ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΕΡΓΑΣΤΗΡΙΟ ΛΟΓΙΚΗΣ ΣΧΕΔΙΑΣΗΣ

ΣΥΝΟΠΤΙΚΟΣ ΟΔΗΓΟΣ ΧΡΗΣΗΣ ΤΟΥ VIVADO

Βασιλόπουλος Διονύσης

ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ
2024-2025

Πρόλογος

Οι εργαστηριακές ασκήσεις του μαθήματος αφορούν τη σχεδίαση ψηφιακών κυκλωμάτων και την υλοποίησή τους σε τεχνολογία Field Programmable Gate Arrays (FPGAs) με τη χρήση των κατάλληλων εργαλείων λογισμικού της XILINX (Vivado Design Suite). Συγκεκριμένα αφορούν τη σχεδίαση με τη γλώσσα περιγραφής υλικού VHDL, τη δημιουργία κατάλληλων περιορισμών χρονισμού (timing constraints) και I/O (I/O constraints), τη σύνθεση (synthesis), υλοποίηση (implementation) και παραγωγή του bitstream στοχεύοντας συγκεκριμένο FPGA καθώς και την επαλήθευση της ορθής σχεδίασης (verification) σε όλα τα διαδοχικά επίπεδα σχεδίασης (πηγαίος κώδικας VHDL, post place & route netlist) με προσομοίωση (simulation), την αποσφαλμάτωση (debug) καθώς και την επαλήθευση της λειτουργικότητας στο υλικό (FPGA validation) με χρήση κατάλληλης αναπτυξιακής κάρτας (development board).

Τα ψηφιακά κυκλώματα που θα υλοποιηθούν θα έχουν κλιμακούμενη πολυπλοκότητα και διδακτική αξία παρόμοια με αντίστοιχα που διδάσκονται στο προπτυχιακό μάθημα κορμού της Λογικής Σχεδίασης (Κ02) το οποίο θεωρείται προαπαιτούμενο ενώ υπάρχει συντονισμός στην διδακτέα ύλη.

Τα FPGAs που θα χρησιμοποιηθούν ως target devices στην εκπαιδευτική διαδικασία θα είναι αυτά που παρέχει η υποδομή του εργαστηρίου σε αναπτυξιακές κάρτες. Η πιο κατάλληλη, σύγχρονη αναπτυξιακή κάρτα που παρέχεται από την XILINX για ακαδημαϊκούς σκοπούς είναι η ZedBoard.

Τέλος θα πρέπει να σημειωθεί ότι ο οδηγός θα ενημερώνεται καθ’ όλη τη διάρκεια του εξαμήνου ώστε να ενσωματώνει τις δυνατότητες του VIVADO όπως αυτές παρουσιάζονται κατά την πρόοδο των μαθημάτων. **Ο οδηγός ισχύει για όποια έκδοση του VIVADO και εάν έχετε.**

Εισαγωγή στο Εργαλείο

Σε αυτή την εισαγωγική παρουσίαση θα δούμε αναλυτικά τη χρήση του εργαλείου Vivado IDE της Xilinx για τη δημιουργία ενός απλού κυκλώματος με χρήση της. Θα δούμε τα βήματα της προσομοίωσης, της σύνθεσης και της υλοποίησης του σχεδίου χρησιμοποιώντας τις προκαθορισμένες ρυθμίσεις. Για τους σκοπούς της παρουσίασης θεωρούμε ότι με τη χρήση του VIVADO θα λύσουμε την άσκηση (που αναλύσουμε και σε διαλέξεις) που ακολουθεί:

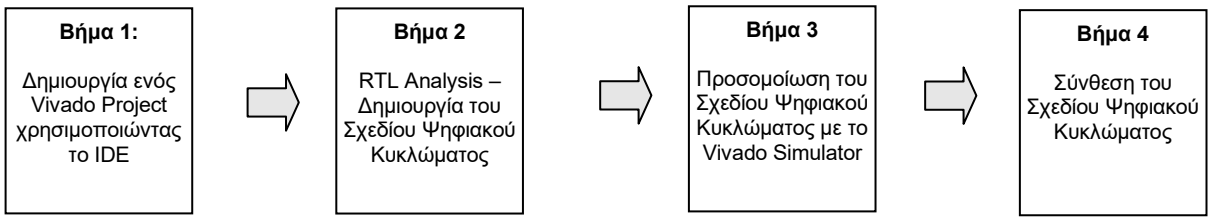
Εργοστάσιο έχει δεξαμενή επεξεργασίας υγρών. Η δεξαμενή πρέπει α)να έχει θερμοκρασία μεταξύ 25 και 30 βαθμών και β) η στάθμη του πρέπει να είναι πάνω από ένα επίπεδο. Σε περίπτωση που το α) ή το β) δεν ικανοποιούνται πρέπει να ενεργοποιηθεί ένα κουδούνι. Παραδοχές: α)Έχουμε θερμόμετρα στη διάθεσή μας που μπορούν να μας ενημερώσουν εάν η θερμοκρασία είναι πάνω από ένα όριο (όποιο όριο θέλουμε) και β)Υπάρχει αισθητήρας που μας ενημερώνει εάν η στάθμη στη δεξαμενή είναι κάτω από ένα επίπεδο.

Στόχοι

Με τη μελέτη του φυλλαδίου θα είστε σε θέση να:

- Δημιουργείτε ένα project στο Vivado, να ορίζετε τα αρχεία HDL που περιγράφουν το μοντέλο του κυκλώματος και να ορίζετε ως target, το ZYNQ FPGA που φιλοξενείται στην αναπτυξιακή κάρτα ZedBoard.
- Προσομοιώνετε το σχέδιο με τον simulator του Vivado
- Κάνετε σύνθεση (synthesis)

Γενική ροή σχεδίασης για πλήρη ανάπτυξη



Δημιουργία ενός Vivado Project με το IDE Βήμα 1

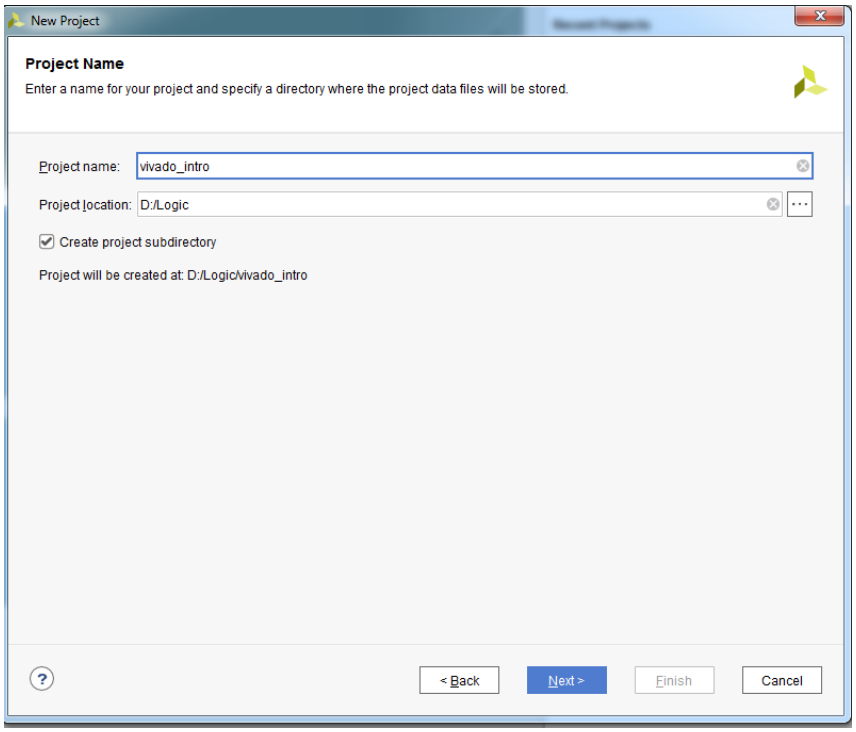
1-1. Ανοίξτε το Vivado και δημιουργείστε ένα project έχοντας ως target το device XC7Z020clg484-1 και χρησιμοποιώντας την VHDL. Χρησιμοποιήστε τον πηγαίο κώδικα που σας παρέχεται lab1.vhd. Ανάλογα με την έκδοση του Vivado και τις επιλογές εγκατάστασης ίσως υπάρχουν διαφορές στις επιλογές των μενού.

1-1-1. Ανοίξτε το Vivado επιλέγοντας **Start > All Programs > Xilinx Design Tools > Vivado 2022.2 > Vivado 2022.2.** Σε περίπτωση που υπάρχει το αντίστοιχο εικονίδιο στο Desktop απλά κάνετε διπλό κλικ (προσοχή ΟΧΙ στο Vivado HLS 2022.2 ή στο Vitis HLS 2022.2)

1-1-2. Πατήστε **Create New Project (ή File-> Project->New)** για να ξεκινήσετε τον wizard. Θα δείτε ένα κουτί διαλόγου *Create A New Vivado Project*. Πατήστε **Next**. Σε περίπτωση που το project έχει ήδη δημιουργηθεί μπορούμε να το επιλέξουμε με το Project_name είτε από το Open Project στο παράθυρο Quick Start, είτε από το παράθυρο Recent Project.

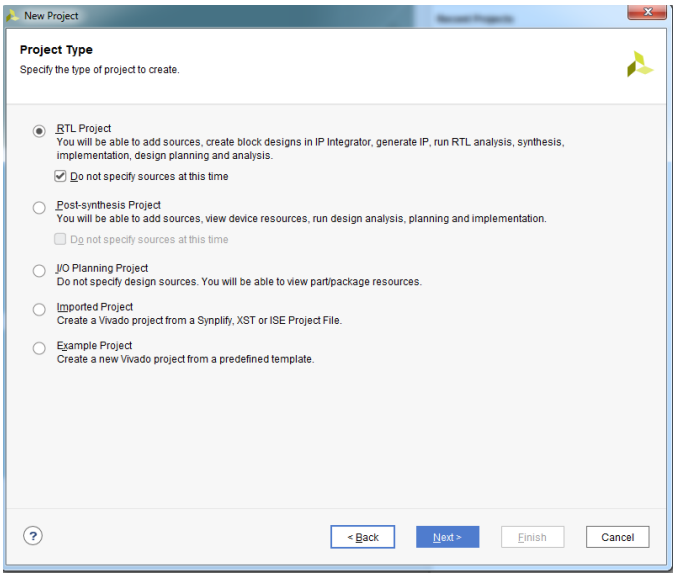
1-1-3. Πατήστε το κουμπί Browse του πεδίου *Project location* της φόρμας **New Project**, και κάνετε browse στο φάκελο που θα τοποθετήσετε το project σας (στο παράδειγμά μας **d:/Logic**) και πατήστε **Select**.

1-1-4. Γράψτε **vivado_intro** στο πεδίο *Project name*. Βεβαιωθείτε ότι το κουτί *Create Project Subdirectory* box είναι επιλεγμένο. Παρατηρήστε ότι στην επόμενη γραμμή φαίνεται ο κατάλογος στον οποίο θα δημιουργηθεί το project. Πατήστε **Next**.



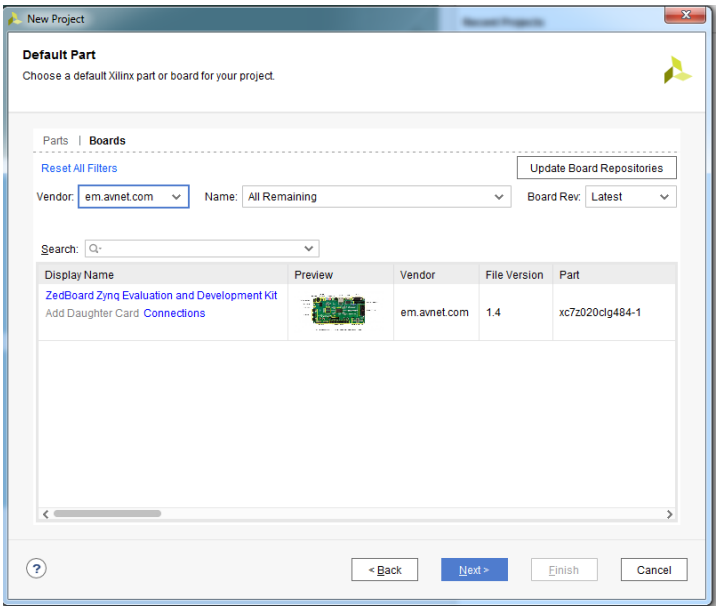
Εικόνα 1. Εισαγωγή Project Name και Project Location

1-1-5. Κάντε κλικ στην επιλογή **RTL Project** στην επόμενη οθόνη (φόρμα *Project Type)*, και επιλέξτε **Next**. Επίσης παρατηρείστε ότι έχουμε κάνει κλικ στην επιλογή **Do not specify sources at this time.** Αυτό σημαίνει ότι δεν έχουμε αρχεία με έτοιμο κώδικα για το project και ότι θα δημιουργήσουμε τα αντίστοιχα αρχεία μέσα από το πρόγραμμα. Σε περίπτωση που έχουμε έτοιμα παρόμοια αρχεία ΔΕΝ θα πρέπει να κάνουμε κλικ.



Εικόνα 2. Επιλογή RTL_Project

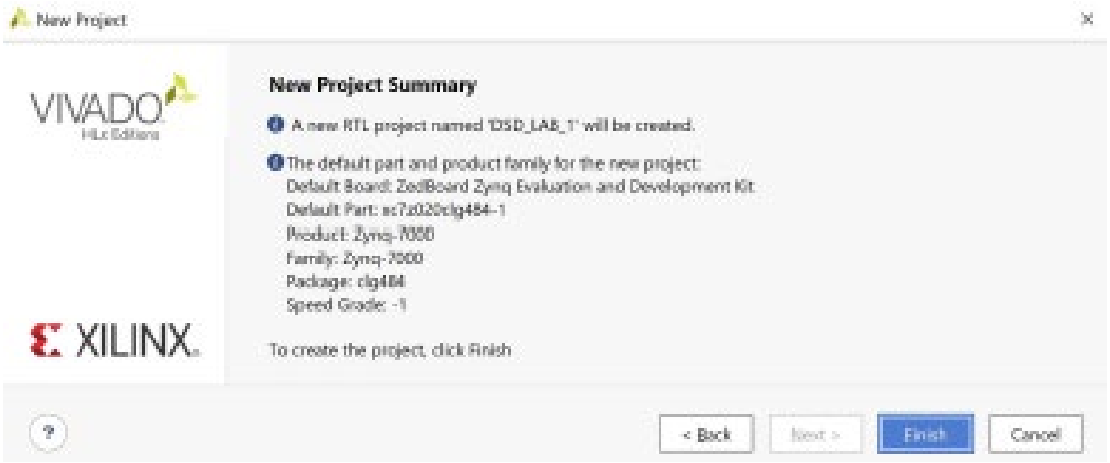
1.1.6. Στη φόρμα *Default Part* , χρησιμοποιήστε την επιλογή **Boards** και το drop-down μενού του φίλτρου **Vendor** επιλέγετε το **em.avnet.com ή Xilinx.com**. Επιλέξτε είτε Zedboard Zynq Evaluation and Development Kit είτε Zynq 7000 ZC702 Evaluation Board. Εναλλακτικά από το το tab Parts επιλέξτε το part **XC7Z020clg484-1**. Παρατηρήστε τις πληροφορίες που υπάρχουν για την κάρτα που επιλέξατε όπως η κατηγορία (Zynq-7020) στην οποία ανήκει ότι έχει 484 ακροδέκτες και ταχύτητα (speed grade) -1 (το πιο αργό της οικογένειας) καθώς και τα διαθέσιμα resources του επιλεγμένου part (IOB = 200, LUT Elements = 53.200, Flip-Flops = 106.400, Block RAMs = 140 και DSPs = 220)



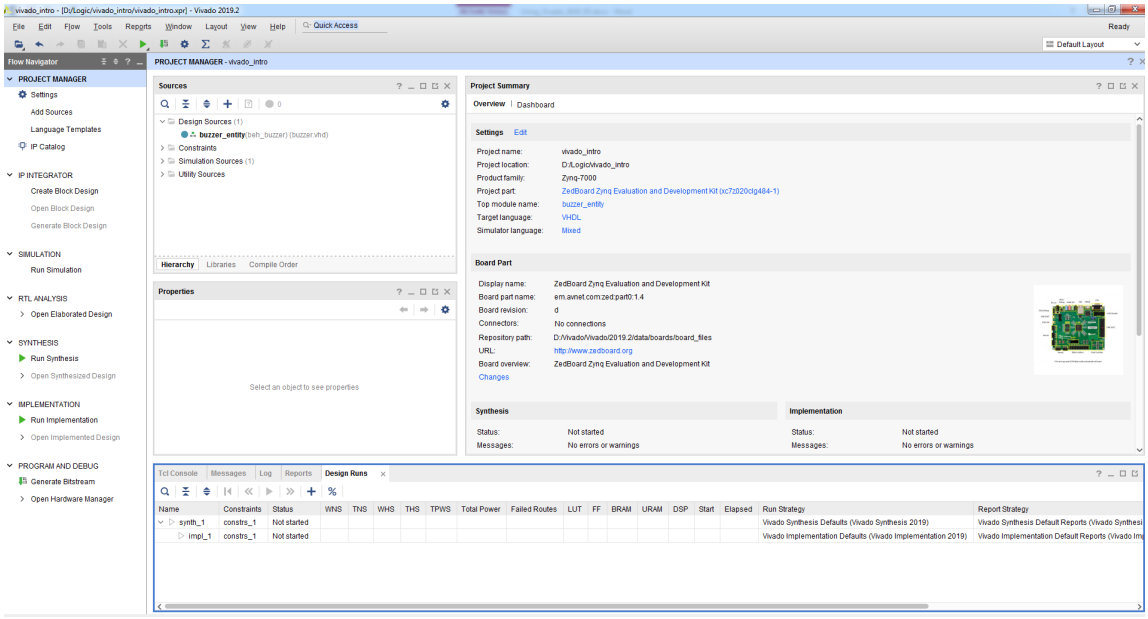
Εικόνα 3. Επιλογή της αναπτυξιακής κάρτας

1-1-7 .Επιλέξτε **Next**.

1-1-8 .Στο παράθυρο διαλόγου N*ew Project Summary* βλέπετε το project_name και τα χαρακτηριστικά της επιλεγμένης (default) διάταξης FPGA.



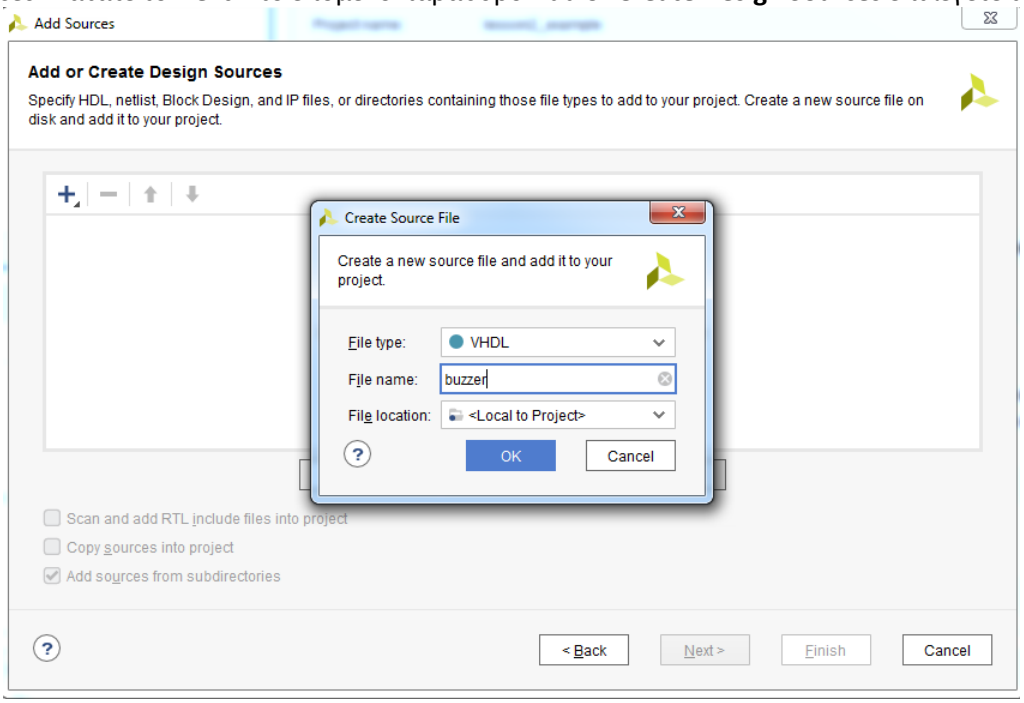
1-1-9 .Πατήστε Finish για να δημιουργήσετε το Vivado project. Παρατηρήστε το περιβάλλον του PROJECT MANAGER πάνω στο οποίο θα σχεδιάσουμε τα ψηφιακά κυκλώματά μας. Αναγνωρίζουμε την οριζόντια μπάρα επάνω με επιλογές File, Edit, Flow, Tools, Reports, Window, Layout, View και Help. Το κατακόρυφο παράθυρο αριστερά του Flow Navigator, όπου με κατάλληλη επιλογή εκτελούνται όλα τα βήματα της ψηφιακής σχεδίασης. Το παράθυρο Sources, όπου φαίνονται όλα τα πηγαία αρχεία του project διαρθρωμένα στα directories: Design Sources, Constraints, Simulation Sources και Utility Sources (αρχικά είναι άδειο). Το παράθυρο Properties. Το παράθυρο Project Summary (σε κάθε βήμα της ψηφιακής σχεδίασης διαφοροποιείται κατάλληλα). Τέλος, το κάτω οριζόντιο παράθυρο πολλαπλών χρήσεων, που χρησιμεύει μεταξύ άλλων ως Tcl Console και για την ανάλυση των αποτελεσμάτων της σύνθεσης και της υλοποίησης.



Εικόνα 4. Βασική οθόνη ενός Project στο Vivado

1-2. Δημιουργία αρχείου κώδικα buzzer.vhd

1-2-1. Στο παράθυρο *Sources*, κάντε δεξί-click στην επιλογή **Design Sources** και κατόπιν κάνετε κλικ στην επιλογή *Add Sources*. Στο παράθυρο **Add Sources** που εμφανίζεται επιλέγετε το *Add or creare design sources*. Πατάτε το **Next**. Στο επόμενο παράθυρο **Add or Create Design Sources** επιλέγετε το *Create File*.

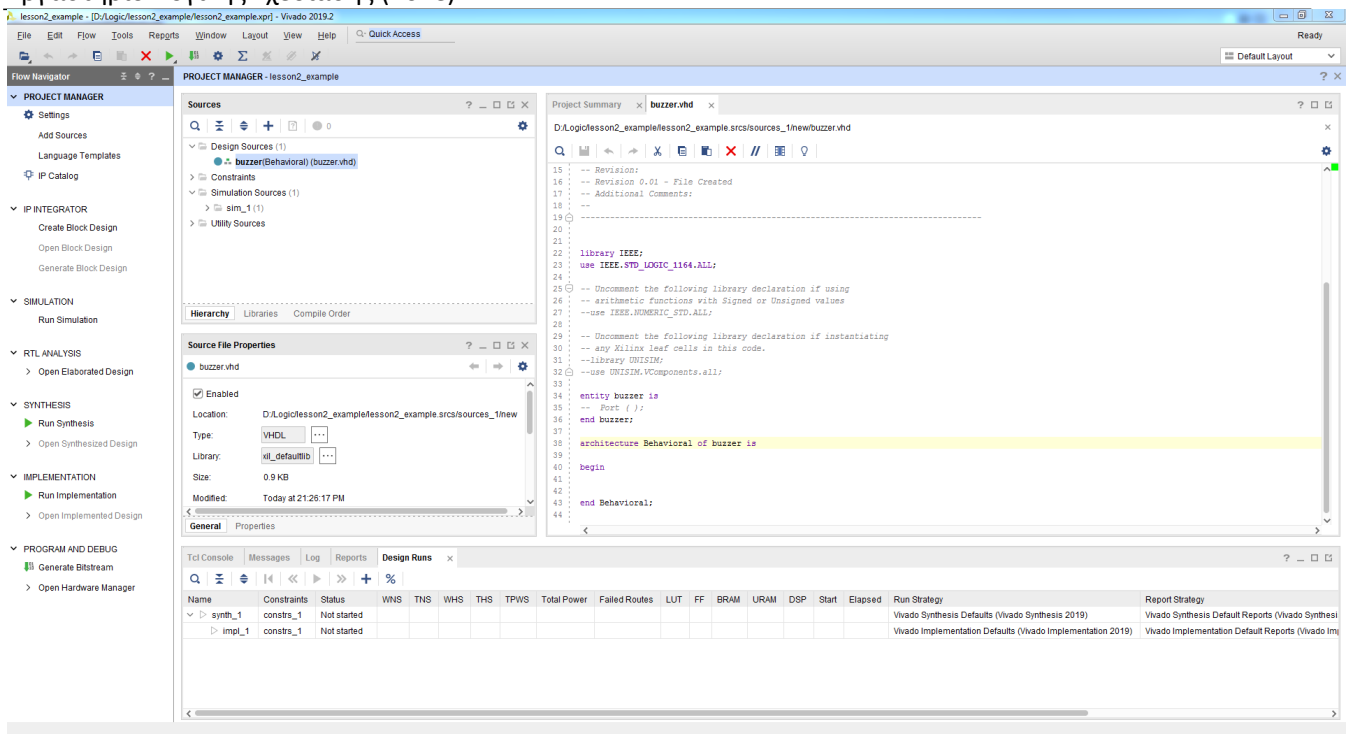


Εικόνα 5. Δημιουργία αρχείου VHDL οντότητας/αρχιτεκτονικής

Στο παράθυρο **Create Source File** επιλέγεται ως File Type το *VHDL*, και πληκτρολογείτε το όνομα του αρχείου που θα περιέχει τον κώδικα του προγράμματός μας (στην περίπτωσή μας το buzzer). Με αυτές τις επιλογές σας έχετε δηλώσει ότι η γλώσσα προγραμματισμού που θα χρησιμοποιήσετε θα είναι η VHDL, και άρα η κατάληξη του αρχείου που θα δημιουργηθεί θα είναι ***.vhd***. Οπότε πατώντας το OK έχετε δημιουργήσει το αρχείο *buzzer.vhd*. Ύστερα επιλέγετε το **Finish**.

1-2-2. Στο παράθυρο **Define Module**, παρατηρήστε ότι το Vivado έχει κάνει δύο προεπιλογές. Σας προτείνει α)να ονομάσετε buzzer την οντότητα (Entity name) του κυκλώματος που θα σχεδιάσετε (όπως δηλαδή το όνομα του αρχείου που θα περιέχει τον κώδικα της VHDL) και β) να ονομάσετε Behavioral την αρχιτεκτονική της οντότητας. Εάν θέλετε μπορείτε να αλλάξετε τα ονόματα, σύμφωνα με τους κανόνες που ισχύουν στην VHDL, για το παράδειγμά μας όμως απλά πατάτε το **OK** και στο επόμενο παράθυρο πατάτε το **YES**.

1-2-3. Πλέον παρατηρείτε ότι στο παράθυρο Sources και κάτω από την επιλογή Design Sources έχει δημιουργηθεί η οντότητα *buzzer*, με την αρχιτεκτονική *behavioral*, στο αρχείο *buzzer.vhd*. Κάντε διπλό κλικ στο γραμμή buzzer και θα ανοίξει ένα παράθυρο που θα έχει το αρχείο buzzer.vhd με προσυμπληρωμένο κάποιο βασικό κώδικα (βασικές βιβλιοθήκες, και βασικοί ορισμοί της οντότητας και της αρχιτεκτονικής της). Πλέον μπορείτε να συμπληρώσετε μόνοι σας κώδικα. Στην περίπτωσή σας μπορείτε να γράψετε τον κώδικα που είναι στο eclass, στα έγγραφα και να αναφέρετε στο 2ο μάθημα.



Εικόνα 6. Εμφάνιση περιεχομένου ενός αρχείου VHDL

1-3. Δημιουργία αρχείου κώδικα προσομοίωσης buzzer_tb.vhd

Η δημιουργία του αρχείου προσομοίωσης ακολουθεί την ανάλογη διαδικασία με αυτή που ακολουθήσαμε για τη δημιουργία του αρχείου buzzer.vhd (entity/architecture), δηλαδή την παράγραφο 1.2.

1-3-1. Ακολουθούμε τα βήματα της παραγράφου 1.2.1 Οι διαφορές είναι:
Στο παράθυρο Sources επιλέγουμε **Simulation και στο** παράθυρο **Add Sources** που εμφανίζεται επιλέγετε το *Add or create simulation sources*. Πατάτε το **Next**. Στο επόμενο παράθυρο **Add or Create Simulation Sources** επιλέγετε το *Create File*. Στο παράθυρο **Create Source File** πληκτρολογείτε το όνομα του αρχείου που θα περιέχει τον κώδικα της προσομοίωσης του προγράμματός μας (στην περίπτωσή μας το buzzer_tb). Μία γενική αρχή που ακολουθούμε είναι ότι το αρχείο της προσομοίωσης έχει το όνομα του αρχείου της οντότητας με την προσθήκη στο τέλος του **_tb**.

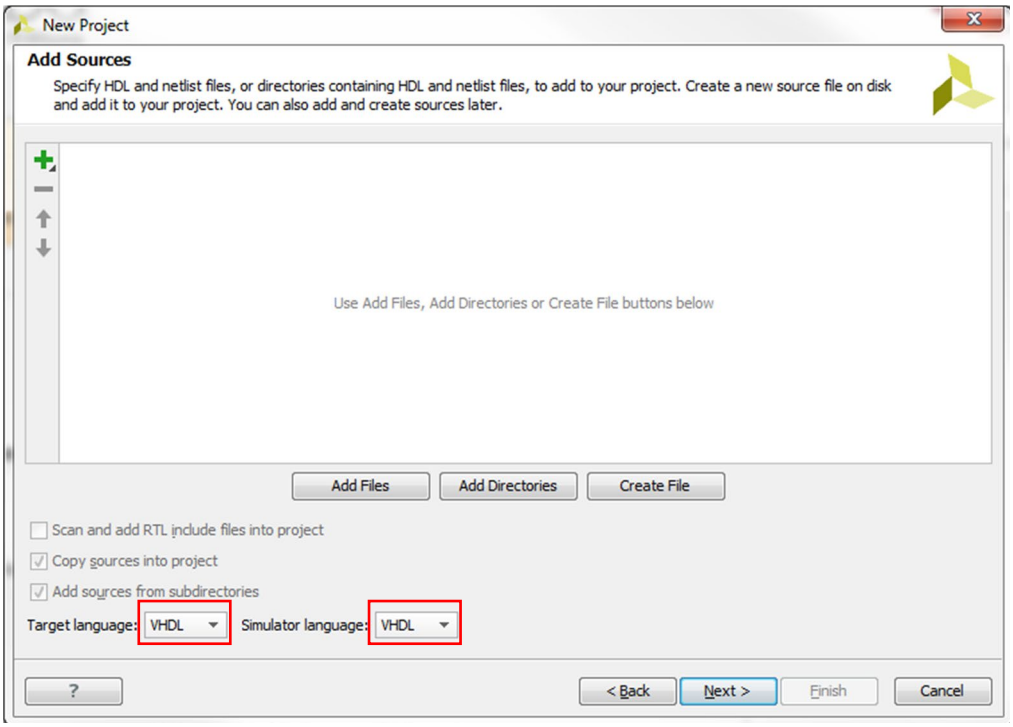
1-3-2. Ακολουθούμε τα βήματα της παραγράφου 1.2.2.

1-3-3. Ακολουθούμε τα βήματα της παραγράφου 1.2.3. Πλέον παρατηρείτε ότι στο παράθυρο Sources και κάτω από την επιλογή Simulation Sources -> sim_1 έχει δημιουργηθεί η οντότητα *buzzer_tb*, με την αρχιτεκτονική *behavioral*, στο αρχείο *buzzer_tb.vhd*. Πλέον μπορείτε να συμπληρώσετε μόνοι σας κώδικα. Στην περίπτωσή σας μπορείτε να γράψετε τον κώδικα που είναι στο eclass, στα έγγραφα και να αναφέρετε στο 2ο μάθημα.

1-4. Ενέργειες σε περίπτωση που έχουμε ήδη δημιουργήσει τα αρχεία του προγράμματος και θέλουμε να τα προσθέσουμε κατά τη δημιουργία του project.

1-4-1. Στο βήμα 1.1.5 ΔΕΝ κάνουμε κλικ στην επιλογή **Do not specify sources at this time.** Αυτό σημαίνει ότι έχουμε αρχεία με έτοιμο κώδικα για το project.

1-4-2. Χρησιμοποιώντας τα κουμπιά drop-down, επιλέξτε **VHDL** ως *Target Language* και *Simulator Language* στην φόρμα *Add Sources*.

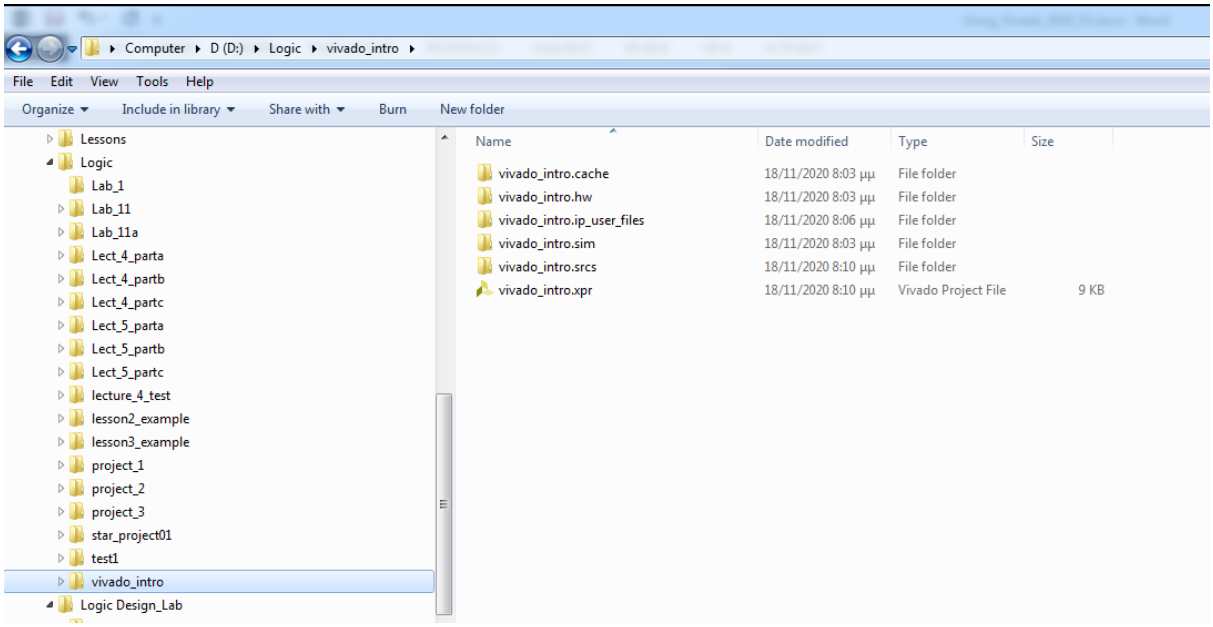


Εικόνα 7. Επιλογή της VHDL ως Target και Simulator language

1-4-3. Πατήστε είτε α)το **Πράσινο Κουμπί +** (πάνω και δεξιά), και μετά **Add Files είτε β) Add files (**κάτω και στο κέντρο) και επιλέξτε το αρχείο με τον κώδικα της οντότητας/αρχιτεκτονικής (στην περίπτωσή μας) **buzzer.vhd.** Πατάτε το **OK.** Ακολούθως επαναλαμβάνετε τη διαδικασία και προσθέτετε και το αρχείο της προσομοίωσης (στην περίπτωσή μας το **buzzer_tb.vhd**). Εάν δεν είναι ήδη επιλεγμένο, επιλέξτε **Copy sources into project.** Τέλος πατήστε το **OK.** Μετά πατήστε **Next** για να πάτε στη φόρμα *Add Constraints*, στην οποία απλά πατάτε το Next. Πλέον συνεχίζεται από το βήμα 1.1.10

1-5. Αρχεία κώδικα VHDL

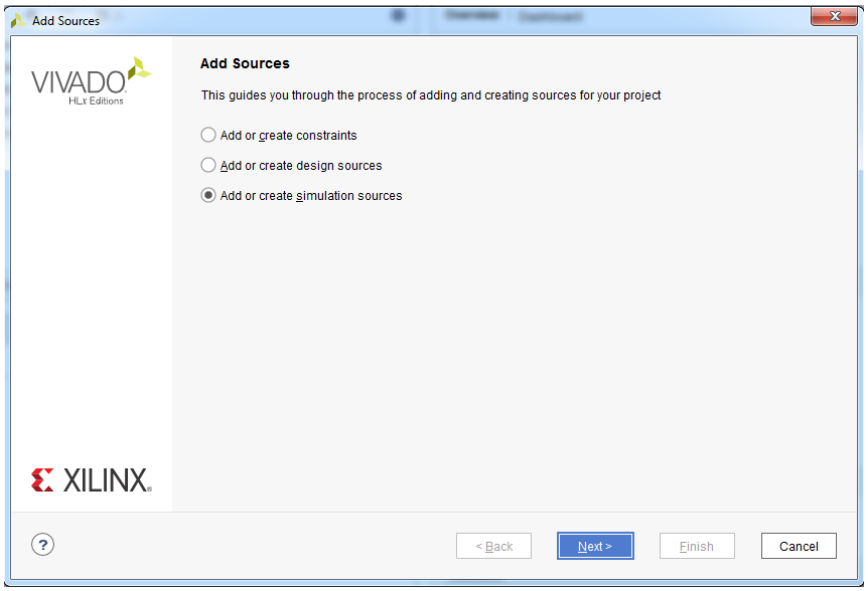
1-5-1. Χρησιμοποιήστε το Windows Explorer και κοιτάξτε στο φάκελο που ορίσατε το project (πχ d:\Logic\vivado_intro). Θα βρείτε ότι έχει δημιουργηθεί, μεταξύ άλλων και, ο φάκελος vivado_intro.srcs και το αρχείο vivado_intro.xpr (αρχείο Vivado: με διπλό κλικ θα ανοίξει μέσα στο Vivado). Δύο φάκελοι sim_1 και sources_1, έχουν δημιουργηθεί μέσα στον φάκελο vivado_intro.srcs που περιέχουν τα αρχεία buzzer_tb.vhd και buzzer.vhd αντίστοιχα.



Εικόνα 8. Επιλογή της VHDL ως Target και Simulator language

1-5-2. Μπορείτε κάθε στιγμή να μεταβείτε στο παράθυρο Sources μέσω του μενού Windows->Sources.

1-5-3. Μπορείτε να προσθέσετε ένα νέο αρχείο στο project πέραν όσων αναφέρθηκαν στις παραγράφους 1.2 και 1.3 και από το παράθυρο Flow Manager->Project Manager->Add Sources, οπότε και οδηγείστε στην ακόλουθη οθόνη όπου επιλέγεται το είδος του αρχείου που θέλετε.



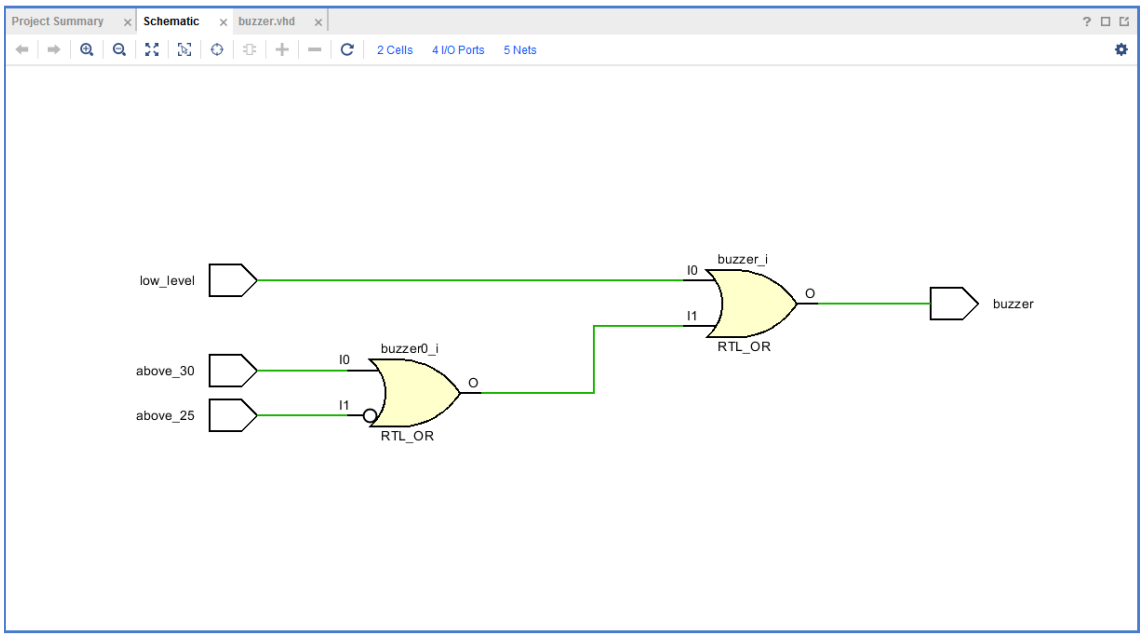
Εικόνα 8. Εισαγωγή αρχείου VHDL στο VIVADO

RTL Ανάλυση Βήμα 2

2-1. Εκτέλεση ανάλυσης RTL στον πηγαίο κώδικα.

2-1-1. Στο παράθυρο Flow_Navigator/PROJECT MANAGER υπάρχει η επιλογή RTL_ANALYSIS. Επεκτείνετε το *Open Elaborated Design* του *RTL Analysis* task και κάνετε κλικ στο **Schematic**.

Το μοντέλο του σχεδίου θα γίνει elaborate και μια παρουσίαση του σχεδίου με λογικές πύλες θα εμφανιστεί στο παράθυρο schematic. (Πατάμε OK αν χρεαστεί). Βλέπετε σχηματικά το κύκλωμα που ισοδυναμεί με το κύκλωμα που έχετε περιγράψει στο αρχείο της οντότητας/αρχιτεκτονικής (buzzer.vhd).



Εικόνα 9. Το σχηματικό διάγραμμα του σχεδίου

Προσομοίωση του Σχεδίου με τον Vivado Simulator Βήμα 3

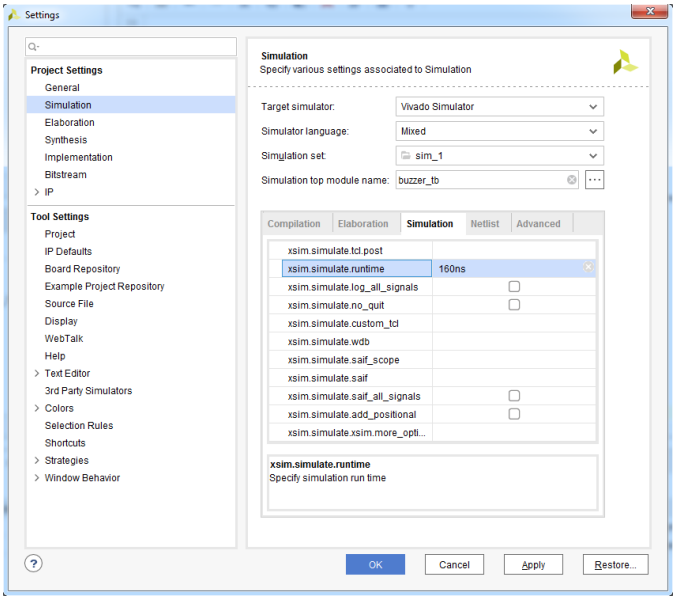
Δοθέντος ότι έχετε δημιουργήσει ή προσθέσει τον κώδικα VHDL για την προσομοίωση από το Βήμα 1, προχωράμε στη χρήση του Vivado Simulator.

3-1. Προσομοίωση του ψηφιακού κυκλώματος χρησιμοποιώντας το Vivado simulator.

3-1-1. Επιλέξτε από το μενού **Flow Navigator**-> **Settings**-> **Simulation Settings (**ή από το πλαίσιο ***Flow Navigator***-> ***Project Manager***->***Settings***) tasks του παραθύρου *Flow Navigator*.

Κάτω από το **Project Settings -> Simulation**

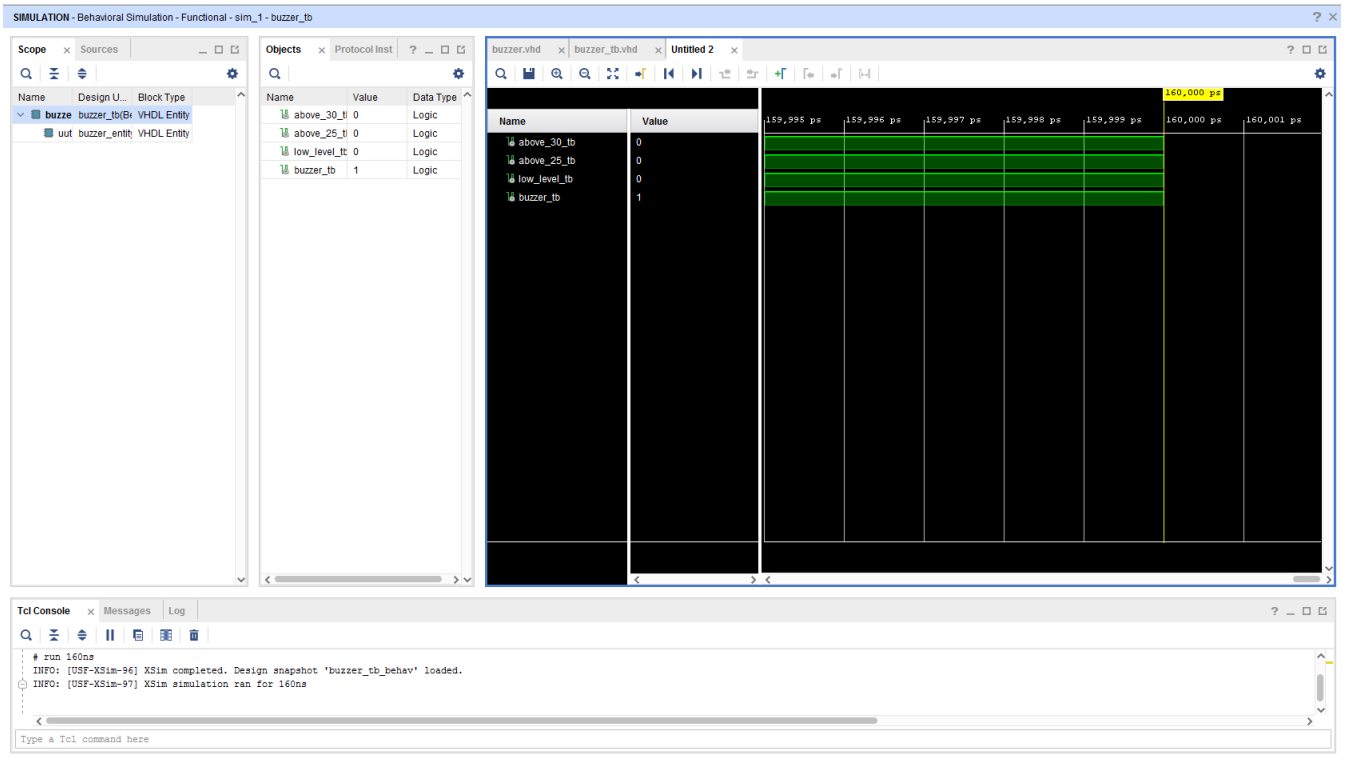
3-1-2. Επιλέξτε την καρτέλα (tab) **Simulation**, θέστε το **Simulation Run Time (xsim.simulate.runtime)** στα 160ns. Παρατηρείστε ότι στο κάτω πλαίσιο του παραθύρου υπάρχει επεξήγηση για τη σημασία της τιμής που δώσαμε στη συγκεκριμένη ρύθμιση του project. Ο χρόνος του simulation εξαρτάται από τον αντίστοιχο κώδικα. Πατήστε **Apply + OK**.



Εικόνα 10. Ορισμός της διάρκειας της προσομοίωσης (simulation run time)

3-1-3. Πατήστε στο **Flow Navigator -> Simulation-> Run Simulation > Run Behavioral Simulation**.

Το testbench και τα αρχεία του πηγαίου κώδικα θα γίνουν compiled και θα τρέξει το Vivado simulator (εφόσον βέβαια δεν υπάρχουν σφάλματα). Θα δείτε την έξοδο του simulator που είναι παρόμοια με την παρακάτω (είστε πλέον σε περιβάλλον simulation).



Εικόνα 11. Έξοδος του Simulator

Θα δείτε τέσσερα κύρια παράθυρα: (i) *Scope,* όπου παρουσιάζεται η ιεραρχία του testbench καθώς και τα instantiations (π.χ. της οντότητας που θέλουμε να προσομοιώσουμε), (ii) *Objects,* όπου εμφανίζονται τα top-level σήματα της ιεραρχίας. Είναι τα εσωτερικά σήματα της οντότητας προσομοίωσης buzzer_tb, που αντιστοιχούν στις εισόδους και εξόδους της οντότητας που προσομοιώνουμε. Προσοχή ότι εάν υπάρχουν και επιπλέον εσωτερικά σήματα θα εμφανιστούν και αυτά, (iii) το παράθυρο με τις κυματομορφές όπου βλέπουμε τη μεταβολή των τιμών των σημάτων, και (iv) *Tcl Console* (κάτω από την κυματομορφή), όπου εμφανίζονται οι ενέργειες που έγιναν κατά το simulation (παρατηρήστε ότι υπάρχουν και άλλα tab: Messages και Log).

Πάνω από το παράθυρο με τις κυματομορφές, θα δείτε διάφορα εικονίδια/κουμπιά που μπορούν να χρησιμοποιηθούν για συγκεκριμένο σκοπό. Αν αφήσετε το δείκτη του ποντικιού πάνω από κάθε εικονίδιο θα εμφανιστεί η π

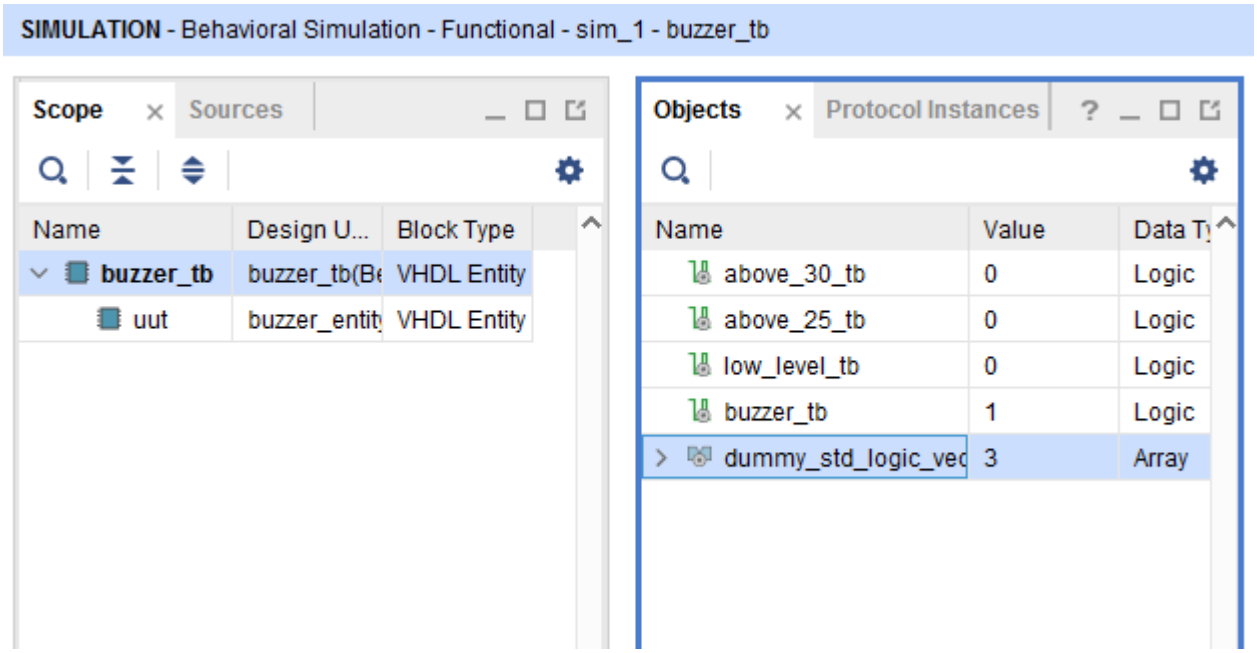
3-1-4. Πατήστε το κουμπί *Zoom Fit* ( ) για να δείτε όλη την κυματομορφή. Παρατηρήστε ότι η έξοδος μεταβάλλεται όταν μεταβάλλεται η είσοδος. Μπορείτε επίσης να ανοίξετε τις κυματομορφές σε ένα νέο μεγάλο παράθυρο πατώντας το κουμπί Float που βρίσκεται στο πάνω δεξιά μέρος. Για να επαναφέρετε το floating παράθυρο, απλά πατήστε στο κουμπί Dock Window.

Εικόνα 12. Κουμπί Float

Εικόνα 13. Κουμπί Dock Window

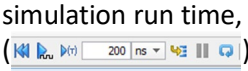
3-3. Προσθέστε περισσότερα σήματα για να παρακολουθήσετε τα σήματα χαμηλότερου επιπέδου και για να συνεχίσετε την προσομοίωση για άλλα 200 ns.

3-3-1. Στο παράθυρο *Scopes* κάντε κλικ στο buzzer_tb. Παρατηρήστε ότι εμφανίζεται στο παράθυρο Objects και το εσωτερικό σήμα dummy_std_logic_vector που είναι δηλωμένο στην οντότητα της προσομοίωσης.

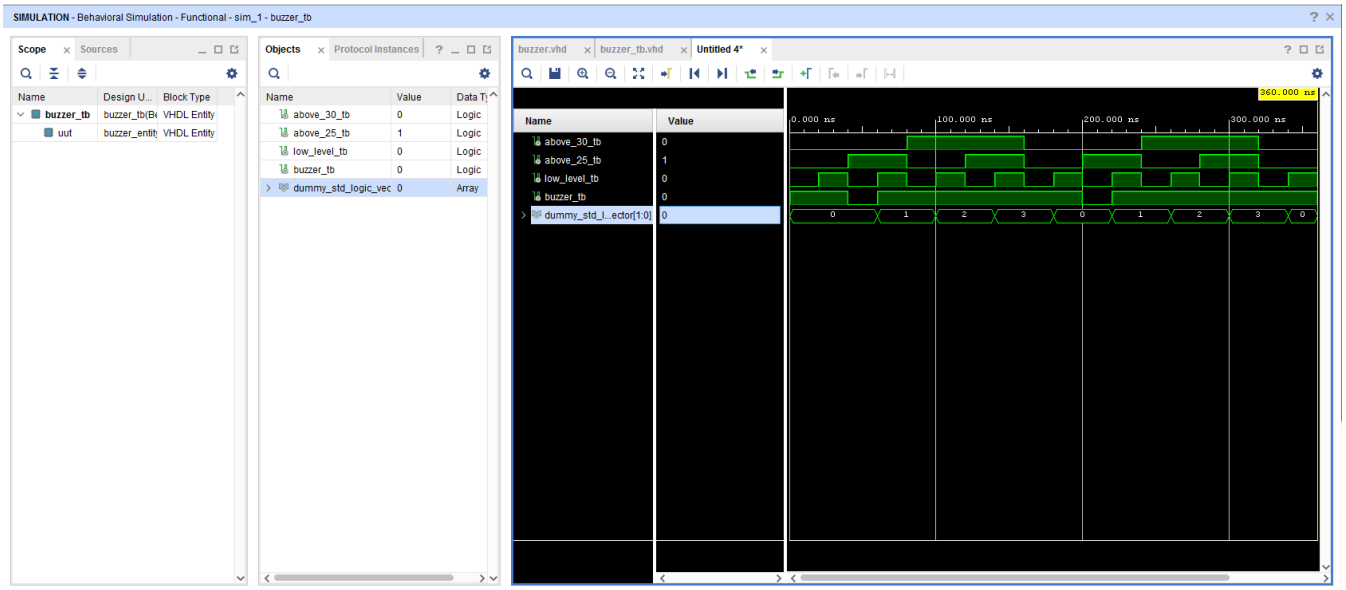


Εικόνα 14. Επιλογή σημάτων χαμηλότερου επιπέδου

3-3-2. Επιλέξτε το σήμα **dummy_std_logic_vector** και κάντε drag στο παράθυρο με τις κυματομορφές για να το παρακολουθήσετε. Εναλλακτικά κάνετε δεξί κλικ ->Add to wave window.

3-3-3. Στο ribbon bar του παραθύρου με τα εργαλεία της προσομοίωσης πληκτρολογήστε 200 στο πεδίο simulation run time, πατήστε click στο drop-down κουμπί του πεδίου με τις μονάδες και επιλέξτε ns () εφόσον θέλουμε να τρέξει η προσομοίωση για άλλα 200 ns (συνολικά 360ns) και πατήστε το κουμπί () button. Η προσομοίωση θα τρέξει για επιπλέον 200 ns.

3-3-4. Πατήστε στο κουμπί *Zoom Fit* και παρατηρείστε την έξοδο.



Εικόνα 15. Εκτέλεση προσομοίωσης για επιπλέον 200 ns

Μπορείτε να παρατηρήσετε το παράθυρο Tcl Console για να δείτε το μήνυμα που εμφανίζει.

3-4. Αλλάξτε το format της παρουσίασης εάν το επιθυμείτε

Παρατηρήστε ότι στην υπάρχουσα κυματομορφή δεν φαίνονται οι τιμές των σημάτων std_logic καθώς προκύπτουν από την κυματομορφή, αντίθετα με το σήμα **dummy_std_logic_vector**. Δίπλα στο όνομα κάθε σήματος υπάρχει η τιμή του σε κάθε δεδομένη χρονική στιγμή της προσομοίωσης. Κάνοντας κλικ σε διάφορα σημεία μέσα στην κυματομορφή παρατηρούμε ότι αλλάζουν και οι τιμές των σημάτων στη στήλη Value.

Πατώντας δεξί click σε κάποιο σήμα (στην περίπτωσή μας μόνο στο **dummy_std_logic_vector)**, επιλέξτε *Radix*, και στη συνέχεια επιλέξτε *Binary* για να βλέπετε τις εισόδους στο δυαδικό ή Hexadecimal για το δεκεξαδικό (υπάρχουν επιλογές για signed/unsigned αναπαραστάσεις, κλπ). Μπορείτε να επιλέξετε ένα μόνο σήμα ή πολλά σήματα μαζί και να κάνετε την αλλαγή της αναπαράστασης. Προσοχή πάντα στην περίπτωση που το σήμα είναι τύπου std_logic, γιατί τότε η επιλογή Radix δεν είναι διαθέσιμη.

3-5. Ολοκλήρωση διαδικασίας simulation

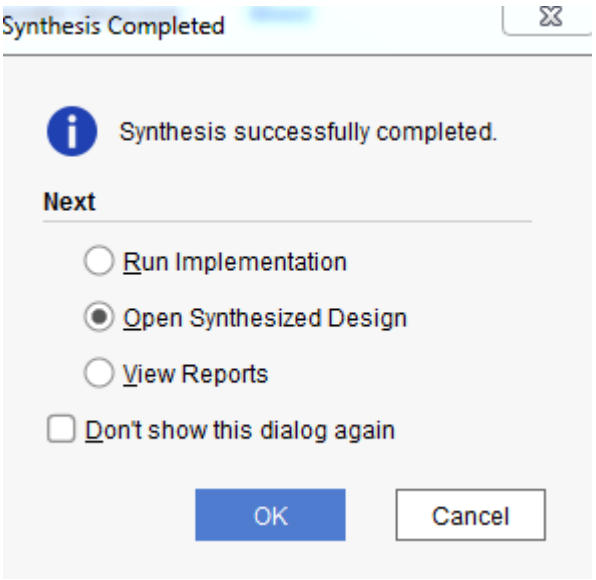
Κλείστε τον simulator επιλέγοντας **File > Close Simulation**. Πατήστε OK και στην συνέχεια πατήστε Discard για να κλείσει χωρίς να σώσετε την κυματομορφή.

Σύνθεση του Σχεδίου Βήμα 4

4-1. Κάντε σύνθεση στο σχέδιο με το εργαλείο σύνθεσης του Vivado και αναλύστε την έξοδο Project Summary

4-1-1. Πατήστε στο **Run Synthesis** που βρίσκεται στα *Synthesis* tasks του παραθύρου *Flow Navigator*. Αν υπάρξει ερώτημα σχετικό με "*Launch runs on local host*" πατάμε OK. (το ίδιο αν ξαναπροκύψει το ίδιο ερώτημα). Δίπλα από το **Number of jobs** μπορείτε να επιλέξετε το πόσα threads θα χρησιμοποιηθούν για την εκτέλεση της εργασίας (κατά μέγιστο 8). Επιλέξτε ανάλογα με τον επεξεργαστή του υπολογιστή σας.

Η διαδικασία της σύνθεσης θα εκτελεστεί στο αρχείο buzzer.vhd (και σε όλα τα αρχεία της ιεραρχίας του εφόσον υπάρχουν, δηλαδή τα component που ίσως υπάρχουν στην αρχιτεκτονική). Όταν η διαδικασία ολοκληρωθεί (η διάρκεια ποικίλει ανάλογα με την ισχύ του υπολογιστή σας), θα εμφανιστεί ένα κουτί διαλόγου *Synthesis Successfully Completed* dialog box με τρεις επιλογές.



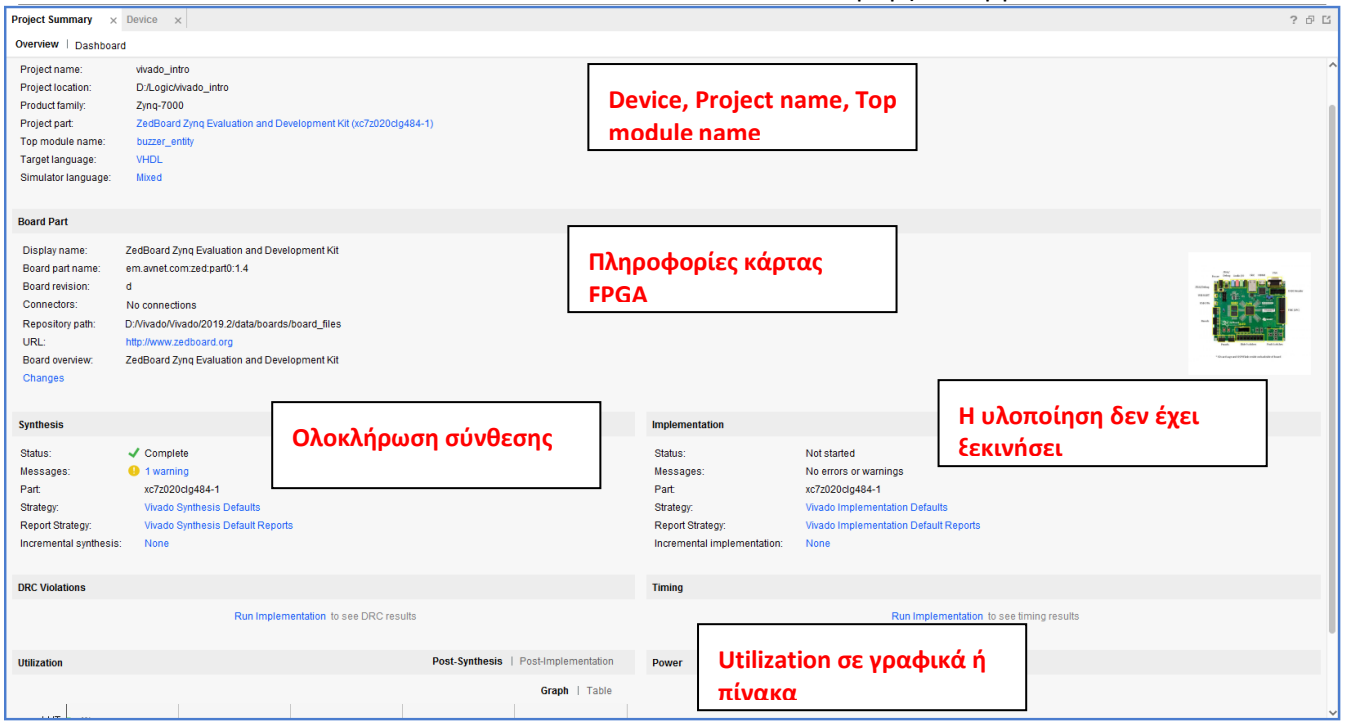
Εικόνα 16. Ολοκλήρωση της Σύνθεσης

4-1-2. Επιλέξτε την επιλογή *Open Synthesized Design* και πατήστε **OK** εφόσον επιθυμούμε να δούμε την έξοδο της σύνθεσης πριν προχωρήσουμε στο στάδιο της υλοποίησης (implementation).

Πατήστε **Yes** για να κλείσετε το elaborated design εάν εμφανιστεί το παράθυρο διαλόγου.

4-1-3. Επιλέξτε την καρτέλα **Project Summary** και μελετήστε τα διάφορα παράθυρα.

Εάν δεν βλέπετε την καρτέλα Project Summary τότε επιλέξτε **Layout > Default Layout,** ή πατήστε στο εικονίδιο **Project Summary**  .



Εικόνα 17. Project Summary για τα αποτελέσματα της σύνθεσης

Πατήστε στα διάφορα links για να δείτε τι πληροφορία παρέχουν και ποια επιτρέπουν να αλλάξετε τις ρυθμίσεις της σύνθεσης.

4-1-4. Πατήστε στην καρτέλα **Table** που βρίσκεται στην καρτέλα **Project Summary**. (Η καρτέλα *Table* είναι στο παράθυρο *Project Summary*, tab *Overview*, κάτω μέρος οθόνης, περιοχή *Utilization*, tab *Table*).

Παρατηρήστε ότι κατ' εκτίμηση έχει χρησιμοποιηθεί 1 LUT και 4 IOs (3 είσοδοι και 1 έξοδος).

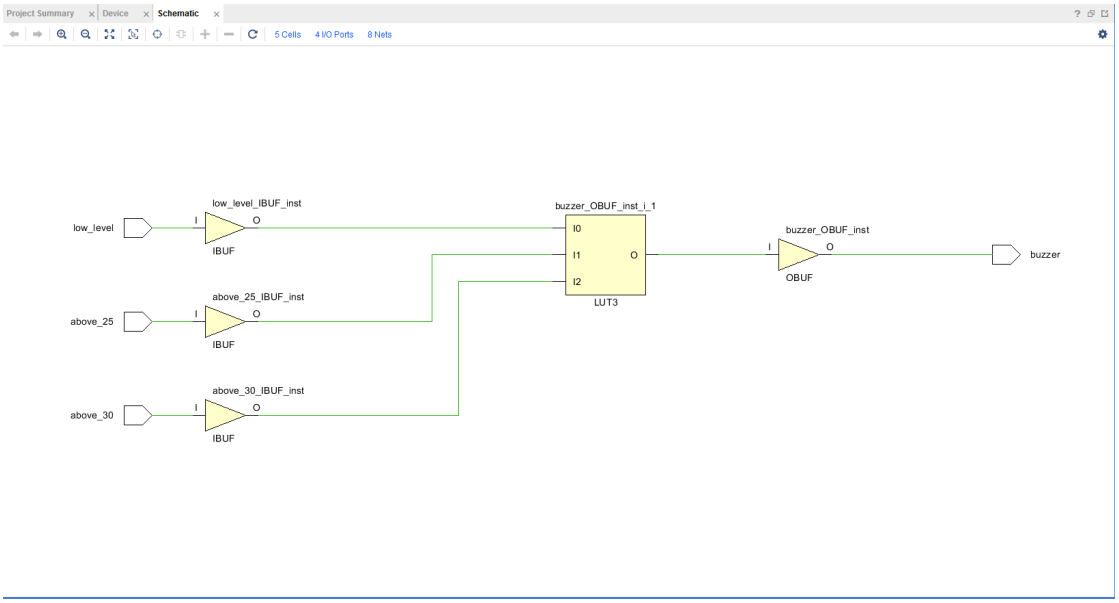
Utilization Post-Synthesis | Post-Implementation

Graph | **Table**

Resource	Estimation	Available	Utilization %
LUT	1	53200	0.01
IO	4	200	2.00

Εικόνα 18. Εκτίμηση χρήσης πόρων για το FPGA του ZedBoard

4-1-5. Στο *Flow Navigator*, μέσα στο *Synthesis* (επεκτείνετε το *Open Synthesized Design* εάν είναι απαραίτητο), πατήστε το **Schematic** για να δείτε το Synthesized design σε σχηματική αναπαράσταση.



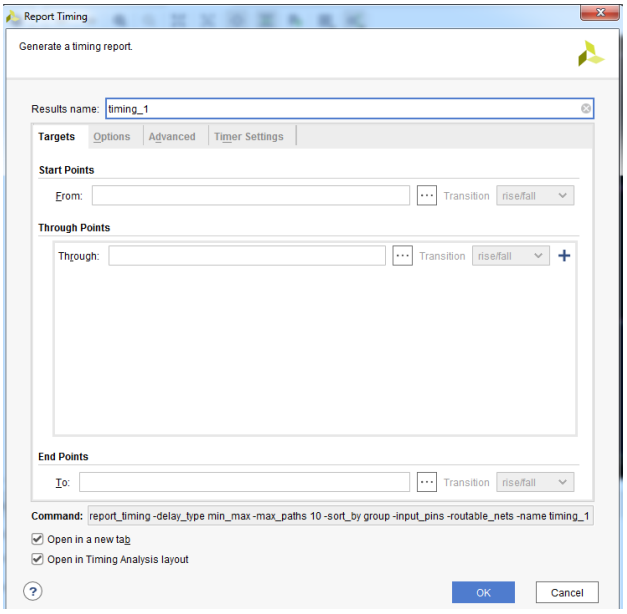
Εικόνα 19. Σχηματική αναπαράσταση του synthesized design

Παρατηρείστε ότι έχουν αυτόματα προστεθεί IBUFs και OBUFs primitives στο σχέδιο αφού οι είσοδοι και οι έξοδοι από το FPGA είναι buffered. Οι λογικές πύλες έχουν υλοποιηθεί σε LUTs (LUT3). Οι δύο λογικές πύλες της ανάλυσης RTL έχουν γίνει mapped σε ένα LUT στην σύνθεση.

Χρησιμοποιώντας τον Windows Explorer, επαληθεύστε ότι έχει δημιουργηθεί ο φάκελος vivado_intro.srcs μέσα στον φάκελο vivado_intro. Μέσα στον φάκελο runs, έχει δημιουργηθεί ο φάκελος synth_1 που περιέχει διάφορα αρχεία σχετικά με την σύνθεση.

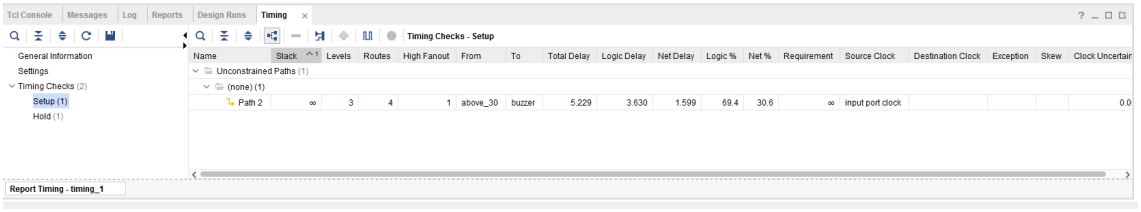
4-2. Χρονική ανάλυση του κυκλώματος κατά τη Σύνθεση

4-2-1. Δοθέντος ότι έχει γίνει πρώτα το βήμα 4.1.2, επιλέγουμε από τη γραμμή του μενού τις επιλογές Reports->Timing->Report Timing. Εμφανίζεται η ακόλουθη οθόνη:



Εικόνα 20. Report Timing

Αν θέλετε μπορείτε να δώσετε ένα όνομα στο Report και πατάτε στο OK. Στο κάτω μέρος της οθόνης εμφανίζεται το report.

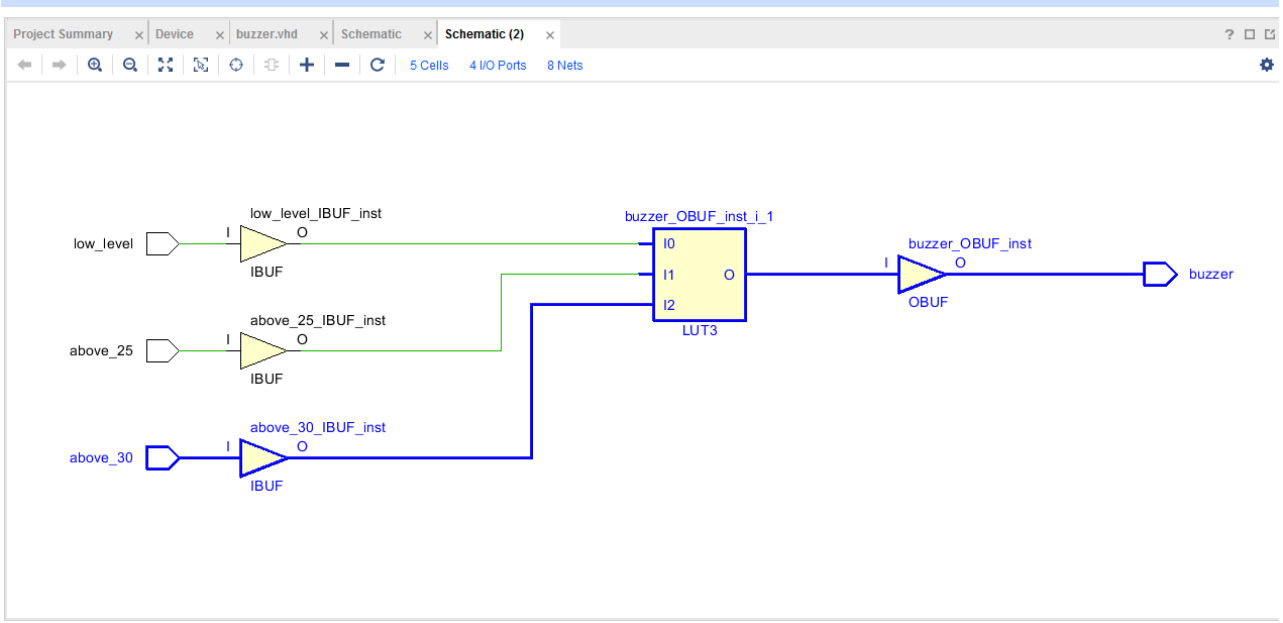


Εικόνα 21. Οι πληροφορίες του Report Timing

Στο κάτω μέρος φαίνεται το όνομα του report (timing_1), ενώ αριστερά και κάτω από την επιλογή *Timing Checks*, υπάρχουν δύο επιλογές. Με την επιλογή **Setup** εμφανίζονται τα μονοπάτια του κυκλώματος που έχουμε φτιάξει με την μεγαλύτερη καθυστέρηση. Το μονοπάτι (**κρίσιμο μονοπάτι**) με την μεγαλύτερη καθυστέρηση (**καθυστέρηση διάδοσης**) είναι αυτό που εμφανίζεται και πρώτο (στην περίπτωσή μας το μονοπάτι είναι μόνο ένα, το **Path 2**). Στη γραμμή δίπλα από το όνομα του μονοπατιού εμφανίζονται και άλλες πληροφορίες, ενδεικτικά:

- **Levels**: Τα ψηφιακά στοιχεία που περιλαμβάνει το μονοπάτι
- **From** – To: Τα στοιχεία αρχής και τέλους του μονοπατιού
- **Total Delay**: Η καθυστέρηση διάδοσης, όπου ο χρόνος του **Loigc Delay** είναι ο χρόνος καθυστέρησης στα ψηφιακά στοιχεία και **Net Delay** ο χρόνος διάδοσης στα σύρματα που συνδέουν τα ψηφιακά στοιχεία.

Αντίστοιχα με την επιλογή **Hold** εμφανίζονται τα μονοπάτια (**σύντομη διαδρομή**) με τη μικρότερη καθυστέρηση (**καθυστέρηση μόλυνσης**) και εμφανίζονται οι αντίστοιχες πληροφορίες.
Εάν θέλουμε να δούμε οπτικά το μονοπάτι (π.χ. το *Path 2*) κάνουμε πρώτα κλικ στο *Flow Navigator->Synthesis->Schematic* και κατόπιν κλικ πάνω στο όνομα του path (*Path 2*).



Εικόνα 22. Το μονοπάτι με την μεγαλύτερη καθυστέρηση (κρίσιμο μονοπάτι).

Για να δούμε επιπλέον πληροφορίες για το μονοπάτι κάνουμε διπλό κλικ στο όνομά του στο report όπου εμφανίζεται το ακόλουθο report:

Project Summary | Device | buzzer.vhd | Schematic | Schematic (2) | Path 2 - timing_1

Summary

Name	Path 2
Slack	∞ns
Source	above_30 (input port)
Destination	buzzer (output port)
Path Group	(none)
Path Type	Max at Slow Process Corner
Requirement	∞ns
Data P...Delay	5.229ns (logic 3.630ns (69.413%) route 1.599ns (30.587%))
Logic Levels	3 (IBUF=1 LUT3=1 OBUF=1)

Data Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
	(r) 0.000	0.000		above_30
net (fo=0)	0.000	0.000		above_30
				above_30_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(r) 0.921	0.921		above_30_IBUF_inst/O
net (fo=1, unplaced)	0.800	1.721		above_30_IBUF
				buzzer_OBUF_inst_i_1/I2
LUT3 (Pro...t3_I2_O)	(r) 0.124	1.845		buzzer_OBUF_inst_i_1/O
net (fo=1, unplaced)	0.800	2.645		buzzer_OBUF
				buzzer_OBUF_inst/I
OBUF (Pr...buf_I_O)	(r) 2.584	5.229		buzzer_OBUF_inst/O
net (fo=0)	0.000	5.229		buzzer
				buzzer

Εικόνα 23. Το μονοπάτι με την μεγαλύτερη καθυστέρηση (κρίσιμο μονοπάτι).

Στην ενότητα **Summary** περιλαμβάνει αρκετές πληροφορίες που υπάρχουν και στην γραμμή του αρχικού report (Εικόνα 21), αλλά έχει και την ενότητα **Data Path,** όπου περιγράφεται αναλυτικά η διαδρομή του μονοπατιού με πλήρη ανάλυση της καθυστέρησης σε κάθε ψηφιακό στοιχείο όσο και της καθυστέρησης στα σύρματα που τα ενώνουν.

Προσομοίωση Χρονική μετά τη Σύνθεση του Σχεδίου Βήμα 5

Εκτελέστε ένα post synthesis timing simulation.

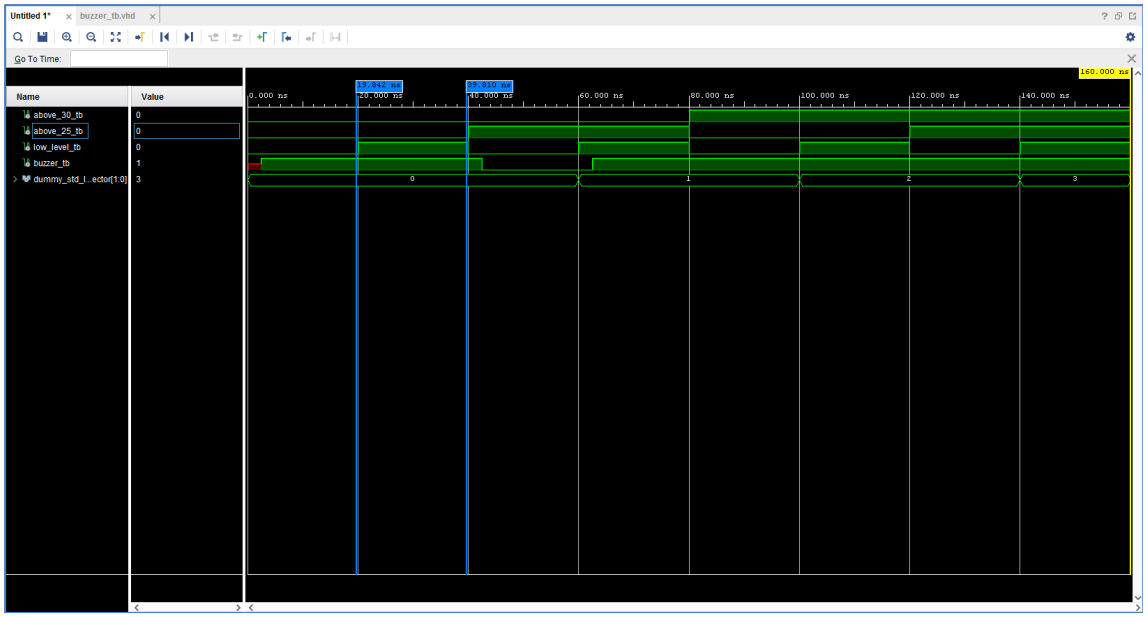
5-1-1. Από το παράθυρο του *Flow Navigator* επιλέξτε *Simulation* και ύστερα **Run Simulation > Run Post-Synthesis Timing Simulation.** Ο Vivado simulator θα ξεκινήσει χρησιμοποιώντας το Synthesized design και το **buzzer_tb** ως top-level module.

Χρησιμοποιώντας τον Windows Explorer, επαληθεύστε ότι ο φάκελος **timing** έχει δημιουργηθεί μέσα στο φάκελο **vivado_intro.sim > sim_1 > synth**. Ο φάκελος **timing** περιέχει τα αρχεία που παρήχθησαν για να εκτελεστεί το timing simulation.

5-1-2. Πατήστε στο κουμπί **Zoom Fit** για να δείτε το παράθυρο με τις κυματομορφές από 0 έως τα 160 ns.

5-1-3. Πατήστε δεξί click στα 20 ns και επιλέξτε **Markers > Add Marker**. Εναλλακτικά, μπορείτε να προσθέσετε ένα marker με click στο κουμπί Add Marker ( ).

5-1-4. Παρόμοια, πατήστε δεξί click για να προσθέσετε ένα marker γύρω στα 40 ns. Παρατηρήστε ότι αν πάτε το ποντίκι πάνω στο marker έχετε τη δυνατότητα να το μετακινήσετε.



Εικόνα 24. Οι κυματομορφές του timing simulation

Παρατηρήστε την πραγματική καθυστέρηση στην αλλαγή των εξόδων σε σχέση με το behavioral simulation. Η μεταβολή στο σήμα εξόδου (buzzer) γίνεται με καθυστέρηση περίπου 2,5 ns.

5-1-5. Κλείστε τον simulator επιλέγοντας **File > Close Simulation** χωρίς να σώσετε κάποιες αλλαγές.