR και ALU καθώς τα επαληθεύσαμε σε DRC,ERC και NCC στην προηγούμενη ενότητα διότι ήταν αναγκαίο για να ολοκληρώσουμε το Datapath.

Η εικόνα 53 μας δείχνει τα επιτυχή αποτελέσματα στο Electric, ενώ στην Εικόνα 54 και 55 βλέπουμε τα αποτελέσματα της προσομοίωσης στο ModelSim. Συγκεκριμένα στο Waveform, διακρίνεται στο /testbench/writedata ότι το 7 έχει καταχωρηθεί σωστά στην θέση μνήμης. Τέλος, εκτός από το Waveform στο παράθυρο Transcript (Εικόνα 55) εμφανίζεται και το μήνυμα “Simulation completely successful” που μας υποδεικνύει ότι η προσομοίωση ολοκληρώθηκε με επιτυχία.

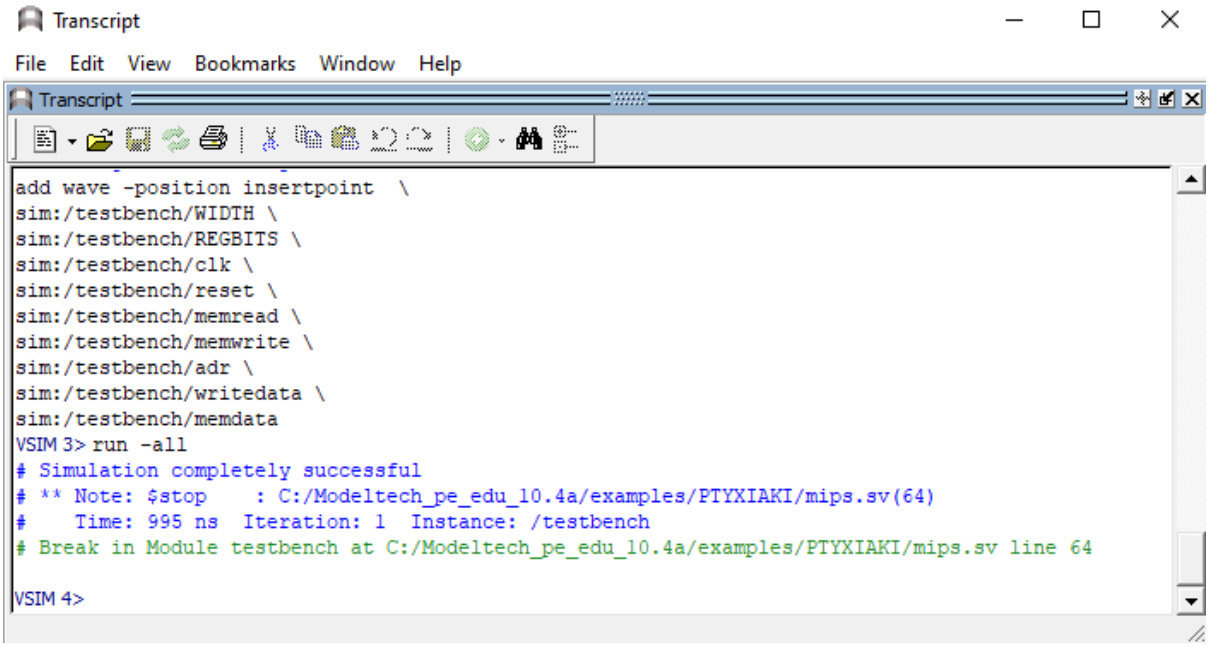
Κλείνοντας το κεφάλαιο αυτό γίνεται αντιληπτό ότι η διαδικασία σχεδίασης γίνεται όλο και πιο πολύπλοκη. Ξεκινώντας από τα Wordslices καταλήξαμε στο Datapath, το οποίο είναι ένα από τα κύρια μέρη του επεξεργαστή μας. Στο επόμενο κεφάλαιο είναι η 3η εργαστηριακή ενότητα κατά την οποία σχεδιάσαμε το ALUdec ώστε να καταλήξουμε στο δεύτερο σημαντικό μέρος του επεξεργαστή που είναι ο Controller.



Εικόνα 55



Εικόνα 56



Εικόνα 57

ΚΕΦΑΛΑΙΟ 5: ΤΡΙΤΗ ΕΡΓΑΣΤΗΡΙΑΚΗ ΕΝΟΤΗΤΑ

5.1 ΕΙΣΑΓΩΓΗ

Ολοκληρώνοντας τη δεύτερη εργαστηριακή ενότητα, έγινε αντιληπτό ένα σημαντικό κομμάτι σχεδίασης ψηφιακών κυκλωμάτων. Τελειοποιήσαμε τα wordslices και στη συνέχεια το Datapath χωρίς να υπάρξουν προβλήματα κατά την υλοποίηση τους. Σε αυτό το κεφάλαιο θα συνεχίσουμε στο άλλο κομμάτι του επεξεργαστή μας που, όπως έχει αναφερθεί, είναι ο Controller. [18]

Ο Controller (‘ελεγκτής’) του επεξεργαστή που δημιουργήσαμε είναι υπεύθυνος για τη δημιουργία σημάτων στο Datapath με τα οποία προσπελάζεται και εκτελείται η κάθε εντολή. Θα δούμε παρακάτω ότι δεν διαθέτει κανονική δομή όπως το Datapath. Αρχικά, σχεδιάσαμε το ALU dec (decoder), βλέποντας ότι ακόμα και ένα μικρό σχέδια είναι επιρρεπές σε σφάλματα που οδηγούν σε λάθος αποτελέσματα ακόμα και στα υψηλότερα επίπεδα σχεδίασης όπως είναι ο Controller. [18]

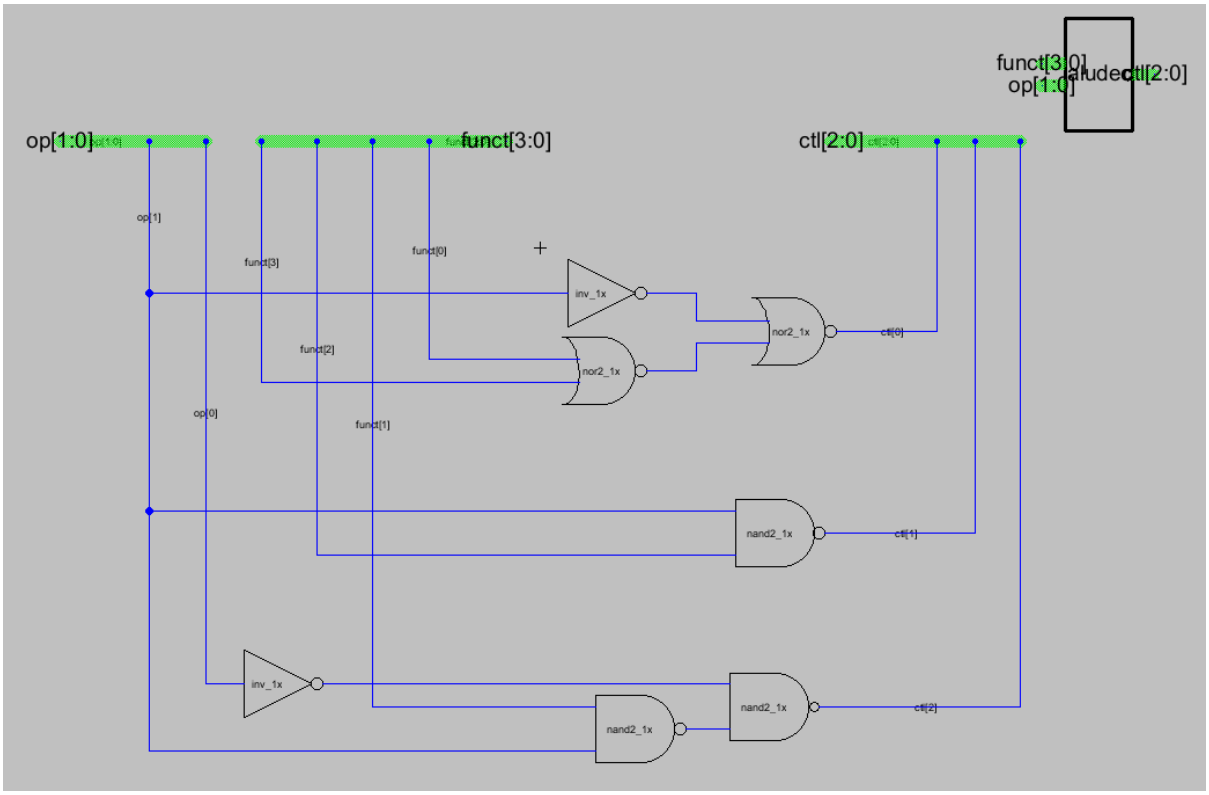
Επίσης, για την ολοκλήρωση του Controller, θα εισάγουμε μέσα στη Verilog του αρχικού Controller τις δύο εντολές που λείπουν διαβάζοντας το FSM του controller το οποίο θα το δούμε παρακάτω. Τέλος, για να γλυτώσουμε μία χρονοβόρα διαδικασία σχεδίασης θα χρησιμοποιήσουμε τον PLA Generator που αναφέραμε σε προηγούμενο κεφάλαιο μαζί με το ALUdec για την ολοκλήρωση, την επαλήθευση αλλά και την προσομοίωση του Controller.

5.2 ΥΛΟΠΟΙΗΣΗ ΤΡΙΤΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ

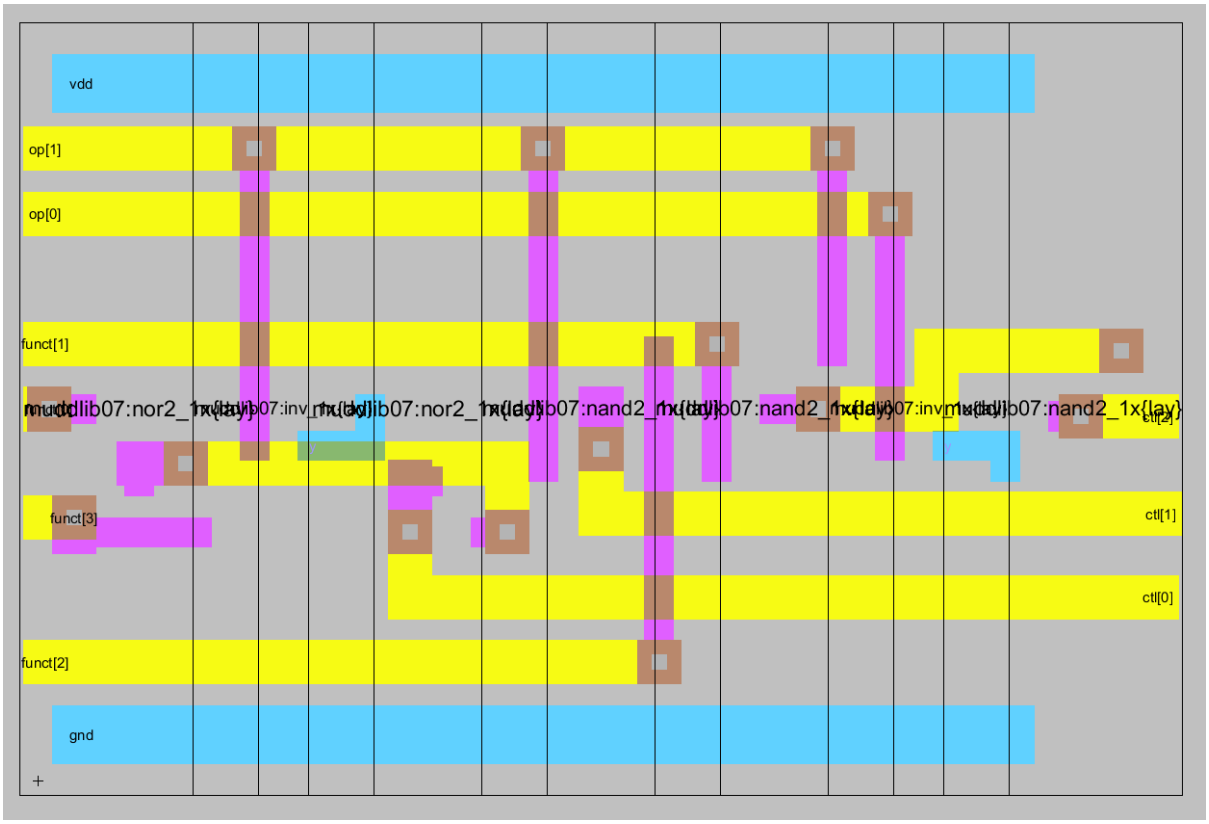
5.2.1 ALUDEC

Η δημιουργία του component ALUdec έγινε βλέποντας τον κώδικα Verilog που δόθηκε, με τον οποίο έγιναν αντιληπτά τα σήματα τα οποία πρέπει το κύκλωμά μας να παράγει προκειμένου να κάνει την δουλειά του. Η λογική του ALUdec, είναι υπεύθυνη για την αποκωδικοποίηση ενός σήματος ALUOp 2-bit και ενός ‘Function’ 6-bit της εντολής για την παραγωγή τριών γραμμών ελέγχου πολυπλέκτη για το ALU. Δύο από τα σήματα επιλέγουν τον τύπο λειτουργίας της ALU και το τρίτο καθορίζει εάν η είσοδος B είναι ανεστραμμένη. [18]

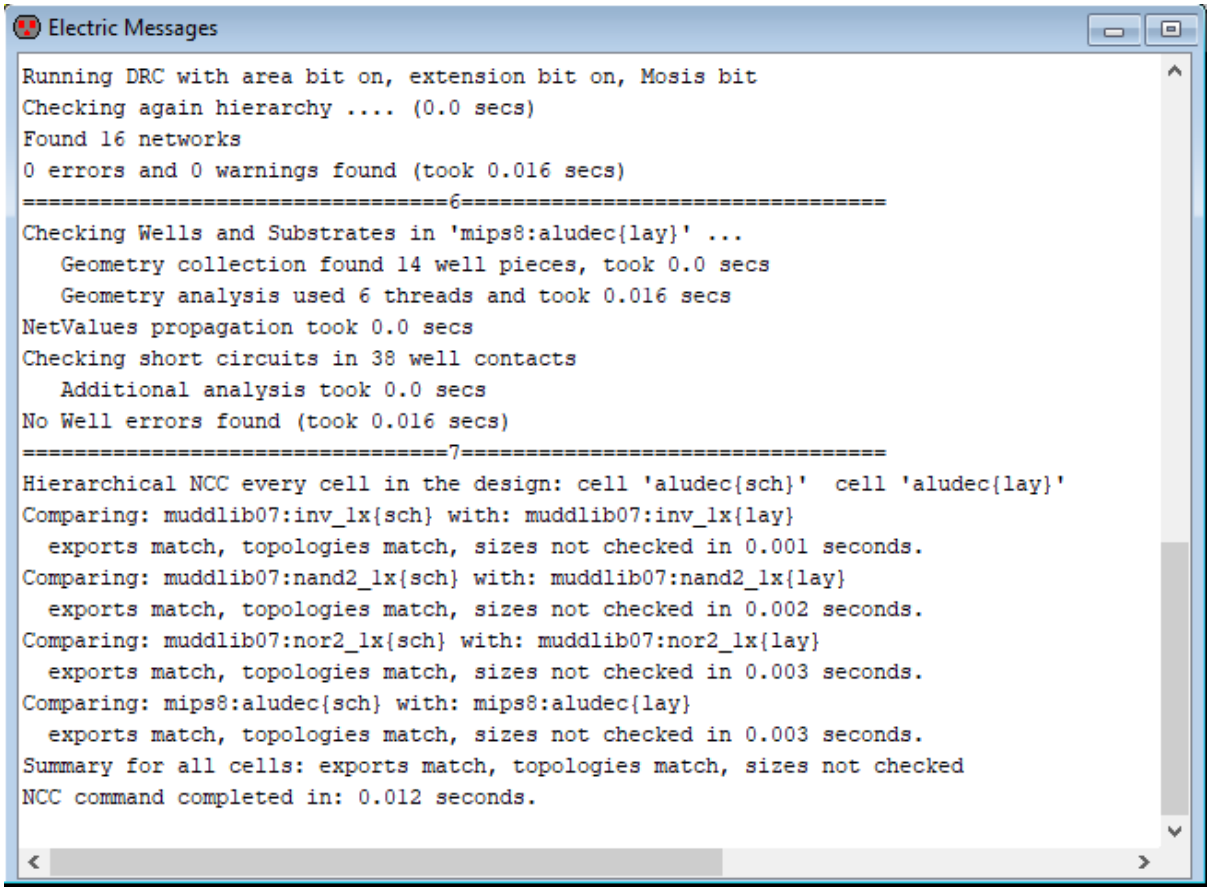
Στις παρακάτω εικόνες θα δούμε την υλοποίηση του ALUdec χρησιμοποιώντας τα components του muddlib07, της ίδιας βιβλιοθήκης που χρησιμοποιήσαμε και στο προηγούμενο κεφάλαιο. Χρησιμοποιήσαμε τα συγκεκριμένα components για να σχεδιάσουμε ένα κύκλωμα το οποίο, όπως αναφέραμε, παράγει σήματα ALUControl [2:0] από σήματα που δίνουμε από ALUOp[1:0] και Funct [5:0]. Να σημειωθεί εδώ πως για να γίνει αυτό δεν υπολογίσαμε 2 σήματα τα οποία είναι ‘don’t cares’ οπότε μπορούν να παραχθούν και μόνο με τέσσερα, δηλαδή, Funct [3:0]. [12][18]



Εικόνα 58



Εικόνα 59



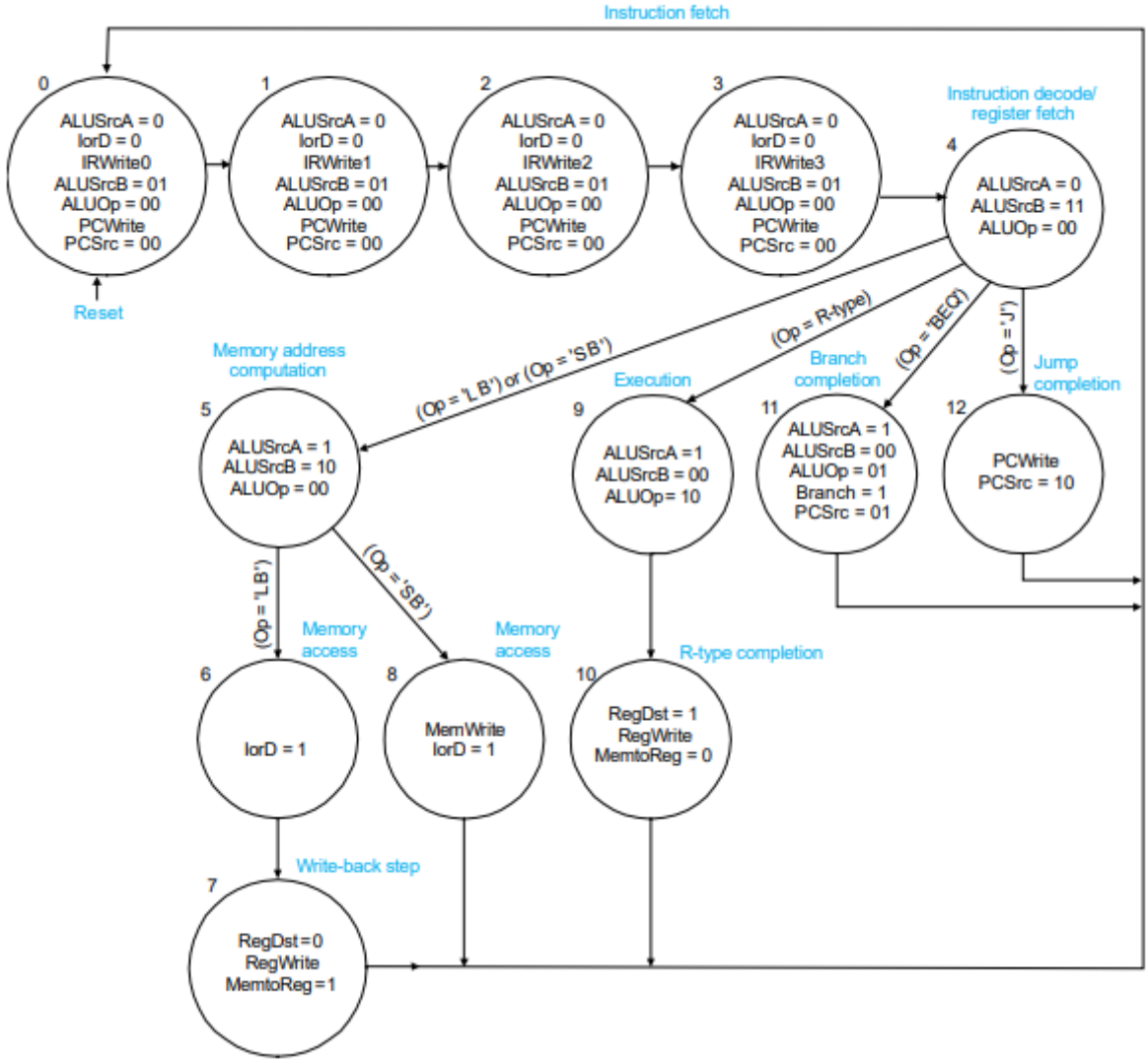
Εικόνα 60

Προς το παρόν το σχέδιο, δεν εμφανίζει κάποιο λάθος όπως είδαμε και από την εικόνα 60. Θα δούμε για τυχόν λάθη του σχεδίου ως προς την προσομοίωσή του στη συνέχεια, όταν θα έχει εισαχθεί στο τελικό component που θα ολοκληρώσουμε σε αυτό το κεφάλαιο, το Controller.

5.2.2 CONTROLLER PLA

Εν συνεχεία της υλοποίησης της 3ης εργαστηριακής ενότητας, χρησιμοποιήσαμε το PLA (Programmable Logic Array) Generator για να συνθέσουμε ένα σχηματικό και layout του Controller.

Με τον όρο σύνθεση εννοούμε την τεχνική σχεδιασμού κυκλώματος ημιαγωγών που μεταφράζει περιγραφές υψηλού επιπέδου απευθείας σε χαρακτηριστικά ‘hardware’ όπως πύλες AND και OR. Γενικά, αντιπροσωπεύει μεγάλο μέρος της σχεδίασης κυκλώματος στους σύγχρονους μικροεπεξεργαστές. Για την ολοκλήρωση, χρησιμοποιήσαμε τον κώδικα Verilog ο οποίος αντιπροσωπεύεται από την παρακάτω εικόνα, η οποία παρουσιάζει μία μηχανή πεπερασμένων καταστάσεων δείχνοντας μας, πως ο Controller προσπελαύνει μία εντολή. Ο κώδικας μας δείχνει τα inputs και τα outputs για την κάθε συνθήκη του controller PLA.[21][18]

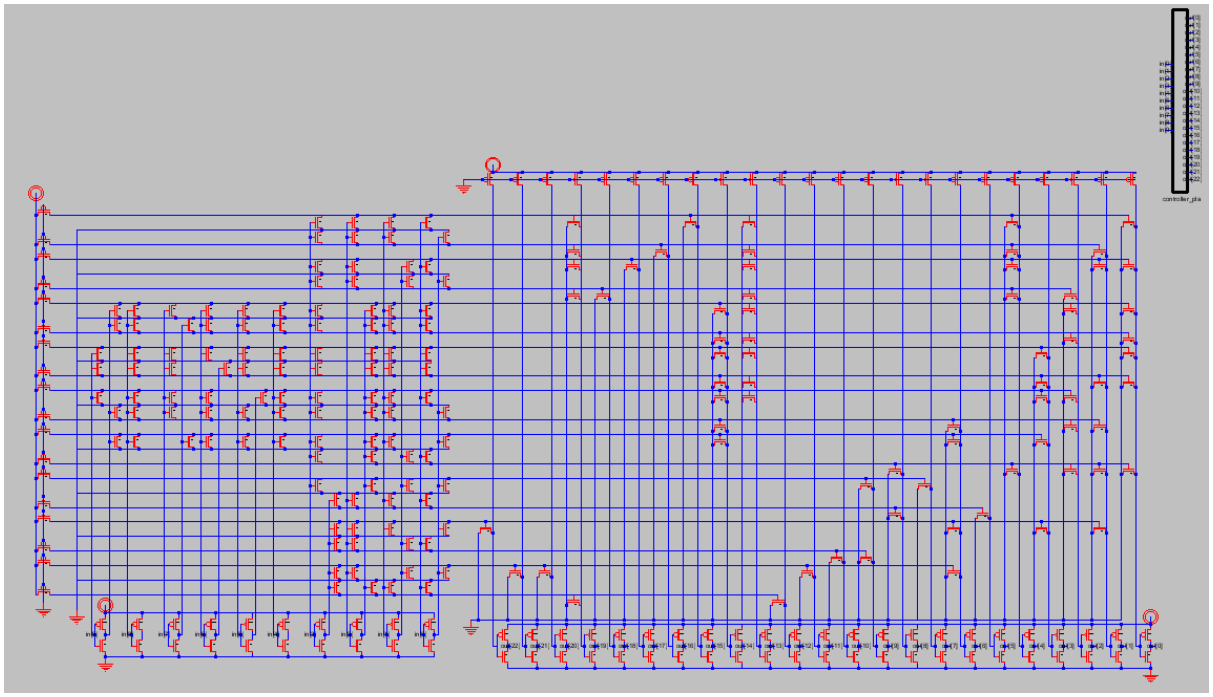


Εικόνα 61

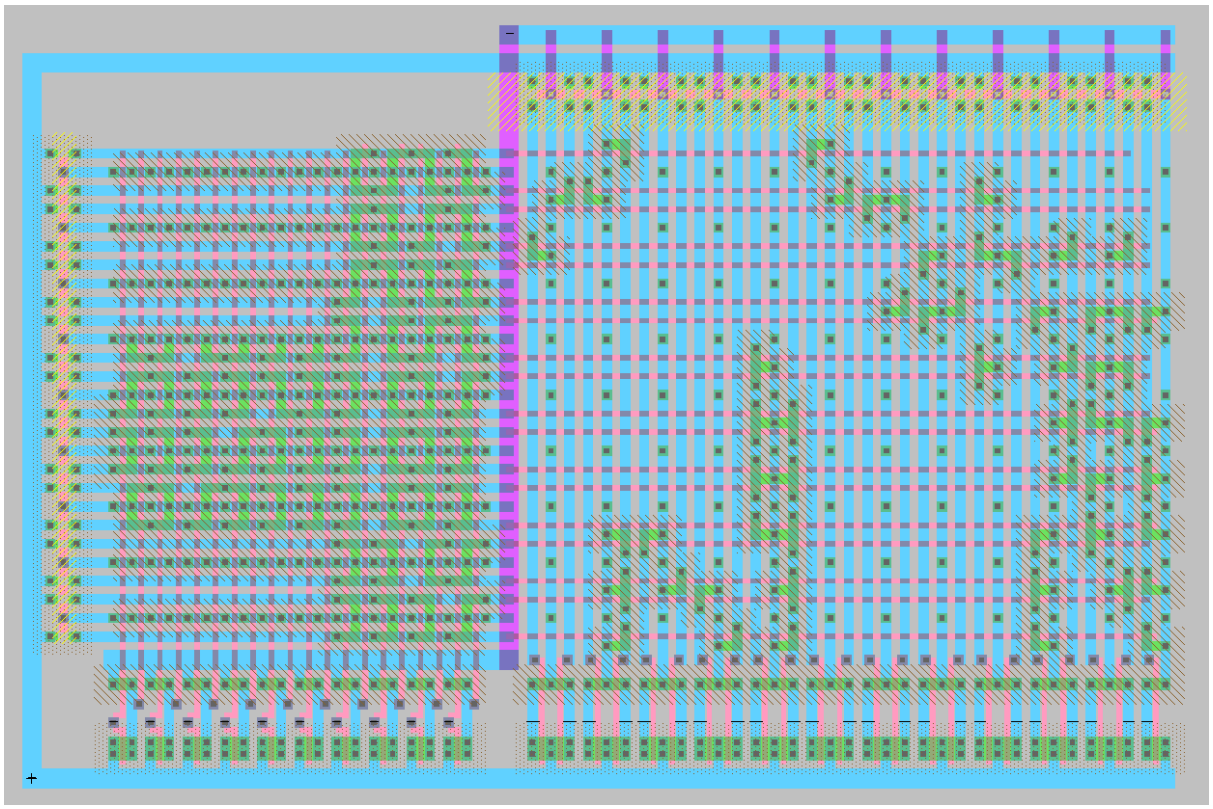
Ο παρακάτω κώδικας είναι αυτός που χρησιμοποιήθηκε στον PLA Generator προκειμένου να έχουμε τα αποτελέσματα που θέλαμε. Κατά την εκφώνηση επισημαίνεται πως τα δύο τελευταία case του κώδικα λείπουν και ότι έπρεπε να συμπληρωθούν από εμάς. Όπως βλέπουμε, οι εντολές που έλειπαν συμπληρώθηκαν, κοιτώντας πάντα τη λογική, που έχει η μηχανή πεπερασμένων καταστάσεων που έχουμε στην παραπάνω εικόνα.

```
// insert lines here for states 11 and 12 of PLA
10'b0001001011: out <= 23'b01100000000100001000000;//beq
10'b0000101100: out <= 23'b00010000001000000000000;//jmp
```

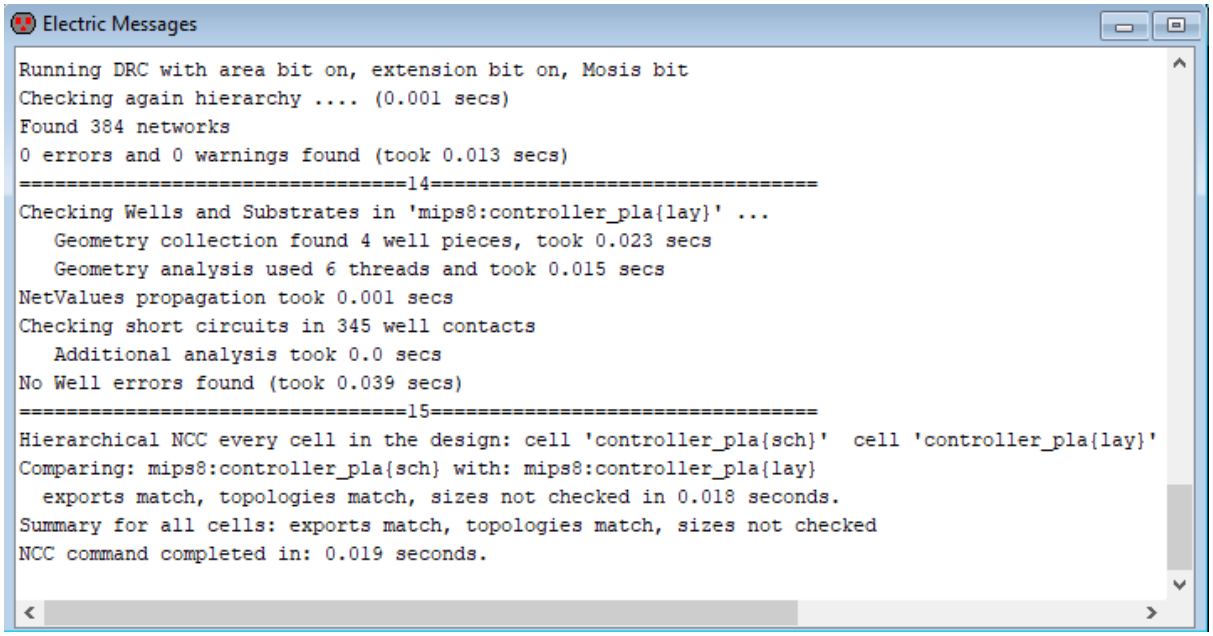
Έχοντας εισάγει τις δύο γραμμές στο αρχείο Verilog, τρέχουμε το πρόγραμμα PLA Generator που δίνεται και εκείνο δημιουργεί το Σχηματικό το icon και το layout του Controller_PLA. Στις παρακάτω εικόνες φαίνονται τα αποτελέσματα του PLA Generator μέσω του Electric.



Εικόνα 62

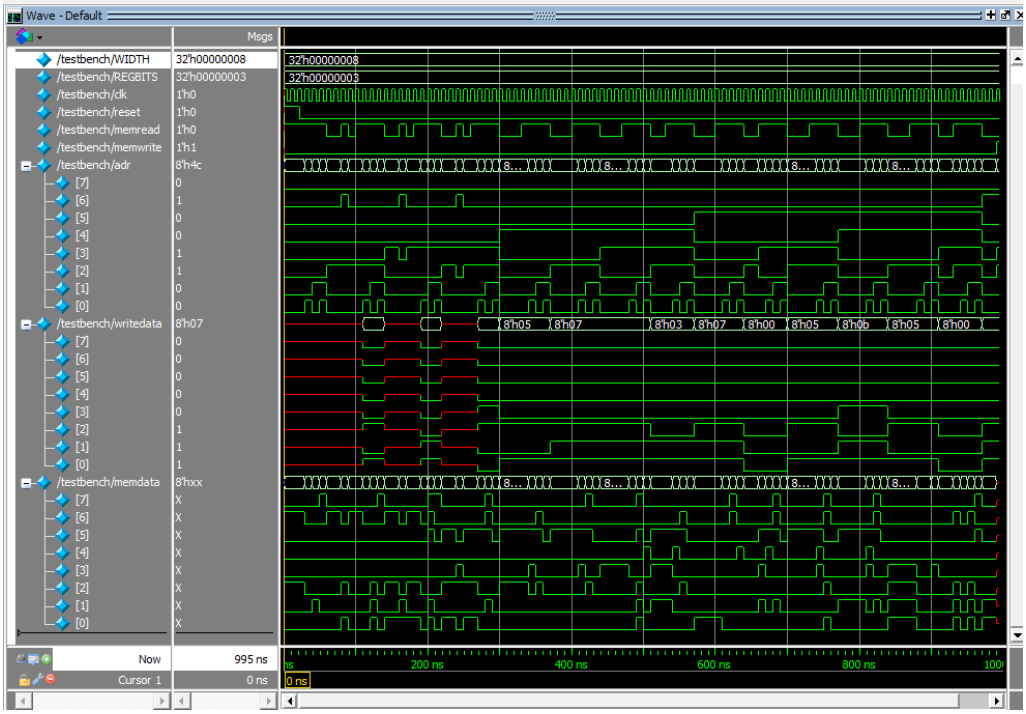


Εικόνα 63



Εικόνα 64

Πριν συνεχίσουμε όμως με το Controller πραγματοποιήσαμε και μία προσομοίωση του controller_PLA προκειμένου να επαληθεύσουμε αν το σχέδιό μας δουλεύει σωστά. Στην παρακάτω εικόνα βλέπουμε το waveform του ModelSim



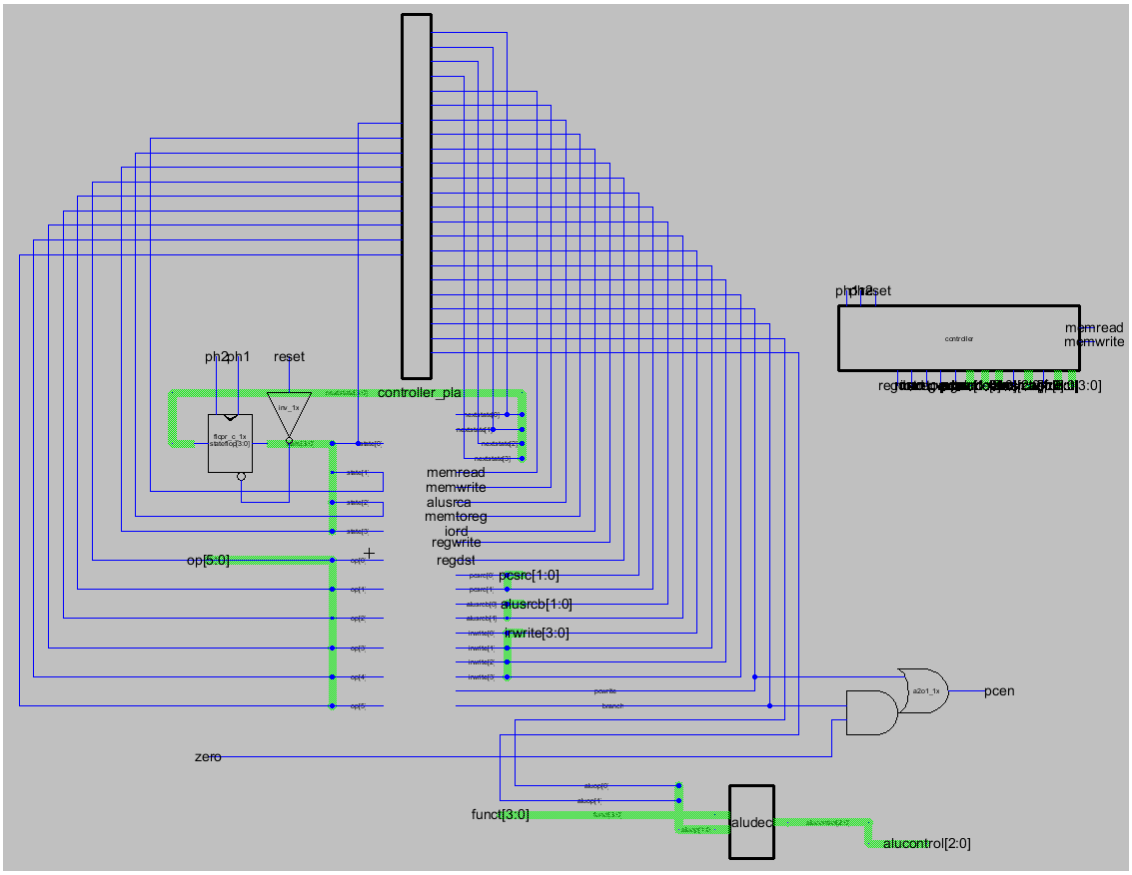
Εικόνα 65

Ολοκληρώνοντας, τον Controller_PLA χωρίς λάθη, συνεχίζουμε με το επόμενο στάδιο της εργαστηριακής ενότητας. Θα εισάγουμε το component στον Controller μαζί με το ALUdec προκειμένου να ολοκληρώσουμε τον Controller, να τον επαληθεύσουμε και τέλος να τον προσομοιώσουμε.

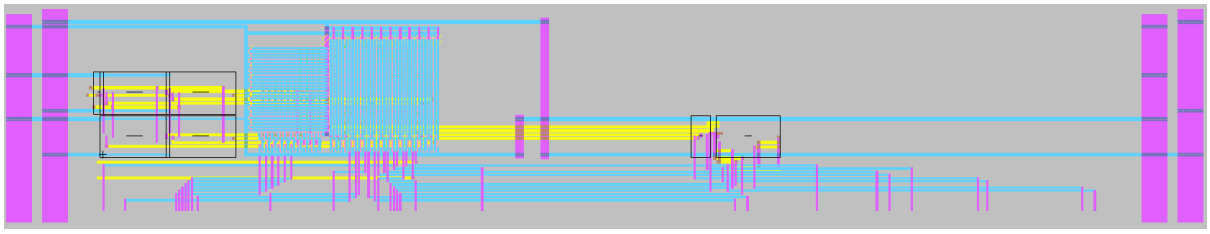
5.2.3 CONTROLLER

Έχοντας υλοποιήσει το ALUdec και τον Controller_PLA, έχουμε φτάσει στο τελικό στάδιο της 3ης εργαστηριακής ενότητας, στο οποίο θα εισάγουμε τα components στο Controller τόσο σε Σχηματικό, όσο και Layout.

Στις παρακάτω εικόνες, βλέπουμε το Σχηματικό (Εικόνα 66) και το Layout (Εικόνα 67). Στο σχηματικό, ο Controller_PLA βρίσκεται στο επάνω μέρος, συνδεδεμένο με όλες τις εισόδους και τις εξόδους και στο κάτω μέρος, διακρίνεται το component ALUdec συνδεδεμένο και αυτό με το υπόλοιπο σχέδιο. Το layout του controller, στην συγκεκριμένη εικόνα δε διακρίνεται καθαρά, αλλά διακρίνονται τα μέρη που μας ενδιαφέρουν που είναι το Controller_PLA, στα αριστερά, και το ALUdec δεξιά, το μεγάλο παραλληλόγραμμο. Το σχέδιο ήθελε ιδιαίτερη προσοχή καθώς είναι εύκολο στο Layout εύκολα να μπερδευτεί κανείς και οι είσοδοι και έξοδοι να μπουν αλλού πάνω στο σχήμα με αποτέλεσμα να εμφανιστούν λάθη.[12]



Εικόνα 66

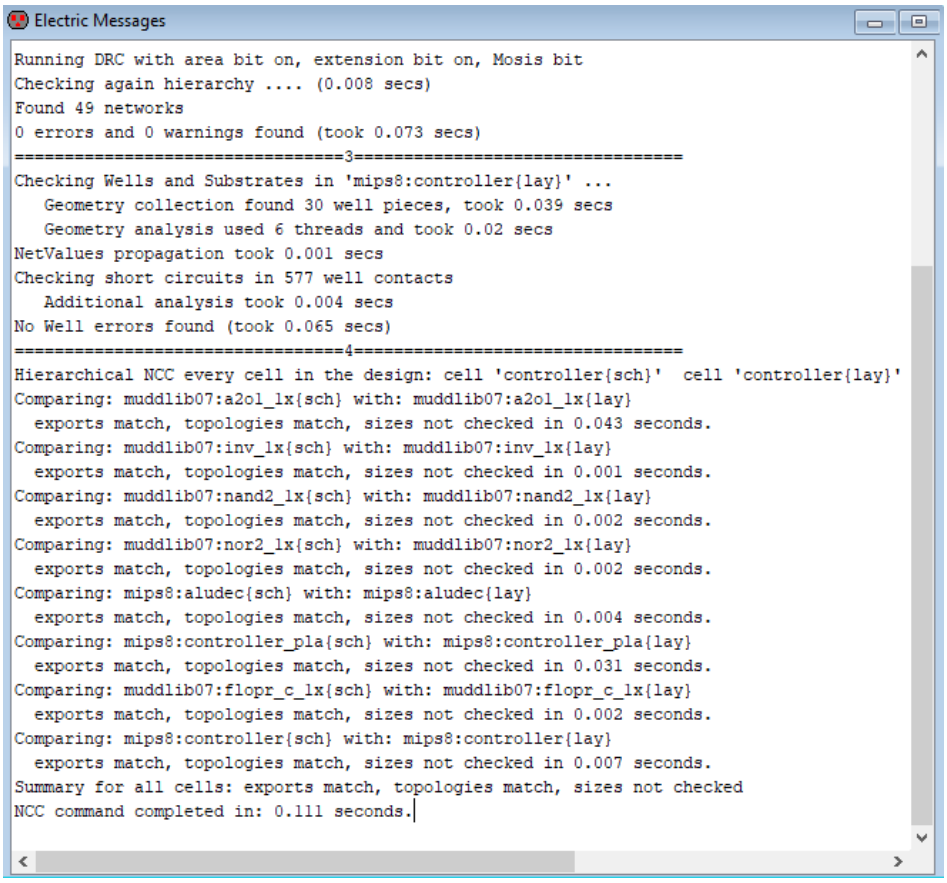


Εικόνα 67

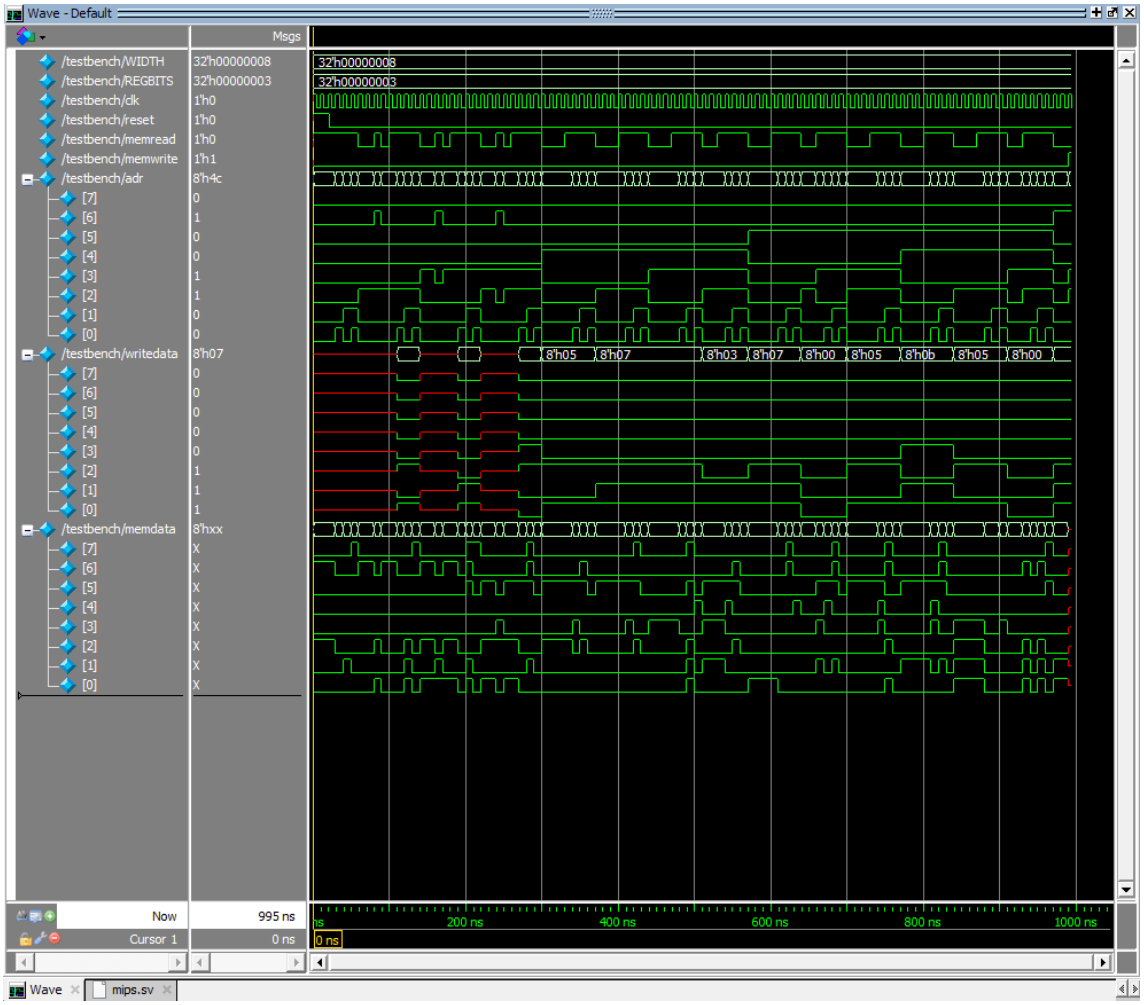
5.3 ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ 3ης ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ

Ολοκληρώνοντας και το τελευταίο κομμάτι του Controller, δηλαδή, τον ίδιο τον Controller μπορούμε να συνεχίσουμε βλέποντας τα αποτελέσματα από τα σχέδια μας κάνοντας την επαλήθευση στο Electric και προσομοίωση στο ModelSim.

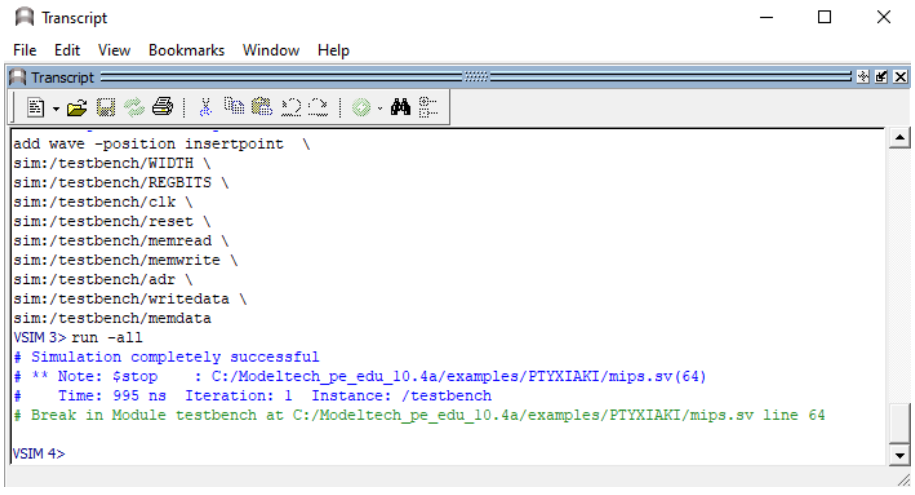
Στην Εικόνα 68, δίνονται τα αποτελέσματα του DRC, ERC και NCC για τον controller και στην εικόνα 69 και 70 βλέπουμε τα αποτελέσματα του ModelSim, στα οποία μπορούμε να δούμε ότι το σχέδιο μας είναι σωστό πραγματοποιώντας την επαλήθευση, γράφοντας στη θέση μνήμης τον αριθμό 7.



Εικόνα 68



Εικόνα 69



Εικόνα 70

Έχοντας τα αποτελέσματα μας σωστά και επαληθευμένα, μπορούμε να συνεχίσουμε στην επόμενη εργαστηριακή ενότητα, που είναι και η καταλυτική διότι ολοκληρώνουμε το σύστημά μας συνδέοντας το Controller και το Datapath για να ολοκληρωθεί ο επεξεργαστής μας και να περάσουμε στις τελευταίες επαληθεύσεις και προσομοιώσεις.

ΚΕΦΑΛΑΙΟ 6: ΤΕΤΑΡΤΗ ΕΡΓΑΣΤΗΡΙΑΚΗ ΕΝΟΤΗΤΑ

6.1 ΕΙΣΑΓΩΓΗ

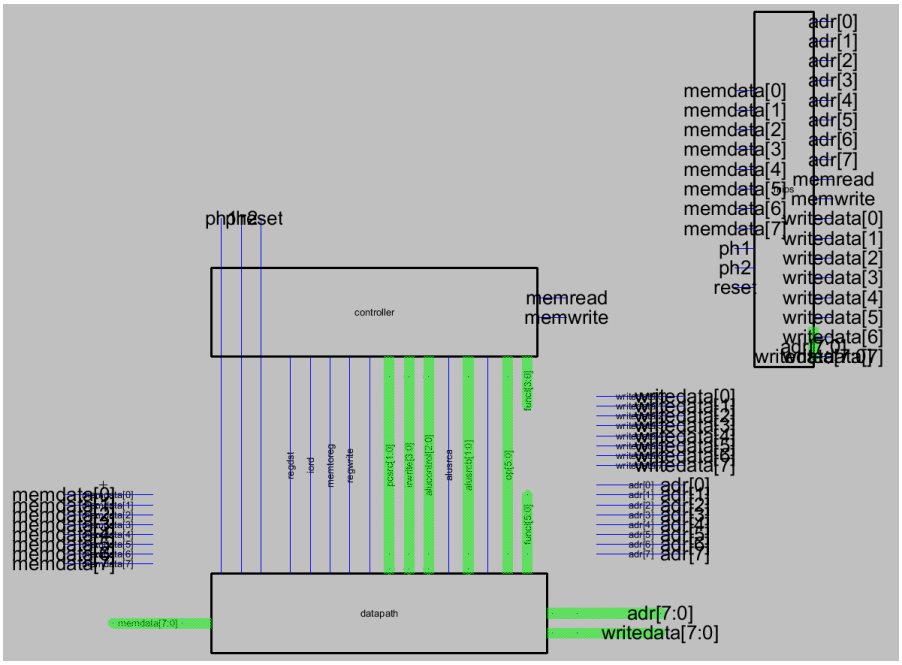
Σε αυτή την τελευταία εργαστηριακή ενότητα συνδέσαμε το Controller με το Datapath και προσομοιώσαμε ολόκληρο τον επεξεργαστή. Έπειτα, μέσω του Electric χρησιμοποιήσαμε το Pad Frame Generator προκειμένου να είμαστε σε θέση να συνδέσουμε τον επεξεργαστή με τον εξωτερικό κόσμο και να βεβαιωθούμε ότι το σύστημα μας είναι έτοιμο.[19]

Τελειώνοντας αυτή την ενότητα και έχοντας ολοκληρώσει τον επεξεργαστή μας, θα κάνουμε το λεγόμενο ‘tapeout’, το οποίο είναι τα αποτελέσματα της διαδικασίας σχεδίασης για το σύστημά μας δημιουργώντας ένα αρχείο που περιέχει τις γεωμετρίες και τις πληροφορίες που χρειάζεται ο κατασκευαστής για να φτιάξει τον επεξεργαστή.[19]

6.2 ΥΛΟΠΟΙΗΣΗ ΤΕΤΑΡΤΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ

6.2.1 ΣΧΗΜΑΤΙΚΟ ΚΑΙ LAYOUT MIPS

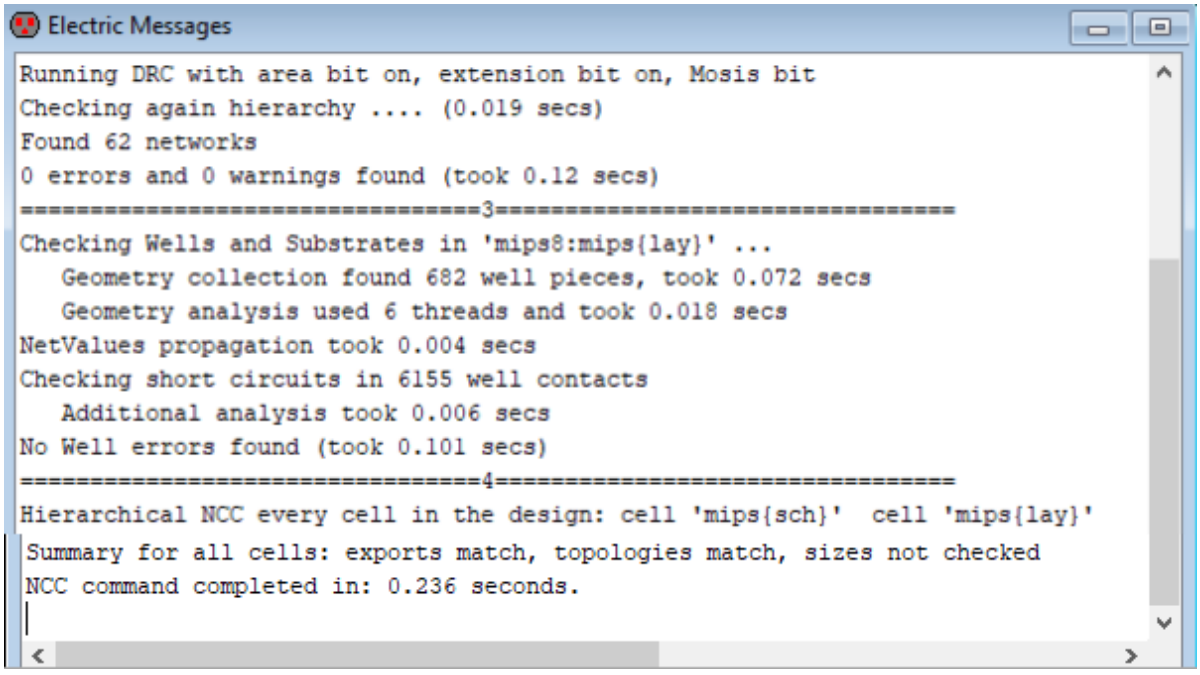
Ξεκινώντας, τελειοποιήσαμε τον επεξεργαστή μας φτιάχνοντας το σχηματικό και το layout της κορυφαίας σχεδίασης, ενώνοντας το Controller με το Datapath. Η διαδικασία είναι αρκετά εύκολη αφού έχουμε βεβαιωθεί ότι τα wires και των 2 είναι σε τέτοια θέση ώστε να κουμπώνει το ένα πάνω στο άλλο. Στις παρακάτω εικόνες βλέπουμε τα σχέδια τόσο σε σχηματικό όσο και σε layout. Στο layout (Εικόνα 72), στο επάνω μέρος είναι ο controller και στο κάτω μέρος το Datapath. Πρέπει πάντα να βεβαιωνόμαστε για τυχόν exports που έχουν ξεφύγει, διότι, προκύπτουν λάθη κατά την επαλήθευση μας σε DRC, ERC και NCC.[12]



Εικόνα 71

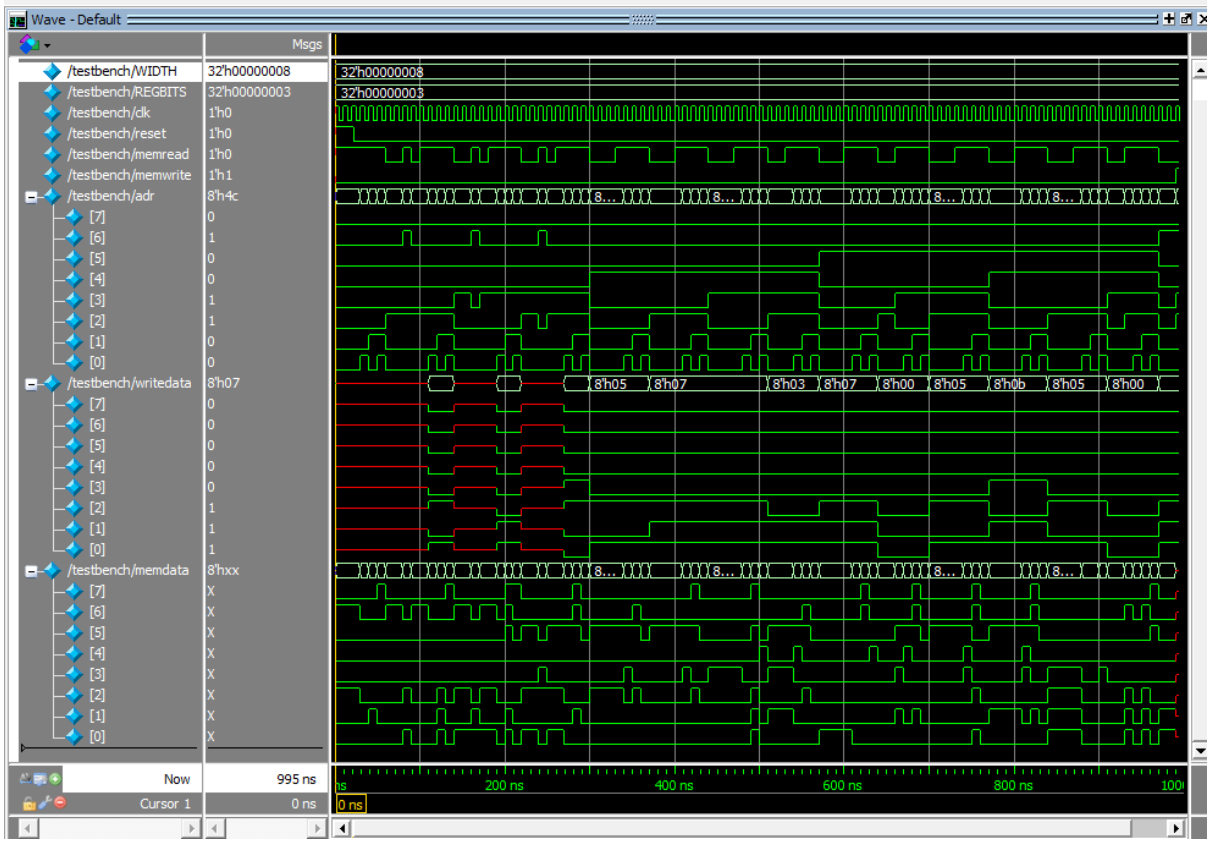
Εικόνα 72

Συνεχίζοντας, πραγματοποιήθηκαν οι επαληθεύσεις για το σχέδιο μας και μπορούμε να δούμε τα αποτελέσματα στην παρακάτω εικόνα, για να μπορέσουμε να συνεχίσουμε με την προσομοίωση του συστήματός μας στο ModelSim για να δούμε ότι ο επεξεργαστής είναι έτοιμος για να μπορέσουμε να συνεχίσουμε στο Pad Frame.

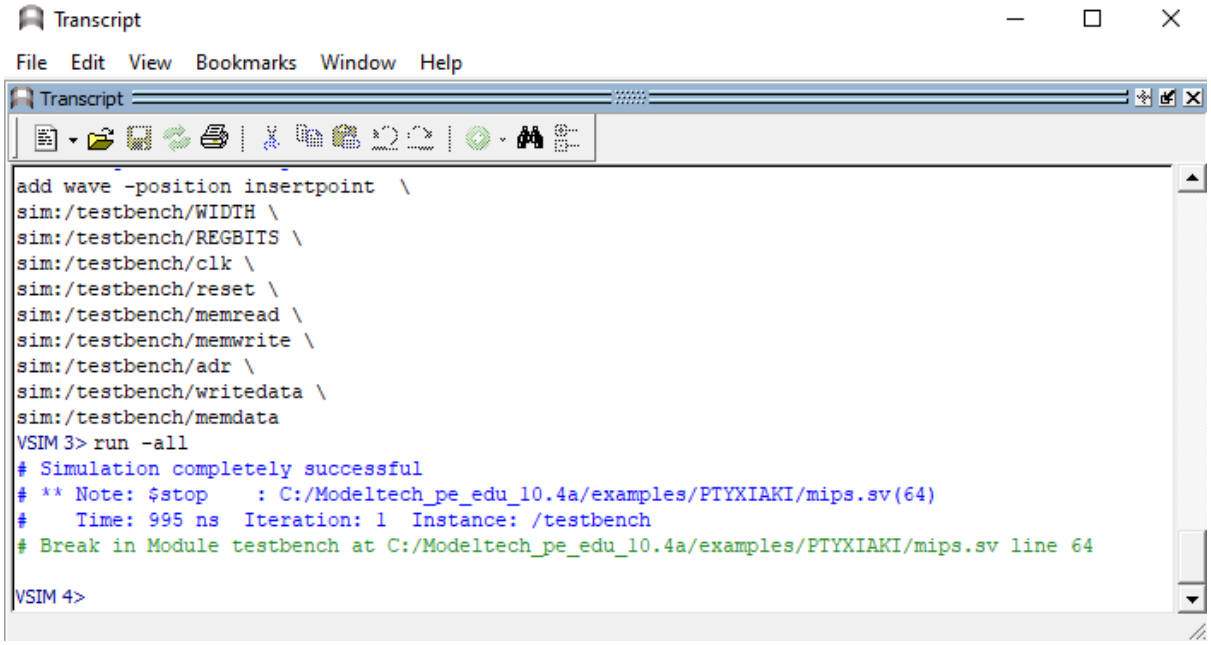


Εικόνα 73

Στις παρακάτω εικόνες βλέπουμε ένα γνώριμο σκηνικό στο ModeSim, με το σύστημα να κάνει την προσομοίωση που του ζητήθηκε με επιτυχία γράφοντας τον αριθμό 7 στη θέση μνήμης που του ζητήθηκε.



Εικόνα 74



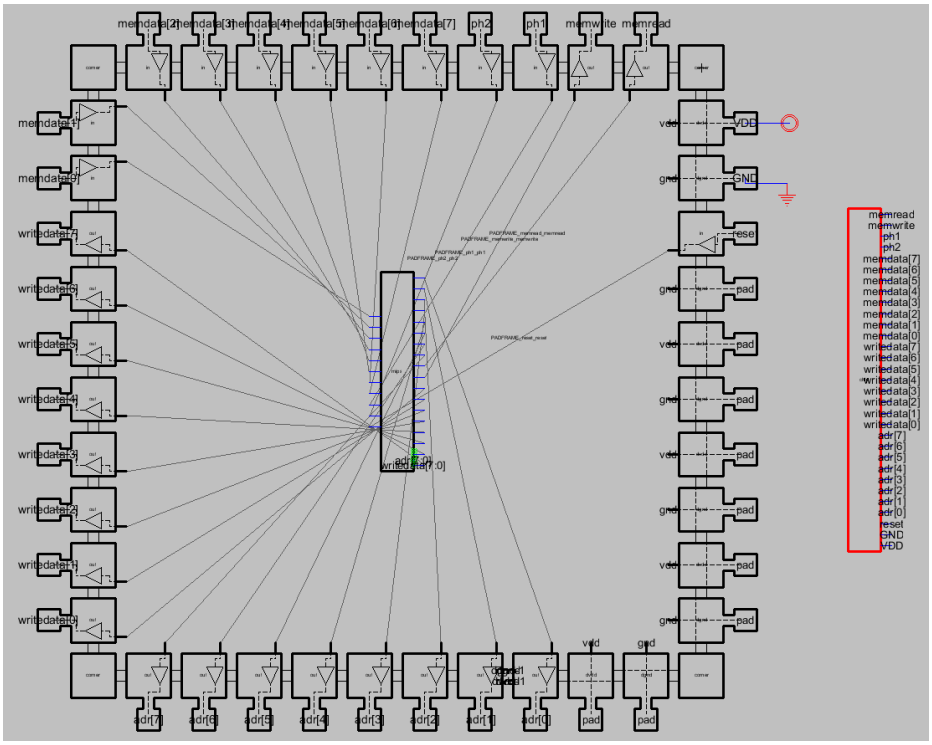
Εικόνα 75

6.2.2 ΣΧΗΜΑΤΙΚΟ ΚΑΙ LAYOUT CHIP

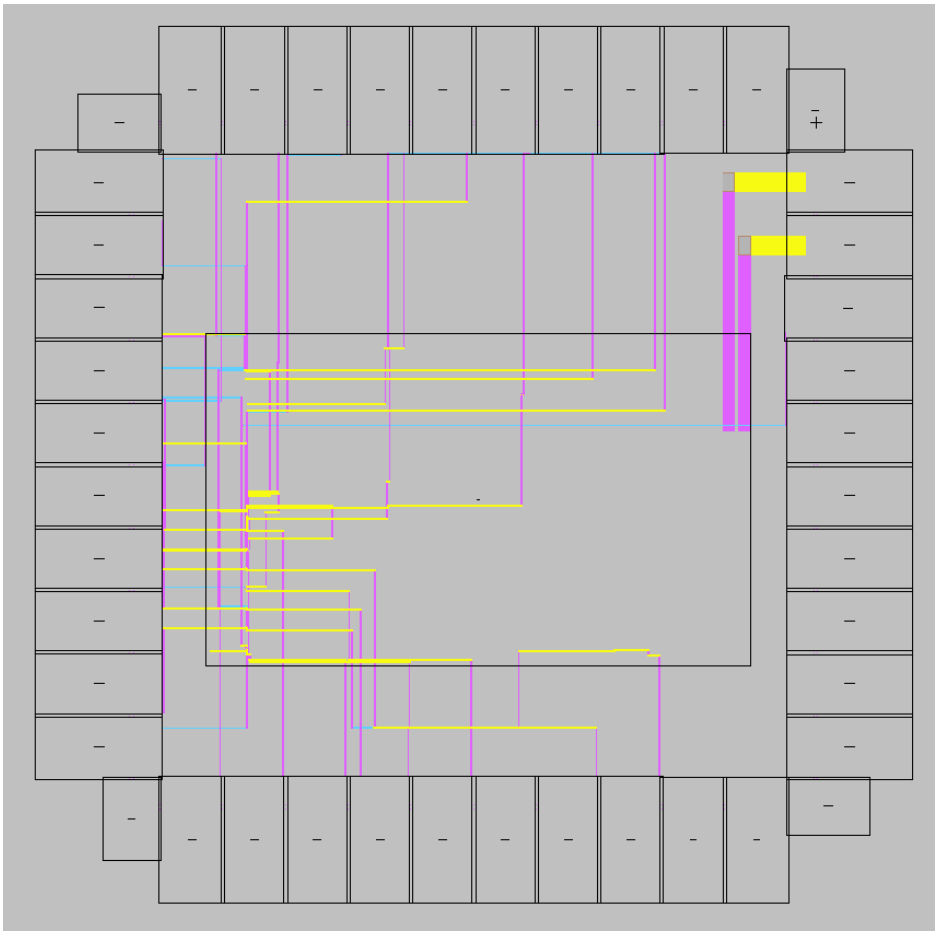
Για τη δημιουργία του chip χρησιμοποιήθηκε η διαδικασία του Electric, Pad Frame Generator. Η συγκεκριμένη διαδικασία, συναρμολογεί αυτόματα το pad frame από τη βιβλιοθήκη. Για να μπορέσει να γίνει αυτό, χρειάστηκε ο MIPS, μια βιβλιοθήκη με pads και ένα pad arrangement file. Η βιβλιοθήκη και το αρχείο δίνονται από τις εργαστηριακές ενότητες. Το pad arrangement file, προσδιορίζει τη βιβλιοθήκη, όπου το pad frame είναι αποθηκευμένο, το όνομα του cell που θα δημιουργηθεί, το όνομα της κορυφαίας σχεδίασής μας και τέλος τα ports που πρέπει να χρησιμοποιηθούν για να βάλει τα pads στη θέση που πρέπει.

Χρησιμοποιώντας την εντολή στο Electric, Tools > Generation > Pad Frame Generator και διαλέγοντας το arrangement file (mips8.arr) δημιουργήθηκαν τρία cells για σχηματικό, layout και icon με ονόματα chip(sch), chip(lay) και chip(ic) αντίστοιχα. Αφού μετακινήσαμε αναλόγως τον επεξεργαστή μας μέσα στο chip για να φαίνεται σωστά στο σχηματικό, συνεχίσαμε συνδέοντας τα power και ground στο επάνω δεξιά μέρος των pads. Έπειτα, εισαγάγαμε power και ground στο σχηματικό μας. Τα αποτελέσματα των παραπάνω διαδικασιών φαίνονται στις παρακάτω εικόνες, όπου βλέπουμε το σχηματικό και το layout του chip που δημιουργήθηκε. [19]

Τέλος, για να ολοκληρωθεί το σχέδιό μας, θα μπορούσαμε να δρομολογήσουμε τα καλώδια από το pad frame στον πυρήνα στο layout, αλλά αυτό θα ήταν κουραστικό. Το Electric περιλαμβάνει έναν αυτόματο δρομολογητή που ονομάζεται Sea-of-Gates router. Το συγκεκριμένο, δρομολογεί αυτόματα τα generic unrouted arcs. Για να διαμορφώσουμε τον router, επιλέγουμε File > Preferences και στη συνέχεια Tools • Routing. Ρυθμίζουμε το μέγιστο πλάτος τόξου σε 4, έτσι ώστε το Electric να μην χρησιμοποιήσει υπερβολικά φαρδιά καλώδια. (Παρόλα αυτά, το metal3 θα δρομολογηθεί σε πλάτος 6 λ). Χωρίς να έχουμε επιλέξει κάτι στο layout επιλέγουμε Tool > Routing > Sea-of-Gates Route.



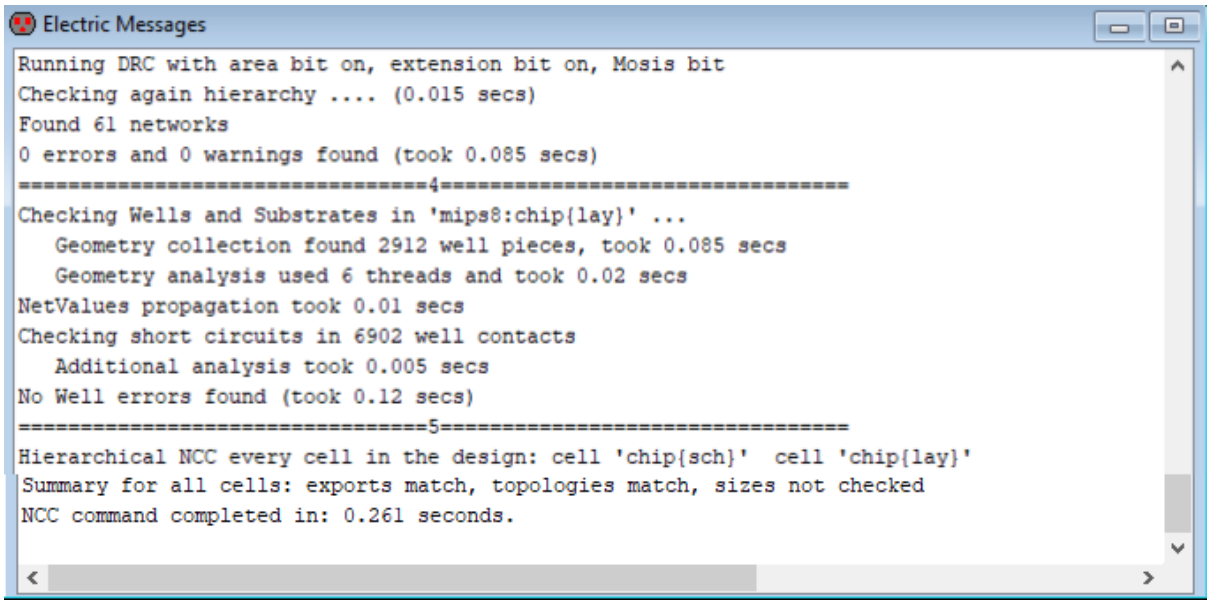
Εικόνα 76



Εικόνα 77

6.3 ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ 4ΗΣ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΕΝΟΤΗΤΑΣ

Έχοντας ολοκληρώσει και το pad frame, θα δούμε τώρα τις επαληθεύσεις του, κάνοντας τις γνωστές διαδικασίες DRC, ERC και NCC. Στην παρακάτω εικόνα 78 βλέπουμε πως ολοκληρώθηκαν και τα τρία χωρίς κάποιο λάθος.



Εικόνα 78

Για να ολοκληρωθεί η σχεδίασή μας και να είμαστε σε θέση να την δώσουμε στον κατασκευαστή, πρέπει να κάνουμε το λεγόμενο Tapeout, που αναφέραμε στο δεύτερο κεφάλαιο. Με τη σειρά του και αυτό θα γίνει μέσω του Electric πατώντας File > Export > CIF και δίνουμε όνομα στο αρχείο mips8.cif. Το αρχείο CIF (Calteck Interchange Format) είναι ένα από τα γνωστά αρχεία που χρησιμοποιούνται για το Tapeout και είναι το πιο εύκολο στην ανάγνωση. Μπορούμε να το ανοίξουμε μέσω ενός text editor και αμέσως μπορούμε να δούμε και να καταλάβουμε κάποια από τα cell που δημιουργήσαμε κατά την εκτέλεση των εργαστηριακών ενοτήτων. Το συγκεκριμένο αρχείο υπάρχει στα αρχεία που δίνονται μαζί με την πτυχιακή εργασία.

Με τη δημιουργία αυτού του αρχείου σηματοδοτείται και το τέλος των εργαστηριακών ενοτήτων. Πλέον είμαστε ικανοί να δημιουργήσουμε σχηματικά και layouts, να επαληθεύουμε και να προσομοιώνουμε σε Electric και ModelSim, όπως επίσης και να δημιουργούμε pad frames και να κάνουμε τους τελευταίους ελέγχους πριν την κατασκευή ενός ψηφιακού κυκλώματος.

ΚΕΦΑΛΑΙΟ 7: ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΠΡΟΤΑΣΕΙΣ ΚΑΤΑ ΤΗ ΣΧΕΔΙΑΣΗ ΚΥΚΛΩΜΑΤΩΝ

7.1 ΣΥΜΠΕΡΑΣΜΑΤΑ

Στο συγκεκριμένο κεφάλαιο της πτυχιακής εργασίας παρατίθενται τα ουσιαστικά συμπεράσματα κατά την σχεδίαση ενός ψηφιακού κυκλώματος, αλλά και της πτυχιακής γενικότερα.

Μέσω μίας μεγάλης και ιστορικής αναδρομής, ανακαλύψαμε την ιστορία των ψηφιακών κυκλωμάτων τα οποία ξεκίνησαν ως ιδέα το 1947 με τη δημιουργία του πρώτου λειτουργικού τρανζίστορ. Η συνεχής και αδιάκοπη ανάπτυξη στον συγκεκριμένο τομέα έχει επιφέρει πολλές αλλαγές, φτάνοντας έτσι στο 2020 όπου οι επεξεργαστές και τα τρανζίστορ συνεχίζουν και γίνονται μικρότερα, ταχύτερα, αποδοτικότερα και με λιγότερη ισχύς.

Η παρούσα πτυχιακή εκπονήθηκε ώστε να κατανοήσουμε τη σχεδίαση ψηφιακών κυκλωμάτων, μέσω της σχεδίασης ενός μικροεπεξεργαστή MIPS 8-bit μέσω τεσσάρων εργαστηριακών ενοτήτων. Μέσα από τις τέσσερεις αυτές ενότητες μάθαμε να χρησιμοποιούμε αναλυτικά τις εφαρμογές Electric και ModelSim, όπως επίσης κατανοήσαμε τις τεχνολογίες που χρησιμοποιήσαμε προκειμένου να ολοκληρώσουμε τον μικροεπεξεργαστή. Με τον όρο τεχνολογίες εννοούμε τις γλώσσες περιγραφής υλικού (Verilog), την RTL σχεδίαση, τους Κανόνες Σχεδίασης αλλά και την τεχνολογία του ίδιου του μικροεπεξεργαστή.

Κατά την πρώτη εργαστηριακή ενότητα κατανοήθηκε η βασική σχεδίαση ψηφιακών κυκλωμάτων καθώς και η επαλήθευση τους μέσω των εφαρμογών Electric και ModelSim. Μέσω της δημιουργίας πέντε απλών κυκλωμάτων, τα οποία είναι τα NAND, NOR, AND, OR και INV εμπεδώθηκε η διαδικασία κανόνων σχεδίασης όπως επίσης και οι βασικές λειτουργίες των εφαρμογών.

Στην δεύτερη εργαστηριακή ενότητα διαπιστώθηκε η λειτουργία του Datapath, ενός από τα τρία βασικά στοιχεία του MIPS 8-bit. Αρχικά ζητήθηκε η δημιουργία 2 bitslices, ενός OR και ενός AND με μέγεθος 8 bit, τα οποία τοποθετήθηκαν στην ALU, ενός στοιχείου του Datapath, για να μπορέσουμε να ολοκληρώσουμε το Datapath. Αφού επαληθεύθηκε το Datapath προχωρήσαμε στην επαλήθευση της σχεδίασής του μέσω ενός testbench σε γλώσσα Verilog προσομοιώνοντας ολόκληρο τον μικροεπεξεργαστή.

Στην επόμενη ενότητα, έγινε αντιληπτός ο Controller του επεξεργαστή, που είναι υπεύθυνος για τη δημιουργία σημάτων προς το Datapath ώστε να μπορεί να γίνεται προσπέλαση και εκτέλεση οποιασδήποτε εντολής. Στο πρώτο σκέλος του δημιουργήσαμε ένα ALU decoder και τέλος χρησιμοποιήσαμε ένα εργαλείο, το PLA Generator ώστε, μαζί με το ALU decoder να κατανοήσουμε, μαζικά, την λειτουργία του Controller στον MIPS 8 bit.

Φτάνοντας στην τελευταία εργαστηριακή ενότητα και αφού έχουμε μάθει για τα components του μικροεπεξεργαστή ξεχωριστά, ολοκληρώσαμε τη σχεδίαση του βάζοντας τα κομμάτια μαζί και επαληθεύθηκε και προσομοιώθηκε ο επεξεργαστής σαν ένα. Έπειτα, έγιναν κατανοητά τα βήματα δημιουργίας ενός PAD frame και τέλος μάθαμε για τους τελευταίους ελέγχους που έπρεπε να γίνουν προκειμένου ο επεξεργαστής να προχωρήσει στην παραγωγή.

7.2 ΠΡΟΤΑΣΕΙΣ ΚΑΤΑ ΤΗ ΣΧΕΔΙΑΣΗ ΚΥΚΛΩΜΑΤΩΝ

Σε αυτή την τελευταία ενότητα δίνονται κάποιες προτάσεις οι οποίες θα μπορούν να βοηθήσουν κατά τη σχεδίαση κυκλωμάτων αλλά επίσης και στην βελτίωσή τους. Θα βοηθήσουν στον εντοπισμό σφαλμάτων, στην προσομοίωση, στην αναφορά και σε πολλά άλλα. Ο σχεδιασμός κυκλωμάτων είναι δύσκολος επειδή απαιτεί εκτενή γνώση κάθε εξαρτήματος που αποτελεί μέρος του κυκλώματος και η επίτευξη «τέλειων» σχεδίων απαιτεί εκτεταμένη πρακτική.

Οι συμβουλές που παρουσιάζονται παρακάτω είναι καλό να λαμβάνονται υπόψη και εφαρμόζονται κατά τη σχεδίαση του κυκλώματός, ώστε τα κυκλώματά να φαίνονται επαγγελματικά, να λειτουργούν σε βέλτιστα επίπεδα απόδοσης και να βελτιώνουν την ποιότητά τους. Αναφορικά κάποιες από αυτές είναι :

- Ένα διάγραμμα μπλοκ πάντα ενισχύει τη σχεδίαση μας, αποτυπώνοντας πάνω σε αυτό τη σχεδίαση που έχουμε στο μυαλό μας αποφεύγοντας λάθη που μπορούμε να κάνουμε στη συνέχεια.
- Κατά την εκτέλεση των παραπάνω εργαστηριακών ασκήσεων είδαμε ότι πήραμε κάθε component ξεχωριστά προκειμένου να φτάσουμε στο τελικό αποτέλεσμα. Είναι καλό αυτός ο διαχωρισμός να συμβαίνει προκειμένου να καταλάβουμε καλύτερα το κάθε κομμάτι ξεχωριστά.
- Όταν σχεδιάζουμε καλό θα ήταν να δίνονται ονόματα στα δίκτυα που συνδέονται μεταξύ των components διότι έτσι, φτάνουμε πιο γρήγορα στη ρίζα του προβλήματος που μπορεί να προκύψει.
- Είναι σημαντικό να κρατάμε σημειώσεις όταν σχεδιάζουμε σε κάθε μας βήμα, να τεκμηριώνουμε κάθε λάθος που μπορεί να προκύψει, κάθε λύση που βρίσκουμε για αυτά οτιδήποτε άλλο σχετίζεται με το κύκλωμα.

Συμπερασματικά, οι παραπάνω προτάσεις είναι σίγουρο ότι θα μας βοηθήσουν να σχεδιάζουμε καλύτερα τα κυκλώματά σας. Θα βοηθήσουν στον εντοπισμό σφαλμάτων, την προσομοίωση, την αναφορά, ακόμα και στην μελλοντική εξέλιξη των κυκλωμάτων. [20]

ΒΙΒΛΙΟΓΡΑΦΙΑ

ΕΛΛΗΝΙΚΗ ΒΙΒΛΙΟΓΡΑΦΙΑ

- [1] ΣΑΝ ΣΗΜΕΡΑ ΣΤΟΥΣ ΥΠΟΛΟΓΙΣΤΕΣ. 23/ 12/ 1947 | ΤΡΑΝΖΙΣΤΟΡ. Ανακτήθηκε από το διαδικτυακό τόπο στις 08/08/2020. https://sansimeracomputers.wordpress.com/2012/12/23/dec23-1947/
- [2] Δημοσθένης Β. Λουκίσσας. Ανοικτοκυκλώματα γραμμών αλληλοσύνδεσης σε τεχνολογία CMOS. Ανακτήθηκε από το διαδικτυακό τόπο στις 06/09/2020. https://docplayer.gr/41447114-Anoiktokyklouata-grauuon-allilosyndesis-se-tehnologia-cmos.html
- [3] Neil H. E. Weste, David M. Harris. Σχεδίαση Ολοκληρωμένων Κυκλωμάτων CMOS VLSI. 2011, Εκδόσεις Παπασωτηρίου.
- [4] Γ. Λιαπέρδος. Εισαγωγή στην Ηλεκτρονική Κεφ. 4. Ανακτήθηκε από το διαδικτυακό τόπο στις 06/09/2020. https://liaperdos.gr/public/intro_to_electronics/xhtml/chapter_04.xhtml#section_04_04
- [5] Editor Nax. Ξαναγράφοντας τον νόμο του Moore (PC MAG Greece). Ανακτήθηκε από το διαδικτυακό τόπο στις 06/09/2020. https://gr.pcmag.com/apopseis/28733/ksanagraphontas-ton-nomo-tou-moore

ΞΕΝΟΓΛΩΣΣΗ ΒΙΒΛΙΟΓΡΑΦΙΑ

- [6] Real Engineering. Transistors - The Invention That Changed The World. Ανακτήθηκε από το διαδικτυακό τόπο στις 08/08/2020. https://www.youtube.com/watch?v=OwS9aTE2Go4
- [7] Computer History Museum. The Impact of Integrated Circuits, lecture by Robert Noyce. Ανακτήθηκε από το διαδικτυακό τόπο στις 09/08/2020. https://www.youtube.com/watch?v=AfxUq_QrpyY
- [8] Texas Instruments. Jack Kilby and the Integrated Circuit. Ανακτήθηκε από το διαδικτυακό τόπο στις 10/08/2020. https://www.youtube.com/watch?v=2mez_HRrsXk
- [9] Gordon E. Moore. Progress In Digital Integrated Electronics. Ανακτήθηκε από το διαδικτυακό τόπο στις 10/08/2020. http://ai.eecs.umich.edu/people/conway/VLSI/BackgroundContext/SMErpt/AppB.pdf
- [10] Static Free Software. Electric History. Ανακτήθηκε από το διαδικτυακό τόπο στις 14/08/2020. https://www.staticfreesoft.com/electricHistory.html
- [11] Mentor Graphics. ModelSim PE Student Edition. Ανακτήθηκε από το διαδικτυακό τόπο στις 14/08/2020. https://www.mentor.com/company/higher_ed/modelsim-student-edition
- [12] John Enyeart, Mohammed Al-Karawi, Ruben Medina. 8-bit MIPS Processor. Ανακτήθηκε από το διαδικτυακό τόπο στις 14/08/2020. http://cmosedu.com/cmos1/electric/Senior%20Design%20Report.pdf

- [13] Science Direct. Testbench. Ανακτήθηκε από το διαδικτυακό τόπο στις 14/08/2020. https://www.sciencedirect.com/topics/computer-science/testbench
- [14] Xilinx® Inc. Test Benches. Ανακτήθηκε από το διαδικτυακό τόπο στις 18/08/2020. https://www.xilinx.com/support/documentation/sw_manuals/xilinx10/isehelp/ise_c_simulation_test_bench.htm
- [15] Computer Hope. Data Path. Ανακτήθηκε από το διαδικτυακό τόπο στις 18/08/2020. https://www.computerhope.com/jargon/d/datapath.htm
- [16] David M. Harris. CMOS VLSI Design Lab1: Cell Design and Verification. Ανακτήθηκε από το διαδικτυακό τόπο στις 18/09/2020. http://pages.hmc.edu/harris/cmosvlsi/4e/cadencelabs/Lab1.pdf
- [17] David M. Harris. CMOS VLSI Design Lab2: Datapath Design and Verification. Ανακτήθηκε από το διαδικτυακό τόπο στις 18/09/2020. http://pages.hmc.edu/harris/cmosvlsi/4e/electriclabs/Lab2.pdf
- [18] David M. Harris. CMOS VLSI Design Lab3: Controller Design and Verification. Ανακτήθηκε από το διαδικτυακό τόπο στις 18/09/2020. http://pages.hmc.edu/harris/cmosvlsi/4e/electriclabs/Lab3.pdf
- [19] David M. Harris. CMOS VLSI Design Lab4: Full Chip Assembly. Ανακτήθηκε από το διαδικτυακό τόπο στις 18/09/2020. http://pages.hmc.edu/harris/cmosvlsi/4e/electriclabs/Lab4.pdf
- [20] 8 Circuit Design Tips by Nichole Heydenburg, 26/ 9/ 2018. Ανακτήθηκε από το διαδικτυακό τόπο στις 25/09/2020. https://www.eeweb.com/8-circuit-design-tips/
- [21] Random Logic by Wikipedia. Ανακτήθηκε από το διαδικτυακό τόπο στις 25/09/2020 https://en.wikipedia.org/wiki/Random_logic
- [22] The System Verilog Evolution by IEEE Ανακτήθηκε από το διαδικτυακό τόπο στις 25/09/2020. https://ieeexplore.ieee.org/document/1214355
- [23] Design Abstraction by Science Direct. Ανακτήθηκε από το διαδικτυακό τόπο στις 25/09/2020. https://www.sciencedirect.com/topics/computer-science/design-abstraction
- [24] Synthesis. Peter Wilson. Design Recipes for FPGAs (Second Edition). 2016. Ανακτήθηκε από το διαδικτυακό τόπο στις 25/09/2020. https://www.sciencedirect.com/topics/engineering/data-path