Ετερογενή Υπολογιστικά Συστήματα

Φυλλάδιο

Εργαστηριακών Ασκήσεων

Μιχάλης Ψαράκης

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ - ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΠΜΣ «ΠΡΟΗΓΜΕΝΑ ΣΥΣΤΗΜΑΤΑ ΠΛΗΡΟΦΟΡΙΚΗΣ»

Εισαγωγή

Οι εργαστηριακές ασκήσεις θα σας βοηθήσουν στην εκμάθηση του εργαλείου Xilinx ISE Design Suite για την εισαγωγή σχεδίασης (design entry), την προσομοίωση (simulation), την σύνθεση (synthesis) ψηφιακών κυκλωμάτων και την υλοποίηση τους σε εκπαιδευτικές πλατφόρμες FPGA (Field Programmable Gate Arrays). Οι στόχοι του εργαστηρίου είναι:

- να εξοικειωθείτε με ένα ολοκληρωμένο περιβάλλον σχεδίασης, προσομοίωσης και σύνθεσης κυκλωμάτων (σημείωση: αυτό το περιβάλλον θα χρησιμοποιήσετε και για να υλοποιήσετε την εργασία του μαθήματος).
- να διδαχτείτε μέσα από μια σειρά εργαστηριακών ασκήσεων με κλιμακούμενη πολυπλοκότητα την σχεδίαση υπολογιστικών συστημάτων με χρήση της γλώσσας περιγραφής υλικού VHDL.
- να έρθετε σε επαφή με μια εκπαιδευτική και αναπτυξιακή πλατφόρμα υλικού και χρησιμοποιώντας την τεχνολογία προγραμματιζόμενης λογικής (programmable logic) να υλοποιήσετε τα κυκλώματά σε συσκευές FPGAs (Field Programmable Gate Arrays).

Οι εργαστηριακές ασκήσεις χωρίζονται στις παρακάτω ενότητες:

- **Εργαστηριακή Άσκηση 1**: *Εισαγωγή στο εργαλείο και εξοικείωση με την χρήση της εκπαιδευτικής πλακέτας FPGA.*

  Θα εξοικειωθείτε με τη χρήση του εργαλείου Xilinx ISE Design Suite σχεδιάζοντας και προσομοιώντας στοιχειώδη κυκλώματα. Επίσης, θα υλοποιήσετε τα κυκλώματα σε μια εκπαιδευτική FPGA πλατφόρμα (Spartan-3E Starter Kit ή Spartan-3E Microblaze Development Kit).

- **Εργαστηριακή Άσκηση 2**: *Υλοποίηση συνδυαστικών και ακολουθιακών κυκλωμάτων*.

  Θα ασχοληθείτε με την σχεδίαση, προσομοίωση και υλοποίηση στην πλακέτα FPGA απλών συνδυαστικών (π.χ. πολυπλέκτες, αποκωδικοποιητές, αθροιστές, κτλ.) και ακολουθιακών κυκλωμάτων (π.χ. μετρητές, καταχωρητές ολίσθησης, κτλ.).

Το εργαλείο Xilinx ISE υποστηρίζει διάφορες γλώσσες περιγραφής υλικού (hardware description languages). Οι εργαστηριακές ασκήσεις επικεντρώνονται στην γλώσσα περιγραφής υλικού VHDL.

Σημείωση: Το παρόν φυλλάδιο αφορά την έκδοση του εργαλείου Xilinx ISE Design Suite 14.7.

Εργαστηριακή Άσκηση 1: Εισαγωγή στο εργαλείο και εξοικείωση με την χρήση της εκπαιδευτικής πλακέτας FPGA

Σε αυτήν την εργαστηριακή άσκηση θα εξοικειωθείτε με το περιβάλλον Xilinx ISE Design Suite και με την χρήση και τον προγραμματισμό της εκπαιδευτικής πλακέτας FPGA. Πιο συγκεκριμένα θα μάθετε:

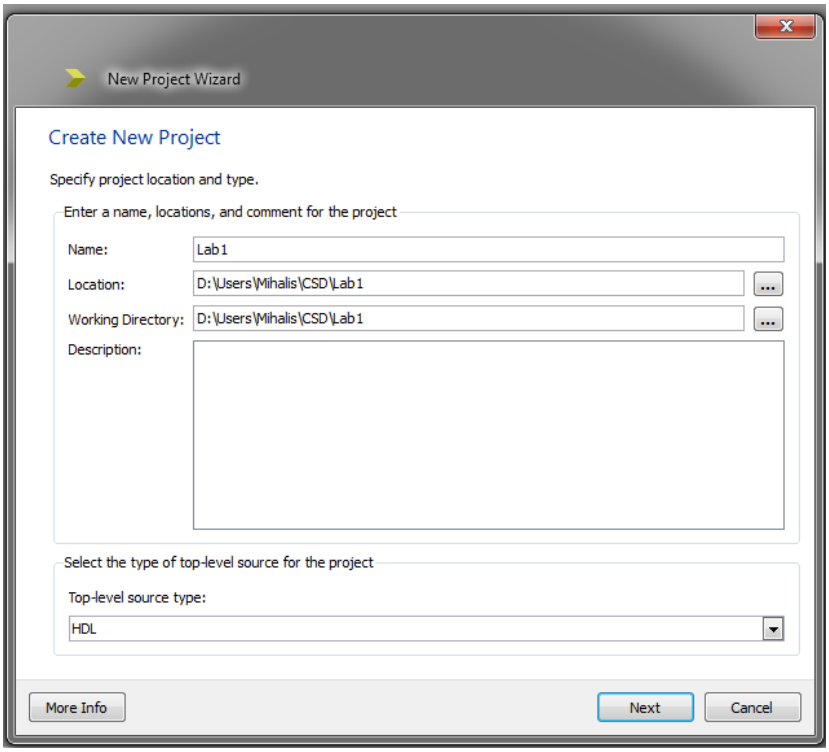
- Πως να χρησιμοποιείτε τα εργαλεία εισαγωγής της σχεδίασης (design entry tools) σχεδιάζοντας κυκλώματα με τη χρήση μιας γλώσσας περιγραφής υλικού (π.χ. VHDL)
- Πώς να προσομοιώνετε το κύκλωμα (functional simulation) με χρήση του προσομοιωτή ISim.
- Τα χαρακτηριστικά της εκπαιδευτικής και αναπτυξιακής πλακέτας (Spartan-3E Starter Kit ή Spartan-3E Microblaze Development Kit) που θα χρησιμοποιήσετε στο εργαστήριο και στις εργασίες
- Πώς να δημιουργείτε ένα αρχείο προγραμματισμού FPGA
- Πώς να προγραμματίζετε μια συσκευή FPGA

Κύκλωμα: 4-bit parity generator (με VHDL)

Το κύκλωμα που θα υλοποιήσετε σε αυτήν την εργαστηριακή άσκηση είναι ένα απλό κύκλωμα υπολογισμού της άρτιας ισοτιμίας ενός μηνύματος των 4-bit. Το κύκλωμα υλοποιείται τρεις πύλες XOR των 2-εισόδων. Η σχεδίαση θα περιγραφεί αρχικά με γλώσσα VHDL και στη συνέχεια με σχηματικό διάγραμμα.

1.1 Δημιουργία νέου έργου (new project)

Κάντε διπλό-κλικ στο εικονίδιο Project Navigator στην επιφάνεια εργασίας ή επιλέξτε All Programs→Xilinx Design Tools→ISE Design Suite 14.x→ISE Design Tools→ 64-bit Project Navigator. Από το Project Navigator, επιλέξτε File→New Project. Θα εμφανιστεί το πρώτο πλαίσιο διαλόγου του New Project, όπως φαίνεται στην Εικόνα 1.



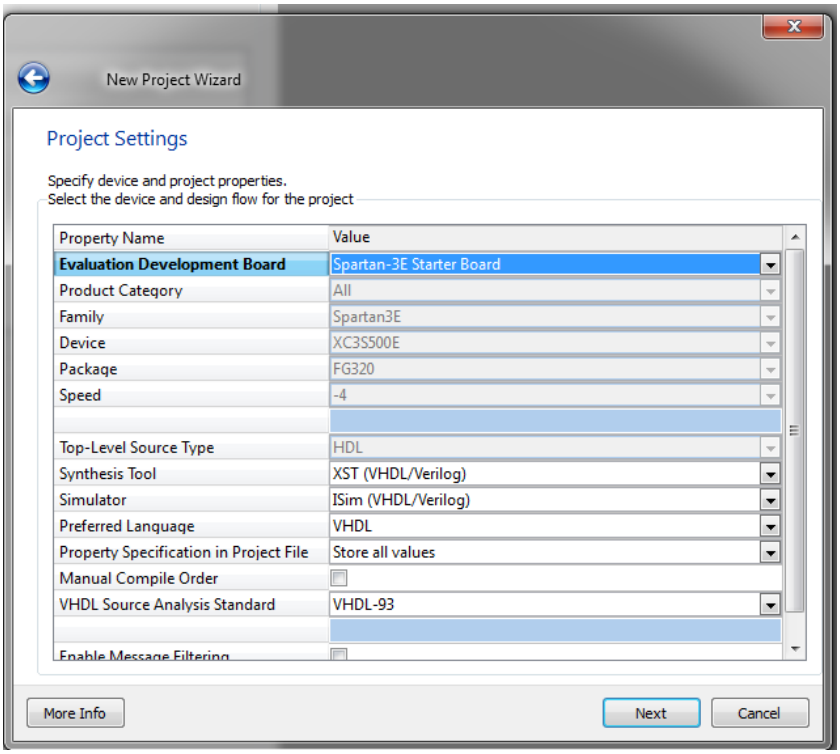
Εικόνα 1: New Project Wizard (1 από 3)

Το πλαίσιο διαλόγου σας προτρέπει να εισάγετε το όνομα του έργου, τη θέση του έργου, και τον τύπο της μονάδας του πιο υψηλού επιπέδου της σχεδίασης, όπως φαίνεται στην Εικόνα 1. Αφού συμπληρώσετε τα στοιχεία, πατήστε Next.

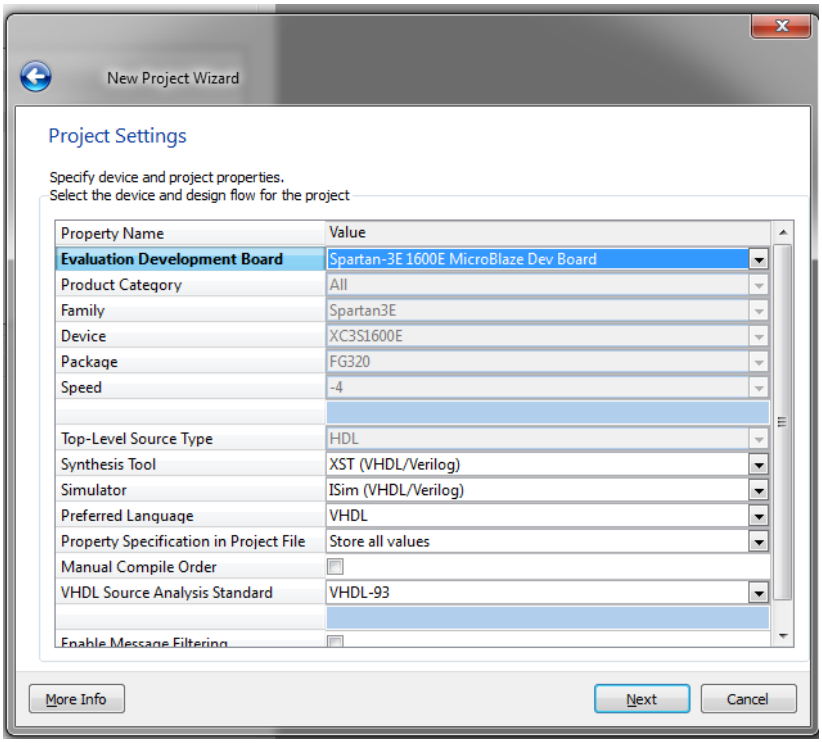
Σημείωση: Μην χρησιμοποιείτε ονόματα αρχείων ή φακέλων που περιέχουν διαστήματα.

Σημείωση: Δεν είναι απαραίτητο οι μονάδες στο υψηλότερο επίπεδο και στα χαμηλότερα επίπεδα να είναι του ίδιου τύπου. Για παράδειγμα, θα μπορούσατε να επιλέξετε τύπο Schematic για το top-level module και τύπο HDL για τα lower-level modules ή και το αντίστροφο. Στο συγκεκριμένο παράδειγμα επιλέγουμε τύπο HDL για όλες τις μονάδες.

Το επόμενο πλαίσιο διαλόγου σας επιτρέπει να θέσετε τις πρόσθετες επιλογές του έργου. Η πρώτη ομάδα ρυθμίσεων καθορίζει τον τύπο της συσκευής FPGA που θα χρησιμοποιήσετε. Οι ρυθμίσεις που παρουσιάζονται στην Εικόνα 2.α αντιπροσωπεύουν τη συσκευή FPGA που βρίσκεται στην εκπαιδευτική πλακέτα Spartan-3E Starter Kit ενώ στην Εικόνα 2.β την πλακέτα Spartan-3E Microblaze Development Kit. Η δεύτερη ομάδα ρυθμίσεων αντιπροσωπεύει τον τρόπο εισαγωγής της σχεδίασης, το εργαλείο σύνθεσης, και τον προσομοιωτή που θα χρησιμοποιήσετε. Αφού θέσετε τις ρυθμίσεις όπως φαίνεται στην Εικόνα 2, πατήστε Next.



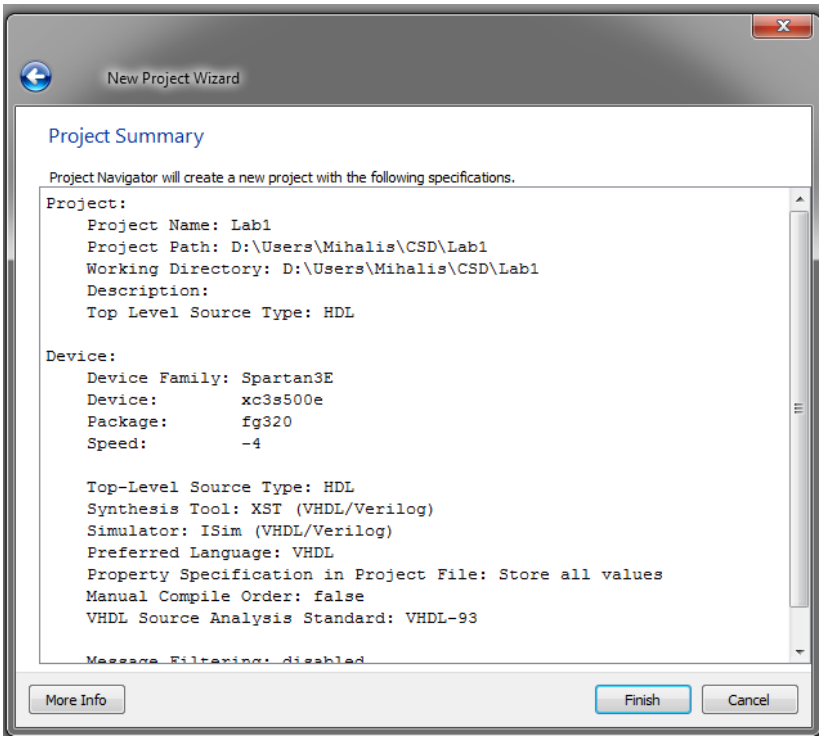
Εικόνα 2.α: Ρυθμίσεις για πλακέτα Spartan-3E Starter Kit



Εικόνα 2.β: Ρυθμίσεις για πλακέτα Spartan-3E Microblaze Development Kit

Εικόνα 2: New Project Wizard (2 από 3)

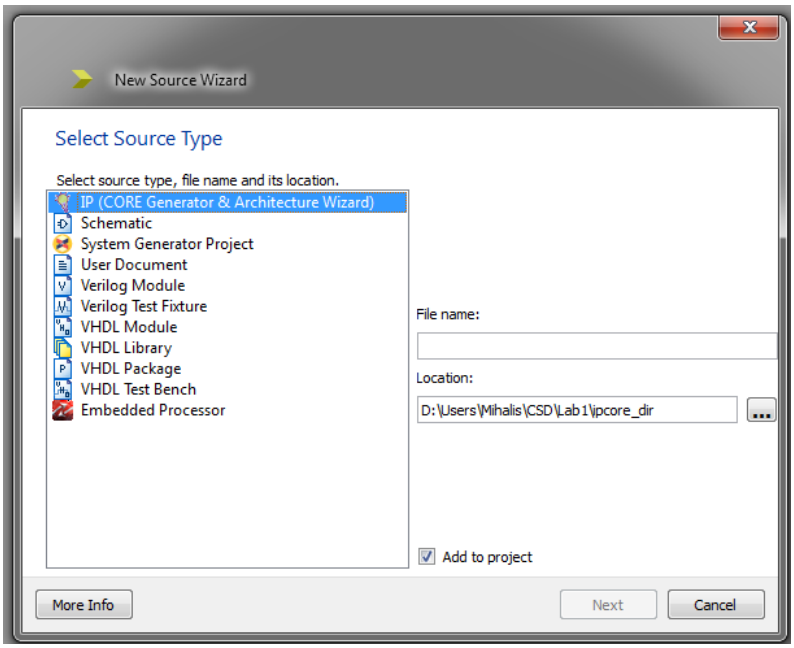
Το τελικό πλαίσιο διαλόγου στη διαδικασία δημιουργίας νέου έργου, που φαίνεται στην Εικόνα 3, παρέχει μια περίληψη του έργου που το Project Navigator θα δημιουργήσει βασισμένο στις ρυθμίσεις σας. Ελέγξτε την περίληψη για να σιγουρευτείτε ότι ταιριάζει με ότι φαίνεται στην Εικόνα 3. Εάν όχι, πατήστε Back για να διορθώσετε οποιοδήποτε λάθος. Διαφορετικά, πατήστε Finish για να ολοκληρώσετε τη διαδικασία.



Εικόνα 3: New Project Wizard (3 από 3)

1.2 Σχεδίαση βασισμένη στη γλώσσα περιγραφής υλικού VHDL

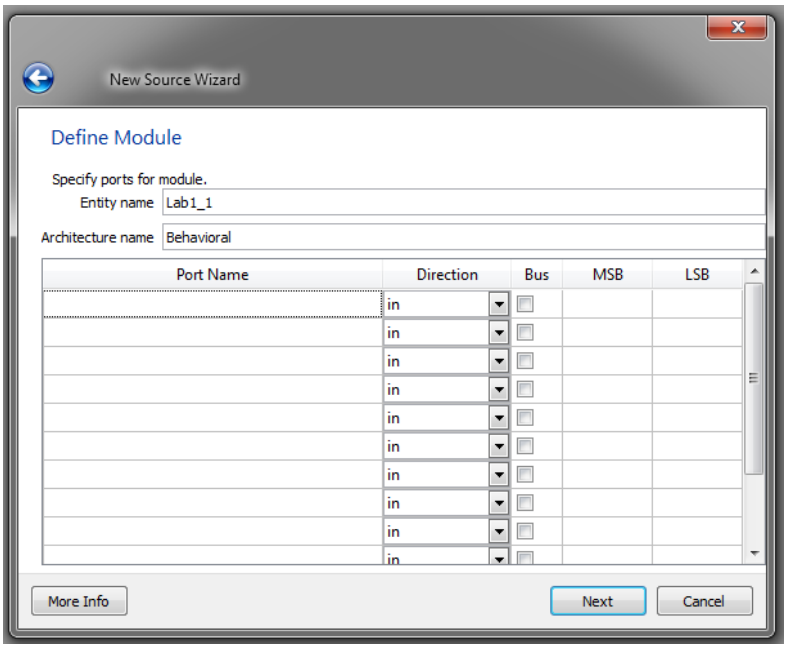
Σε αυτό το σημείο, το έργο που έχει δημιουργηθεί δεν περιέχει κανένα πηγαίο αρχείο. Δημιουργήστε ένα νέο πηγαίο αρχείο για τη σχεδίαση του κυκλώματος υπολογισμού άρτιας ισοτιμίας. Επιλέξτε Project→New Source από το κυρίως μενού. Το πρώτο από τα νέα κουτιά διαλόγου θα εμφανιστεί, όπως φαίνεται στην Εικόνα 4.



Εικόνα 4: New Source Wizard (1 από 3)

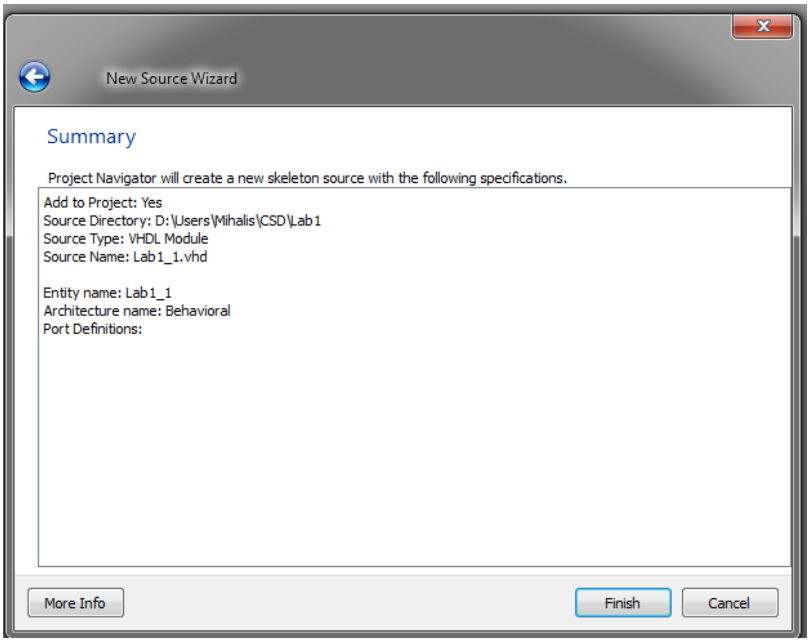
Επιλέξτε VHDL Module για να δηλώσετε ότι δημιουργείτε μια μονάδα σχεδίασης βασισμένη στη γλώσσα VHDL. Κατόπιν, δώστε ένα όνομα αρχείου όπως φαίνεται στην Εικόνα 4. Δεν χρειάζεται να αλλάξετε τη θέση του αρχείου, η οποία θα είναι μέσα στον κατάλογο του έργου που δημιουργήσατε προηγουμένως. Πατήστε Next.

Το επόμενο πλαίσιο διαλόγου σας επιτρέπει (προαιρετικά) να καθορίσετε τις θύρες (ports) της μονάδας. Αυτό μπορεί επίσης να γίνει στον επεξεργαστή κειμένου, κατά την επεξεργασία της μονάδας, έτσι αγνοήστε το σε αυτή τη φάση. Απλά επιβεβαιώστε ότι οι ρυθμίσεις ταιριάζουν με εκείνες που φαίνονται στην Εικόνα 5 και πατήστε Next.



Εικόνα 5: New Source Wizard (2 από 3)

Το τελικό πλαίσιο διαλόγου (Εικόνα 6) παρέχει μια περίληψη του πηγαίου αρχείου που θα δημιουργήσει το Project Navigator βασισμένο στις ρυθμίσεις σας. Ελέγξτε την περίληψη για να σιγουρευτείτε ότι ταιριάζει με την Εικόνα 6. Εάν όχι, πατήστε Back για να διορθώσετε οποιοδήποτε λάθος. Διαφορετικά, πατήστε Finish για να ολοκληρώσετε τη διαδικασία. Το νέο πηγαίο αρχείο θα ανοίξει αυτόματα στον επεξεργαστή κειμένου.



Εικόνα 6: New Source Wizard (3 από 3)

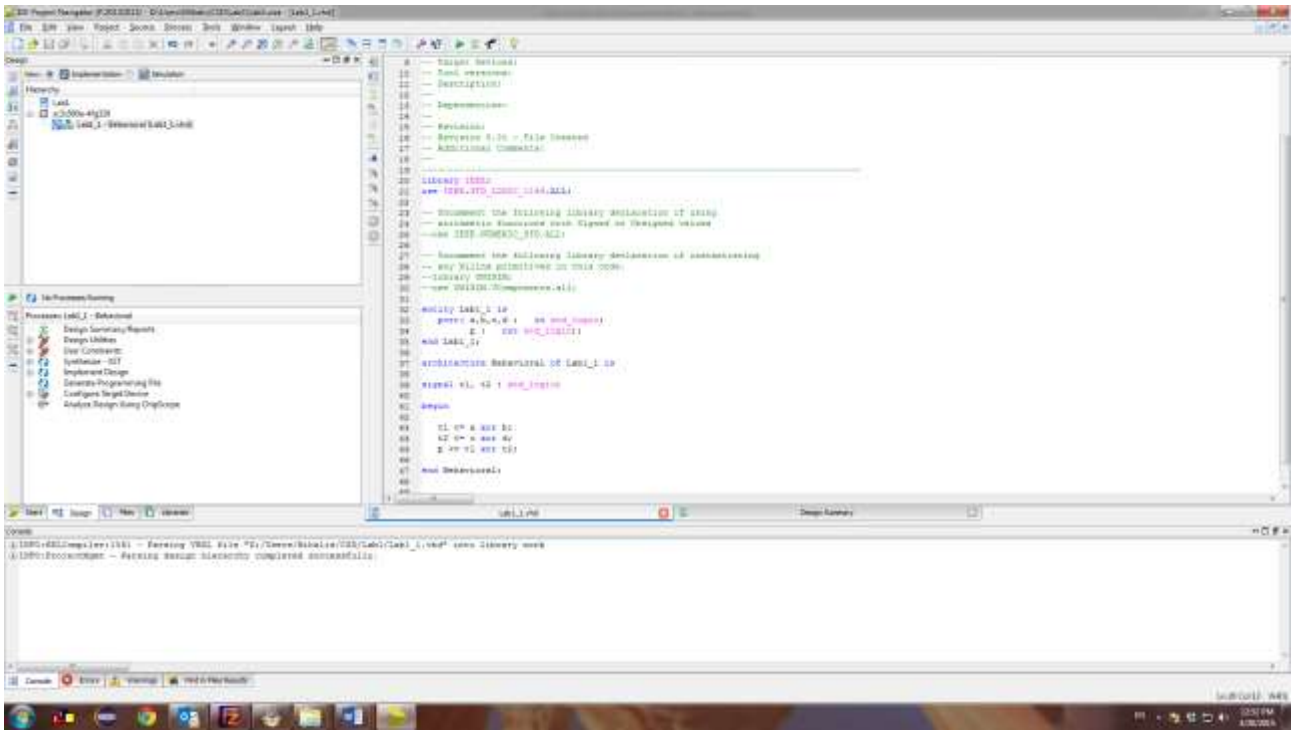
Στον επεξεργαστή κειμένου, μερικές από τη βασικές δομές του VHDL αρχείου είναι ήδη γραμμένες. Οι λέξεις κλειδιά (keywords) της γλώσσας απεικονίζονται με μπλε χρώμα, οι τύποι δεδομένων με κόκκινο, τα σχόλια με πράσινο, και οι τιμές με μαύρο. Αυτή η κωδικοποίηση ανάλογα με το χρώμα ενισχύει την αναγνωσιμότητα του VHDL αρχείου και την εύρεση τυπογραφικών λαθών. Τώρα, εισάγετε την περιγραφή του κυκλώματος υπολογισμού άρτιας ισοτιμίας.

Σημείωση: Μπορείτε να κατεβάσετε το παρακάτω μοντέλο (Lab1_1.vhd) από την ιστοσελίδα του μαθήματος (περιέχεται στο Lab1.zip).

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity Lab1_1 is
   port(  a,b,c,d :     in std_logic;
          p :    out std_logic);
end Lab1_1;

architecture Behavioral of Lab1_1 is

signal t1, t2 : std_logic;
```

```
begin

   t1 <= a xor b;
   t2 <= c xor d;
   p <= t1 xor t2;

end Behavioral;
```

Σε αυτό το σημείο, πρέπει να καταλήξετε με ένα παράθυρο που μοιάζει με αυτό που φαίνεται στην Εικόνα 7. Μόλις τελειώσετε, αποθηκεύστε το αρχείο και κλείστε το παράθυρο. Υπάρχουν επιλογές στο κυρίως μενού για να σώσετε είτε μεμονωμένα αρχεία είτε όλο το έργο.

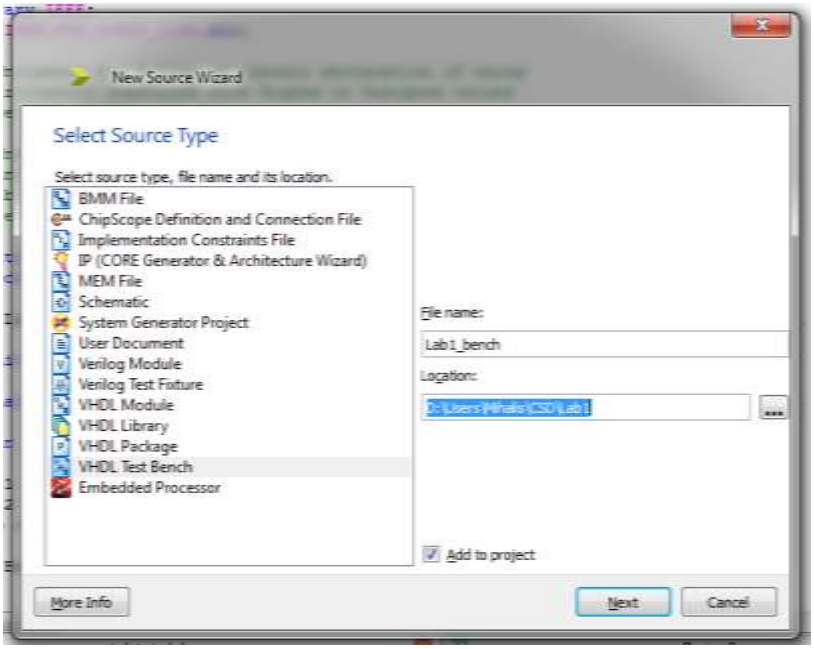


Εικόνα 7: Ολοκληρωμένη σχεδίαση

1.3 Λειτουργική προσομοίωση (functional simulation)

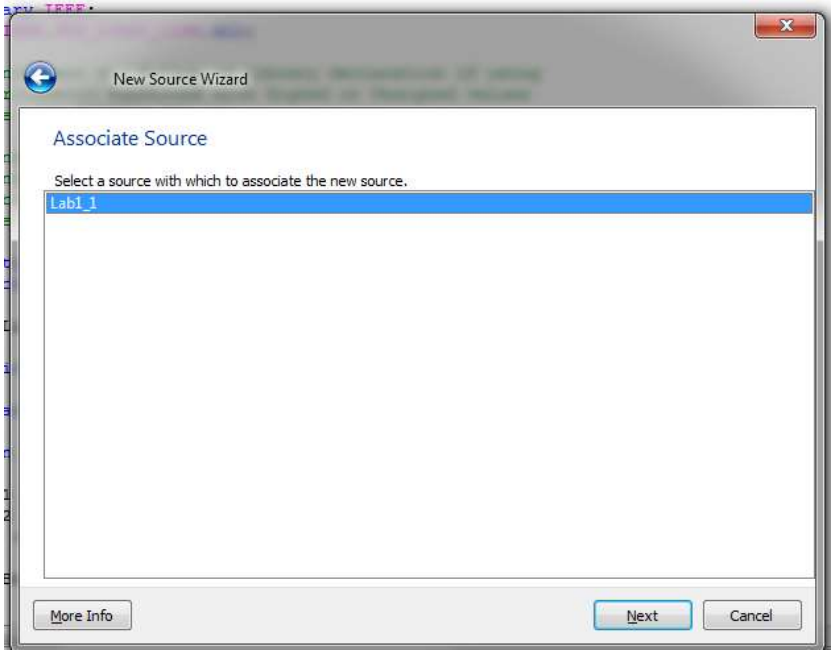
Η λειτουργική προσομοίωση εκτελείται προτού να γίνει σύνθεση της σχεδίασης για να επαληθεύσει ότι η λογική του κυκλώματος είναι σωστή. Αυτό επιτρέπει σε ένα σχεδιαστή να βρει και να διορθώσει τυχόν λάθη στη σχεδίαση προτού ξοδέψει χρόνο στα επόμενα βήματα. Το Project Navigator ενσωματώνει τον προσομοιωτή **ISim** και μας επιτρέπει να εκτελέσουμε τις προσομοιώσεις από το Project Navigator.

Προκειμένου να προσομοιώσουμε τη σχεδίαση, απαιτείται ένα πρόγραμμα δοκιμής (test bench) για να παράγει τα απαραίτητα ερεθίσματα εισόδου (input stimulus) στη σχεδίαση. Δημιουργήστε ένα νέο πηγαίο αρχείο για το testbench επιλέγοντας Project→New Source από το κυρίως μενού είτε χρησιμοποιήστε την ισοδύναμη διαδικασία στο παράθυρο Processes for Current Source. Το πρώτο από τα κουτιά διαλόγου New Source θα εμφανιστεί, όπως φαίνεται στην Εικόνα 8.



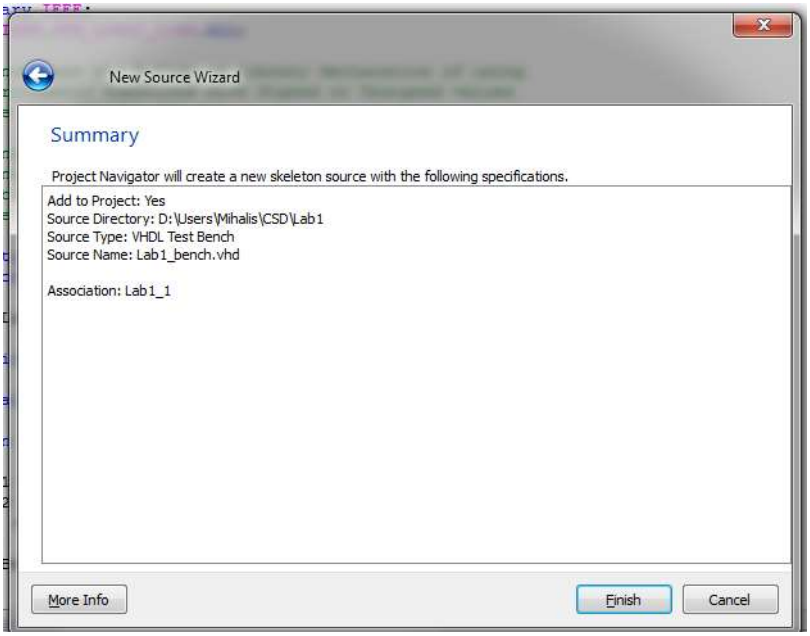
Εικόνα 8: New Source Wizard (1 από 3)

Επιλέξτε VHDL Test Bench για να δηλώσετε ότι δημιουργείτε μια μονάδα δοκιμής (testbench) βασισμένη στη γλώσσα VHDL. Κατόπιν, δώστε ένα όνομα αρχείου όπως φαίνεται στην Εικόνα 8. Δεν χρειάζεται να αλλάξετε τη θέση του αρχείου, η οποία θα είναι μέσα στον κατάλογο του έργου που δημιουργήσατε προηγουμένως. Πατήστε Next.



Εικόνα 9: New Source Wizard (2 από 3)

Το δεύτερο πλαίσιο διαλόγου, που φαίνεται στην Εικόνα 9, σας ζητά να προσδιορίσετε μια μονάδα σχεδίασης με την οποία πρέπει να συσχετιστεί το test bench. Επιλέξτε τη μονάδα Lab1_1 όπως φαίνεται στην Εικόνα 9 και πατήστε Next.



Εικόνα 10: New Source Wizard (3 από 3)

Το τελικό πλαίσιο διαλόγου (Εικόνα 10) παρέχει μια περίληψη του πηγαίου αρχείου που θα δημιουργήσει το Project Navigator βασισμένο στις ρυθμίσεις σας. Ελέγξτε την περίληψη για να σιγουρευτείτε ότι ταιριάζει με την Εικόνα 10. Εάν όχι, πατήστε Back για να διορθώσετε οποιοδήποτε λάθος. Διαφορετικά, πατήστε Finish για να ολοκληρώσετε τη διαδικασία. Το νέο πηγαίο αρχείο θα ανοίξει αυτόματα στον επεξεργαστή κειμένου.

Στον επεξεργαστή κειμένου, μερικές από τη βασικές δομές του αρχείου testbench είναι ήδη γραμμένες. Τώρα, εισάγετε την περιγραφή του testbench.

Σημείωση: Μπορείτε να κατεβάσετε το testbench (Lab1_bench.vhd) από την ιστοσελίδα του μαθήματος (περιέχεται στο Lab1.zip).

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY Lab1_bench IS
END Lab1_bench;

ARCHITECTURE behavior OF Lab1_bench IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT Lab1_1
    PORT(
         a : IN  std_logic;
         b : IN  std_logic;
         c : IN  std_logic;
         d : IN  std_logic;
         p : OUT  std_logic
        );
```

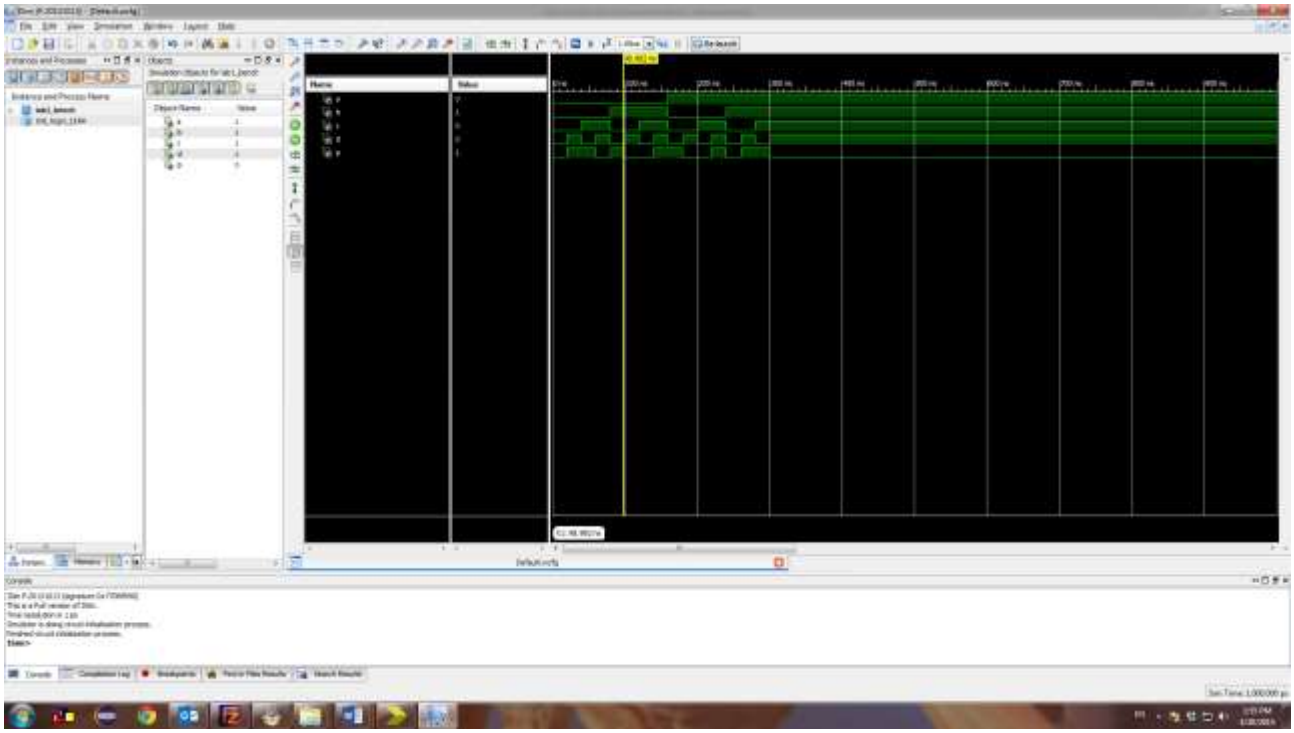
```
    END COMPONENT;


   --Inputs
   signal a : std_logic := '0';
   signal b : std_logic := '0';
   signal c : std_logic := '0';
   signal d : std_logic := '0';

  --Outputs
   signal p : std_logic;


BEGIN

  -- Instantiate the Unit Under Test (UUT)
   uut: Lab1_1 PORT MAP (
          a => a,
          b => b,
          c => c,
          d => d,
          p => p
        );

   -- Stimulus process
   stim_proc: process
   begin
      -- insert stimulus here
  a <= '0'; b <= '0'; c <= '0'; d <= '0';
  wait for 20 ns;
  a <= '0'; b <= '0'; c <= '0'; d <= '1';
  wait for 20 ns;
  a <= '0'; b <= '0'; c <= '1'; d <= '0';
  wait for 20 ns;
  a <= '0'; b <= '0'; c <= '1'; d <= '1';
  wait for 20 ns;
  a <= '0'; b <= '1'; c <= '0'; d <= '0';
  wait for 20 ns;
  a <= '0'; b <= '1'; c <= '0'; d <= '1';
  wait for 20 ns;
  a <= '0'; b <= '1'; c <= '1'; d <= '0';
  wait for 20 ns;
  a <= '0'; b <= '1'; c <= '1'; d <= '1';
  wait for 20 ns;
  a <= '1'; b <= '0'; c <= '0'; d <= '0';
  wait for 20 ns;
  a <= '1'; b <= '0'; c <= '0'; d <= '1';
  wait for 20 ns;
  a <= '1'; b <= '0'; c <= '1'; d <= '0';
  wait for 20 ns;
  a <= '1'; b <= '0'; c <= '1'; d <= '1';
  wait for 20 ns;
  a <= '1'; b <= '1'; c <= '0'; d <= '0';
```

```
    wait for 20 ns;
    a <= '1'; b <= '1'; c <= '0'; d <= '1';
    wait for 20 ns;
    a <= '1'; b <= '1'; c <= '1'; d <= '0';
    wait for 20 ns;
    a <= '1'; b <= '1'; c <= '1'; d <= '1';

    wait; -- will wait forever
     end process;

END;
```

Τώρα που έχετε ένα testbench στο έργο σας, μπορείτε να εκτελέσετε λειτουργική προσομοίωση στη σχεδίασή σας. Οι διεργασίες προσομοίωσης σας επιτρέπουν να τρέξετε προσομοίωση στη σχεδίαση χρησιμοποιώντας τον προσομοιωτή ISim. Για να εντοπίσετε τις διεργασίες του προσομοιωτή ISim, επιλέξτε το **Simulation View** στο παράθυρο Design. Κατόπιν, επιλέξτε το αρχείο Lab1_bench και στο παράθυρο Processes (για το module Lab1_bench) εμφανίζονται οι διαθέσιμοι προσομοιωτές (ISim) και για κάθε προσομοιωτή ποιες διεργασίες είναι διαθέσιμες:

- Behavioral Check Syntax. Ελέγχει εάν το test bench είναι συντακτικά σωστή.
- Simulate Behavioral Model. Προσομοιώνει το μοντέλο συμπεριφοράς του κυκλώματος.

Ελέγξτε πρώτα την ορθότητα του testbench και στη συνέχεια ξεκινήστε την προσομοίωση με διπλό-κλικ στο Simulate Behavioral Model. Το ISim μεταγλωττίζει τα πηγαία αρχεία, φορτώνει τη σχεδίαση, και εκτελεί την προσομοίωση για τον καθορισμένο χρόνο (βλέπε process properties του Simulate Behavioral Model). Θα εμφανιστεί το παράθυρο του ISim, όπως φαίνεται στην Εικόνα 11 (πατήστε Zoom to Full View).



Εικόνα 11: ISim Environment

Στο περιβάλλον του ISim εμφανίζονται τα παρακάτω παράθυρα:

- Console: Εμφανίζει τα μηνύματα του προσομοιωτή, π.χ. προειδοποιήσεις, λάθη, συν οποιοδήποτε μήνυμα εξόδου παραχθεί από τη σχεδίαση που προσομοιώνετε.
- Instances and Process name: Αυτό το παράθυρο σας επιτρέπει να αναζητήσετε την ιεραρχία του testbench και της σχεδίασης υπό δοκιμή. Στις μεγάλες ιεραρχικές σχεδιάσεις, αυτό είναι πολύ χρήσιμο.
- Objects: Αυτό το παράθυρο δείχνει τα σήματα που περιέχονται στο instance/process της σχεδίασης που έχετε επιλέξει στο παράθυρο Instances and Process name
- Wave: Απεικονίζει τις κυματομορφές που προσομοιώνετε.

Για να προσθέσετε σήματα στο παράθυρο wave, μπορείτε είτε να τα μεταφέρετε από το παράθυρο Objects (με drag and drop), ή να τα επιλέξετε στο παράθυρο Objects και έπειτα να επιλέξετε Add to Wave Window.

Όταν προσθέτετε τα νέα σήματα στο παράθυρο wave (π.χ. επιλέξτε το instance uut και προσθέστε τα σήματα t1 και t2), θα παρατηρήσετε ότι οι κυματομορφές δεν εμφανίζονται αυτόματα. Αυτό είναι επειδή το ISim δεν κατέγραψε τα δεδομένα προσομοίωσης για αυτά τα σήματα. Το Isim καταγράφει δεδομένα μόνο για τα σήματα που έχουν προστεθεί στο παράθυρο wave πριν ή κατά τη διάρκεια της προσομοίωσης. Επομένως, όταν προστίθενται νέα σήματα στο παράθυρο wave, η προσομοίωση πρέπει να ξαναξεκινήσει και να επαναληφθεί για το επιθυμητό χρονικό διάστημα. Για να ξαναξεκινήσει η προσομοίωση, κάντε κλικ στο Restart.

Ο προσομοιωτής Isim παρέχει την ικανότητα της αποθήκευσης του καταλόγου σημάτων στο παράθυρο wave. Αυτό μπορεί να είναι σημαντικό όταν προστίθενται νέα σήματα ή ερεθίσματα, και η προσομοίωση ξαναξεκινά.

Προγραμματισμός της FPGA πλακέτας

Αφού έχετε ολοκληρώσει και επαληθεύσει (με προσομοίωση) την σχεδίαση, τα επόμενα βήματα ώστε να προγραμματίσετε την συσκευή FPGA είναι: σύνθεση σχεδίασης (design synthesis), υλοποίηση σχεδίασης (design implementation) και προγραμματισμός FPGA.

1.4 Πλακέτα FPGA Spartan-3E

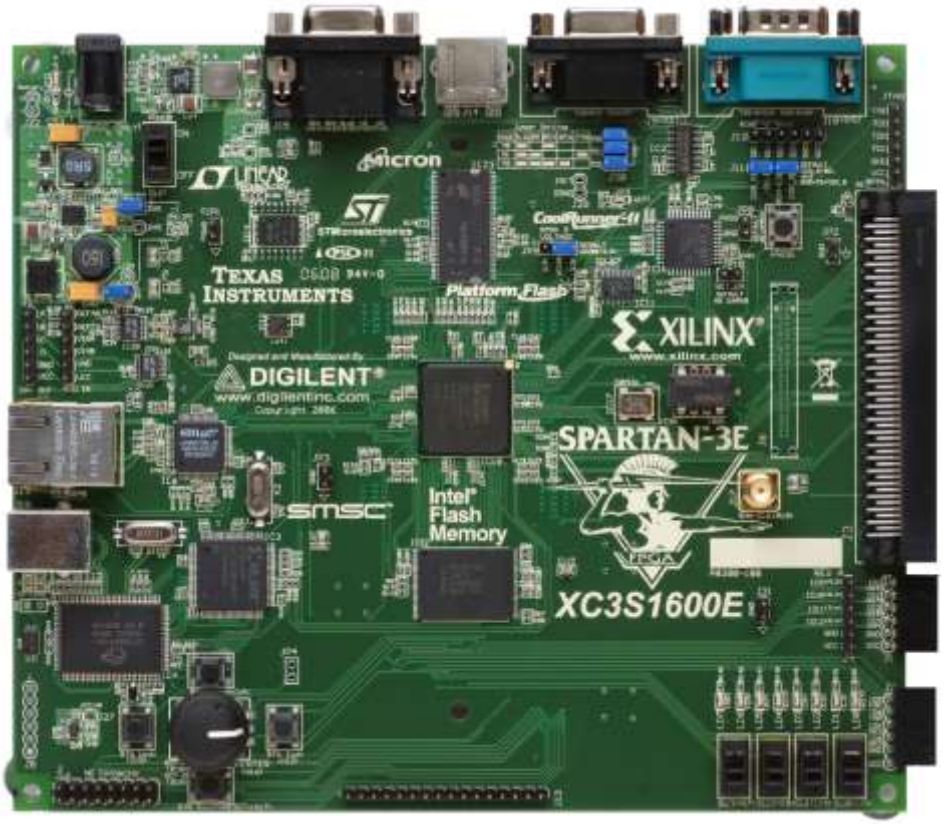
Πρώτα όμως εξοικειωθείτε με την πλακέτα FPGA που θα χρησιμοποιήσετε. Υπάρχουν 2 είδη πλακετών: η πλακέτα Spartan-3E Starter Kit (βλέπε Εικόνα 12) και η πλακέτα Spartan-3E Microblaze Development Kit (βλέπε Εικόνα 13). Στην ιστοσελίδα του εργαστηρίου μπορείτε να βρείτε εγχειρίδια χρήσης των πλακετών.

Προσοχή: Ελέγξτε ποια από τις 2 πλακέτες χρησιμοποιείτε και σιγουρευτείτε ότι στις ιδιότητες του έργου (project settings) έχετε επιλέγει την σωστή συσκευή FPGA.

Εικόνα 12 Spartan-3E Starter Kit

Η πλακέτα Spartan-3E Starter Kit περιέχει τα εξής (στα πλαίσια του εργαστηρίου θα χρησιμοποιήσετε ότι είναι με bold):

- **Ένα τσιπάκι Xilinx XC3S500E Spartan-3E FPGA** (έχει μέχρι 232 ακροδέκτες I/O και πάνω από 10,000 λογικά στοιχεία)
- Ένα τσιπάκι Xilinx 64-macrocell XC2C64A CoolRunner™ CPLD
- Μια μνήμη DDR SDRAM 64 MByte (512 Mbit), 100+ MHz
- Μια μνήμη Flash Xilinx 4 Mbit για προγραμματισμό του FPGA
- Μια μνήμη NOR Flash 16 MByte (128 Mbit) (Intel StrataFlash)
- Μια μνήμη SPI Serial Flash 16 Mbits (STMicro). Οι μνήμες Flash μπορούν να χρησιμοποιηθούν για το προγραμματισμό του FPGA και την αποθήκευση κώδικα του επεξεργαστή MicroBlaze code shadowing
- Οθόνη LCD 2-line, 16-character
- Θύρα PS/2 για mouse ή keyboard
- Θύρα VGA
- Θύρα 10/100 Ethernet PHY
- Δύο θύρες RS-232 9-pin
- **Διασύνδεση USB για τον προγραμματισμό του FPGA**
- **Ρολόι 50 MHz**
- **Περιστροφικός διακόπτης**
- **Οκτώ LEDs**
- **Τέσσερις διακόπτες 2 θέσεων**
- **Τέσσερις πιεστικοί διακόπτες**
- Και άλλα συστατικά που δεν θα χρησιμοποιηθούν στα πλαίσια του εργαστηρίου



Εικόνα 13: Spartan-3E Microblaze Development Kit

Η πλακέτα Spartan-3E Microblaze Development Kit περιέχει τα εξής (στα πλαίσια του εργαστηρίου θα χρησιμοποιήσετε ότι είναι με bold):

- **Ένα τσιπάκι Xilinx XC3S1600E Spartan-3E FPGA** (έχει μέχρι 250 ακροδέκτες I/O και πάνω από 33,000 λογικά στοιχεία)
- Ένα τσιπάκι Xilinx 64-macrocell XC2C64A CoolRunner™ CPLD
- Μια μνήμη DDR SDRAM 64 MByte (512 Mbit), 100+ MHz
- Δύο μνήμες Flash Xilinx 4 Mbit για προγραμματισμό του FPGA
- Μια μνήμη NOR Flash 16 MByte (128 Mbit) (Intel StrataFlash)
- Μια μνήμη SPI Serial Flash 16 Mbits (STMicro). Οι μνήμες Flash μπορούν να χρησιμοποιηθούν για το προγραμματισμό του FPGA και την αποθήκευση κώδικα του επεξεργαστή MicroBlaze code shadowing
- Οθόνη LCD 2-line, 16-character
- Θύρα PS/2 για mouse ή keyboard
- Θύρα VGA
- Θύρα 10/100 Ethernet PHY
- Δύο θύρες RS-232 9-pin
- **Διασύνδεση USB για τον προγραμματισμό του FPGA**
- **Ρολόγια 50 και 66 MHz**
- **Περιστροφικός διακόπτης**
- **Οκτώ LEDs**
- **Τέσσερις διακόπτες 2 θέσεων**
- **Τέσσερις πιεστικοί διακόπτες**
- Και άλλα συστατικά που δεν θα χρησιμοποιηθούν στα πλαίσια του εργαστηρίου

1.5 Σύνθεση σχεδίασης (design synthesis)

Μετά την προσομοίωση και την επαλήθευση ότι το κύκλωμά σας λειτουργεί σωστά, το επόμενο βήμα είναι να χρησιμοποιήσετε ένα εργαλείο σύνθεσης (synthesis tool) για να μετασχηματίσετε την περιγραφή σας σε μια λίστα συνδέσεων (netlist). Μια λίστα συνδέσεων (netlist) είναι μια σχηματική αναπαράσταση που μπορεί να διαβαστεί από αυτόματα εργαλεία. Σε αυτήν την κατηγορία, θα χρησιμοποιήσουμε ένα εργαλείο που ονομάζεται XST, το οποίο είναι ενσωματωμένο στον Project Navigator και μπορεί να στοχεύσει μόνο FPGA συσκευές Xilinx.

Επιλέξτε το tab Implementation στο παράθυρο Design και κατόπιν επιλέξτε την top-level σχεδίαση του έργου (στην προκειμένη περίπτωση το κύκλωμα 4-bit parity generator). *Προσοχή: Μην επιλέξετε το testbench.*

Κατόπιν, κάντε διπλό-κλικ στη διαδικασία Synthesize - XST στο παράθυρο Processes. Το Project Navigator θα συνθέσει τη σχεδίαση και θα εμφανίσει πληροφορίες στο παράθυρο Console.

Δεν πρέπει να δείτε κανένα λάθος στο παράθυρο Console. Εντούτοις, πρέπει πάντα να διαβάζετε το αρχείο log, γιατί μπορεί να περιέχει μηνύματα που επισημαίνουν κάποιο σχεδιαστικό λάθος. Εάν δεν καταλαβαίνετε ένα συγκεκριμένο μήνυμα, μπορείτε να ψάξετε στην ιστοσελίδα υποστήριξης της Xilinx.

Διαβάζοντας την αναφορά μπορείτε επίσης να βρείτε ποιους (και πόσους) πόρους της συσκευής χρησιμοποίησε το εργαλείο σύνθεσης. Σε αυτό το σημείο, πρέπει να έχετε ένα πράσινο σημάδι (√) δίπλα στη διαδικασία Synthesize—XST.

1.6 Υλοποίηση σχεδίασης (design implementation)

Η υλοποίηση της σχεδίασης είναι η ακολουθία γεγονότων που μεταφράζει τη λίστα συνδέσεων της σχεδίασης που έχετε ήδη συνθέσει (synthesized design netlist) σε ένα αρχείο προγραμματισμού για τη συσκευή FPGA. Η περιγραφή του κυκλώματός σας, που έχετε συνθέσει τώρα, έχει έναν αριθμό θυρών (ports) στο υψηλότερο επίπεδο (top level). Τα εργαλεία υλοποίησης (implementation tools) πρέπει να γνωρίζουν πώς θα αναθέσουν τις θύρες στο υψηλότερο επίπεδο της σχεδίασής σας στους φυσικούς ακροδέκτες (pins) του FPGA, οι οποίοι συνδέονται με διάφορους πόρους της πλακέτας Spartan-3E. Εάν δεν ορίσετε ρητές αναθέσεις, τα εργαλεία θα ορίσουν τυχαία τους ακροδέκτες για σας. Προφανώς, αυτό είναι μια κακή ιδέα αφού οι τυχαίες αναθέσεις θα είναι λανθασμένες.

Το υψηλότερο επίπεδο της σχεδίασης του παραδείγματος έχει 4 θύρες εισόδου (a, b, c, d), και μια θύρα εξόδου (p). Άρα, θέλουμε να **έχουμε 4 διακόπτες, SW0, SW1, SW2 και SW3**, που συνδέονται με τις εισόδους. Επιπλέον, θέλουμε την έξοδο να συνδέεται με μια ενδεικτική λυχνία (LED) έτσι ώστε να μπορούμε να την παρατηρήσουμε – το **LD0** είναι κατάλληλο για αυτόν τον σκοπό.

Εάν επιθεωρήσετε την πάνω πλευρά της πλακέτας σας, θα παρατηρήσετε ότι σχεδόν κάθε πόρος έχει σχολιαστεί με κάποιο κείμενο που προσδιορίζει με ποιους ακροδέκτες του FPGA συνδέεται. Αυτές οι πληροφορίες είναι επίσης διαθέσιμες στον οδηγό χρήσης της πλακέτας (User Guide). Προσπαθήστε να προσδιορίσετε στην πλακέτα σας ποιοι ακροδέκτες του FPGA χρησιμοποιούνται για τα SW0, SW1, SW2, SW3 και LD0, και ελέγξτε έπειτα τα αποτελέσματά σας με αυτά που παρουσιάζονται παρακάτω:

Για την Spartan-3E Starter Kit	Για την Spartan-3E Microblaze Development Kit
SW0 → FPGA Pin L13	SW0 → FPGA Pin L13
SW1 → FPGA Pin L14	SW1 → FPGA Pin L14
SW2 → FPGA Pin H18	SW2 → FPGA Pin H18
SW3 → FPGA Pin N17	SW3 → FPGA Pin N17
LD0 → FPGA Pin F12	LD0 → FPGA Pin D4

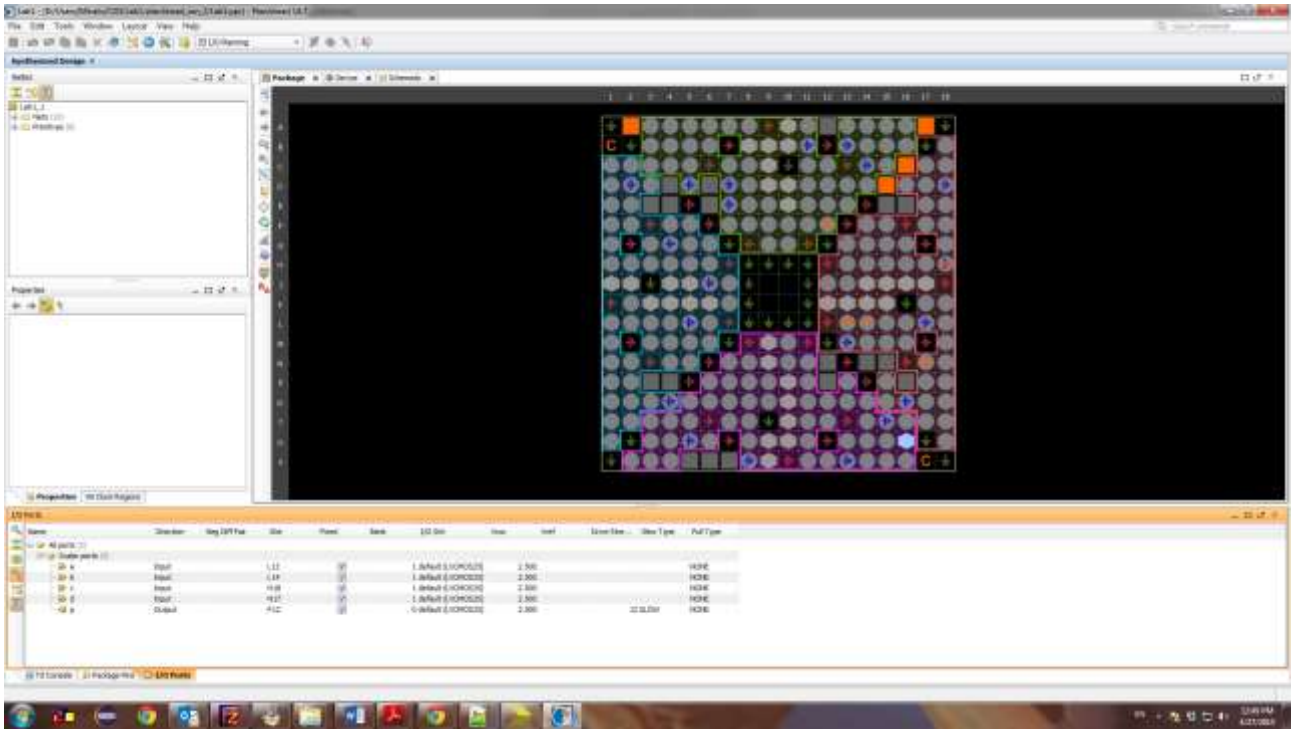
Έχετε τώρα αρκετές πληροφορίες για να δημιουργήσετε αυτό που ονομάζεται αρχείο περιορισμών του χρήστη **(user constraint file), ή UCF**. Αυτό το αρχείο περιέχει τους περιορισμούς της σχεδίασης που δεν καθορίσατε στην περιγραφή VHDL, όπως οι περιορισμοί θέσης των ακροδεκτών (pin location) και

απόδοσης της σχεδίασης (design performance). Είναι βολικό να παρασχεθούν σε ένα UCF παρά στην VHDL περιγραφή. Για παράδειγμα, εάν κάνετε ένα λάθος στις αναθέσεις των ακροδεκτών, δεν χρειάζεται να επιστρέψετε και να επανασυνθέσετε το κύκλωμά σας.

Μπορείτε να προσθέσετε ένα UCF στο έργο χρησιμοποιώντας την ίδια διαδικασία που χρησιμοποιήσατε για την προσθήκη της σχεδίασης και του testbench. Δημιουργήστε ένα νέο πηγαίο αρχείο: επιλέξτε Project→New Source από το κυρίως μενού. Στη συνέχεια, επιλέξτε Implementation Constraints File για να δηλώσετε ότι θέλετε να δημιουργήσετε ένα αρχείο περιορισμών και δώστε ένα όνομα αρχείου, π.χ. Lab1_1. Πατήστε Next.

Το User Constraints File (UCF) είναι ένα αρχείο κειμένου στο οποίο μπορεί άμεσα να γράψει ο χρήστης μέσω του ενσωματωμένου κειμενογράφου του ISE. Επίσης, ένας άλλος εύκολος τρόπος για την επεξεργασία του αρχείου UCF είναι με την βοήθεια δύο γραφικών εργαλείων, του Constraint Editor και του PlanAhead™. Χρησιμοποιώντας το PlanAhead είναι δυνατόν να κάνουμε τις αντιστοιχίες των I/O Pins του FPGA με τα σήματα της σχεδίασης μας. Το PlanAhead είναι ένα αρκετά ισχυρό εργαλείο με πολλές δυνατότητες αλλά θα χρησιμοποιήσουμε μόνο μια μικρή μερίδα των δυνατοτήτων του στο εργαστήριο.

Για να ξεκινήσουμε το PlanAhead, έχοντας ενεργοποιήσει το tab Implementation του μενού Design, επεκτείνουμε το μενού User Constraints και επιλέγουμε την εντολή I/O Pin Planning (PlanAhead) – Post-Synthesis. Όπως παρατηρήσατε υπάρχει και η επιλογή I/O Pin Planning (PlanAhead) – Pre-Synthesis. Είναι προτιμότερο να χρησιμοποιείτε την επιλογή Post-Synthesis, δεδομένου ότι μετά την σύνθεση παρέχεται περισσότερη πληροφορία για τα I/O του κάθε design που έχετε δημιουργήσει. Η Εικόνα 14 παρουσιάζει το πρόγραμμα PlanAhead.



Εικόνα 14: PlanAhead Software

Στο μενού I/O Ports επεκτείνετε την επιλογή Scalar Ports κάτω από το All Ports. Σε αυτό το σημείο θα πρέπει να βλέπετε τις τέσσερις εισόδους (a, b, c, d) και τη μία έξοδο (p) της σχεδίασής σας. Επιλέξτε μία προς μία τις εισόδους και τις εξόδους της σχεδίασής σας και πραγματοποιήστε drag and drop στα αντίστοιχα pin του Package Tool.

- a → FPGA Pin L13
- b → FPGA Pin L14

- c → FPGA Pin H18
- d → FPGA Pin N17
- p → FPGA Pin F12

Αποθηκεύστε τις αλλαγές σας και κλείστε το εργαλείο PlanAhead. Επιστρέψτε στο Project Navigator, επιλέξτε το αρχείο Lab1_1.ucf και εκτελέστε την εντολή Edit Constraints (Text) ώστε να επαληθεύσετε τις αντιστοιχίες των ακροδεκτών που μόλις κάνατε. Σε αυτό το σημείο πρέπει να δείτε τα παρακάτω:

```
# PlanAhead Generated physical constraints


NET "a" LOC = L13;
NET "b" LOC = L14;
NET "c" LOC = H18;
NET "d" LOC = N17;
NET "p" LOC = F12;
```

Τώρα που έχετε ένα αρχείο περιορισμών στο έργο σας, μπορείτε να υλοποιήσετε τη σχεδίαση. Κατόπιν, κάντε διπλό-κλικ στη διεργασία Implement Design στο παράθυρο Processes. Το Project Navigator θα υλοποιήσει τη σχεδίαση και θα εμφανίσει πληροφορίες στο παράθυρο Console. Σαν ενημερωτική σημείωση, είναι δυνατό να αλλάξετε τις επιλογές υλοποίησης κάνοντας δεξί κλικ στο Implement Design και έπειτα επιλέγοντας Process Properties. Για αυτό το εργαστήριο, αφήστε τις επιλογές στις προεπιλεγμένες ρυθμίσεις τους.

Δεν πρέπει να δείτε κανένα λάθος στο παράθυρο Console. Εντούτοις, πρέπει πάντα να διαβάζετε τα τρία αρχεία log, τα οποία είναι διαθέσιμα κάνοντας δεξί click πάνω στα Translate, Map, και Place and Route και εκτελώντας την εντολή View Text Report. Εάν δεν καταλαβαίνετε ένα συγκεκριμένο μήνυμα, ψάξτε στην ιστοσελίδα υποστήριξης της Xilinx. Σε αυτό το σημείο, πρέπει να έχετε ένα πράσινο σημάδι (√) δίπλα στη διεργασία Implement Design.

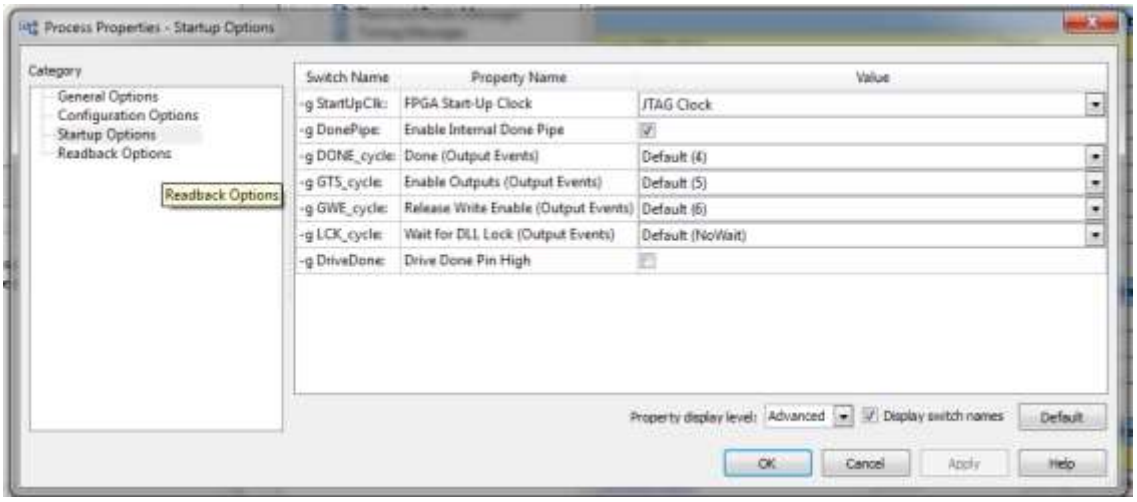
1.7 Προγραμματισμός του FPGA (FPGA programming)

Σε αυτό το σημείο, είστε έτοιμοι να προγραμματίσετε το FPGA με τη σχεδίασή σας. Η πλακέτα σας μπορεί να προγραμματιστεί με δύο διαφορετικές μεθόδους:

- Να προγραμματιστεί το FPGA με ένα usb καλώδιο (download cable).
- Να προγραμματιστεί η συσκευή PROM με το usb καλώδιο (download cable), και έπειτα να έχουμε την PROM να προγραμματίζει το FPGA.

Θα χρησιμοποιήσουμε την πρώτη μέθοδο, που είναι ένας βολικός τρόπος για να δοκιμαστεί μια σχεδίαση. Αυτή η μέθοδος είναι χρήσιμη όταν θέλετε να δοκιμάσετε γρήγορα κάτι, ή αν δεν είστε σίγουροι ότι η σχεδίασή σας είναι τελική. Για παράδειγμα, σε αυτό το σημείο είστε αρκετά βέβαιοι ότι το κύκλωμά σας είναι σωστό. Εντούτοις, πρέπει να κατανοήσετε ότι τα σύνθετα κυκλώματα σπάνια δουλεύουν με την πρώτη δοκιμή. Ένα από τα μεγάλα πλεονεκτήματα των FPGAs έναντι των ASICs είναι ότι το κόστος που πληρώνετε εάν έχετε κάνει λάθος στην πρώτη δοκιμή είναι ελάχιστο.

Το πρώτο βήμα είναι να δημιουργήσετε ένα αρχείο προγραμματισμού για το Στο παράθυρο Processes, κάντε δεξί-κλικ στο Generate Programming File έπειτα επιλέξτε Process Properties. Το πλαίσιο διαλόγου Process Properties εμφανίζεται. Επιλέξτε την ετικέτα Startup Options, όπως φαίνεται στην Εικόνα 15.

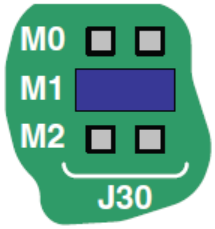


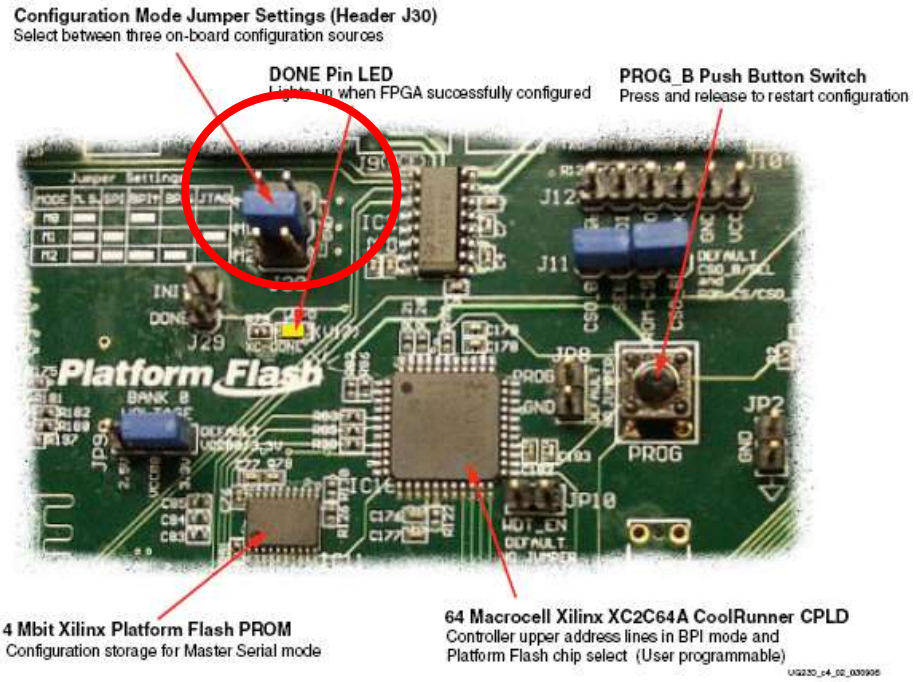
Εικόνα 15: Programming File Properties (Startup Options)

Αλλάξτε την επιλογή FPGA Start-Up Clock σε JTAG Clock. Οι άλλες ρυθμίσεις πρέπει ήδη να είναι σωστές, αλλά σιγουρευτείτε ότι ταιριάζουν με την Εικόνα 15. Πατήστε OK για να αποθηκεύσετε τις ρυθμίσεις.

Έπειτα, κάντε διπλό-κλικ στην διαδικασία Generate Programming File στο παράθυρο Processes. Το Project Navigator θα παράγει ένα αρχείο προγραμματισμού και θα εμφανίσει πληροφορίες στο παράθυρο Console.

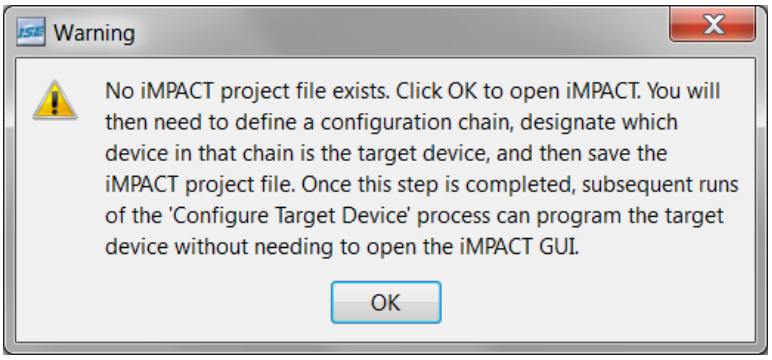
Πριν συνεχίσετε, πρέπει να έχετε διαθέσιμα την πλακέτα, την τροφοδοσία, και το καλώδιο προγραμματισμού. Συνδέστε το καλώδιο προγραμματισμού στην usb θύρα του PC που χρησιμοποιείτε. Ελέγξτε αν οι διακλαδωτήρες (jumpers) J30 της πλακέτας Spartan-3E είναι τοποθετημένοι σωστά σύμφωνα με την Εικόνα 16 ώστε ο προγραμματισμός να πραγματοποιηθεί μέσω του JTAG.





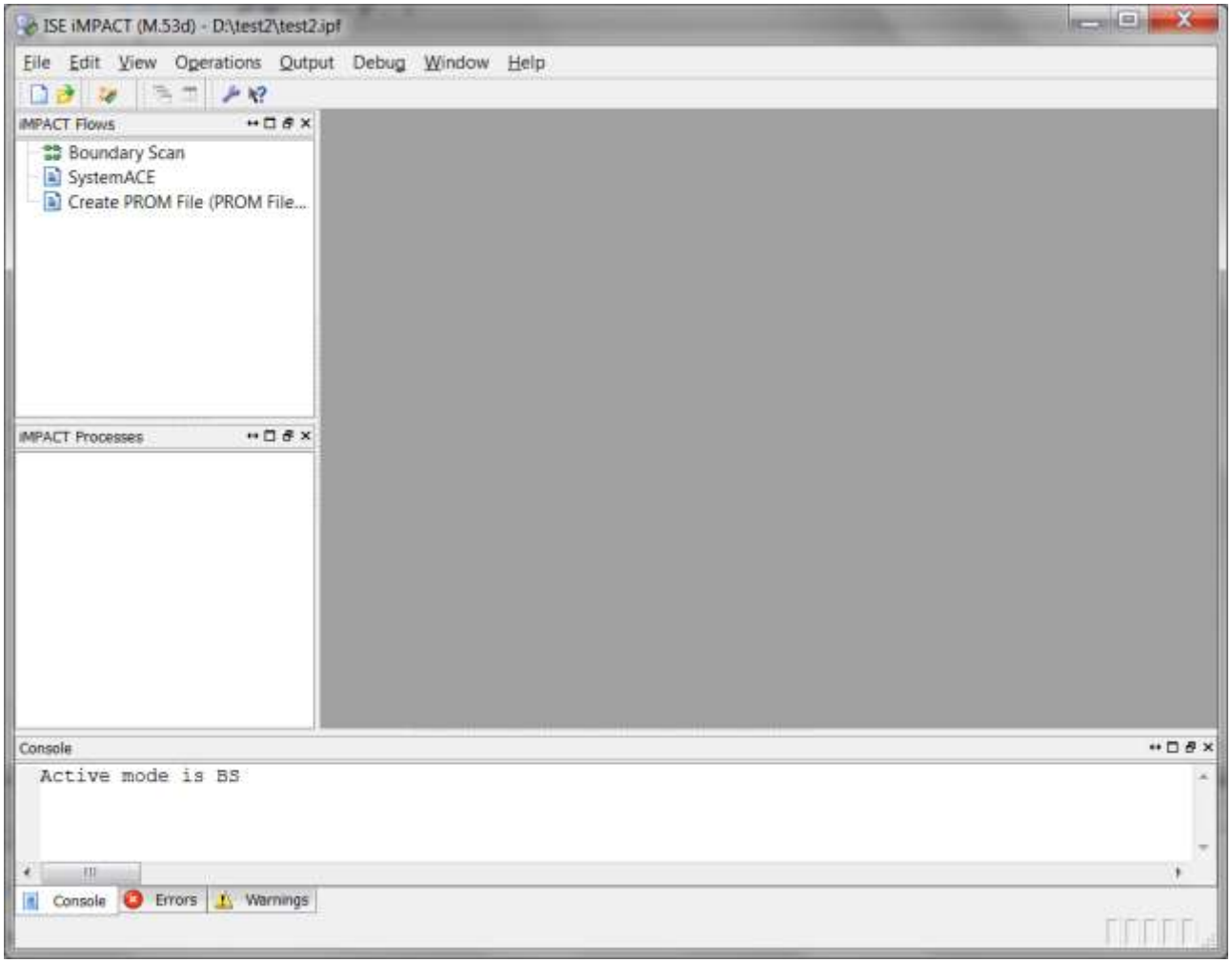
Εικόνα 16: Τοποθέτηση jumper της αναπτυξιακής πλακέτας

Για να κατεβάσετε (προγραμματίσετε) το αρχείο προγραμματισμού στη συσκευή FPGA, κάντε διπλό click στο Configure Target Device. Αυτό θα φορτώσει το πρόγραμμα ISE iMPACT σε άλλο παράθυρο. Την πρώτη φορά που θα εκτελέσετε αυτή την εντολή θα λάβετε ένα μήνυμα όπως αυτό που φαίνεται στην Εικόνα 17 σχετικά με τον καθορισμό του iMPACT αρχείου.



Εικόνα 17: Μήνυμα σχετικά με το iMPACT αρχείο

Κατόπιν θα εμφανιστεί στην οθόνη σας ένα νέο εργαλείο, το iMPACT όπως φαίνεται στην Εικόνα 18 το οποίο σας επιτρέπει να προγραμματίσετε το FPGA.



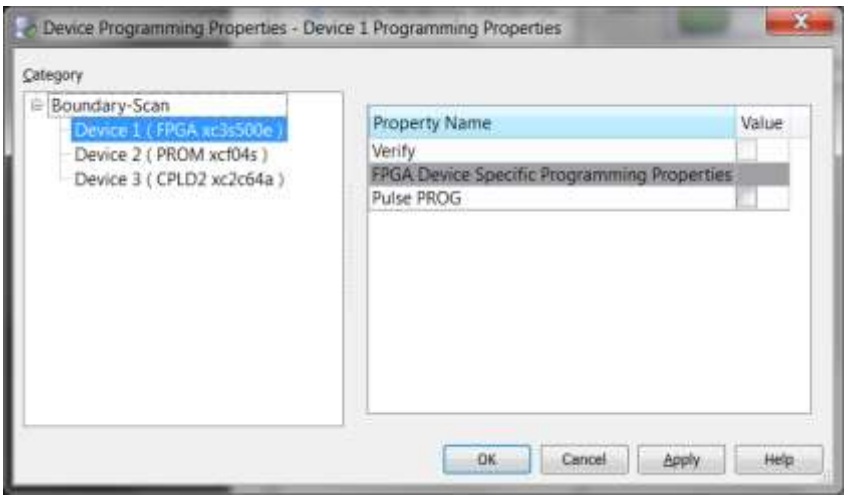
Εικόνα 18: Εργαλείο προγραμματισμού iMPACT

Κάντε διπλό click στο boundary Scan και έπειτα δεξί click στο Right Click to add Device or Initialize JTAG chain και δώστε την εντολή Initialize Chain.

Έπειτα, πρέπει να σας ζητηθούν τρία αρχεία. Για το πρώτο αρχείο επιλέξτε το Lab1_1.bit που δημιουργήσατε με τη διαδικασία υλοποίησης. Αυτό είναι το αρχείο προγραμματισμού του FPGA. Για το δεύτερο αρχείο, σας ζητείτε ένα αρχείο προγραμματισμού της PROM. Δεν προγραμματίζουμε την PROM αυτήν τη στιγμή, επομένως επιλέξτε Bypass όπως ακριβώς και στο τρίτο πλαίσιο διαλόγου.

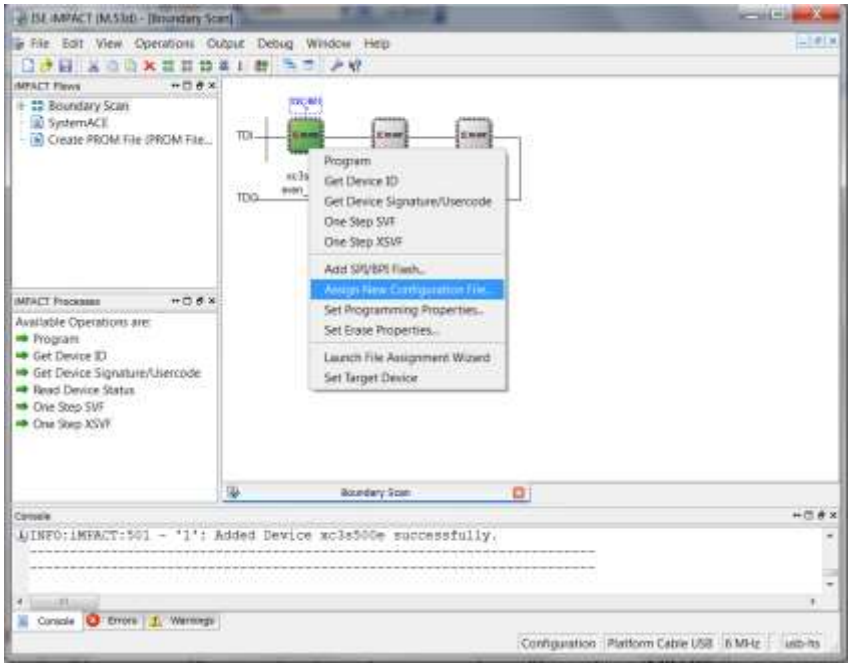
Σημείωση: Εάν δεν σας ζητηθούν σε αυτό το σημείο τα δύο αρχεία, αγνοήστε το και συνεχίστε με τα παρακάτω βήματα.

Στο τέλος της διαδικασίας λαμβάνεται ένα πλαίσιο διαλόγου όπως αυτό που φαίνεται στην Εικόνα 19.



Εικόνα 19: Τελικές ρυθμίσεις προγραμματισμού του FPGA

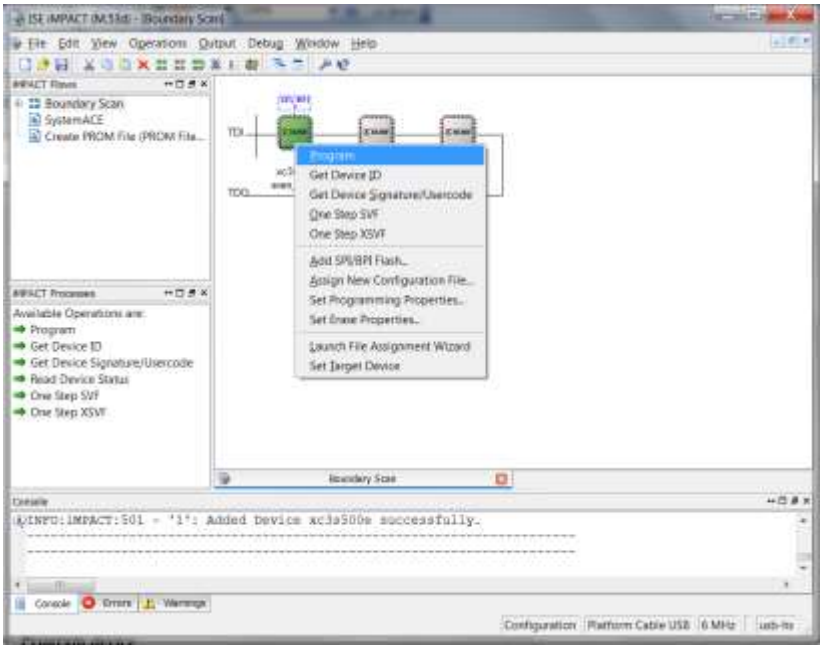
Σε αυτό το σημείο, είστε έτοιμοι να προγραμματίσετε το FPGA. Εάν έχετε κάνει κάποιο λάθος, μπορείτε να διορθώσετε τις αναθέσεις σας με τη χρησιμοποίηση της τεχνικής που απεικονίζεται στην Εικόνα 20.



Εικόνα 20: Εναλλακτική μέθοδος ανάθεσης των αρχείων προγραμματισμού

Για να διορθώσετε μια ανάθεση αρχείου, ή να κάνετε τις αρχικές αναθέσεις εάν το iMPACT δεν σας προτρέψει αυτόματα για τα αρχεία, επιλέξτε το εικονίδιο του FPGA στο παράθυρο iMPACT. Κάντε δεξί κλικ και επιλέξτε Assign New Configuration File. Θα σας ζητηθεί ένα αρχείο όπως αναφέραμε παραπάνω.

Τέλος, θα φθάσετε στο σημείο που φαίνεται στην Εικόνα 21. Το iMPACT είναι έτοιμο να προγραμματίσει το FPGA. Επιλέξτε το εικονίδιο του FPGA στο παράθυρο και έπειτα χρησιμοποιήστε το δεξί κουμπί του ποντικιού για να ενεργοποιήσετε το μενού και επιλέξτε την επιλογή Program.



Εικόνα 21: Program device

Ένας δείκτης προόδου θα εμφανιστεί. Μόλις ο προγραμματισμός ολοκληρωθεί, το πρόγραμμα θα σας ενημερώσει εάν ο προγραμματισμός πέτυχε ή απέτυχε. Εάν ο προγραμματισμός έχει αποτύχει, επανελέγξτε τις συνδέσεις των καλωδίων σας, τη σύνδεση τροφοδοσίας και προσπαθήστε ξανά. Εάν συνεχίσει να αποτυγχάνει, ζητήστε τη βοήθεια του καθηγητή.

Τώρα, μπορείτε να δοκιμάσετε τη σχεδίασή σας στο υλικό. Εντοπίστε τους διακόπτες SW0-SW3 στην πλακέτα, και εξετάστε το κύκλωμά σας δοκιμάζοντας τους 16 πιθανούς συνδυασμούς των τιμών των διακοπτών και παρατηρώντας το LD0. Το κύκλωμα σας συμπεριφέρεται όπως αναμένετε; Εάν όχι, ζητήστε τη βοήθεια του καθηγητή. Εάν λειτουργεί σωστά, έχετε ολοκληρώσει με επιτυχία την άσκηση.

Εργαστηριακή Άσκηση 2: Υλοποίηση κυκλωμάτων

Σε αυτήν την εργαστηριακή άσκηση θα ασχοληθείτε με την σχεδίαση διάφορων συνδυαστικών και ακολουθιακών κυκλωμάτων. Πιο συγκεκριμένα θα σχεδιάσετε:

- Έναν αποκωδικοποιητή 3-σε-8
- Έναν συγκριτή δύο αριθμών των 2-bit
- Έναν δυαδικό μετρητή
- Ένα κύκλωμα διαίρεσης συχνότητας
- Έναν συσσωρευτή (accumulator)

Σημείωση: Υιοθετήστε τα ονόματα των οντοτήτων (entities) και των θυρών (ports) που αναφέρονται στην εκφώνηση της άσκησης για να μην έχετε προβλήματα ασυμβατότητας με τα vhdl αρχεία που σας δίδονται.

Κύκλωμα 1: Αποκωδικοποιητής 3-σε-8

Σχεδιάστε έναν αποκωδικοποιητή 3-σε-8 (3x8 decoder) χρησιμοποιώντας τους διακόπτες SW2-SW0 ως εισόδους και τα leds LED7-LED0 ως εξόδους.

Σημείωση: Ένας αποκωδικοποιητής n-σε-2^n για κάθε συνδυασμό εισόδου ενεργοποιεί (θέτει στην ενεργή τιμή, π.χ. 1) μία έξοδο και απενεργοποιεί (θέτει στο 0) όλες τις υπόλοιπες.

- Δημιουργήστε ένα νέο project.
- Δημιουργήστε ένα νέο αρχείο (vhdl module) με όνομα lab3_decoder_3x8.vhd.
- Η οντότητα της μονάδας είναι η εξής:

```
entity lab3_decoder_3x8 is
   port (
        d : in STD_LOGIC_VECTOR (2 downto 0);
        q : out STD_LOGIC_VECTOR (7 downto 0)
   );
end lab3_decoder_3x8;
```

- Προσθέστε στο αρχείο τις παρακάτω βιβλιοθήκες (περιέχουν συναρτήσεις για τον τύπο δεδομένων std_logic):

```
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
```

- Εισάγετε την σχεδίαση του αποκωδικοποιητή χρησιμοποιώντας είτε την εντολή ανάθεσης **with-select** είτε την εντολή ανάθεσης **when-else**.
- Εισάγετε στο project το testbench lab3_decoder_3x8_tb.vhd. Σημείωση: Μπορείτε να κατεβάσετε το testbench από την ιστοσελίδα του μαθήματος (περιέχεται στο Lab3.zip).
- Προσομοιώστε το κύκλωμα.
- Υλοποιήστε το κύκλωμα εισάγοντας το κατάλληλο ucf file.
- Προγραμματίστε την πλακέτα και ελέγξτε την λειτουργία του κυκλώματος.

Κύκλωμα 2: Συγκριτής δυαδικών αριθμών

Σχεδιάστε έναν συγκριτή 2 απρόσημων δυαδικών αριθμών των 2-bit. Το κύκλωμα θα συγκρίνει τους δυαδικούς αριθμούς A=A1A0 (για την είσοδο A χρησιμοποιήστε τους διακόπτες SW3, SW2) και B=B1B0 (για την είσοδο B χρησιμοποιήστε τους διακόπτες SW1, SW0) και θα παράγει 3 εξόδους: LT (μικρότερο από, LED2), GT (μεγαλύτερο από, LED1) και EQ (ίσο, LED0).

- Σχεδιάστε το κύκλωμα χρησιμοποιώντας την εντολή **if-then-else**.
- Συνθέστε και υλοποιήστε το κύκλωμα εισάγοντας το κατάλληλο ucf file.
- Προγραμματίστε την πλακέτα και ελέγξτε την λειτουργία του κυκλώματος.

Κύκλωμα 3: Δυαδικός μετρητής των 4-bit

Σχεδιάστε έναν δυαδικό μετρητή των 4-bit που έχει τις εξής λειτουργίες: (α) μηδενίζεται όταν ενεργοποιηθεί το σήμα εισόδου RESET, (β) μετράει προς τα πάνω ή προς τα κάτω ανάλογα με την τιμή ενός σήματος εισόδου UP_DOWN και (γ) παγώνει όταν ενεργοποιηθεί το σήμα εισόδου FREEZE. Για τις εισόδους και εξόδους του κυκλώματος να χρησιμοποιήσετε τα παρακάτω:

- RESET: BTN_SOUTH
- FREEZE: BTN_NORTH
- UP_DOWN: SW0
- COUNT (counter outputs): LED3-LED0
- CLK: για το ρολόι του μετρητή να χρησιμοποιήσετε το 50MHz ρολόι της πλακέτας

- Δημιουργήστε ένα νέο project.
- Δημιουργήστε ένα νέο αρχείο (vhdl module) με όνομα counter_4b.vhd.
- Η οντότητα της μονάδας είναι η εξής:

```
entity counter_4b is
   port (
        clk, reset      : in STD_LOGIC;
        freeze, up_down : in STD_LOGIC;
        count           : out unsigned(3 downto 0)
   );
end counter_4b;
```

- Σχεδιάστε την αρχιτεκτονική του μετρητή (Σημείωση: Συμβουλευτείτε τις διαφάνειες του μαθήματος).
- Δημιουργήστε ένα vhdl testbench με όνομα counter_4b_tb.vhd και συσχετίστε το με την μονάδα counter_4b. Το ρολόι θα πρέπει να έχει περίοδο 20ns (συχνότητα 50 MHz). Προσομοιώστε το κύκλωμα.
- Υλοποιήστε το κύκλωμα εισάγοντας το κατάλληλο ucf file.
- Προγραμματίστε την πλακέτα και δοκιμάστε όλες τις λειτουργίες του κυκλώματος. Τι παρατηρείτε;

Κύκλωμα 4: Διαιρέτης ρολογιού

Σε αυτήν την άσκηση θα χρησιμοποιήσετε το αρχείο freq_div.vhd. Η οντότητα freq_div διαιρεί την συχνότητα του ρολογιού.

Σημείωση: Μπορείτε να κατεβάσετε τα αρχεία freq_div.vhd και top_level_counter.vhd από την ιστοσελίδα του μαθήματος (περιέχεται στο Lab4.zip).

- Προσθέστε στο project της Άσκησης 1 το αρχείο freq_div.vhd
- Προσθέστε στο project το αρχείο top_level_counter.vhd. Σημείωση: Το αρχείο έχει την ίδια οντότητα (ports) με το counter_4b και περιέχει τα στιγμιότυπα (instances) του counter_4b και του freq_div. Ουσιαστικά συνδέει στον μετρητή το διαιρεμένο ρολόι αντί του ρολογιού των 50MHz της πλακέτας.
- Μπορείτε να ρυθμίσετε τον διαιρέτη της συχνότητας ανάλογα με την τιμή της σταθεράς CLK_DIVISOR. Το ρολόι εισόδου έχει οριστεί ως 50MHz και το ρολόι εξόδου 2Hz.
- Συνθέστε και υλοποιήστε το κύκλωμα χρησιμοποιώντας το ίδιο ucf file.
- Προγραμματίστε την πλακέτα και ελέγξτε την λειτουργία του κυκλώματος.
- Μπορείτε να αλλάξετε τον διαιρέτη του ρολογιού σε άλλη συχνότητα εξόδου και να ελέγξετε την λειτουργία του κυκλώματος.

Κύκλωμα 5: Συσσωρευτής των 4-bit

Σχεδιάστε έναν συσσωρευτή των 4-bit (4-bit accumulator) που εκτελεί την πράξη: ACC = ACC + DIN. Η είσοδος DIN έχει μέγεθος 4 bit και η έξοδος ACC έχει μέγεθος 8 bit. Το κύκλωμα θα πρέπει να έχει τις εξής λειτουργίες: (α) μηδενίζεται όταν ενεργοποιηθεί το σήμα εισόδου RESET, (β) έχει είσοδο επίτρεψης (ENABLE) και διαβάζει την είσοδο DIN μόνο όταν η είσοδος ENABLE είναι 1. Για τις εισόδους και εξόδους του κυκλώματος να χρησιμοποιήσετε τα παρακάτω:

- RESET: BTN_SOUTH
- ENABLE: BTN_NORTH
- DIN (accumulator inputs): SW3-SW0
- COUNT (counter outputs): LED7-LED0
- CLK: on-board 50MHz clock (Προσοχή: Μην χρησιμοποιήσετε το διαιρεμένο ρολόι)

Σημείωση: Για κάθε νέα είσοδο που θέλετε να προσθέσει ο συσσωρευτής θα πρέπει να ενεργοποιήσετε την είσοδο (push button) ENABLE αφού πρώτα θέσετε την είσοδο DIN στους διακόπτες SW3-SW0. Για κάθε πάτημα του ENABLE ο συσσωρευτής θα πρέπει να αντιλαμβάνεται μόνο ένα παλμό ρολογιού (των 50MHz) ώστε να διαβάζει και να προσθέτει μόνο μία τιμή.

- Δημιουργήστε ένα νέο project.
- Δημιουργήστε ένα νέο αρχείο (vhdl module) με όνομα acc_4b.vhd.
- Η οντότητα της μονάδας είναι η εξής:

```
entity acc_4b is
   port (
        clk, reset      : in STD_LOGIC;
        enable          : in STD_LOGIC;
```

```
        din             : in unsigned(3 downto 0);
        acc             : out unsigned(7 downto 0)
    );
end acc_4b;
```

- Σχεδιάστε την αρχιτεκτονική του συσσωρευτή.
- Δημιουργήστε ένα vhdl testbench με όνομα acc_4b_tb.vhd και συσχετίστε το με την μονάδα acc_4b. Το ρολόι θα πρέπει να έχει περίοδο 20ns (συχνότητα 50 MHz). Προσομοιώστε το κύκλωμα.
- Υλοποιήστε το κύκλωμα εισάγοντας το κατάλληλο ucf file.
- Προγραμματίστε την πλακέτα και δοκιμάστε την λειτουργία του κυκλώματος.