Σχεδίαση Ψηφιακών Κυκλωμάτων

Μεθοδολογίες σχεδίασης και η ροή λογικής σύνθεσης κυκλωμάτων σε FPGA

Νικόλαος Καββαδίας
nkavv@uop.gr

19 Ιανουαρίου 2011

Σκιαγράφηση της διάλεξης

- Η τυπική ροή της λογικής σχεδίασης
- Λογική σύνθεση ψηφιακών κυκλωμάτων (επιλογή υλικού από διαλέξεις **Χρ. Καβουσιανού**, Λέκτορα Πανεπ. Ιωαννίνων)
  - Διαδικασίες που λαμβάνουν χώρα στο frontend
  - Διαδικασίες που λαμβάνουν χώρα στο backend
- Η αναπτυξιακή πλακέτα Xilinx Spartan-3 Starter Kit
- Παραδείγματα με τη ροή Xilinx (ISE Webpack)

Λογική σύνθεση

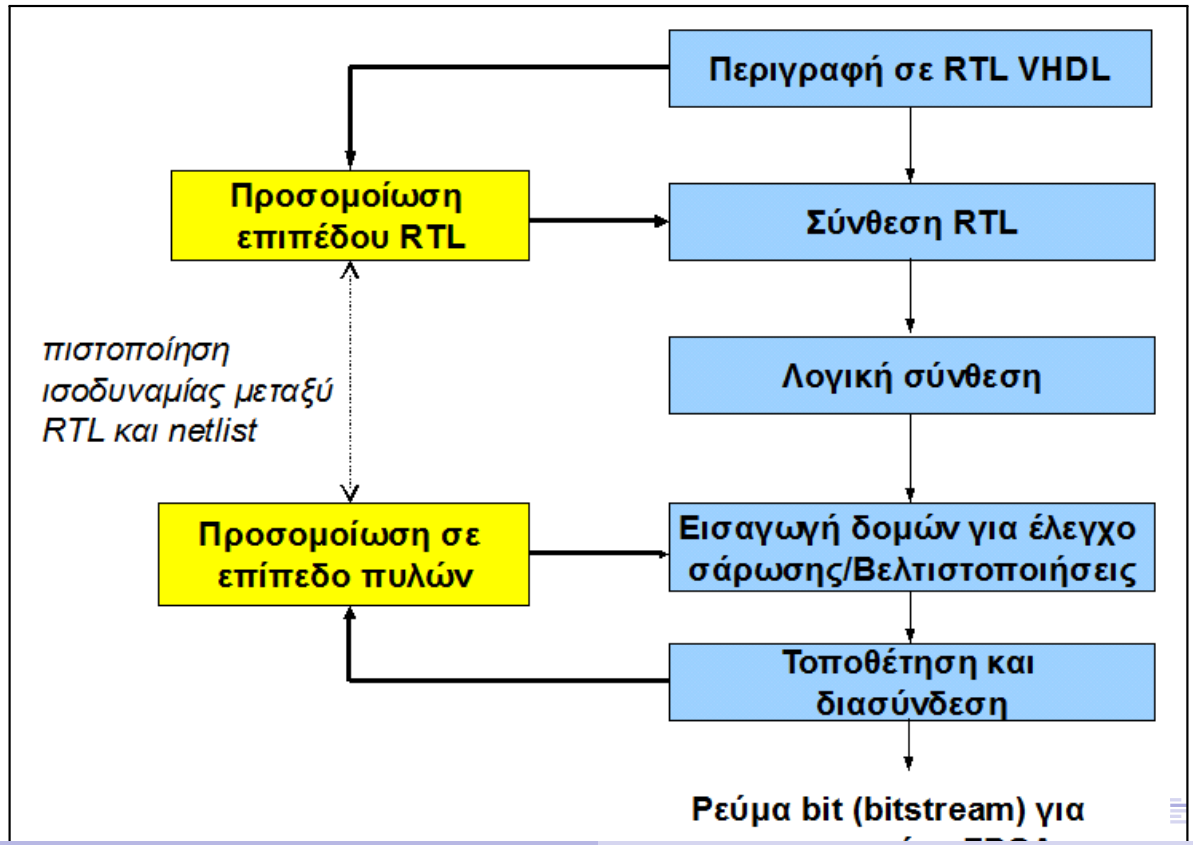
- Σκοπός της περιγραφής ψηφιακών κυκλωμάτων είναι η υλοποίησή τους σε ολοκληρωμένο
- Η λογική σύνθεση είναι είδος μεταγλώττισης από το υψηλό επίπεδο μιας HDL (δομική, RTL ή μικτή) περιγραφής στο χαμηλό επίπεδο της λίστας κόμβων (netlist) με τα στοιχειώδη κυκλωματικά στοιχεία της τεχνολογίας
- Δημοφιλείς τεχνολογίες: διεργασίες τυποποιημένου κελιού (standard cell VLSI), FPGA
- Περιορισμοί στον τρόπο σχεδιασμού με μια HDL ώστε η τελική περιγραφή/κώδικας να είναι συνθέσιμη
- Κατάλληλες τεχνικές στην ανάπτυξη του κώδικα οδηγούν στην επίτευξη καλύτερων επιδόσεων (ως προς ταχύτητα επεξεργασίας, επιφάνεια ολοκληρωμένου, κατανάλωση ισχύος/ενέργειας)
- Για τη σύνθεση χρησιμοποιούνται εμπορικά εργαλεία (ISE Webpack, LeonardoSpectrum, Synopsys DC) ή εργαλεία ανοικτού κώδικα (ABC, Alliance, OCEAN, Signs, VPR) με ισχυρούς αυτοματισμούς (βελτιστοποίηση άκυκλων γράφων, παραγωγή διανυσμάτων ελέγχου, κ.α.)

Επιλογή της κατάλληλης γλώσσας περιγραφής υλικού: VHDL και Verilog HDL

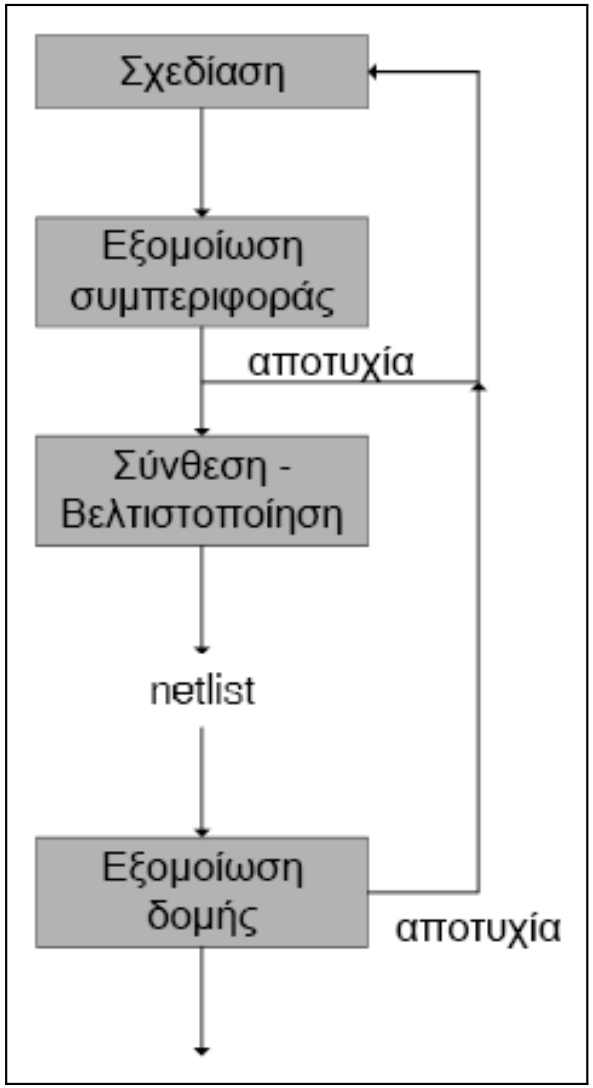
- Η VHDL αρχικά σχεδιάστηκε με σκοπό την **αυτοτεκμηρίωση (self-documentation) ψηφιακών συστημάτων**
- Με την ανάπτυξη κατάλληλων εργαλείων λογισμικού χρησιμοποιήθηκε για την **προσομοίωση** και τη **λογική σύνθεση** κυκλωμάτων
- Εν γένει, οποιοδήποτε κύκλωμα μπορεί να μοντελοποιηθεί σε VHDL μπορεί να μοντελοποιηθεί και στη Verilog HDL και το αντίστροφο
- Βασικά κριτήρια στην επιλογή γλώσσας περιγραφής υλικού (HDL) στην ψηφιακή σχεδίαση είναι:
  - Διαθεσιμότητα εργαλείων ανάπτυξης
  - Δυνατότητα επαναχρησιμοποίησης υπάρχοντος κώδικα
  - Υποκειμενικά κριτήρια όπως οικειότητα με τις συντακτικές δομές της γλώσσας

Η ροή της διαδικασίας λογικής σύνθεσης

- Η συγκεκριμένη ροή εστιάζει στις τεχνολογίες FPGA. Τα περισσότερα εργαλεία CAD χωρίζονται σε frontend και backend
  - frontend: μετατροπή της περιγραφής του κυκλώματος σε netlist
  - backend: λογική σύνθεση της netlist



Διαδικασίες στο frontend



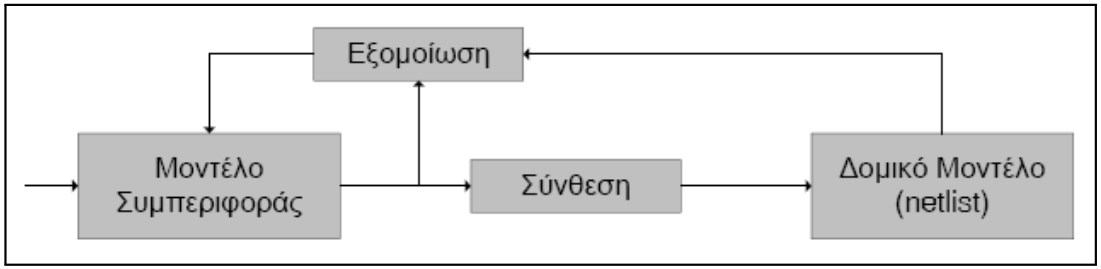
- Στόχος: η δημιουργία της netlist που υλοποιεί το κύκλωμα που επιθυμούμε
- Η netlist είναι μία λίστα διασυνδεδεμένων λογικών κυττάρων
- Οι διασυνδέσεις δεν είναι πραγματικές αλλά λογικές (περιγράφουν πως θα διασυνδεθούν τα κύτταρα)
- Η φυσική διασύνδεσή τους ακολουθεί την τοποθέτησή τους στην επιφάνεια του ολοκληρωμένου

Διαδικασίες στο backend

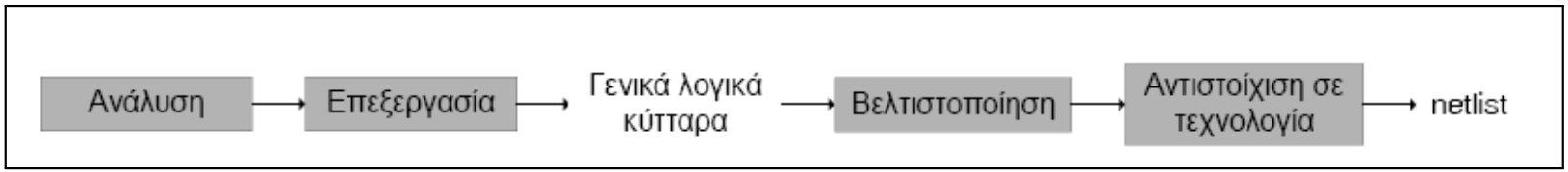
- Σημασιολογική ανάλυση της netlist (κατανόηση και αποδόμηση της πληροφορίας της)
- Διαμέριση (partitioning)
- Χωροθέτηση (floorplanning)
- Τοποθέτηση (placement)
- Διασύνδεση (routing)

Λογική σύνθεση

- Η σύνθεση λαμβάνει ως είσοδο μία συνθέσιμη περιγραφή HDL και παράγει μία λίστα (netlist) από κύτταρα βιβλιοθήκης, με τις διασυνδέσεις τους



- Ένα από τα μεγάλα πλεονεκτήματα της σύνθεσης είναι η δυνατότητα βελτιστοποίησης του τελικού κυκλώματος

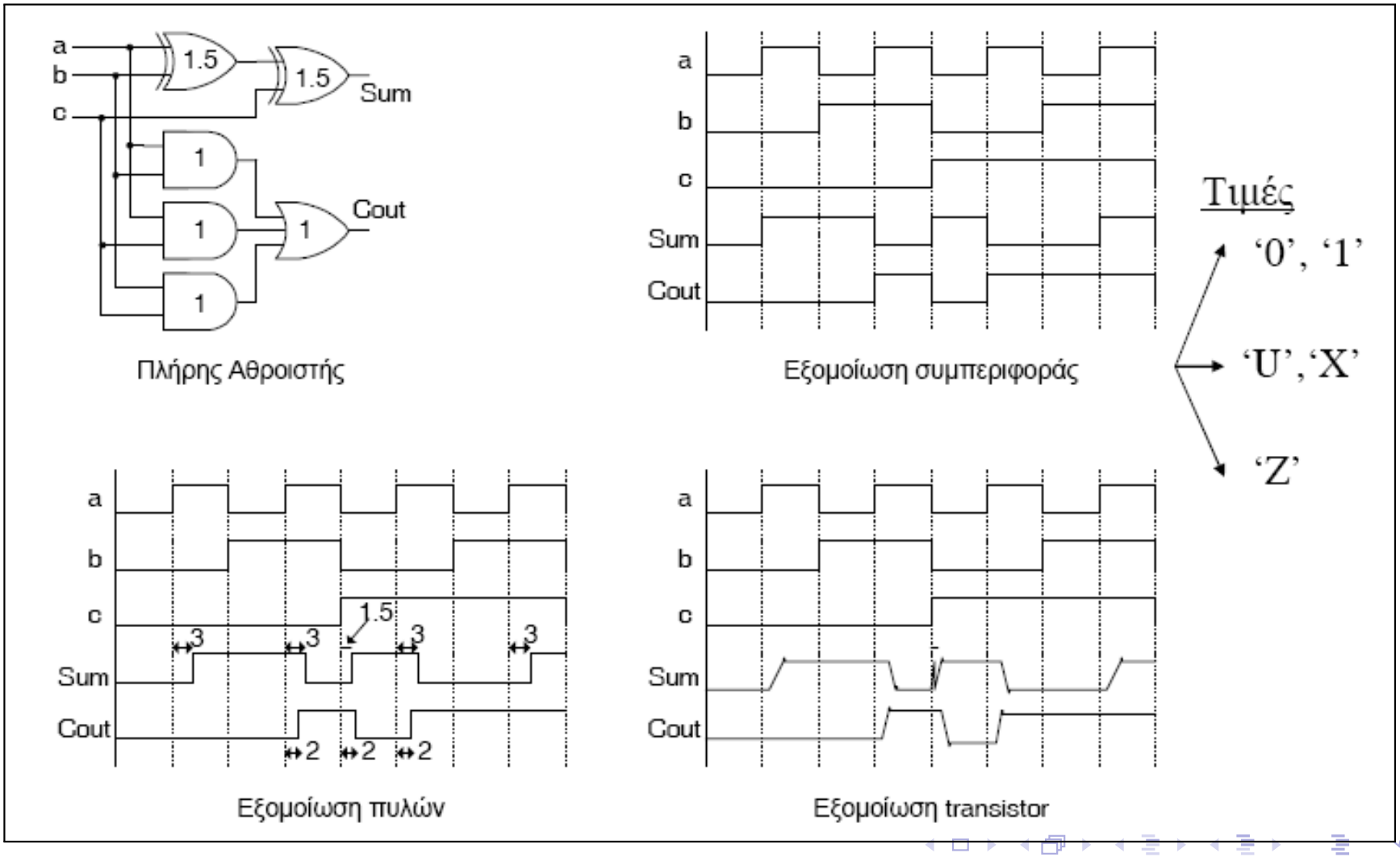


- Βελτιστοποίηση κυκλώματος
  - Ελαχιστοποίηση της επιφάνειας
  - Μεγιστοποίηση της ταχύτητας
  - Ικανοποίηση περιορισμών λειτουργίας
- Η αντιστοίχιση σε τεχνολογία (technology mapping, library

Προσομοίωση (1)

- Η προσομοίωση είναι το βασικότερο εργαλείο επιβεβαίωσης της ορθότητας σχεδίασης ενός κυκλώματος
- Είδη προσομοίωσης
  - Προσομοίωση συμπεριφοράς (behavioral simulation)
  - Προσομοίωση λειτουργίας (functional simulation-unit delay)
  - Στατική χρονική ανάλυση (static timing analysis), χωρίς διανύσματα εισόδου
  - Προσομοίωση σε επίπεδο πύλης (gate-level simulation)
  - Προσομοίωση διακοπτών (switch-level simulation)
  - Προσομοίωση transistor (transistor-level simulation)
- Η πιο λεπτομερής προσομοίωση γίνεται μετά τη σχεδίαση του layout (post-layout simulation)

Προσομοίωση (2)

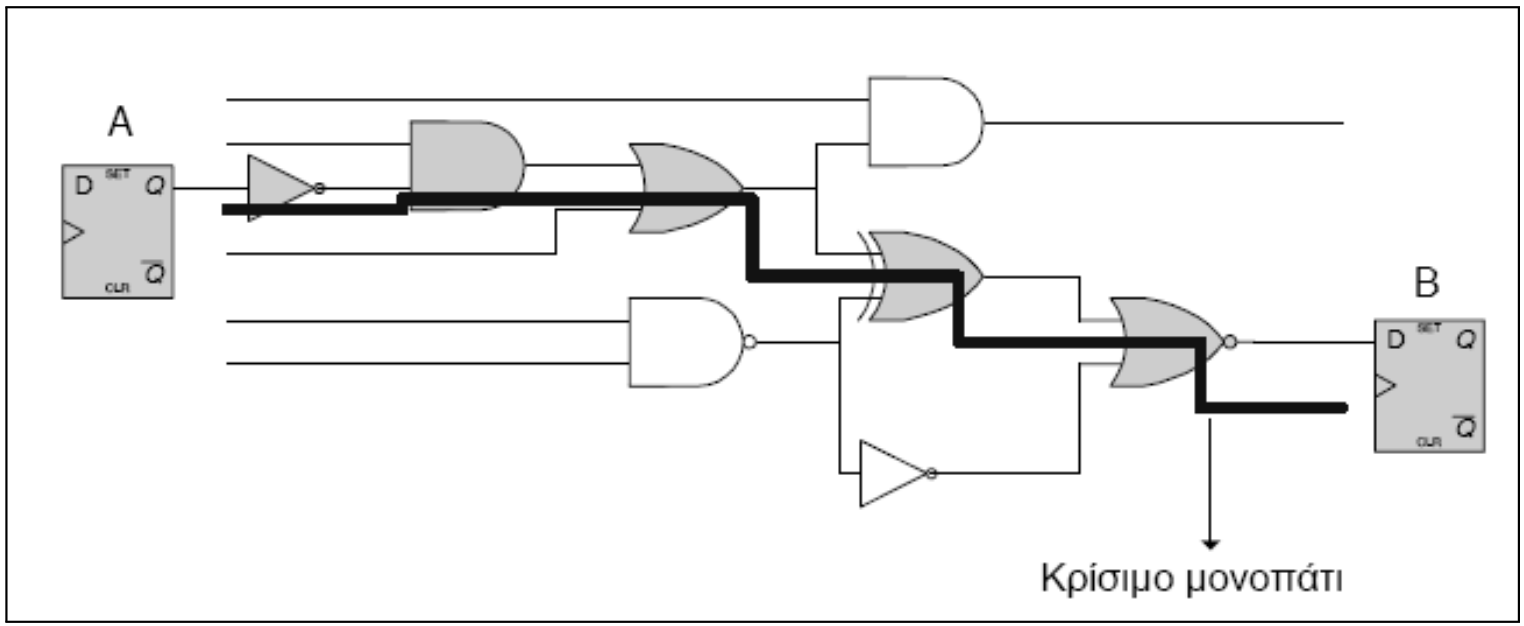


Στατική χρονική ανάλυση (1)

- Ο στατικός αναλυτής βρίσκει το/τα κρίσιμα μονοπάτια καθυστέρησης στο κύκλωμα
- Δεν απαιτεί από το σχεδιαστή να παρέχει τα διανύσματα προσομοίωσης
- Η ανάλυση του κυκλώματος γίνεται στατικά (από αναπαράσταση τύπου γράφου)
  - Αλγόριθμοι όπως all pairs longest paths
- Δεν δίνει τις συνθήκες ενεργοποίησης του κρίσιμου μονοπατιού (διάνυσμα εισόδου και εσωτερικών καταστάσεων)
- Ένα κρίσιμο μονοπάτι μπορεί να είναι λανθάνον
- Υποτίμηση της συχνότητας του ρολογιού (όταν το κρίσιμο μονοπάτι είναι κάποιο μικρότερο)

Στατική χρονική ανάλυση (2)

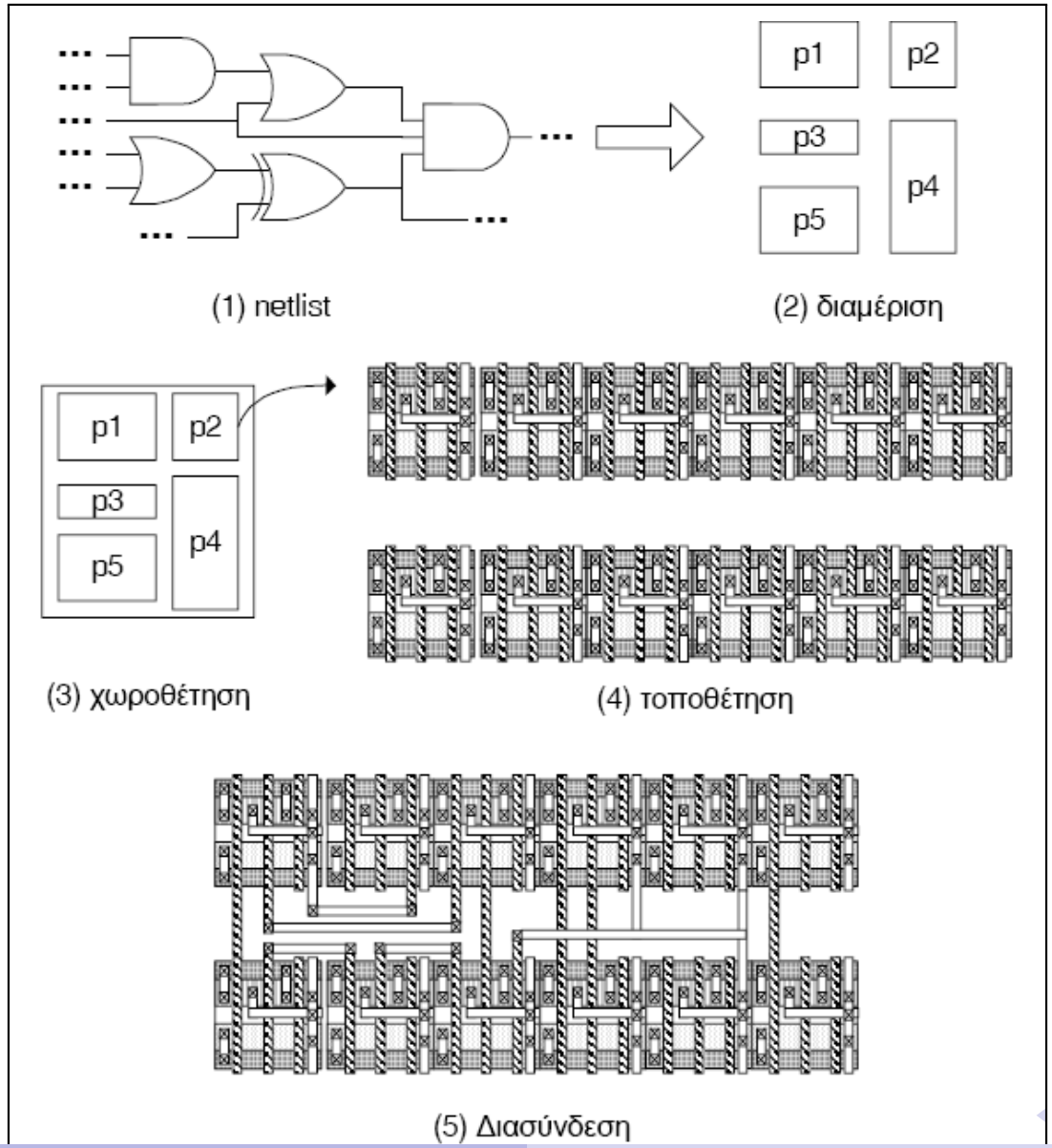
- Η στατική χρονική ανάλυση πρέπει να λαμβάνει υπόψη και τις καθυστερήσεις γραμμών που οφείλονται σε χωρητικότητες
- Θα πρέπει να εκτελείται μετά και από την τοποθέτηση και διασύνδεση των κυττάρων στο ολοκληρωμένο



Προσομοίωση διακοπτών και τρανζίστορ

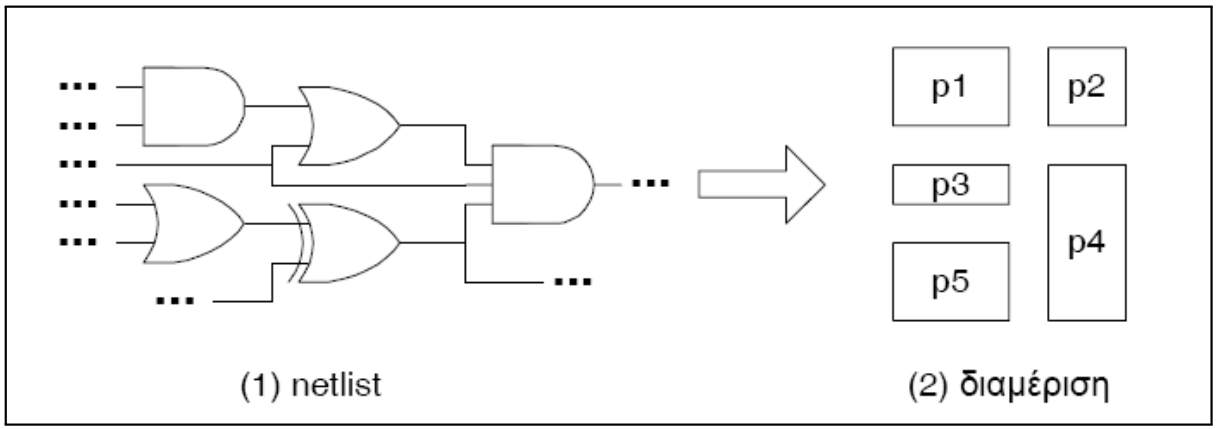
- Είναι οι τύποι προσομοιώσεων με την υψηλότερη ακρίβεια, καθώς αντιμετωπίζουν το κύκλωμα σαν δίκτυο των βασικών δομικών του στοιχείων (transistor)
- Προσομοίωση διακοπτών: θεωρούμε το transistor ως ψηφιακό στοιχείο, το οποίο μπορεί να είναι on ή off
- Προσομοίωση transistor: θεωρείται ως μη-γραμμικό αναλογικό στοιχείο
- Απαιτούν μεγάλο χρόνο εκτέλεσης, ειδικά η προσομοίωση transistor
- Προσομοιωτής transistor: SPICE = Simulation Program with Integrated Circuit Emphasis
  - Δυνατότητα προσομοίωσης σε διάφορα επίπεδα μοντελοποίησης (Levels)
  - Κάθε επίπεδο με διαφορετική λεπτομέρεια/ακρίβεια
  - Κάθε επίπεδο με διαφορετική μοντελοποίηση transistor

Γενική άποψη του backend



Διαμέριση

- Διαμέριση του δικτύου
  - Διαιρούμε τη σχεδίαση σε μικρότερα τμήματα (ομάδες διαμέρισης)
  - Απλοποιούμε ένα πρόβλημα, διαιρώντας το σε υποπροβλήματα
  - Στόχος 1: ελαχιστοποίηση των διασυνδέσεων μεταξύ των ομάδων διαμέρισης
  - Στόχος 2: διατήρηση του μεγέθους κάθε ομάδας διαμέρισης κάτω από ένα προκαθορισμένο όριο



Χωροθέτηση και τοποθέτηση

- Χωροθέτηση
  - Τοποθετούμε τις ομάδες διαμέρισης στο χώρο του ολοκληρωμένου
  - Τοποθέτηση ομάδων με πολλές διασυνδέσεις μεταξύ τους, σε κοντινά σημεία στο ολοκληρωμένο, για ελαχιστοποίηση μήκους διασυνδέσεων
- Τοποθέτηση
  - Καθορίζει τη θέση των λογικών κυττάρων στο χώρο της διαμέρισης
  - Εξαρτάται από την αρχιτεκτονική:
    - Σε gate arrays και standard cells υπάρχουν οριζόντιες διατάξεις γραμμών
    - Στα FPGAs οι θέσεις είναι προκαθορισμένες
  - Στόχος: ελαχιστοποίηση διασυνδέσεων που θα γίνουν σε επόμενο στάδιο

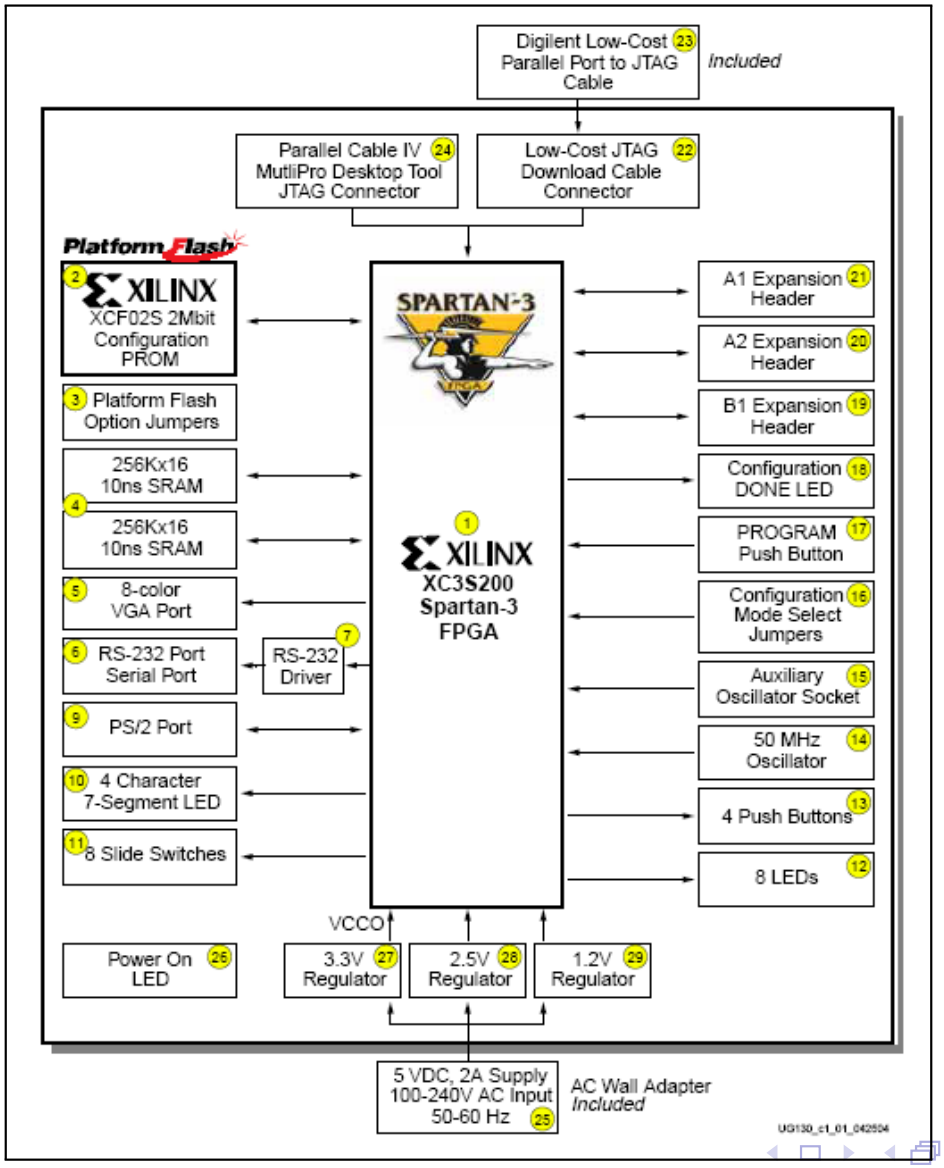
Διασύνδεση

- Πραγματοποιούνται οι διασυνδέσεις ανάμεσα στα κύτταρα
- Αρχικά αποφασίζεται από πού θα περάσουν οι διασυνδέσεις (global routing)
- Κατόπιν αποφασίζεται η ακριβής διαδρομή των διασυνδέσεων (local routing)
- Στόχος είναι η ελαχιστοποίηση της συνολικής επιφάνειας διασυνδέσεων και του μήκους της κάθε διασύνδεσης
- Πρόβλημα μη-επιλύσιμο σε πολυωνυμική χρονική πολυπλοκότητα (NP-completeness)
- Απαιτείται η χρήση ευριστικών αλγορίθμων για την επίτευξη προσεγγιστικής λύσης σε εύλογο χρονικό διάστημα

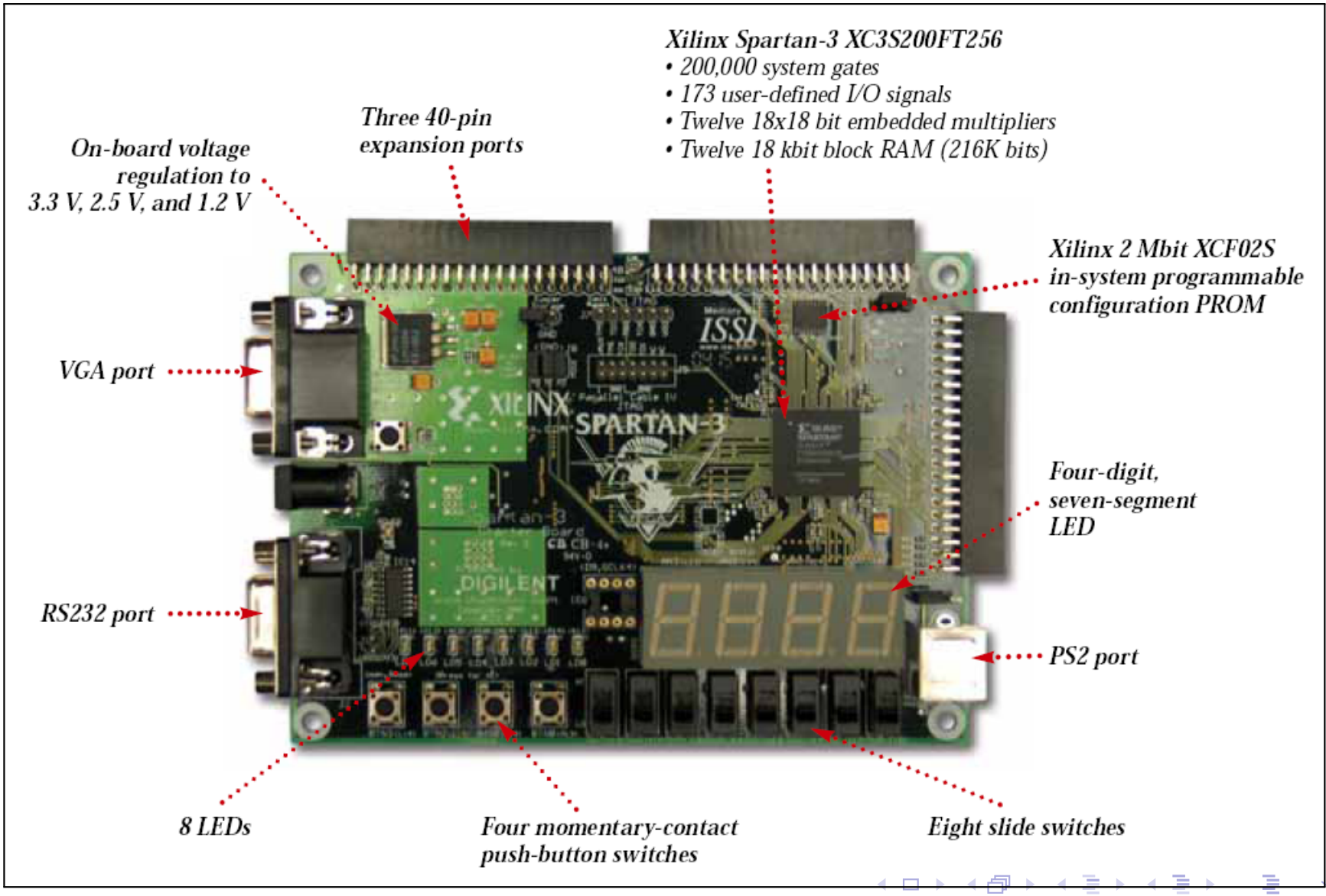
Η αναπτυξιακή πλακέτα Xilinx Spartan-3 Starter Kit (1)

- 200,000-gate Xilinx Spartan-3 XC3S200 FPGA in a 256-ball thin Ball Grid Array package (XC3S200FT256)
- 4,320 logic cell equivalents
- 12 18K-bit BRAMs (216K bits) and 12 18x18 multipliers
- Four Digital Clock Managers (DCMs)
- 1M-byte of Fast Asynchronous SRAM (as two 256Kx16 ISSI IS61LV25616AL-10T 10 ns SRAMs)
- 3-bit, 8-color VGA display port
- 9-pin RS-232 Serial Port (two connections), PS/2-style mouse/keyboard port
- Four-character, seven-segment LED display, 8 slide switches, 8 individual LED outputs, 4 push button switches
- 50 MHz crystal oscillator clock source
- On-board 3.3V, 2.5V, and 1.2V regulators

Η αναπτυξιακή πλακέτα Xilinx Spartan-3 Starter Kit (2)



Η αναπτυξιακή πλακέτα Xilinx Spartan-3 Starter Kit (3)



Η αναπτυξιακή πλακέτα Xilinx Spartan-3 Starter Kit (4)

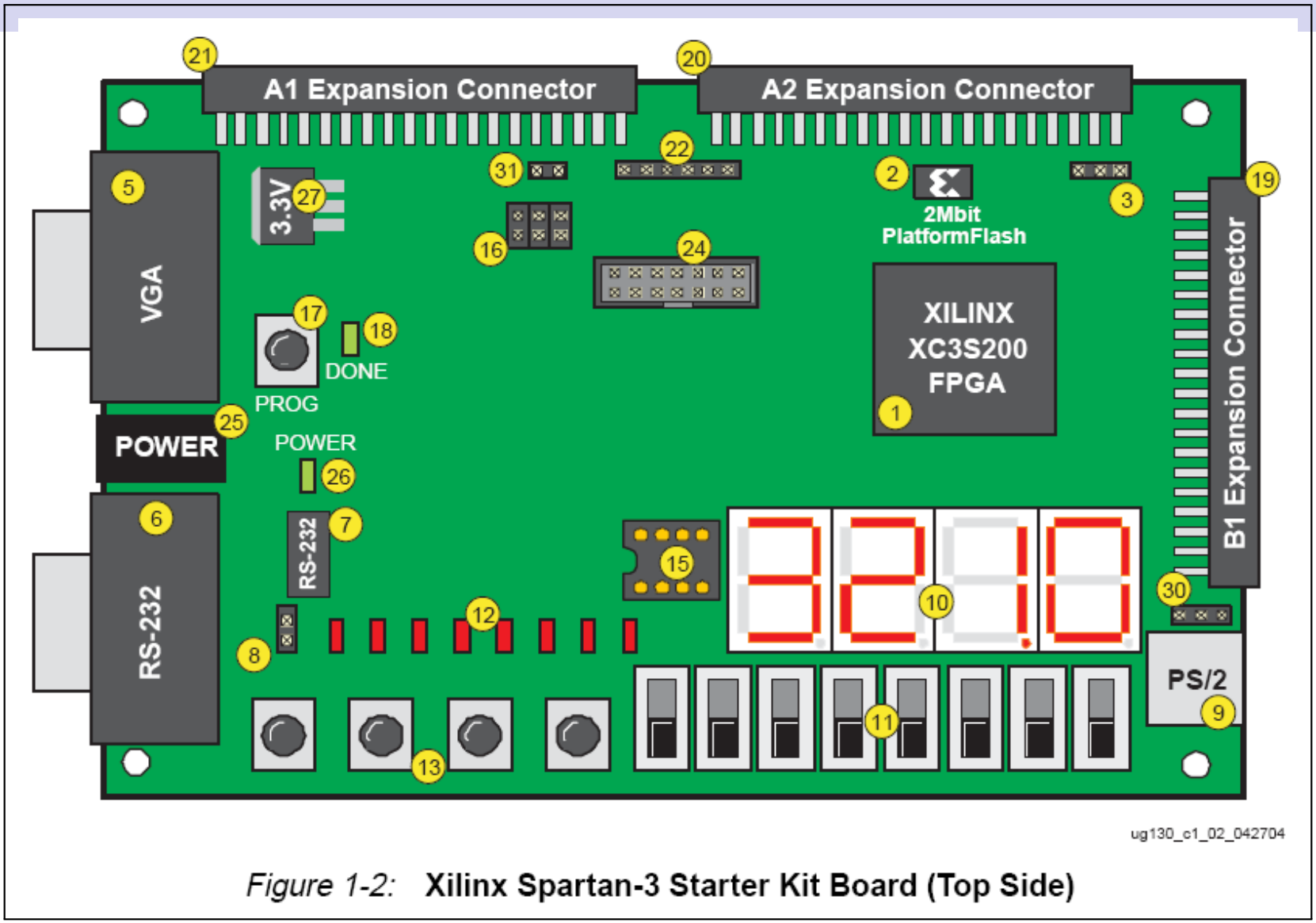


Figure 1-2: **Xilinx Spartan-3 Starter Kit Board (Top Side)**

Η αναπτυξιακή πλακέτα Xilinx Spartan-3 Starter Kit (5)

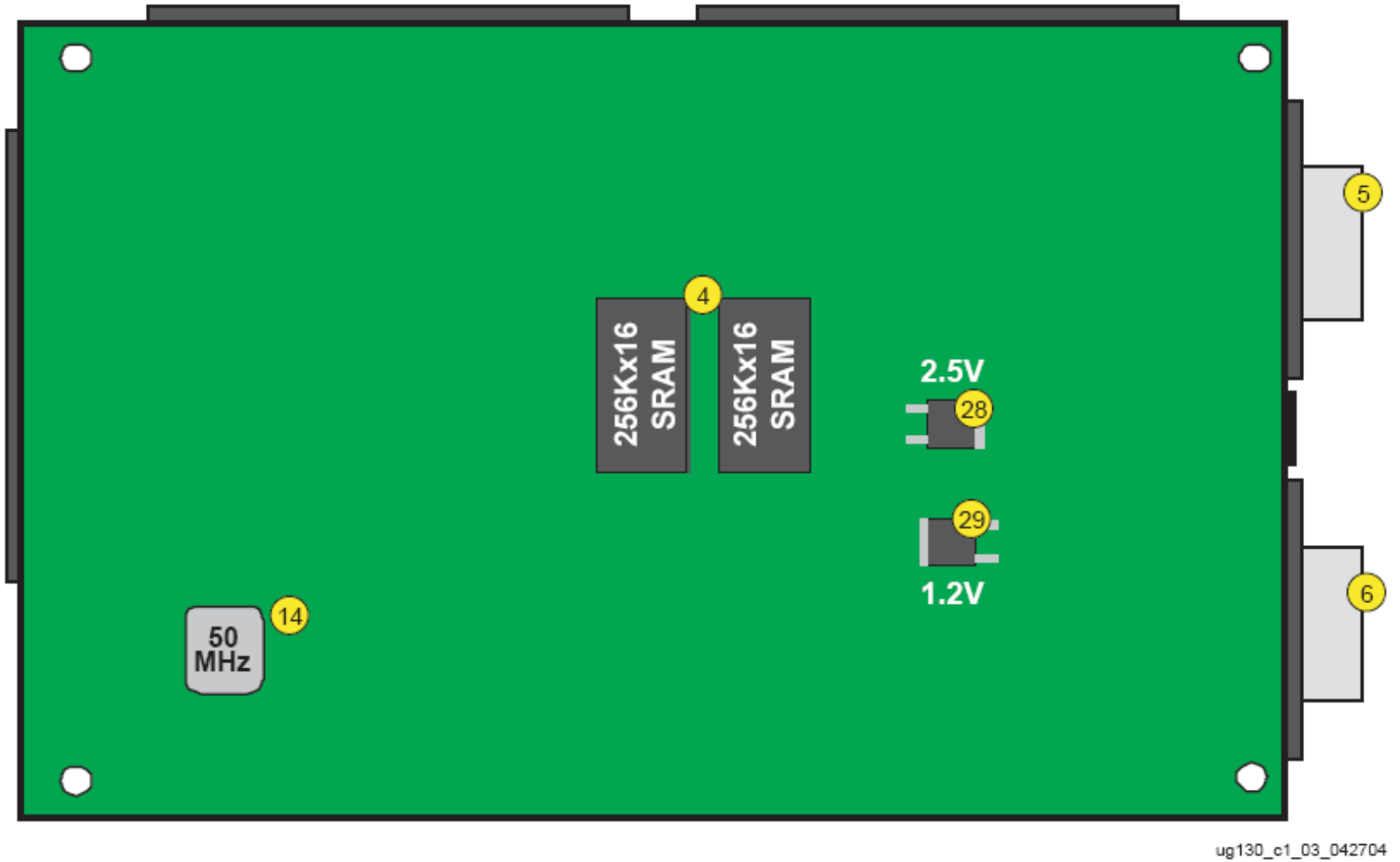
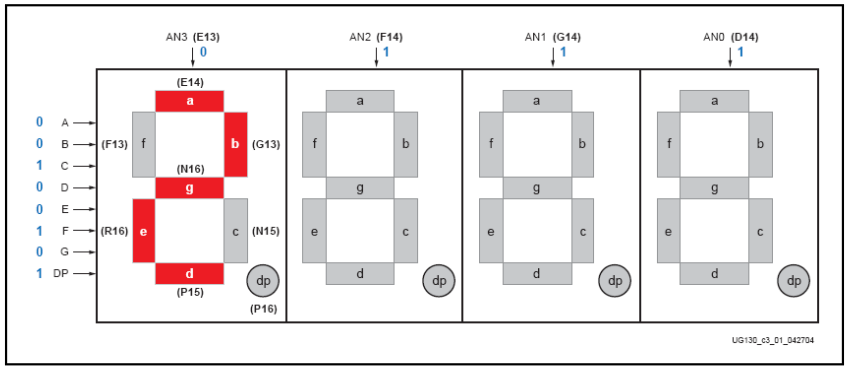


Figure 1-3: **Xilinx Spartan-3 Starter Kit Board (Bottom Side)**

Απλοί τρόποι διεπαφής της αναπτυξιακής πλακέτας Xilinx Spartan-3 Starter Kit με το χρήστη

LED display 7 τομέων



Slide switches

Switch	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
FPGA Pin	K13	K14	J13	J14	H13	H14	G12	F12

Push button switches

Push Button	BTN3 (User Reset)	BTN2	BTN1	BTN0
FPGA Pin	L14	L13	M14	M13

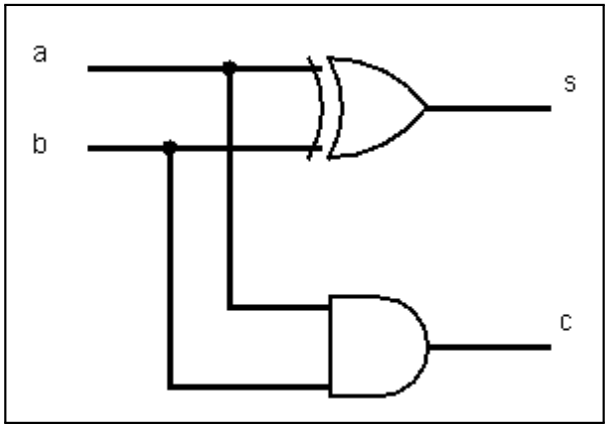
LEDs

LED	LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0
FPGA Pin	P11	P12	N12	P13	N14	L12	P14	K12

Παραδείγματα λογικής σύνθεσης με τη ροή Xilinx

- Στοχευόμενη συσκευή: Xilinx Spartan-3
- Αναπτυξιακή πλακέτα: Xilinx Spartan-3 starter kit
- Παραδείγματα
  - Ημιαθροιστής
  - Αθροιστής ριπής κρατουμένου των 4-bit
- Χρήση του Xilinx ISE Webpack

Περιγραφή του ημιαθροιστή (half adder)




```
library IEEE;
use IEEE.std_logic_1164.all;

entity half_adder is
  port (
    a : in  std_logic;
    b : in  std_logic;
    s : out std_logic;
    c : out std_logic
  );
end half_adder;

architecture structural of half_adder is
begin
  s <= a xor b;
  c <= a and b;
end structural;
```

Αρχείο UCF (User Constraints File) για την τοποθέτηση και διασύνδεση του ημιαθροιστή

```
NET "a"   LOC = "F12";
NET "b"   LOC = "J14";

NET "s"   LOC = "K12";
NET "c"   LOC = "N12";
```

Αναφορά για τη δημιουργία νέου project στο ISE Webpack

```
Project:
    Project Name: half_adder
    Project Path: C:\My Documents\Work\DCD\code\08\half_adder\half_adder
    Top Level Source Type: HDL

Device:
    Device Family: Spartan3
    Device:        xc3s200
    Package:       ft256
    Speed:         -5

    Synthesis Tool: XST (VHDL/Verilog)
    Simulator: ISE Simulator (VHDL/Verilog)
    Preferred Language: VHDL

    Enhanced Design Summary: enabled
    Message Filtering: disabled
    Display Incremental Messages: disabled

Existing Sources:
     half_adder.vhd
     xcs3sk_lab.ucf
```

Αναφορά λογικής σύνθεσης (1)

```
--> Reading design: half_adder.prj
=========================================================================
*                      Synthesis Options Summary                        *
=========================================================================
---- Source Parameters
Input File Name                    : "half_adder.prj"
Input Format                       : mixed
Ignore Synthesis Constraint File   : NO

---- Target Parameters
Output File Name                   : "half_adder"
Output Format                      : NGC
Target Device                      : xc3s200-5-ft256

---- Source Options
Top Module Name                    : half_adder
Automatic FSM Extraction           : YES
FSM Encoding Algorithm             : Auto
Safe Implementation                : No
FSM Style                          : lut
RAM Extraction                     : Yes
RAM Style                          : Auto
ROM Extraction                     : Yes
Mux Style                          : Auto
Decoder Extraction                 : YES
Priority Encoder Extraction        : YES
Shift Register Extraction          : YES
Logical Shifter Extraction         : YES
XOR Collapsing                     : YES
```

Αναφορά λογικής σύνθεσης (2)

```
ROM Style                          : Auto
Mux Extraction                     : YES
Resource Sharing                   : YES
Asynchronous To Synchronous        : NO
Multiplier Style                   : auto
Automatic Register Balancing       : No

---- Target Options
Add IO Buffers                     : YES
Global Maximum Fanout              : 500
Add Generic Clock Buffer(BUFG)     : 8
Register Duplication               : YES
Slice Packing                      : YES
Optimize Instantiated Primitives   : NO
Use Clock Enable                   : Yes
Use Synchronous Set                : Yes
Use Synchronous Reset              : Yes
Pack IO Registers into IOBs        : auto
Equivalent register Removal        : YES
```

Αναφορά λογικής σύνθεσης (3)

```
Compiling vhdl file "C:/My Documents/Work/DCD/code/08/half_adder/half_adder.vhd" in
Library work.
Entity <half_adder> compiled.
Entity <half_adder> (Architecture <structural>) compiled.
Analyzing hierarchy for entity <half_adder> in library <work> (architecture <structural>).
Analyzing Entity <half_adder> in library <work> (Architecture <structural>).
Entity <half_adder> analyzed. Unit <half_adder> generated.
Synthesizing Unit <half_adder>.
    Related source file is "C:/My Documents/Work/DCD/code/08/half_adder/half_adder.vhd".
    Found 1-bit xor2 for signal <s>.
Unit <half_adder> synthesized.
=========================================================================
HDL Synthesis Report

Macro Statistics
# Xors                                                  : 1
 1-bit xor2                                            : 1
```

Αναφορά λογικής σύνθεσης (3)

```
Optimizing unit <half_adder> ...
Mapping all equations...
Building and optimizing final netlist ...
Found area constraint ratio of 100 (+ 5) on block half_adder, actual ratio is 0.
Final Results
RTL Top Level Output File Name     : half_adder.ngr
Top Level Output File Name         : half_adder
Output Format                      : NGC
Optimization Goal                  : Speed
Keep Hierarchy                     : NO

Design Statistics
# IOs                              : 4

Cell Usage :
# BELS                             : 2
#      LUT2                        : 2
# IO Buffers                       : 4
#      IBUF                        : 2
#      OBUF                        : 2
=========================================================================

Device utilization summary:
---------------------------
Selected Device : 3s200ft256-5
 Number of Slices:                        1  out of   1920     0%
 Number of 4 input LUTs:                  2  out of   3840     0%
 Number of IOs:                           4
 Number of bonded IOBs:                   4  out of    173     2%
```

Αναφορά λογικής σύνθεσης (4)

```
TIMING REPORT

Clock Information:
------------------
No clock signals found in this design

================================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 4 / 2
--------------------------------------------------------------------------------
Delay:               7.824ns (Levels of Logic = 3)
  Source:            a (PAD)
  Destination:       c (PAD)

  Data Path: a to c
                                   Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             2   0.715   1.040  a_IBUF (a_IBUF)
     LUT2:I0->O            1   0.479   0.681  c1 (c_OBUF)
     OBUF:I->O                 4.909          c_OBUF (c)
    ----------------------------------------
    Total                      7.824ns (6.103ns logic, 1.721ns route)
                                       (78.0% logic, 22.0% route)
```

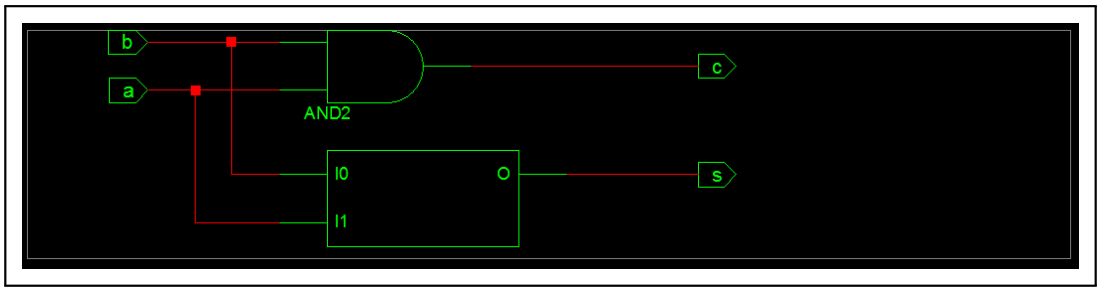
Λίστα κόμβων για τον ημιαθροιστή

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
library UNISIM;
use UNISIM.VCOMPONENTS.ALL;
use UNISIM.VPKG.ALL;

entity half_adder is
  port (
    a : in STD_LOGIC := 'X';
    b : in STD_LOGIC := 'X';
    c : out STD_LOGIC;
    s : out STD_LOGIC
  );
end half_adder;

architecture Structure of half_adder is
  signal a_IBUF_0, b_IBUF_1 : STD_LOGIC;
  signal c_OBUF_2, s_OBUF_3 : STD_LOGIC;
  signal N2, N3 : STD_LOGIC;
begin
  c1 : LUT2
    generic map(
      INIT => X"8"
    )
    port map (
      I0 => a_IBUF_0,
      I1 => b_IBUF_1,
      O => c_OBUF_2
    );
```

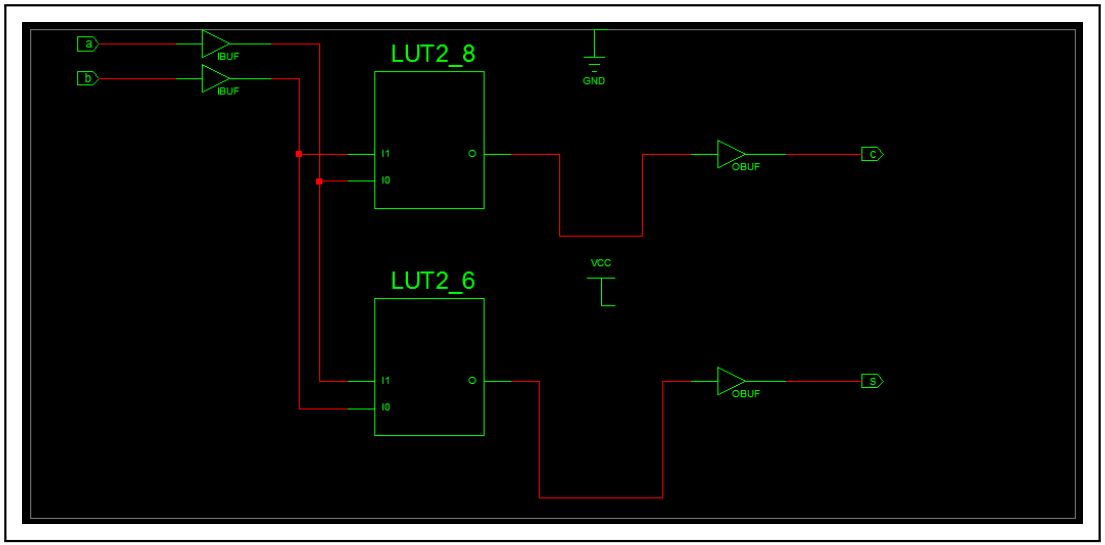
```
  Mxor_s_Result1 : LUT2
    generic map(
      INIT => X"6"
    )
    port map (
      I0 => b_IBUF_1, I1 => a_IBUF_0,
      O => s_OBUF_3
    );
  a_IBUF : IBUF
    port map (
      I => a, O => a_IBUF_0
    );
  b_IBUF : IBUF
    port map (
      I => b, O => b_IBUF_1
    );
  c_OBUF : OBUF
    port map (
      I => c_OBUF_2,  O => c
    );
  s_OBUF : OBUF
    port map (
      I => s_OBUF_3, O => s
    );
  XST_GND : GND
    port map (G => N2);
  XST_VCC : VCC
    port map (P => N3);
end Structure;
```

Αυτόματα παραγόμενα σχηματικά του κυκλώματος

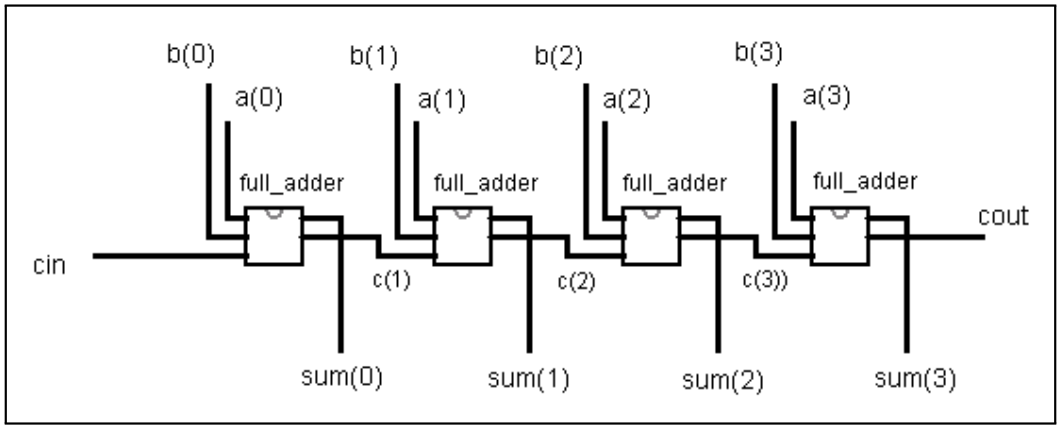
Σε επίπεδο RTL (ανεξάρτητα από την τεχνολογία)



Σε επίπεδο λίστας κόμβων (για την τεχνολογία)



Περιγραφή του αθροιστή ριπής κρατουμένου (4-bit ripple carry adder)




```
library IEEE;
use IEEE.std_logic_1164.all;

entity rca4 is
  port (
    a, b : in std_logic_vector(3 downto 0);
    cin : in std_logic;
    sum : out std_logic_vector(3 downto 0);
    cout : out std_logic
  );
end rca4;

architecture structural of rca4 is
  component full_adder is
    port (
      a, b, cin : in std_logic;
      sum, cout : out std_logic
  );
end component;
  signal c : std_logic_vector(4 downto 0);
begin
  c(0) <= cin;
  G1: for i in 0 to 3 generate
    fa_i : full_adder port map (
      a => a(i), b => b(i), cin => c(i),
      sum => sum(i), cout => c(i+1)
    );
  end generate G1;
  cout <= c(4);
end structural;
```

Αρχείο UCF (User Constraints File) για την τοποθέτηση και διασύνδεση του αθροιστή ριπής κρατουμένου

```
NET "a<0>"  LOC = "F12";
NET "a<1>"  LOC = "G12";
NET "a<2>"  LOC = "H14";
NET "a<3>"  LOC = "H13";

NET "b<0>"  LOC = "J14";
NET "b<1>"  LOC = "J13";
NET "b<2>"  LOC = "K14";
NET "b<3>"  LOC = "K13";

NET "cin"  LOC = "M13";

NET "sum<0>"  LOC = "K12";
NET "sum<1>"  LOC = "P14";
NET "sum<2>"  LOC = "L12";
NET "sum<3>"  LOC = "N14";

NET "cout"  LOC = "N12";
```

Αποσπάσματα από την αναφορά λογικής σύνθεσης (1)

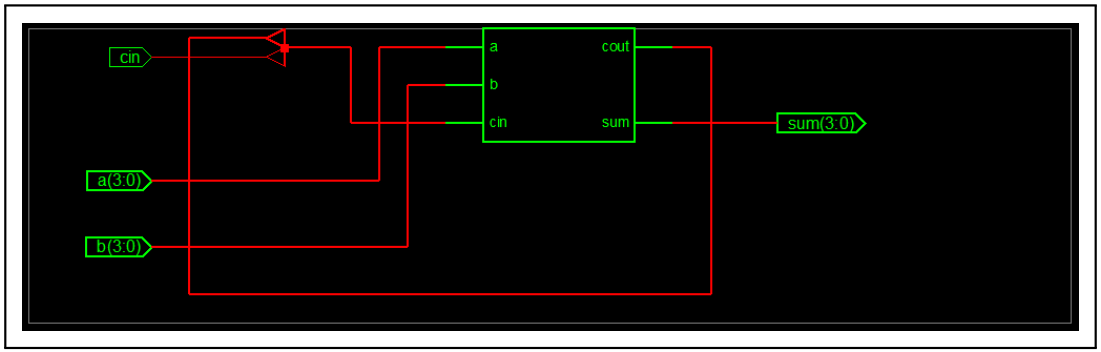
```
RTL Top Level Output File Name     : rca4.ngr
Top Level Output File Name         : rca4
Output Format                      : NGC
Optimization Goal                  : Speed
Keep Hierarchy                     : NO

Design Statistics
# IOs                              : 14

Cell Usage :
# BELS                             : 8
#      LUT3                        : 8
# IO Buffers                       : 14
#      IBUF                        : 9
#      OBUF                        : 5
=========================================================================

Device utilization summary:
---------------------------

Selected Device : 3s200ft256-5

 Number of Slices:                        4  out of   1920     0%
 Number of 4 input LUTs:                  8  out of   3840     0%
 Number of IOs:                          14
 Number of bonded IOBs:                  14  out of    173     8%
```

Αποσπάσματα από την αναφορά λογικής σύνθεσης (2)

```
Timing Summary:
---------------
   Minimum period: No path found
   Maximum combinational path delay: 12.008ns

=========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 33 / 5
-------------------------------------------------------------------------
Delay:               12.008ns (Levels of Logic = 6)
  Source:            b<0> (PAD)
  Destination:       cout (PAD)

  Data Path: b<0> to cout
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             2   0.715   1.040  b_0_IBUF (b_0_IBUF)
     LUT3:I0->O            2   0.479   0.915  G1[0].fa_i/cout1 (c<1>)
     LUT3:I1->O            2   0.479   0.915  G1[1].fa_i/cout1 (c<2>)
     LUT3:I1->O            2   0.479   0.915  G1[2].fa_i/cout1 (c<3>)
     LUT3:I1->O            1   0.479   0.681  G1[3].fa_i/cout1 (c<4>)
     OBUF:I->O                 4.909          cout_OBUF (cout)
    ----------------------------------------
    Total                     12.008ns (7.540ns logic, 4.468ns route)
                                       (62.8% logic, 37.2% route)
```

Αυτόματα παραγόμενα σχηματικά για τον αθροιστή ριπής κρατουμένου

Σε επίπεδο RTL (ανεξάρτητα από την τεχνολογία)



Σε επίπεδο λίστας κόμβων (για την τεχνολογία)