4η Εργαστηριακή ΄Άσκηση

# Κοτσιμπού Φωτεινή ΑΜ 1059567 Σιγούρου ΄Άλκηστις-Αικατερίνη ΑΜ 1059661

28 Μαΐου 2021



# Εργασία στα πλαίσια του µαθήµατος Σχεδιασμός Συστημάτων VLSI

Τμήμα Μηχ. Ηλεκτρονικών Υπολογιστών και Πληροφορικής

# Περιεχόμενα

1. [Ενότητα Α](#_bookmark0) 1
   1. [Α1 - Σύνθεση και Αποτελέσµατα](#_bookmark1) 1

|  |  |  |  |
| --- | --- | --- | --- |
| [1.1.1](#_bookmark2) | [Πύλη Α - 8 bit](#_bookmark2) | . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 1 |
| [1.1.2](#_bookmark4) | [Πύλη Β - 16 bit](#_bookmark4) | . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 1 |
| [1.1.3](#_bookmark5) | [Πύλη Γ - 32 bit](#_bookmark5) | . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 1 |
| [1.1.4](#_bookmark8) | [Πύλη .6 - 64 bit](#_bookmark8) | . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 2 |

1. [Ενότητα Β](#_bookmark11) 3
   1. [Logic Unit](#_bookmark12) 3
   2. [Σύνθεση και Αποτελέσµατα](#_bookmark24) 9

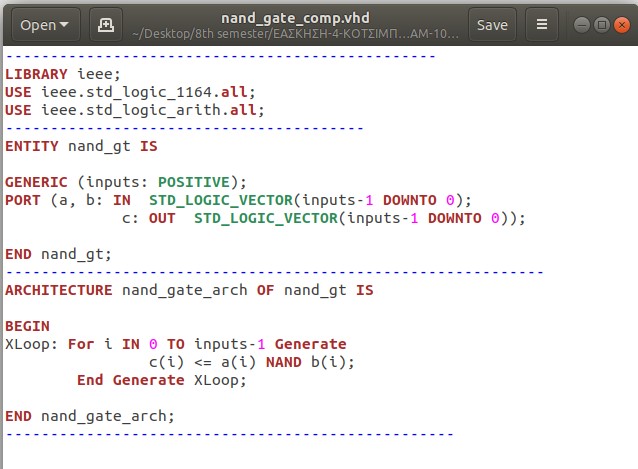
# Ενότητα Α

## Α1 - Σύνθεση και Αποτελέσ ατα

* + 1. Πύλη Α - 8 **bit**

Dς πύλη Α αποφασίσαμε να υλοποιήσουμε µία NAND των 8 bit . Για τη χρήση της στην ενότητα Β, κατασκευάσαμε 2 διαφορετικά αρχεία, ένα component Σχήμα [1](#_bookmark3) και ένα main Σχήμα [2.](#_bookmark6) Στη main δηλώσαµε µε τη χρήση του generic το πλάτος των λέξεως που θα δέχεται η συγκεκριµένη πύλη ως είσοδο και έξοδο, δηλαδή στη προκειµένη περίπτωση ¨8¨. Παράλληλα, στο αρχείο comp που στη συνέχεια θα χρησιµοποιηθεί για την ενότητα Β, µε τη χρήση ενός Loop εξασφαλίζουµε τη πράξη NAND για κάθε ένα από τα 8 bit της λέξης µας.

Επίσης το σχηµατικό σύνθεσης της πύλης παρουσιάζεται στο Σχήµα [3.](#_bookmark7)



Σχήµα 1: NAND Component code

* + 1. Πύλη Β - 16 **bit**

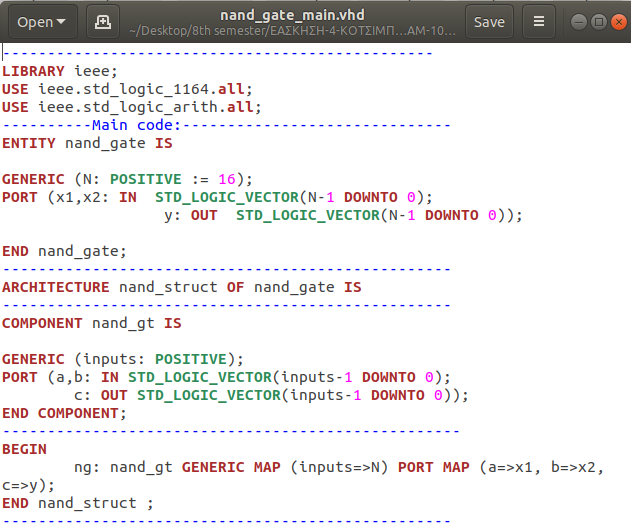
Για τη πύλη Β θελήσαµε να κατασκευάσουµε µία AND η οποία όπως ζητάει και η εκφώνηση, θα έχει πλάτος 16 bit . ΄Οπως και στη προηγούµενη πύλη, προχωρήσαµε στη δηµιουργία 2 ξεχωριστών αρχείων για να εκπληρώσουµε το σκοπό της ενότητας Β. Στο ένα αρχείο µε τη χρήση του generic δηλώσαµε το πλάτος -16 bit των εισόδων και της εξόδου, ενώ στο άλλο αρχείο µε ένα Loop πραγµατοποιήσαµε τη πράξη για κάθε ένα από τα bit .

Στο Σχήµα [4](#_bookmark9) βλέπουµε τον κώδικα για το component , στο Σχήµα [5](#_bookmark10) για την main συνάρτηση, καθώς και το σχηµατικό σύνθεσης της πύλης στο Σχήµα [6.](#_bookmark13)

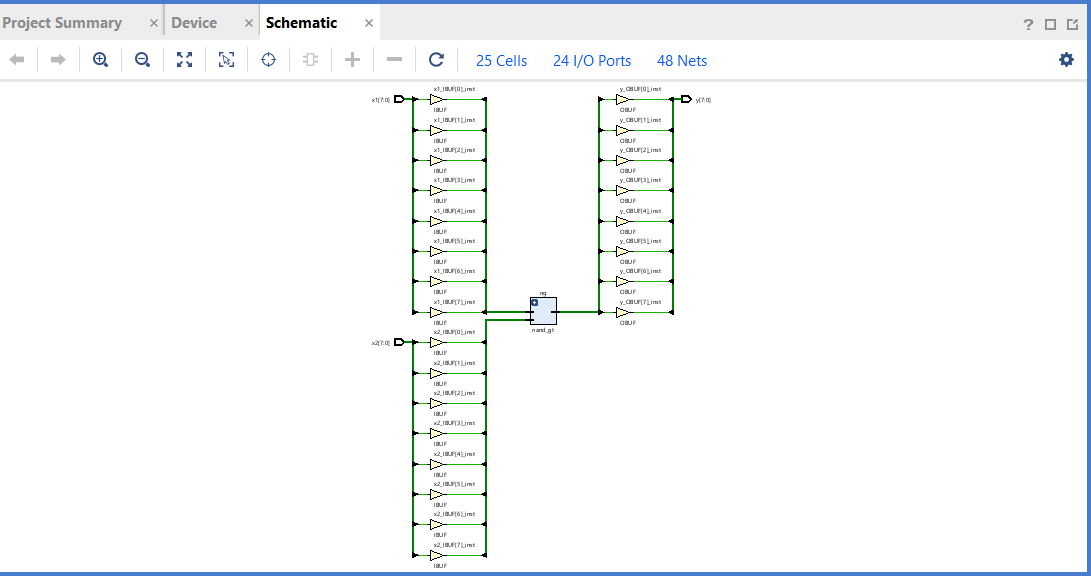
* + 1. Πύλη Γ - 32 **bit**

Στη Πύλη Γ αποφασίσαµε να κάνουµε µία πύλη OR ακολουθόντας ακριβώς τα ίδια βήµατα µε τα απο πάνω, προσαρµόζοντας φυσικά το generic και το Loop στα 32 bit .

Στο Σχήµα [7](#_bookmark14) βλέπουµε τον κώδικα για το component , στο Σχήµα [8](#_bookmark16) για την main συνάρτηση, καθώς και το σχηµατικό σύνθεσης της πύλης στο Σχήµα [9.](#_bookmark17)



Σχήµα 2: NAND Main code



Σχήµα 3: NAND Schematic

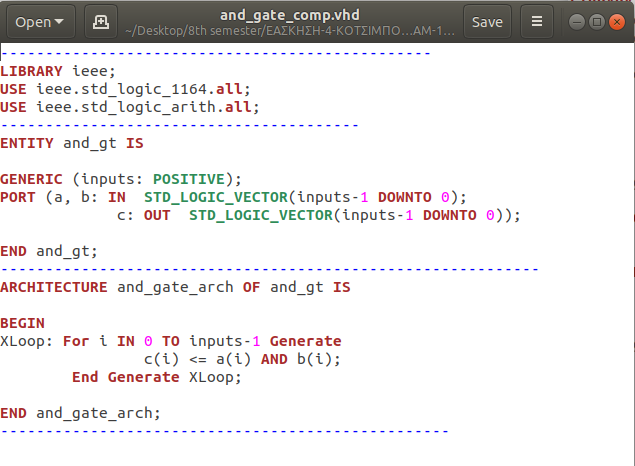
* + 1. Πύλη Δ - 64 **bit**

Τέλος για πύλη .6 καταλήξαµε να φτιάχνουµε µία πύλη XOR των 64 bit . Για τη δηµιουργία της ακολουθήσαµε τη παραπάνω διαδικασία µεταβάλλοντας των αριθµό των επαναλήψεων αλλά και του πλάτους, µε τη χρήση του generic στα 64 bit .

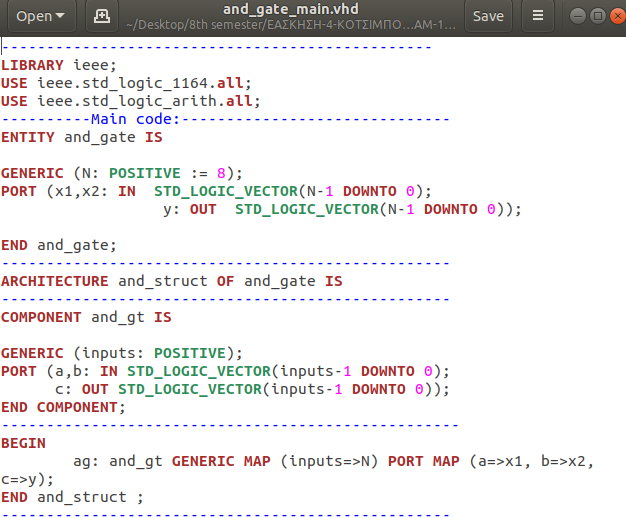
Στο Σχήµα [10](#_bookmark18) βλέπουµε τον κώδικα για το component , στο Σχήµα [11](#_bookmark19) για την main συνάρτηση, καθώς και το σχηµατικό σύνθεσης της πύλης στο Σχήµα [12.](#_bookmark20)

Στην συνέχεια εκτελούµε την σύνθεση για κάθε µία απο τις πύλες και λαµβάνουµε τα αποτελέσµατα που φαίνονται στον Πίνακα [1.](#_bookmark15)

Για λεπτοµερή αναφορά των αποτελεσµάτων της σύνθεσης, µπορείτε να µεταβείται στο spreadsheet µε όνοµα SyntesisReport.xlsx, που θα βρείτε στο συµπιεσµένο αρχείο που παραδώσαµε.



Σχήµα 4: AND Component code

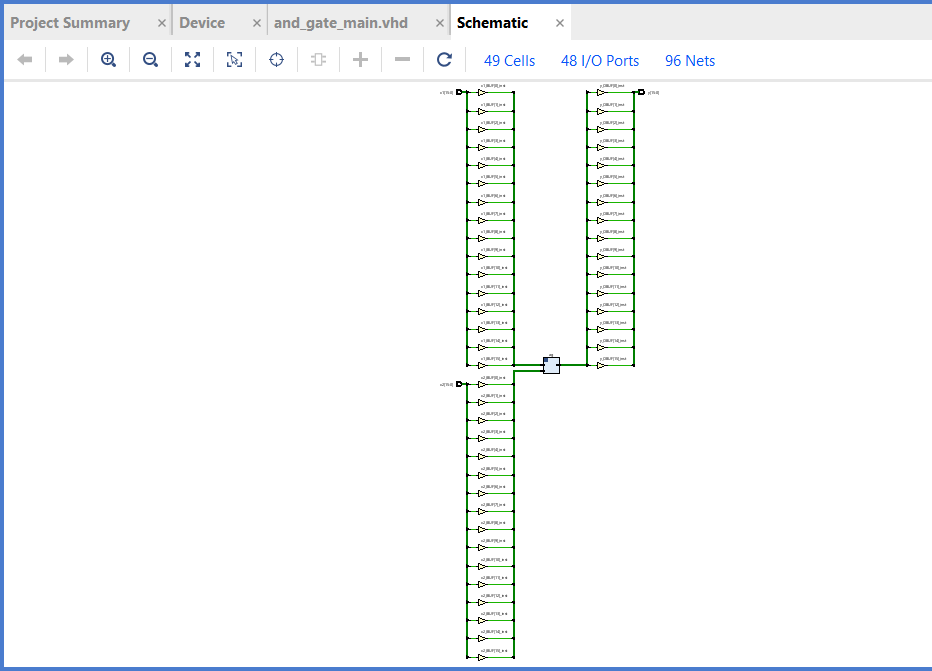


Σχήµα 5: AND Main code

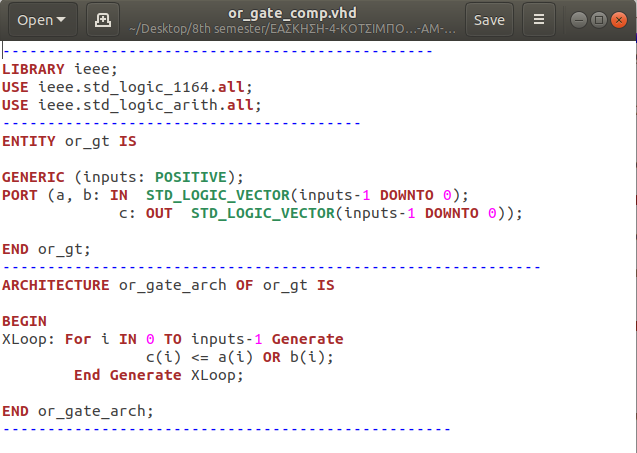
# Ενότητα Β

* 1. **Logic Unit**

Στον συγκεκριµένο κώδικα έχουµε αναλάβει τη δηµιουργία µίας λογικής µονάδας που θα εκτελεί τις 4 πράξεις που έχουµε ήδη επιλέξει στην ενότητα Α. Η Λογική Μονάδα µας θα παίρνει σαν είσοδο 2 εισόδους των 120 bit η καθεµία. Στη συνέχεια θα αναλαµβάνει να εκτελέσει κάθε πράξη µε τα bit είσόδου που απαιτούνται ανάλογα µε το πλάτος που έχουµε επιλέξει να της δώσουµε στην ενότητα Α. ΄Ετσι, στο µήκος των 120 bit , η



Σχήµα 6: AND Schematic

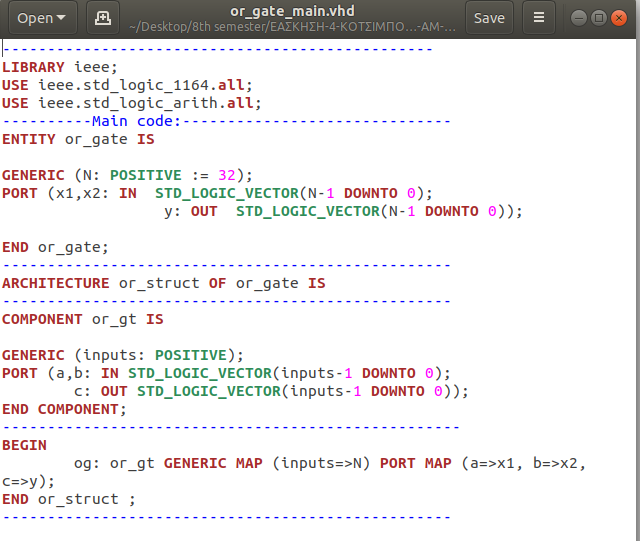


Σχήµα 7: OR Component code

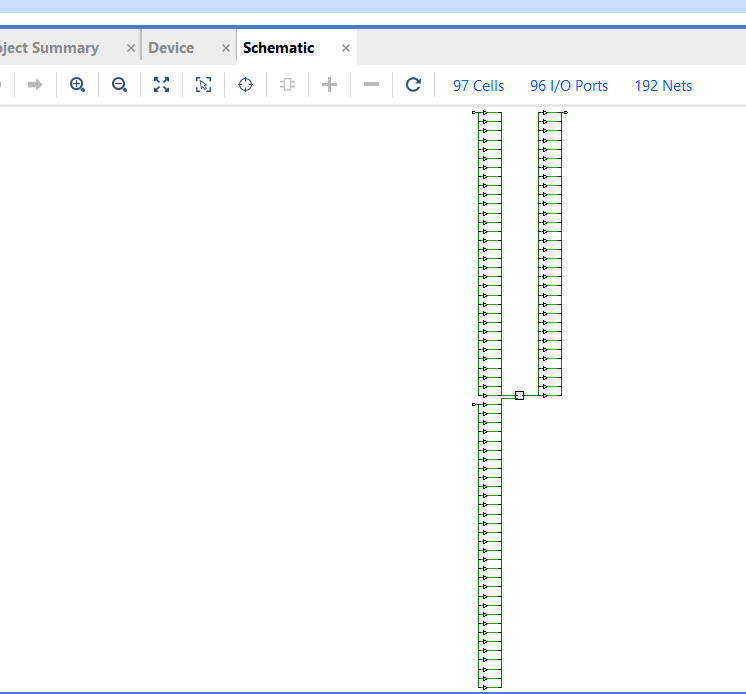
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| File Code | Part | % Used IOB | % Used LUT | Worst Delay | Total Power |
| nand gate.vhd | xc7k70tfbv484-2L | 8.42% | 0.02% | 4,063ns | 2.047W |
| and gate.vhd | xc7k70tfbv484-2L | 16.84% | 0.04% | 4,063ns | 1.014W |
| or gate.vhd | xc7k70tfbv484-2L | 33.68% | 0.08% | 4,063ns | 7.953W |
| xo gate.vhd | xc7k70tfbv484-2L | 67.37% | 0.16% | 4,063ns | 28.833W |

Πίνακας 1: Ενδεικτικά αποτελέσµατα Σύνθεσης

πράξη NAND θα πάρει τα πρώτα 8 bit , δηλαδή τα πρώτα 2 δεκαεξαδικά ψηφία και θα εκτελέσει τη πράξη της, δίνοντας µας τα πρώτα 8 bit της εξόδου µας (τα 2 δεξιότερα δεκαεξαδικά ψηφία). Συνεχίζουµε τις πράξεις µας µε τη πύλη AND των 16 bit η οποία θα κληθή να επιλέξει τα επόµενα 16 bit από κάθε είσοδο και να

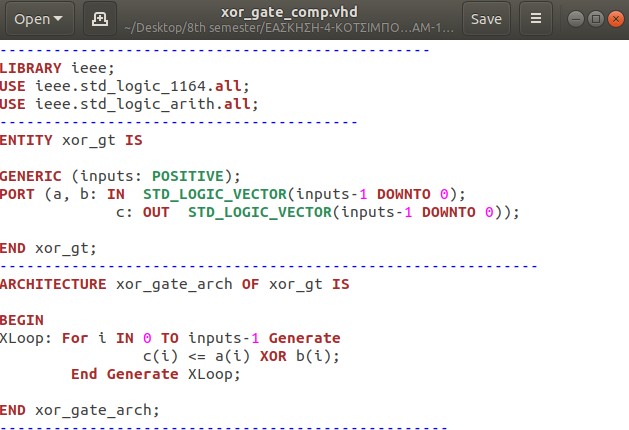


Σχήµα 8: OR Main code



Σχήµα 9: OR Schematic

µας επιστρέψει το αποτέλεσµα των 16 bit που θα τοποθετηθεί αµέσως αριστερότερα από τα ήδη υπάρχοντα 2 δεκαεξαδικά ψηφία που προέκυψαν από τη πύλη NAND . Η αµέσως επόµενη πράξη που εκτελείται είναι αυτή της OR που ακολουθεί το παράδειγµα των 2 προηγούµενων και αφου πάρει τα 32 bit που απαιτούνται για να γίνει η πράξη από κάθε είσοδο, τυπώνει το αποτέλεσµα των 32 bit της αµέσως πριν (αριστερότερα) από τα ήδη



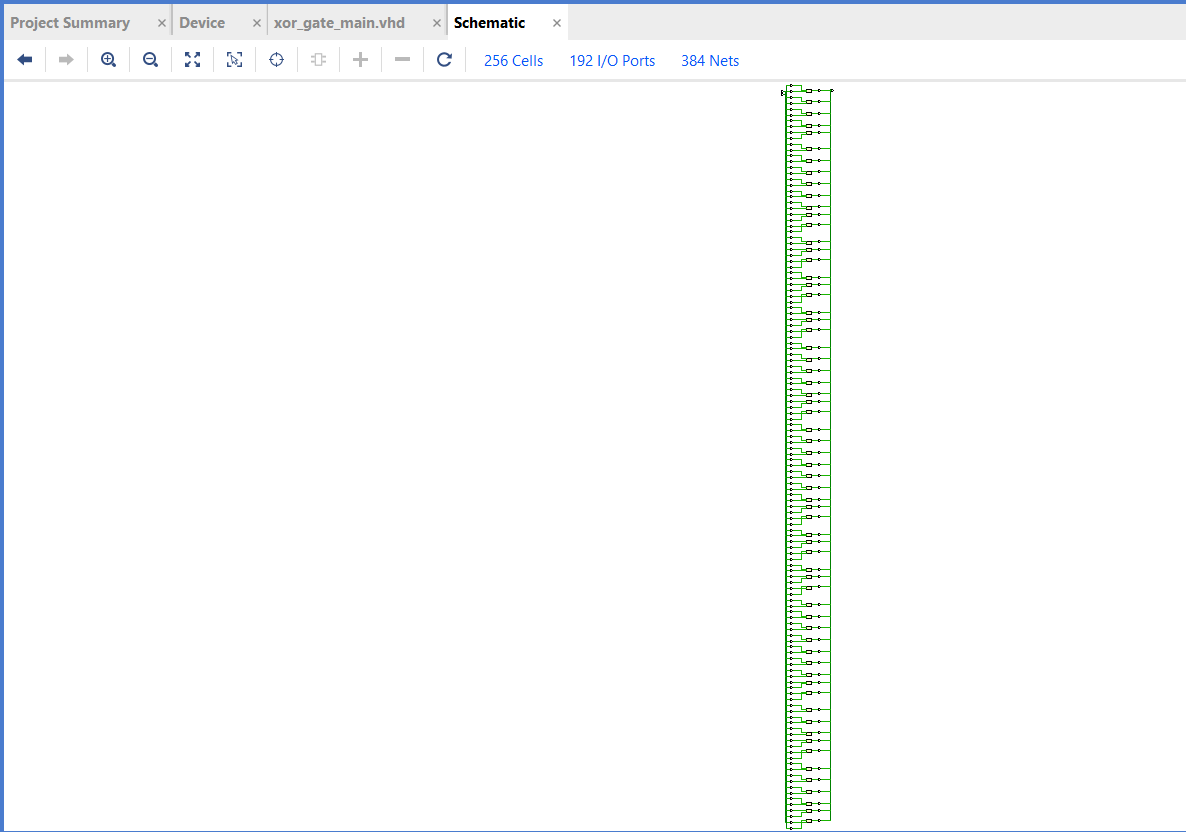
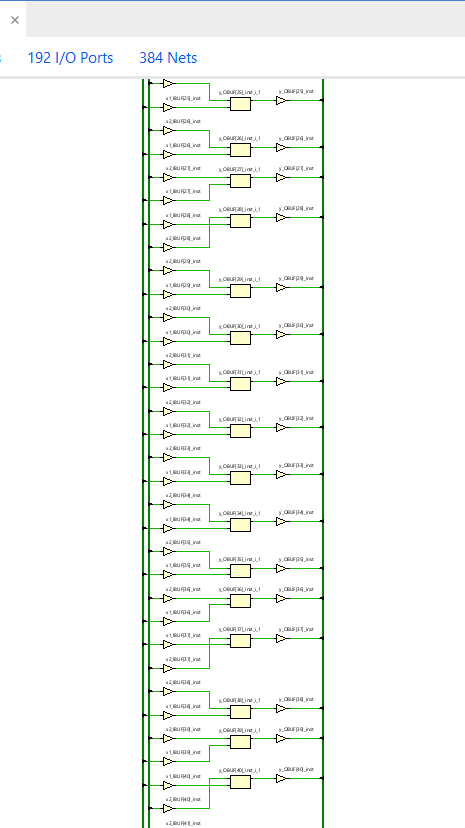
Σχήµα 10: XOR Component code



Σχήµα 11: XOR Main code

τυπωµένα 6 δεκαεξαδικά ψηφία των πράξεων NAND και AND . Τέλος, η µόνη πράξη που αποµένει είναι αυτή της XOR η οποία απαιτεί 64 bit για να πραγµατοποιηθεί και έτσι αφού επιλέξει τα 64 τελευταία δυαδικά ψηφία από κάθε είσοδο τυπώνει το αποτέλεσµά της, προσδίδοντας έτσι στην έξοδό µας τα τελευταία 64 bit που χρειάζεται για να ολοκληρωθούν και τα 120 bit εξόδου.

Παρατηρώντας τον κώδικα µπορούµε να διακρίνουµε τον τρόπο που εισάγαµε µέσα σε ένα ενιαίο αρχείο, ως components την κάθε πύλη που χρειάζεται η λογική µονάδα µας. Στην αρχή του αρχείου που περιέχει τον κώδικα για την υλοποίηση της λογικής µονάδας παρατηρούµε πως έχουµε γράψει τον κώδικα των componentsπου έχουµε ήδη αναφέρει πως κατασκευάσαµε στην ενότητα Α. Αυτό έγινε ώστε να διασφαλίσουµε τη λειτουργικότητα του κώδικα καθώς δεν µπορούµε να τρέξουµε πολλά διαφορετικά αρχεία µε κώδικες ταυτόχρονα. Με το

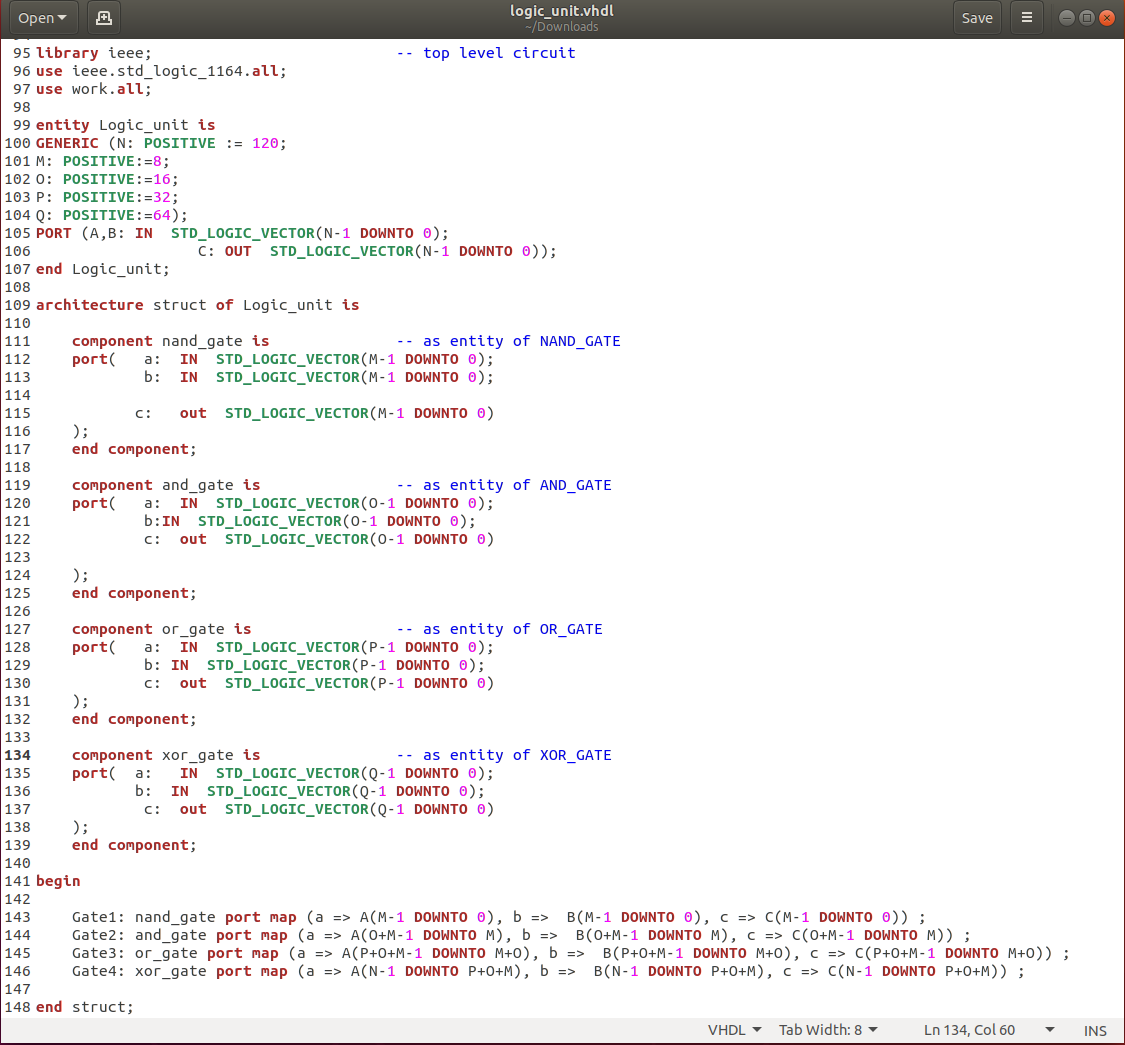
 

Σχήµα 12: XOR Schematic

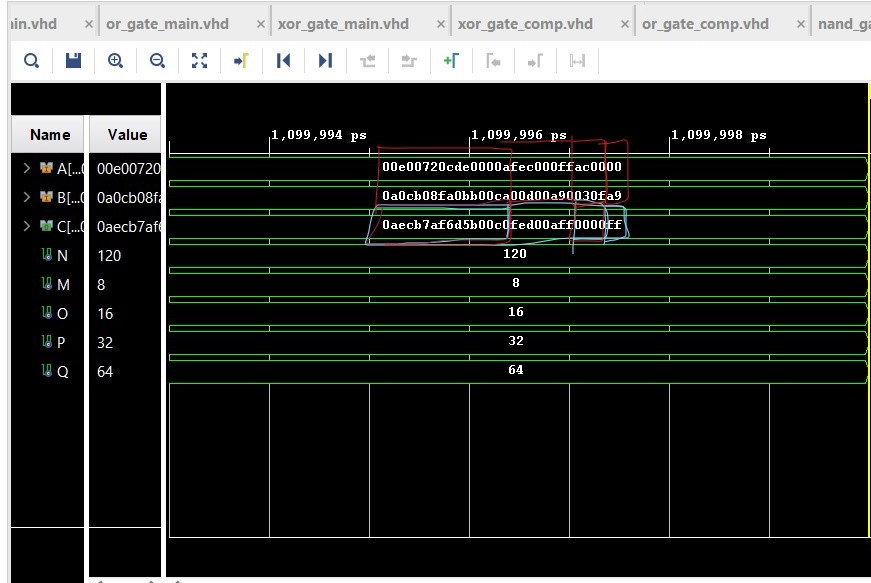
τρόπο αυτό δηλώνουµε εξάρχής τη κάθε πράξη που θα ζητήσουµε να πραγµατοποιηθεί στη συνέχεια και δεν συναντάµε σφάλµατα κατα τη διάρκεια του simulation και synthesis . Αφού αντιγράψουµε τον κώδικα του κάθε component , συνεχίζουµε µε τον κώδικα της λογικής µονάδας. Συγκεκριµένα, ορίζουµε 5 διαφορετικές µεταβλητές, η καθε µία ορίζοντας το εύρος πλάτους, σε διαδικά ψηφία, των πράξεων που περιλαµβάνει η λογική µονάδα. Στο τέλος του κώδικα γίνεται και η επιλογή των bit που θα χρησιµοποιήσει κάθε πράξη για τη εκτέλεσή της. Πιο ειδικά, δηλώνουµε πως η πράξη NAND θα πραγµατοποιηθεί πρώτη και θα διαβάσει τα bit από 7 εως 0 από τις εισόδους Α και Β αντίστοιχα ενώ θα εκτυπώσει τα bit 7 εως 0 του αποτελέσµατος. Συνεχίζουµε µε τα bit 23 εως 8 των εισόδων και της εξόδου που χρησιµοποιεί η πύλη AND και µε τα bit 55 εως 24 για τη πύλη OR . Τα τελευταία 64 bit της εξόδου προέρχονται από τη πράξη XOR ανάµεσα στα bit 119 εώς 56 των εισόδων Α και Β. Με τις παραπάνω εντολές ολοκληρώνεται η αρχιτεκτονική της Λογικής Μονάδας και το πρόγραµµα είναι έτοιµο για το simulation .

Η κεντρική δοµή του κώδικα της Logic Unit παρουσιάζεται στο Σχήµα [13.](#_bookmark21)

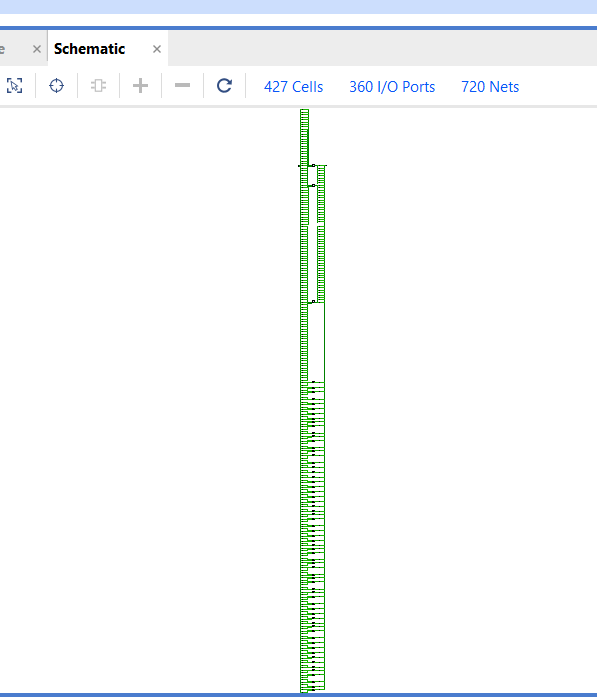
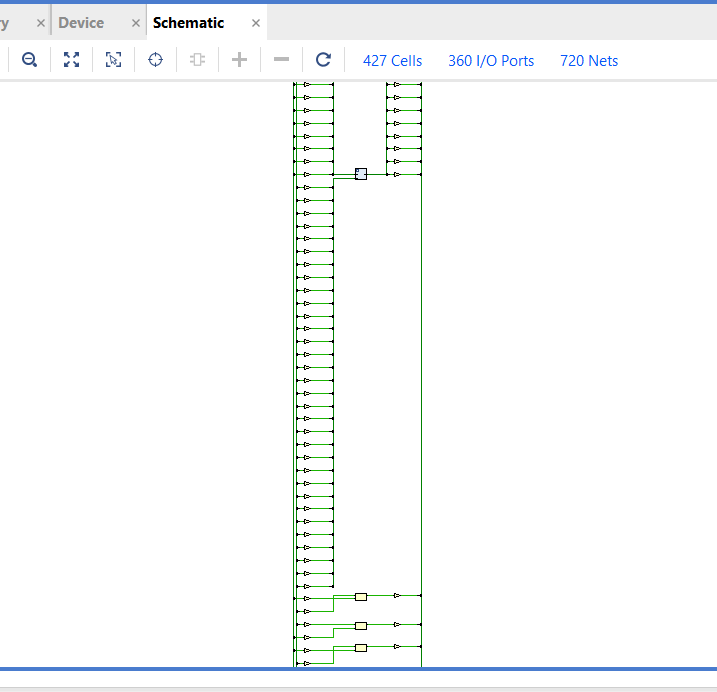
Μετά το simulation παίρνουµε τις εξής κυµατοµορφές που φαίνονται στο Σχήµα [14](#_bookmark22) Το σχηµατικό σύνθεσης της Λογικής Μονάδας παρουσιάζεται στο Σχήµα [15](#_bookmark23)



Σχήµα 13: Logic Unit code



Σχήµα 14: Simulation Results

Σχήµα 15: Logic Unit Schematic

## Σύνθεση και Αποτελέσ ατα

Μετά το simulation παίρνουµε τα εξής αποτελέσµατα στο Σχήµα.΄Οπως περιγράψαµε και παραπάνω, τα 2 δεξιότερα δεκαεξαδικά ψηφία (ff) είναι το αποτέλεσµα της λογικής πράξης NAND , τα 4 αµέσως επόµενα (0000) είναι το αποτέλεσµα της πράξης AND . Τα 8 ψηφία που ακολουθούν (fed00aff) προκύπτουν απο τη πράξη OR και τέλος τα ψηφία (0aecb7af6d5b00c0) είναι αποτέλεσµα της πράξης XOR .

Στην συνέχεια εκτελούµε την σύνθεση για κάθε µία απο τις πύλες και λαµβάνουµε τα αποτελέσµατα που

φαίνονται στον Πίνακα [2.](#_bookmark25)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| File Code | Part | % Used IOB | % Used LUT | Worst Delay | Total Power |
| logic*unit.vhd* | xc7a200tffv1156-1 | 72% | 0.09% | 5,378ns | 42.694W |

Πίνακας 2: Ενδεικτικά αποτελέσµατα Σύνθεσης

Για λεπτοµερή αναφορά των αποτελεσµάτων της σύνθεσης, µπορείτε να µεταβείται στο spreadsheet µε όνοµα SyntesisReport.xlsx, που θα βρείτε στο συµπιεσµένο αρχείο που παραδώσαµε.