3η Εργαστηριακή ΄Ασκηση

Κοτσιµπού Φωτεινή ΑΜ 1059567 Σιγούρου ΄Αλκηστις-Αικατερίνη ΑΜ 1059661

26 Απριλίου 2021



Εργασία στα πλαίσια του µαθήµατος Σχεδιασµός Συστηµάτων VLSI

Τµήµα Μηχ. Ηλεκτρονικών Υπολογιστών και Πληροφορικής

Περιεχό ενα

1. [Ενότητα Α](#_bookmark0) 1
   1. [Α1 - Σύνθεση και Αποτελέσµατα](#_bookmark1) 1
      1. [1η Εργαστηριακή ΄Ασκηση](#_bookmark2) 1
      2. [2η Εργαστηριακή ΄Ασκηση](#_bookmark3) 2
2. [Ενότητα Β](#_bookmark4) 3

[2.1 B1- RAM-32](#_bookmark5) 3

* 1. [B2- FPGA RAM](#_bookmark6) 3
  2. [Β3- Σύνθεση και Αποτελέσµατα](#_bookmark8) 4
  3. [Β4- Επεξήγηση Σύνθεσης](#_bookmark9) 5

# Ενότητα Α

* 1. Α1 - Σύνθεση και Αποτελέσ ατα

#### 1η Εργαστηριακή ΄Ασκηση

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| File Code | Part | % Used IOB | % Used LUT | Worst Delay | Total Power |
| Nand642*.vhd* |  | % | % |  |  |
| 1bit*F ullAdder.vhd* | xa7z010clg225-1I | 9% | 1% | 5,352 | 1.03W |
| Logic*unit*1*.vhd* | xa7z010clg225-1I | 59% | 1% | 5,352 | 5.75W |
| Logic*unitRegister.vhd* | xa7z010clg225-1I | 93% | 1% | 5,352 | 11.533W |

Πίνακας 1: Ενδεικτικά αποτελέσµατα Σύνθεσης

**Nand64**2 Για τον συγκεκριµένο κώδικα παρατηρείται πως έχει γίνει χρήση διαφορετικού FPGA καθώς όταν έγινε χρήση του ίδιου που έχουν και τα υπόλοιπα, παρατηρήθηκε χρήση στο used IOB µεγαλύτερη του 100% ,γεγονός που δηλώνει πως είναι απαραίτητη η χρήση ενός διαφορετικού FPGA µε µεγαλύτερο πλήθος σε IOB.

**1bit***F A* Παρατηρείται πως χρησιµοποιείται ο µικρότερος αριθµός σε IOB από όλα τα υπόλοιπα κυκλώµατα της πρώτης εργαστηριακής άσκησης γεγονός που είναι απόλυτα φυσιολογικό καθώς έχει µικρότερη πολυπλοκότητα από τα άλλα κυκλώµατα και απιτούνται πολύ λιγότερα δεδοµένα σε IOB.

**Logic unit** Η συγκεκριµένη υλοποίηση της λογικής µονάδας χωρίς τη χρήση του ρεγιστερ είναι προφανές πως θα απαιτεί λιγότερα IOB και αυτό επαλυθεύεται από τον πίνακα.

**Logic Unit Register** Στο τελευταίο κώδικα παρατηρούµε σχεδόν τα ίδια αποτελέσµατα µε τον παραπάνω κώδικα κάτι που είναι λογικό καθώς υλοποιούν παρόµοιο κύκλωµα. Dστόσο, παρατηρούµε πως η θερµοκρασία έφτασε τους 125 βαθµούς Κελσίου, γεγονός που µαρτυρά πως το σύστηµα υπερθερµάνθηκε.

#### 2η Εργαστηριακή ΄Ασκηση

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| File Code | Part | % Used IOB | % Used LUT | Worst Delay | Total Power |
| mux.vhd | xa7z010clg225-1I | 48% | 1% | 5,94 | 2.286W |
| mux2.vhd | xa7z010clg225-1I | 48% | 1% | 5,94 | 2.281W |
| ZfuncB.vhd | xa7z010clg225-1I | 7% | 1% | 5,33 | 0.402W |
| ZfuncS.vhd | xa7z010clg225-1I | 7% | 1% | 5,326 | 0.402W |

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.

# Ενότητα Β

## 2.1 B1- RAM-32

Στον συγκεκριµένο κώδικα έχουµε αναλάβει τη δηµιουργία µίας µνήµης RAM 32 θέσεων, µε µήκος λέξης τα 8 bit . Ξεκινάµε λοιπόν ορίζοντας τις τιµές που θέλουµε για το πλήθος των bits κάθε λέξης και το πληθος των ψηφίων που θα χρειαστούµε για να αναπαραστήσουµε τις 32 θέσεις της µνήµης µας. Στη συνέχεια ορίζουµε τις εισόδους ως std logic vector µε το address να παίρνει τιµές από 4 εώς 0 (δηλαδή 5 bits για την αναπαραστάση 25 = 32 θέσεων µνήµης) και τα data in και data out να παιρνουν τιµές από το 7 εώς το 0, όσο δηλαδή είναι και το απαιτούµενο µήκος για κάθε λέξη. Τέλος στο όρισµα των µεταβλητών δεν πρέπει να παραλείψουµε τα WE (Write Enable), enable, reset και φυσικά clk (clock) που κάνουν την µνήµη µας λειτουργική.

Προχωρόντας στην γραµµή 24 του κώδικα ξεκιναει η αρχιτεκτονική της µνήµης µας. Η µνήµη απεικονίζεται σε µορφή ενός array που έχει µέγεθος 32 *∗* 8. Στην πρώτη θετική ακµή του ρολογιού και εφόσον η µεταβλητή reset βρίσκεται στη τιµή 1 , πραγµατοποιείται reset στην µνήµη, καθαρίζοντας τη από τα προηγούµενα

περιεχόµενά της, βγάζοντας ως έξοδο 0. Στη περίπτωση που το reset έχει τη τιµή ¨0’ και το σήµα enable είναι στη τιµή ¨1¨, άρα είναι ενεργό, τότε προχωράµε στον ελέγχο της τιµής του WE . Αν και το WE είναι ενεργό, τότε το σύστηµα λαµβάνει τη εντολή να κάνει εγγραφή στη µνήµη. Στην αντίθετη περίπτωση που το WE είχε τη τιµή ¨0’ τότε η µνήµη θα µπορούσε να προσπελαστεί µόνο για ανάγνωση των περιεχοµένων της µνήµης στη συγκεκριµένη θέση που δηλώνει το address . Τέλος στη περίπτωση που το enable έχει τιµή 0 , καµία διαδικασία δεν είναι δυνατό να πραγµατοποιηθεί καθώς η µνήµη δεν ειναι προσπελάσιµη.

Ο κώδικας παρατίθεται στο Σχήµα [1.](#_bookmark7)

Για τη λειτουργία του testbench ξεκινάµε τον κώδικά µας µε µία αρχικοποίηση των µεταβλητών µας σε 0 εκτός από το σήµα reset το οποίο έχει τη τιµή 1 ώστε να καθαρίσει τη µνήµη πριν ξεκινήσει η νέα εγγραφή. Επιπλέον ορίζουµε ως περίοδο για το simulation στα 100ns ως τη περίοδο για ενα κύκλο ρολογιού. Στη συνέχεια στην γραµµή 40 του κώδικά µας, γίνεται η αρχικοποίηση των µοναδων που θα χρησιµοποιήσουµε και δηµιουργούµε τη κυµατοµορφή του ρολογιού, δίνοντας του τις κατάλληλες τιµές για κάθε µισό της περιόδου. Στο τελευταίο τµήµα του κώδικα δίνουµε στις µεταβλητές µας τις καταλληλες τιµές ώστε να εξετάσουµε όλες τις πιθανές περιπτώσεις για τον έλεγχο της ορθής λειτουργίας της µνήµης.

Συγκεκριµένα τα πρώτα 80ns του simulation , το σήµα reset βρίσκεται στο 1 και κατόπιν µένει κολληµένο στο 0 για το υπόλοιπο του simulation ενώ το enable γίνεται 1 µόλις 20ns αργότερα ώστε να εξετάσουµε και τη περίπτωση που η µνήµη δεν είναι προσπελάσιµη για καµία ενέργεια. Ταυτόχρονα µε το σήµα enable, ενεργοποιείται και το σήµα WE καθώς επιθυµούµε πρώτα την εγγραφή της µνήµης µας και στη συνέχεια την ανάγνωση. Το σήµα WE θα µηδενιστεί και θα παραµείνει σε αυτη τη κατάσταση µετά τη χρονική στιγµή 550ns . Στο σήµα data in δηλώνουµε τις τιµές που θελουµε να εισάγουµε ως περιεχόµενα στη µνήµη µας τις χρονικές στιγµές που φαινονται στη γραµµη 73 και 74 ενώ στη µεταβλητή address δηλώνουµε σε ποιες θέσεις θέλουµε να γίνει η εγγραφή ή η ανάγνωση των δεδοµένων µας όπως φαινεται και στις γραµµές 75-78 του κώδικα.

Ο κώδικας του testbench παρατίθεται στο Σχήµα [2.](#_bookmark10)

Μετά το simulation παίρνουµε τις εξής κυµατοµορφές που φαίνονται στο Σχήµα [3](#_bookmark11)

## B2- FPGA RAM

Στο επόµενο ερώτηµα κληθήκαµε να προγραµµατίσουµε µια υπάρχουσα µνήµη του FPGA µας. Αρχικά θα σχεδιάσουµε την µνήµη µας µέσω του IP Integrator , επιλέγουµε µία µεµορψ βλοςκ γενερατορ και προσδιορίζουµε τα µεγέθη των λέξεων µας όπως και το πλήθος θέσεων της µνήµης µας. Στην συνέχεια µέσω της εντολής Create VHDL wrapper καταλήγουµε στον κώδικα που έχουµε στο Σχήµα [4.](#_bookmark12) Παρατηρείται οτι ο κώδικας φέρει µεγάλες οµοιότητες µε τον δικό µας, όσο αφορά τα σήµατα ενεργοποίησης της µνήµης, το µόνο που παραλείπεται είναι το σήµα reset. Μια ακόµα διαφορα µε τον κώδικα του ερωτήµατος Β1, είναι ότι δεν ξεχωρίζεται η διαδικασία του read write κάτι που οδηγεί σε διαφορετικά αποτελεσµάτα στην εξοµοίωση



Σχήµα 1: RAM32 code

που παρουσιάζονται στο Σχήµα [6.](#_bookmark14)

Προκειµένου να εξοµοιώσουµε τον κώδικα µας, όµως, χρειαζόµαστε και το αντίστοιχο testbench Ο κώδικας του testbench παρατίθεται στο Σχήµα [5.](#_bookmark13) Για να είµαστε σε θέσεις να κλανουµε συγκρίσεις, χρησιµοποιούµε τα ίδια data inputs, όπως και στο πρώτο ερώτηµα.

Μετά το simulation παίρνουµε τις εξής κυµατοµορφές που φαίνονται στο Σχήµα [6](#_bookmark14) . Παρατηρείται διαφοροποίηση µε τα αποτελέσµατα του πρώτου ερωτήµατος στα σηµεία των δεδοµένων που διαβάζει η µνήµη. Στην πρώτη περίπτωση διαβάζει η µνήµη τις θέσεις µνήµης 0x00, 0x08, 0x1b, 0x15 που περιλαµβάνει την πληροφορία 0x02 0x04,στις περιπτώσεις που το σήµα WE=0 ενώ στην 2η περίπτωση η µνήµη διαβάζει την πληροφορία και καθώς αυτή γράφεται.

* 1. Β3- Σύνθεση και Αποτελέσ ατα

Μετά το synthesis παίρνουµε τα εξής αποτελέσµατα στο Σχήµα ;;, τις εξής κυµατοµορφές όσον αφορά την κατανάλωση ενέργειας του chip και για τις 2 εκδοχές φαίνονται στο Πίνακα ;;

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| File Code | Part | % Used IOB | % Used LUT | Worst Delay | Total Power |
| RAM*testv.*3*.vhd* | xa7s6ftgb196-2I | 25% | 3% | 4,06 | 1.85W |
| B2*RAMwrapper.vhd* | xa7s6ftgb196-2I | 24% | - | 3,71 | 15,663W |

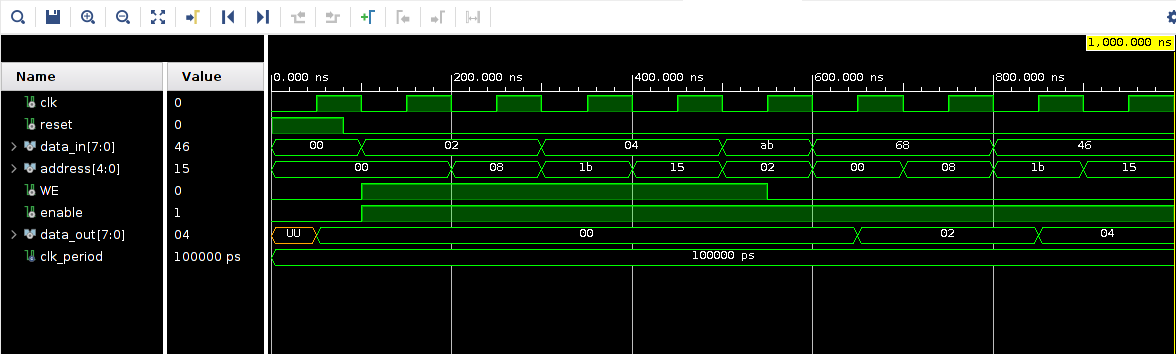
Πίνακας 3: Ενδεικτικά αποτελέσµατα Σύνθεσης

### Β4- Επεξήγηση Σύνθεσης

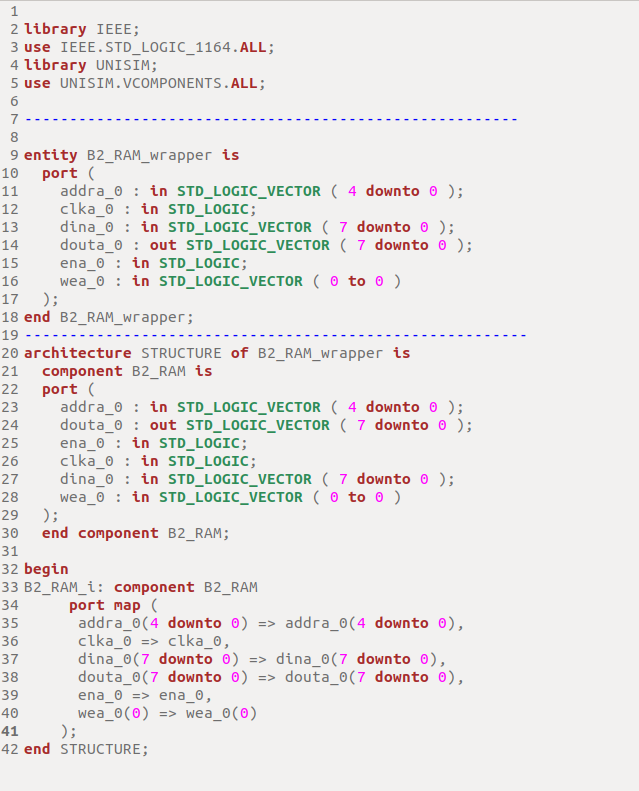
Τα αποτελέσµατα της σύνθεσης είναι πανοµοιτύπα µεταξυ τους ανάµεσα στα 2 ερωτήµατα µε εξαίρεση τις τιµές του Power Report που παρατηρούµε σχεδόν 10πλάσια κατανάλωση ενέργεια στο σχεδιασµό χωρίς τα LUTs.



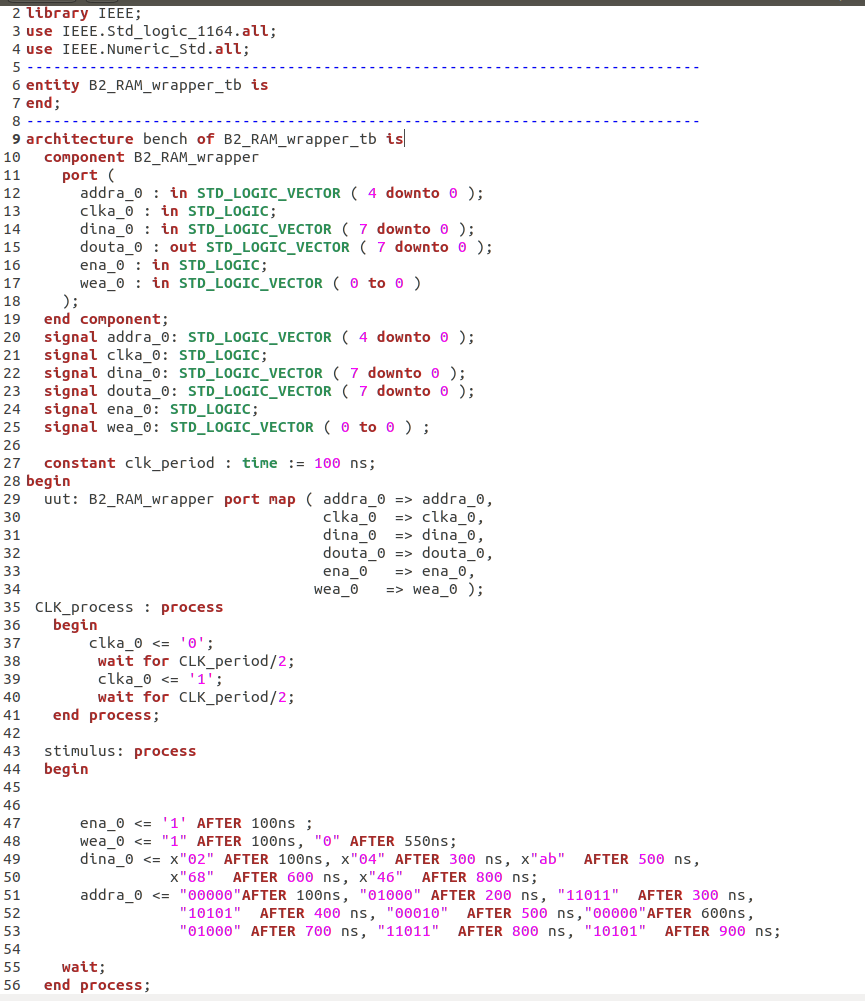
#### Σχήµα 2: Testbench RAM32 code



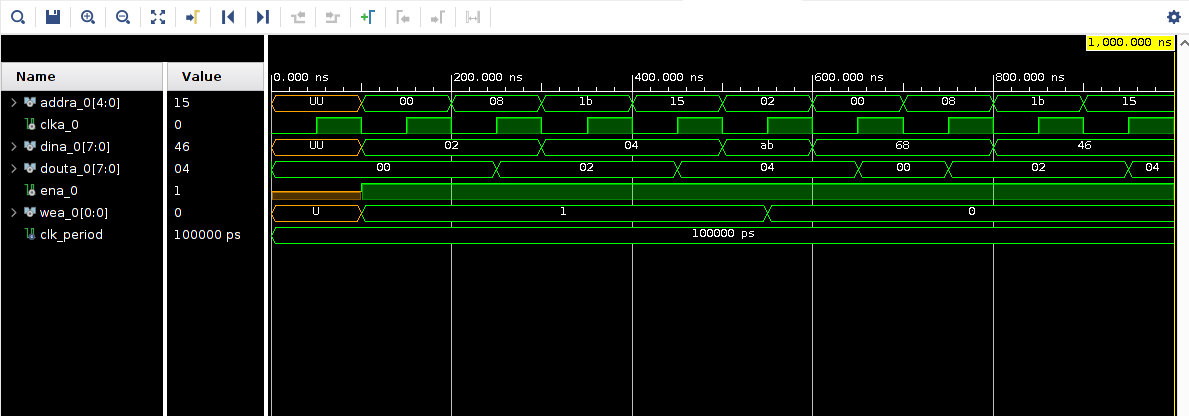
Σχήµα 3: Simulation Results



Σχήµα 4: RAM32 code



Σχήµα 5: Testbench RAM32 code



Σχήµα 6: Simulation Results