Digital Design Laboratory

Dr. Oniga István
University of Debrecen, Faculty of Informatics

This work was supported by the construction EFOP-3.4.3-16-2016-00021.
The project was supported by the European Union, co-financed by the European Social Fund.

1. Laboratory assignments

- ISE tutorial
- 2 variable logic function implementation - schematic design
- 2 variable logic function implementation - HDL design entry
- Command of 8 LEDs using 8 switches - HDL design entry

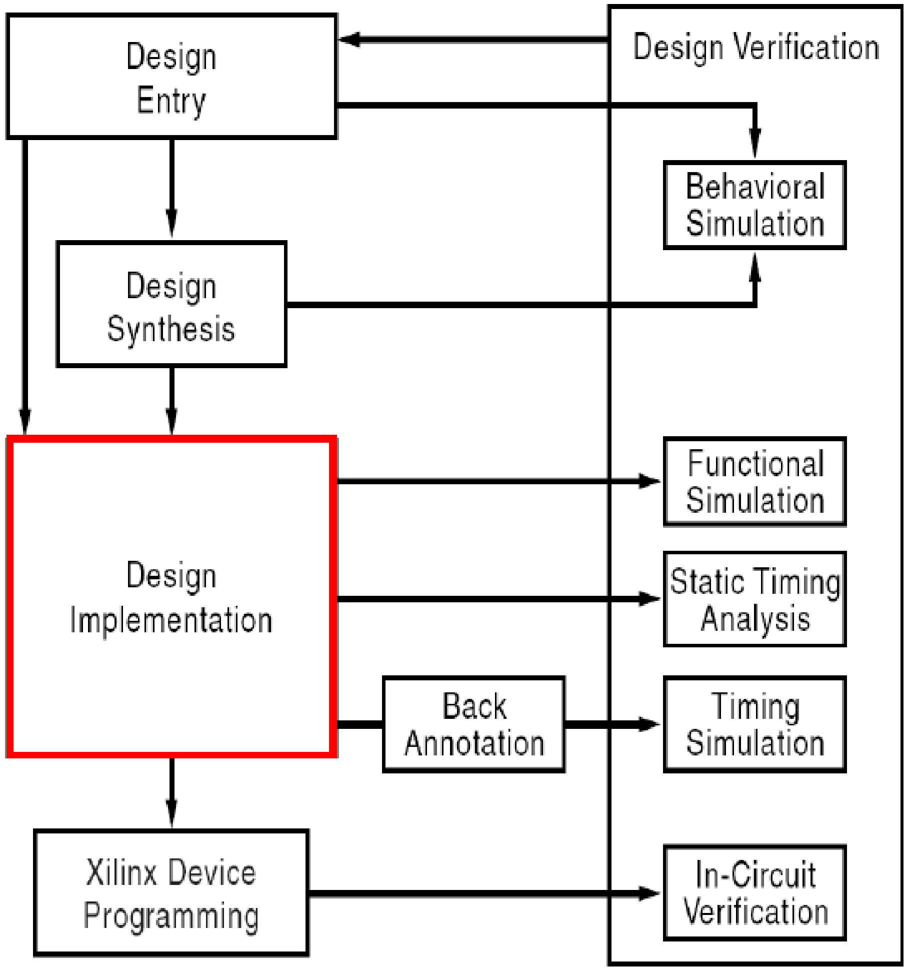
Xilinx ISE Design Suite 14.7

- ***ISE WebPack –*** ISE (Integrated software enviroment) for design with programmable logic devices.
  - In the laboratory practice, we use the free version of the Xilinx ISE 14.7 development environment called WebPack.
  - The installer can be downloaded after registration from the Xilinx website

    https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/design-tools/v2012_4---14_7.html
  - Full Installer for Windows 7/XP/Server (TAR/GZIP - 6.18 GB)
  - The WebPack (free) licence could be obtained also from the Xilinx product licensing site.
  - The software doesn't work without problem on 64 bites Windows 8, 8.1 and 10. The problem can be solved as is presented here (There are also two youtube links that present the solution), or you can try the program that can be downloaded from here.

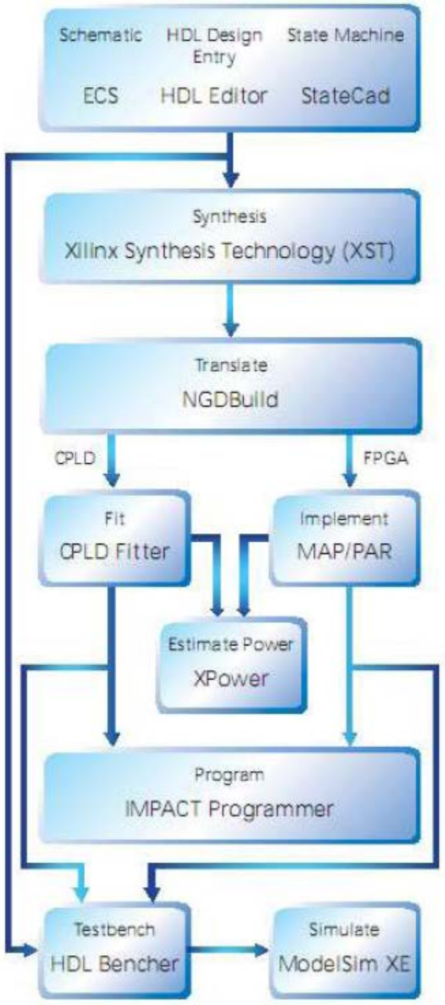
Software enviroments

- Design entry:
  - Xilinx Foundation ISE
  - Alternatives
    - Mentor Graphics: FPGA Advantage
    - Celoxica: DK Design Suite
- Design Synthesis:
  - XST: Xilinx Synthesis Technology
  - Mentor: Leonardo Spectrum
  - Synplicity: Synplify Pro
  - Celoxica: DK Design Suite
- Simulation:
  - Mentor: Modelsim
  - Aldec: Active-HDL
  - Celoxica: DK Design Suite
- In Circuit verification
  - Xilinx: ChipScope



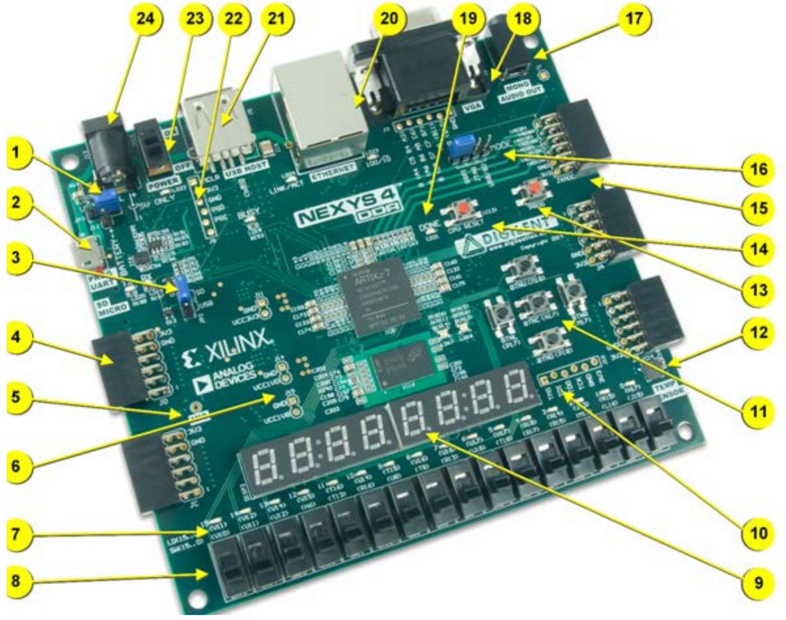
ISE Design flow

- Project Navigator
  - Design entry + constraints)
  - RTL simulation- (Testbench)
  - Sinthesys
  - Implementation: TRANSLATE →MAP →PAR (place & route)
  - Static timing analisys: (max clock frequency, propagation delays etc.)
  - Bitstream generate and download (configuration file - .bit)



Nexys 4 DDR Artix-7 FPGA

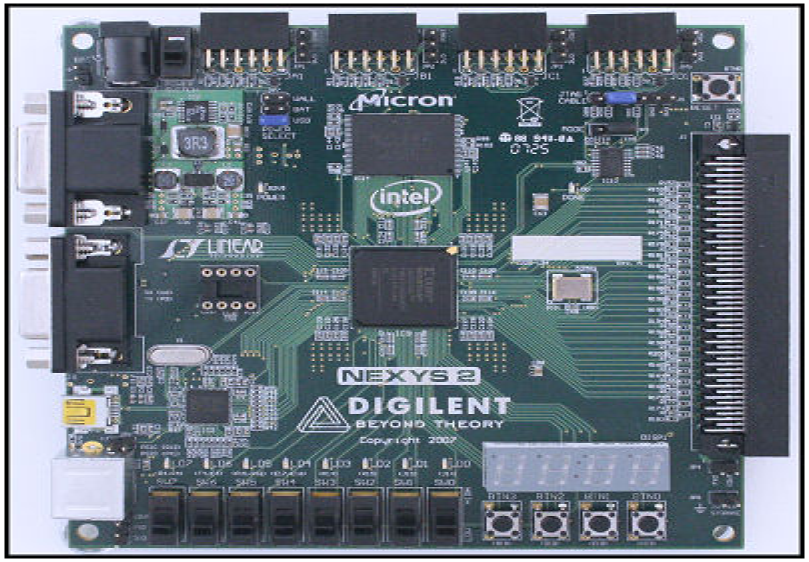
- •15,850 logic slices, each with four 6-input LUTs and 8 flip-flops
- •4,860 Kbits of fast block RAM
- •Six clock management tiles, each with phase-locked loop (PLL)
- •240 DSP slices
- •Internal clock speeds exceeding 450 MHz
- •On-chip analog-to-digital converter (XADC)
- •16 user switches
- •USB-UART Bridge
- •12-bit VGA output
- •3-axis accelerometer
- •128MiB DDR2
- •Pmod for XADC signals
- •16 user LEDs
- •Two tri-color LEDs
- •PWM audio output
- •Temperature sensor
- •Serial Flash
- •Digilent USB-JTAG port for FPGA programming and communication
- •Two 4-digit 7-segment displays
- •Micro SD card connector
- •PDM microphone
- •10/100 Ethernet PHY
- •Four Pmod ports
- •USB HID Host for mice, keyboards and memory sticks

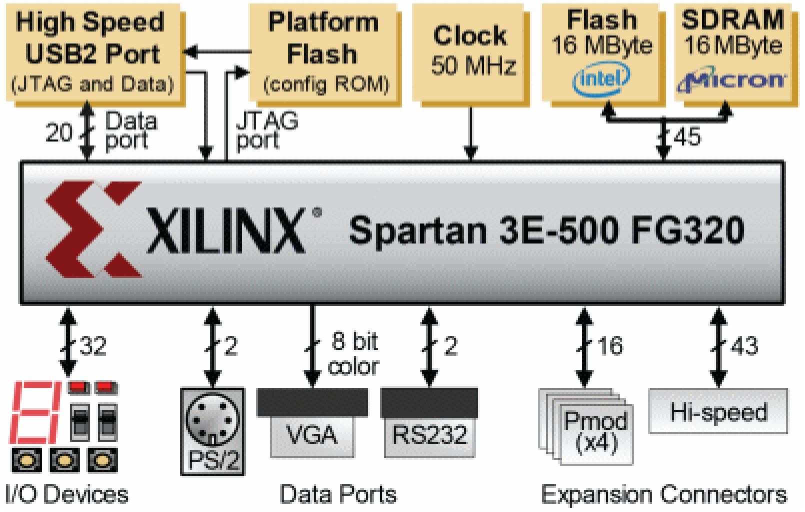


Nexys 4 DDR Reference Manual
Nexys 4 DDR Master UCF

Digilent Nexys 2

- Xilinx Spartan-3E FPGA, 500K or 1200K gate
- USB2 port providing board power, device configuration, and high-speed data transfers
- Works with ISE/Webpack and EDK
- 16MB fast Micron PSDRAM
- 16MB Intel StrataFlash Flash R
- Xilinx Platform Flash ROM
- High-efficiency switching power supplies (good for battery-powered applications
- 50MHz oscillator, plus a socket for a second oscillator
- 75 FPGA I/O's routed to expansion connectors (one high-speed Hirose FX2 connector with 43 signals and four 2x6 Pmod connectors)
- All I/O signals are ESD and short-circuit protected, ensuring a long operating life in any environment.
- On-board I/O includes eight LEDs, four-digit seven-segment display, four pushbuttons, eight slide switches
- Ships in a DVD case with a high-speed USB2 cable
- *Requires Adept 2.0 or later for operation*





Nexys 2 reference manual https://reference.digilentinc.com/reference/programmable-logic/nexys-2/reference-manual
Digilent Nexys 2 kártya https://store.digilentinc.com/nexys-2-spartan-3e-fpga-trainer-board-retired-see-nexys-4-ddr/

Digilent Adept suite

• Digilent Adept is a unique and powerful solution which allows you to communicate with Digilent system boards and a wide assortment of logic devices.

ADEPT for Windows

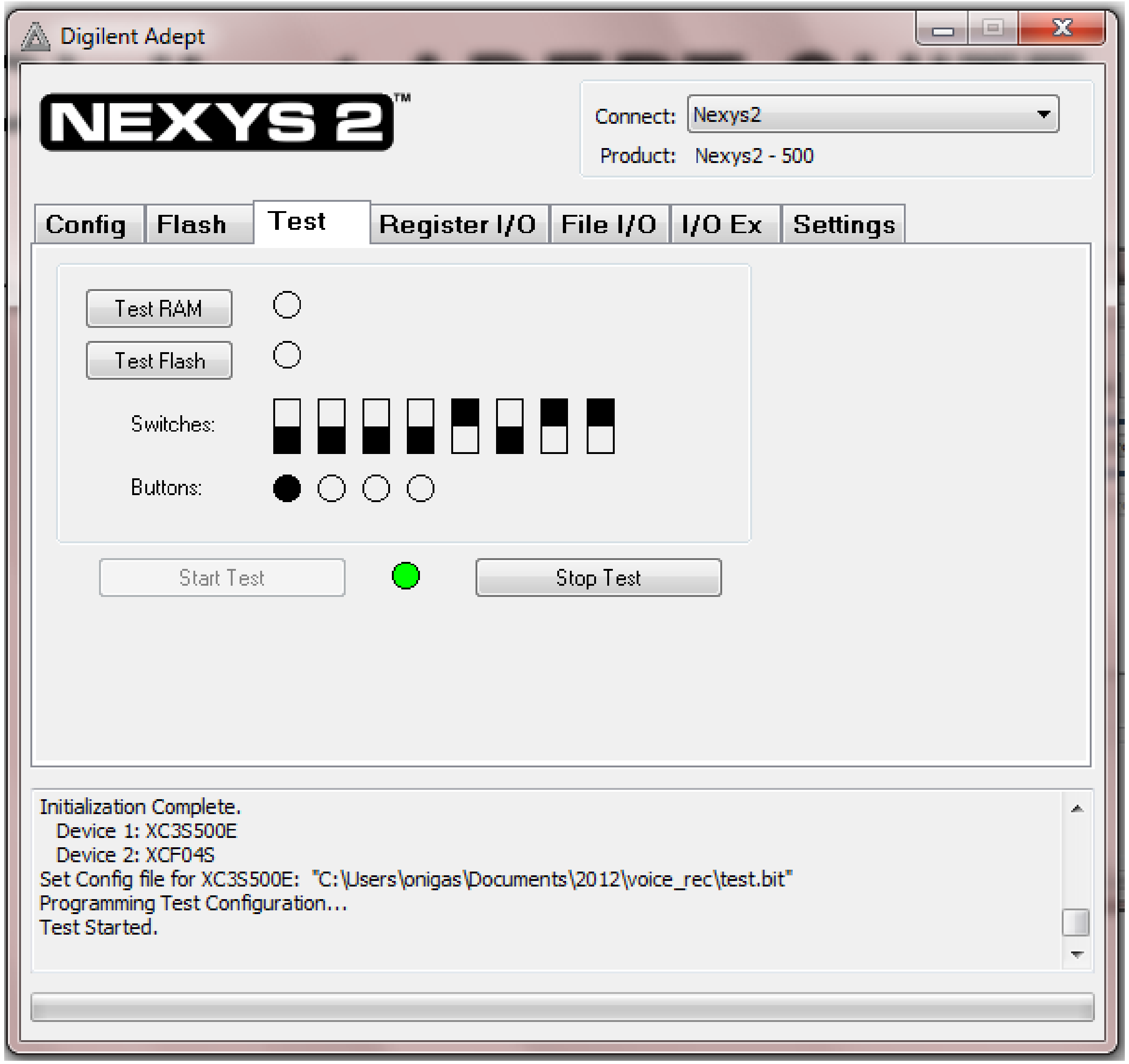
Adept 2 provide JTAG configuration and data transfering
•Also adds board verification and I/O expansion features.

• Configure the Xilinx logic devices. Initialize a scan chain, program FPGAs, CPLDs, and PROMs, organize and keep track of your configuration files
• Transfer data to and from the onboard FPGA on your system board. Read from and write to specified registers. Load a stream of data to a register or read a stream of data from a register.
• Organize and quickly connect to your communications modules.
• Program Xilinx XCFS Platform Flash devices using .bit or .mcs files.
• Program Xilinx CoolRunner2 CPLDs using .jed files.
• Program most Spartan and Virtex series FPGAs with .bit files

Board testing

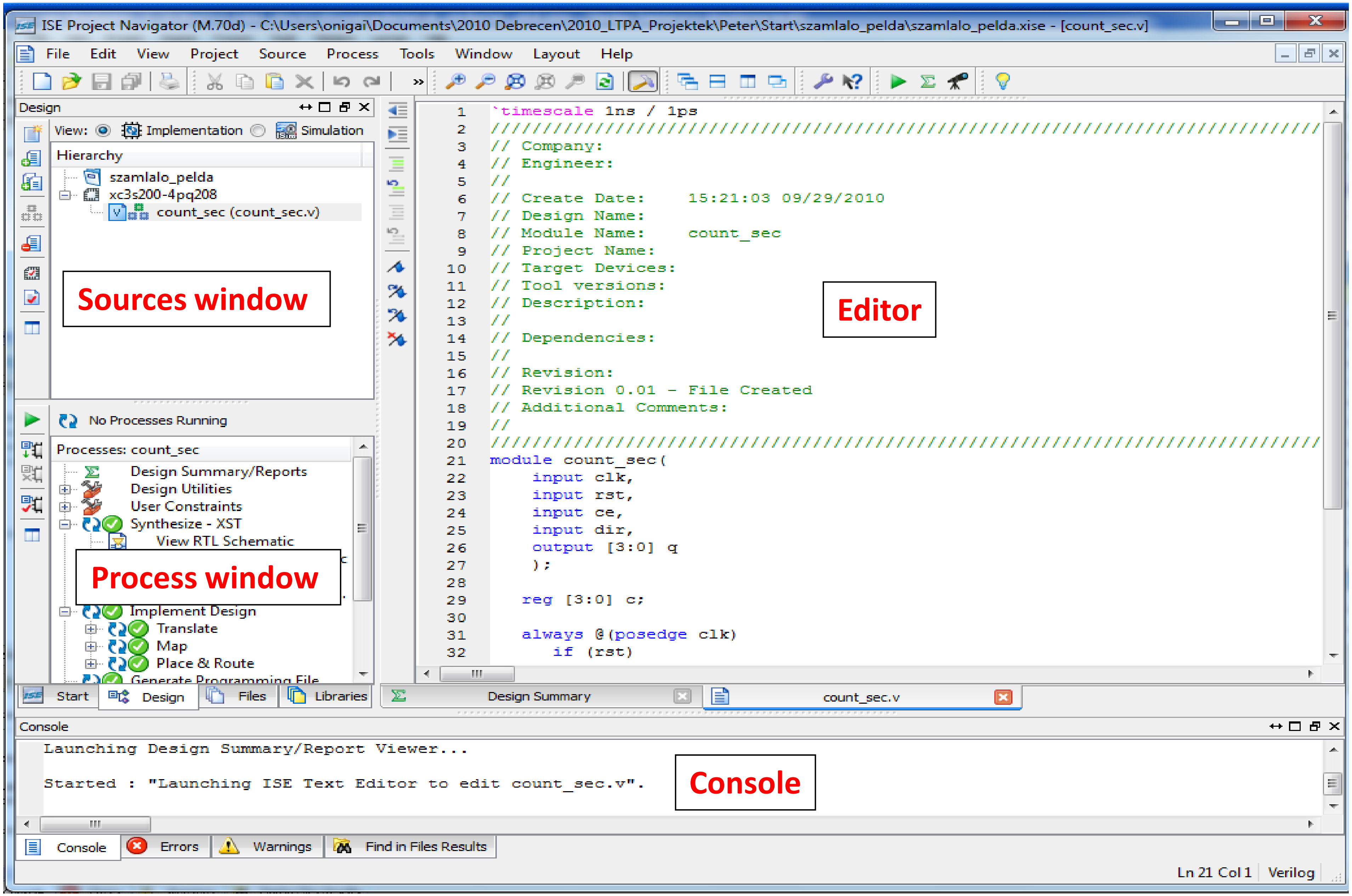
Start Test:

- RAM
- Flash
- Switches
- Push buttons
- LEDs
- 7 segment display



```
Initialization Complete.
  Device 1: XC3S500E
  Device 2: XCF04S
Set Config file for XC3S500E:  "C:\Users\onigas\Documents\2012\voice_rec\test.bit"
Programming Test Configuration...
Test Started.
```

Xilinx ISE



ISE Project Navigator (M.70d) - C:\Users\onigai\Documents\2010 Debrecen\2010_LTPA_Projektek\Peter\Start\szamlalo_pelda\szamlalo_pelda.xise - [count_sec.v]

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    15:21:03 09/29/2010
// Design Name:
// Module Name:    count_sec
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module count_sec(
    input clk,
    input rst,
    input ce,
    input dir,
    output [3:0] q
    );

   reg [3:0] c;

   always @(posedge clk)
      if (rst)
```

Console

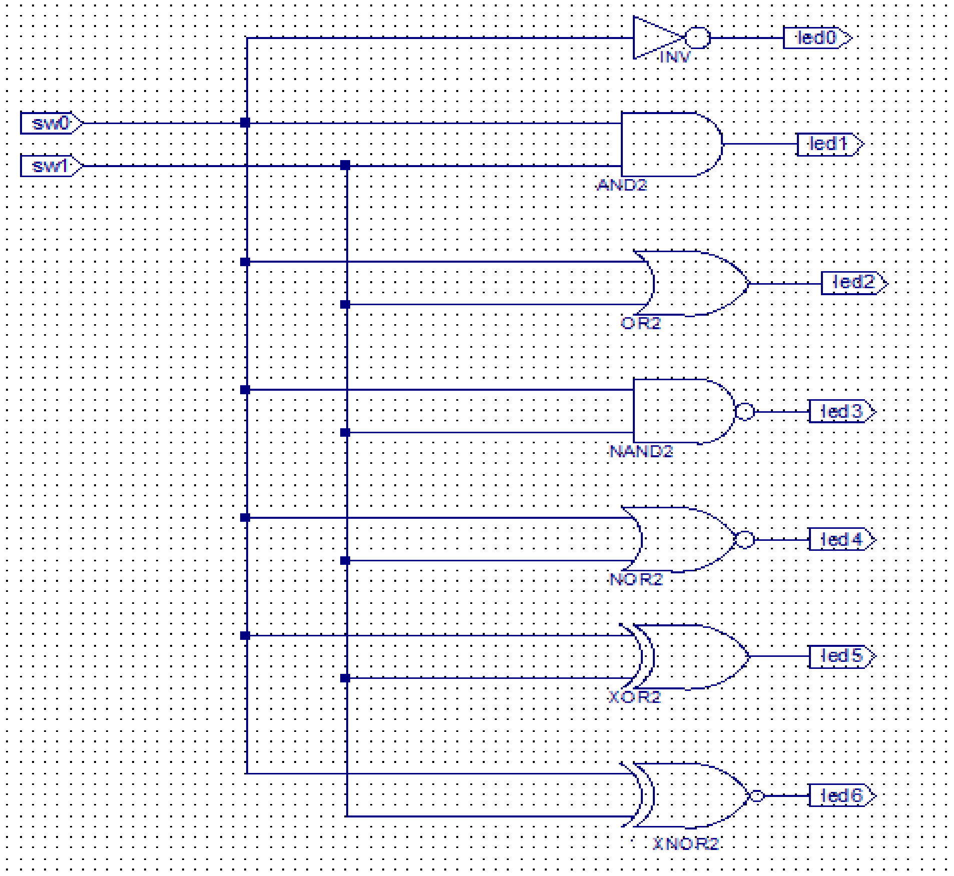
```
Launching Design Summary/Report Viewer...

Started : "Launching ISE Text Editor to edit count_sec.v".
```

Lab1_1 assignment

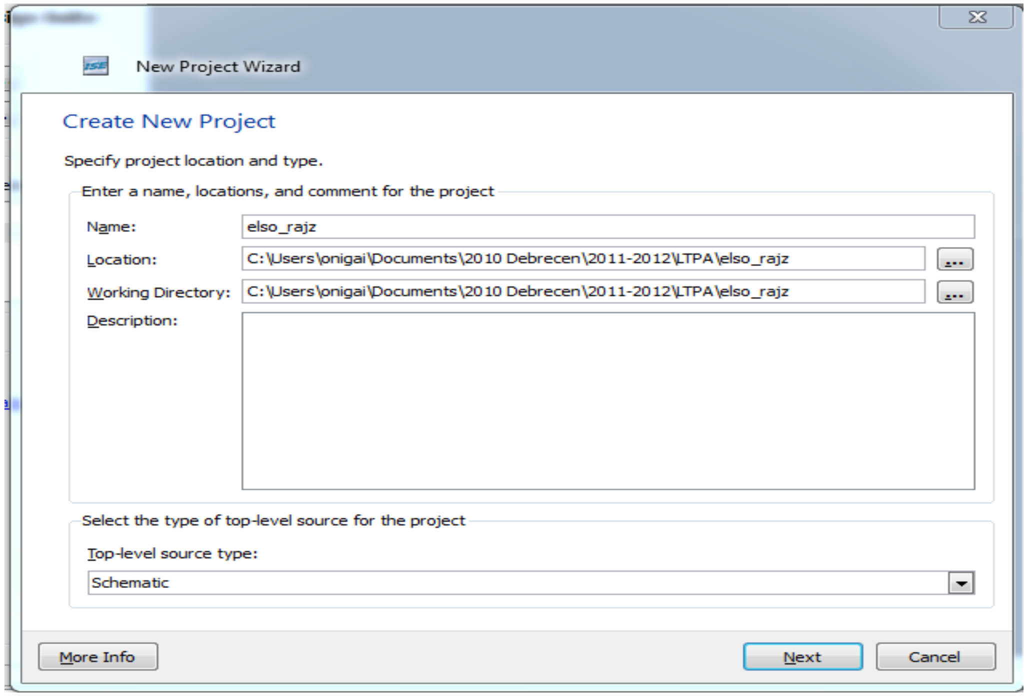
2 variable logic function implementation

- schematic design -



Project creation

- Start -> Programs\Xilinx ISE Design Suite 14.7\ISE Design Tools\Project Navigator.
- *File→New Project*,
- Name „*first_sch*” ,
  - For the folder and the file name don’t use white-spaces.
  - The name could not start with numbers, but could contain numbers
  - For easier readability of error messages use different names for folders and files
- **Top level source type:** schematic!



New Project Wizard

Create New Project

Specify project location and type.

Enter a name, locations, and comment for the project

Name: elso_rajz

Location: C:\Users\onigai\Documents\2010 Debrecen\2011-2012\LTPA\elso_rajz

Working Directory: C:\Users\onigai\Documents\2010 Debrecen\2011-2012\LTPA\elso_rajz

Description:

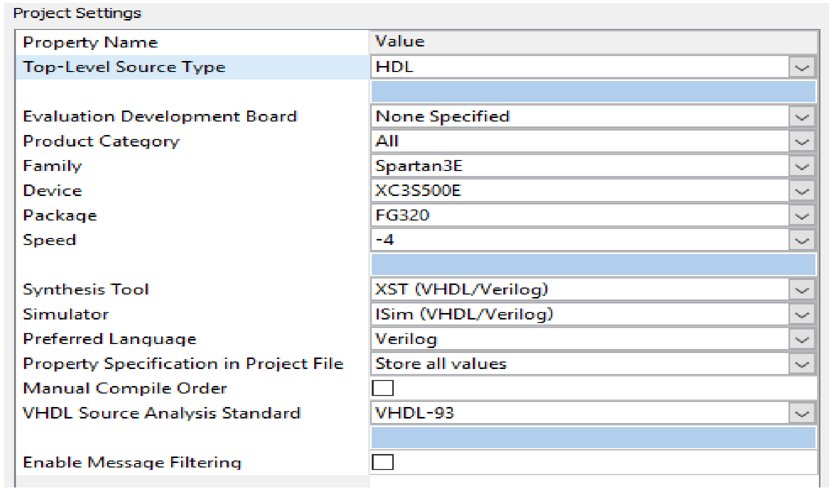
Select the type of top-level source for the project

Top-level source type: Schematic

Project settings

- **Next -> Device Properties -> Value :**

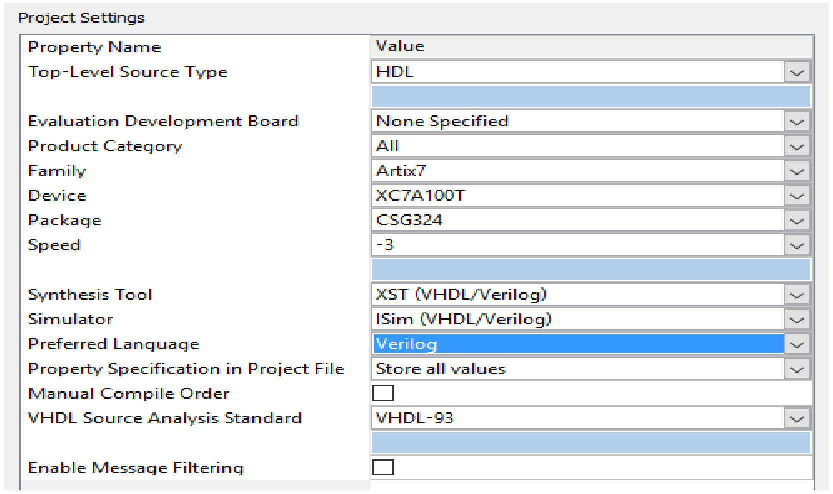
<u>NEXYS 2 board</u>
Device Family: Spartan3E
Device: xc3s500E
Package: FG320
Speed Grade: -4
Synthesis Tool: XST (VHDL/Verilog)
Simulator: ISim (VHDL/Verilog)
Preffered Language: Verilog



Project Settings

Property Name	Value
Top-Level Source Type	HDL
Evaluation Development Board	None Specified
Product Category	All
Family	Spartan3E
Device	XC3S500E
Package	FG320
Speed	-4
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	Verilog
Property Specification in Project File	Store all values
Manual Compile Order	☐
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	☐

<u>NEXYS 4 DDR board</u>
Family: Artix7
Device: XC7A100T
Package: CSG324
Speed Grade: -3
Synthesis Tool: XST (VHDL/Verilog)
Simulator: ISim (VHDL/Verilog)
Preffered Language: Verilog



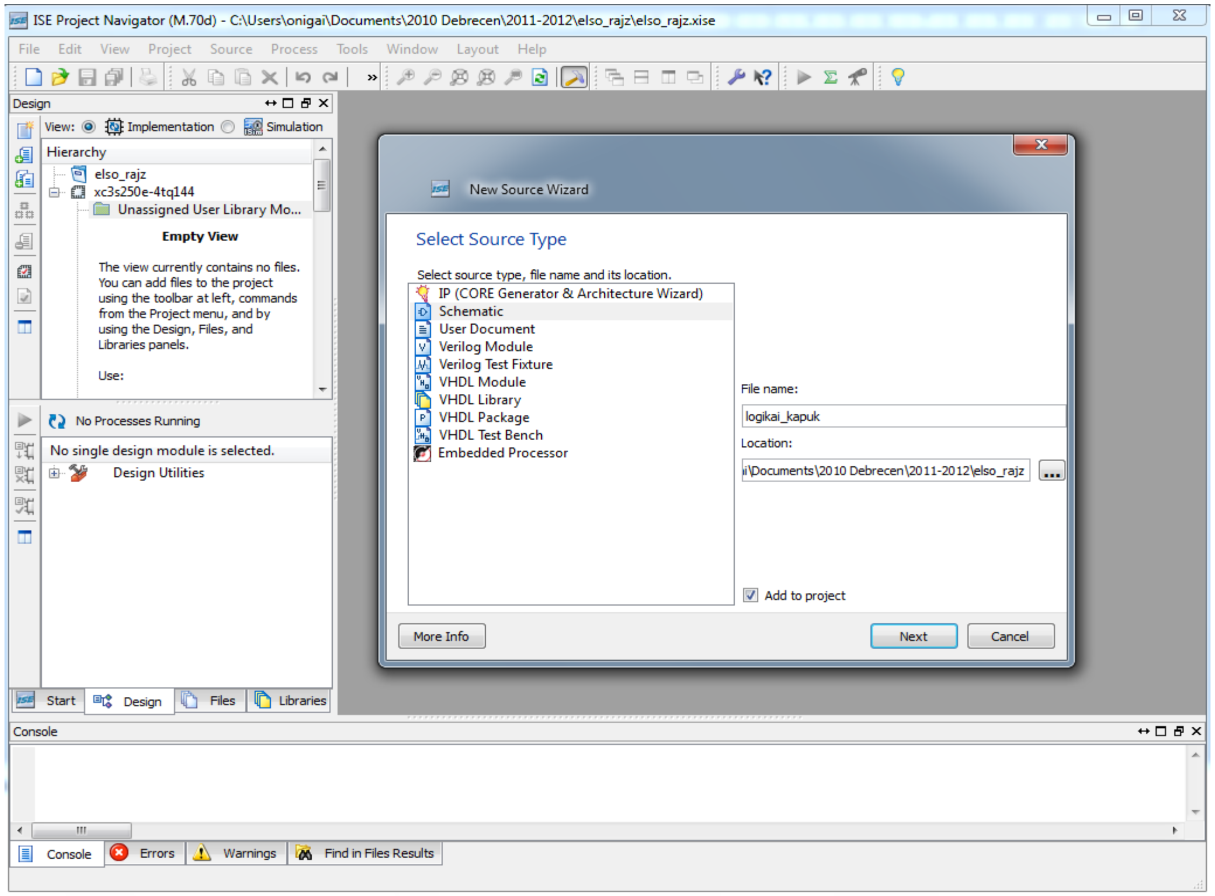
Project Settings

Property Name	Value
Top-Level Source Type	HDL
Evaluation Development Board	None Specified
Product Category	All
Family	Artix7
Device	XC7A100T
Package	CSG324
Speed	-3
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	Verilog
Property Specification in Project File	Store all values
Manual Compile Order	☐
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	☐

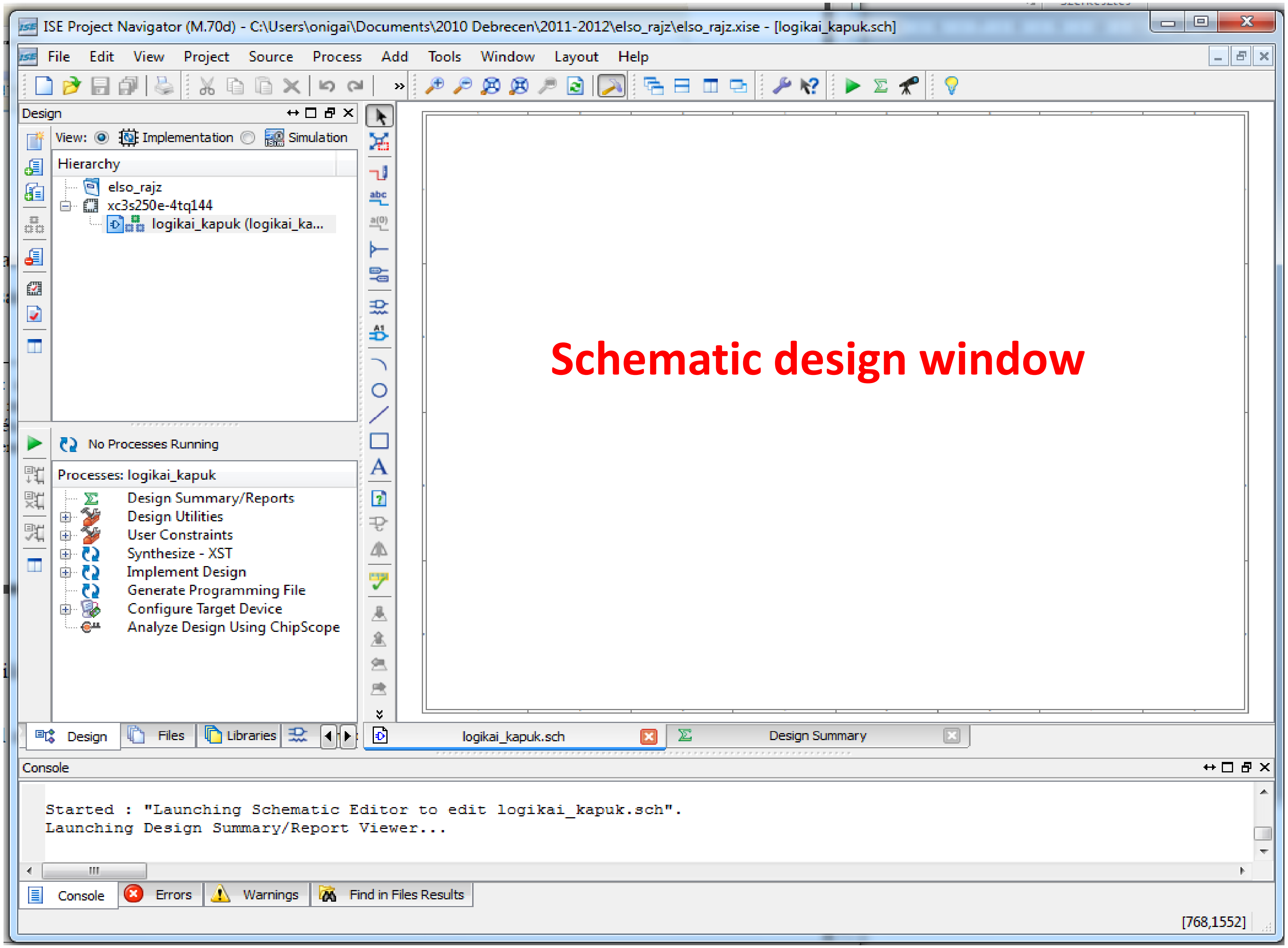
- **Next, Finish .**

Adding new souce

- *Project→New Source...*!
- Type: **schematic**, name logic_gates!
- Iff neccesary (*Project→Add Source...*)


ISE Project Navigator (M.70d) - C:\Users\onigai\Documents\2010 Debrecen\2011-2012\elso_rajz\elso_rajz.xise
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Hierarchy
elso_rajz
xc3s250e-4tq144
Unassigned User Library Mo...
Empty View
The view currently contains no files. You can add files to the project using the toolbar at left, commands from the Project menu, and by using the Design, Files, and Libraries panels.
Use:
No Processes Running
No single design module is selected.
Design Utilities
Start
Design
Files
Libraries
New Source Wizard
Select Source Type
Select source type, file name and its location.
IP (CORE Generator & Architecture Wizard)
Schematic
User Document
Verilog Module
Verilog Test Fixture
VHDL Module
VHDL Library
VHDL Package
VHDL Test Bench
Embedded Processor
File name:
logikai_kapuk
Location:
i\Documents\2010 Debrecen\2011-2012\elso_rajz
Add to project
More Info
Next
Cancel
Console
Errors
Warnings
Find in Files Results

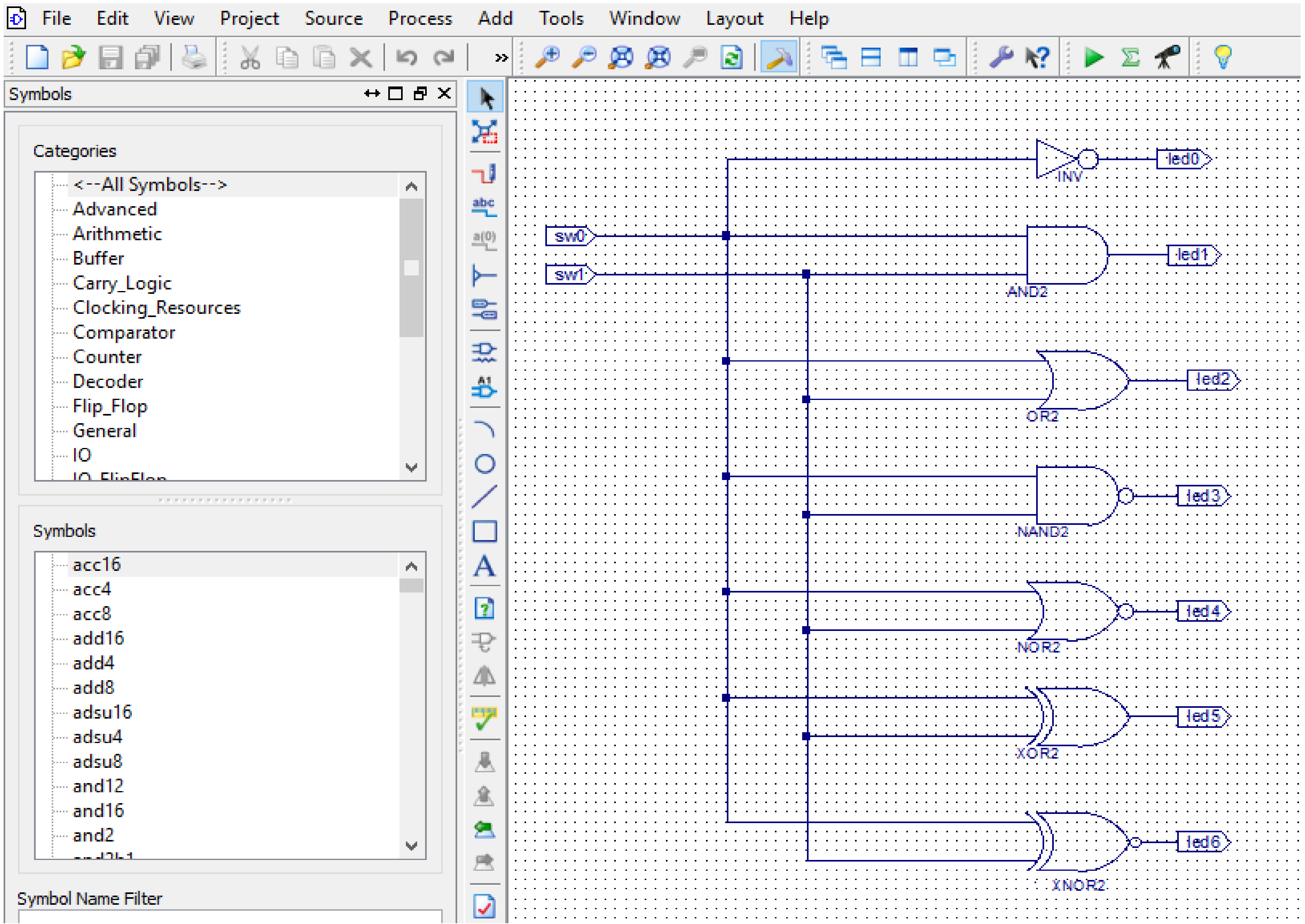
Schematic project


ISE Project Navigator (M.70d) - C:\Users\onigai\Documents\2010 Debrecen\2011-2012\elso_rajz\elso_rajz.xise - [logikai_kapuk.sch]

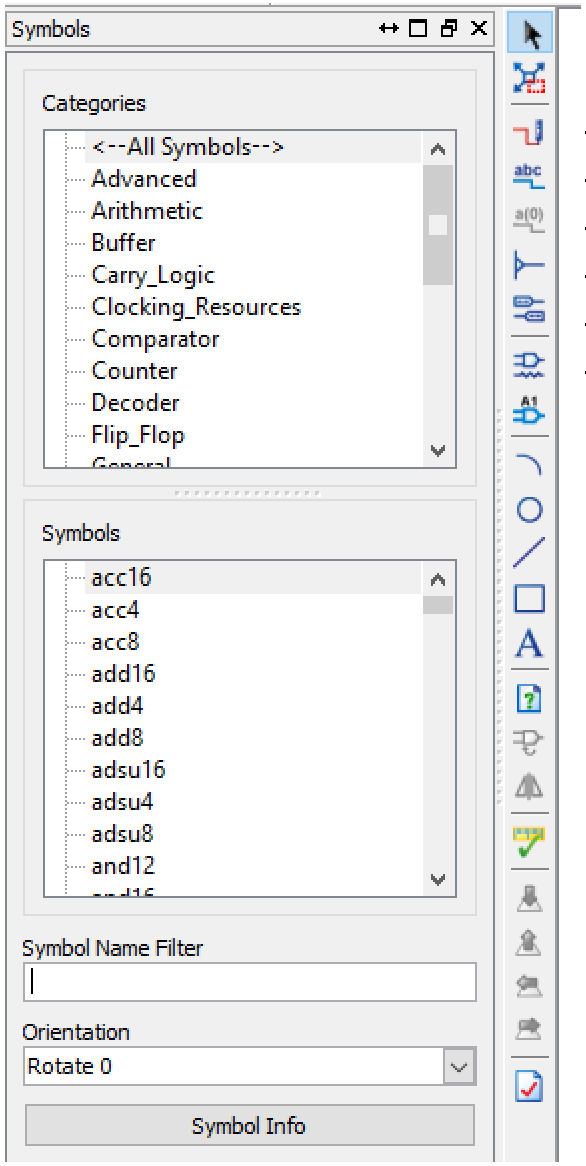
```
Started : "Launching Schematic Editor to edit logikai_kapuk.sch".
Launching Design Summary/Report Viewer...
```

Task

- **Logic gates implementation**



Tools for schematic design


Symbols

Categories

<--All Symbols-->
Advanced
Arithmetic
Buffer
Carry_Logic
Clocking_Resources
Comparator
Counter
Decoder
Flip_Flop
General

Symbols

acc16
acc4
acc8
add16
add4
add8
adsu16
adsu4
adsu8
and12
and16

Symbol Name Filter

Orientation: Rotate 0

Symbol Info

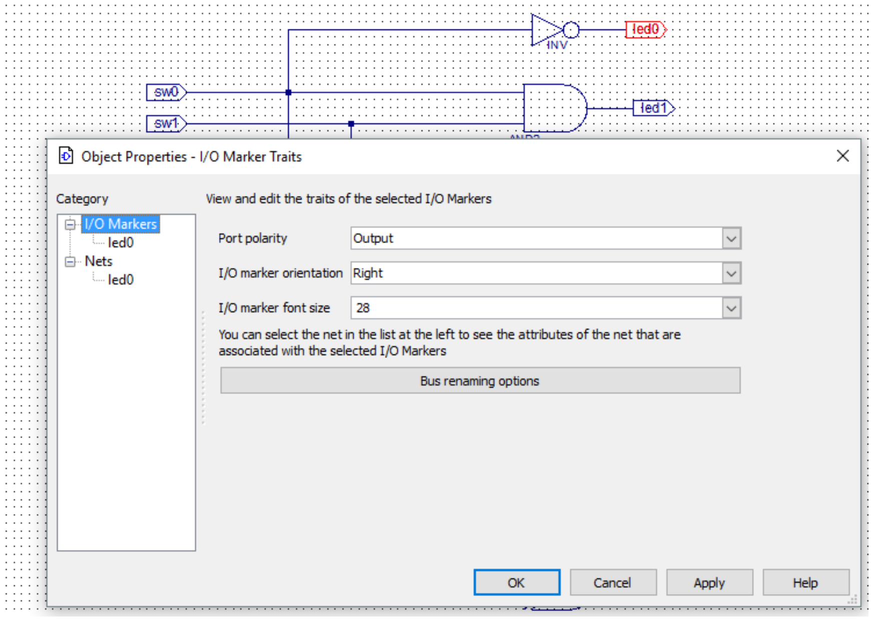
- Add wire
- Add net name
- Rename selected bus
- Add bus tap
- Add I/O marker
- Add symbol

- Select components from the **Symbols** tab of the upper left pane and drag them to your schematic.
- You can narrow down your choices using the **Categories**, or by typing the first few characters of the symbol you're looking for in the **Symbol Name Filter**, or just scroll through the lists. The important category for now is **Logic:** General logic gates.
- Use the wiring tool to wire up the components. It is in the tool bar and looks like a red line and a pencil.

- Place **I/O Markers** to the inputs and outputs using the **I/O Marker** widget.

Tools for schematic design

Change the name of the marker to what you se on the assignment. You should double click the marker. Then click on “Nets” and then edit the “Name”, also observe the Port Polarity, then click OK.

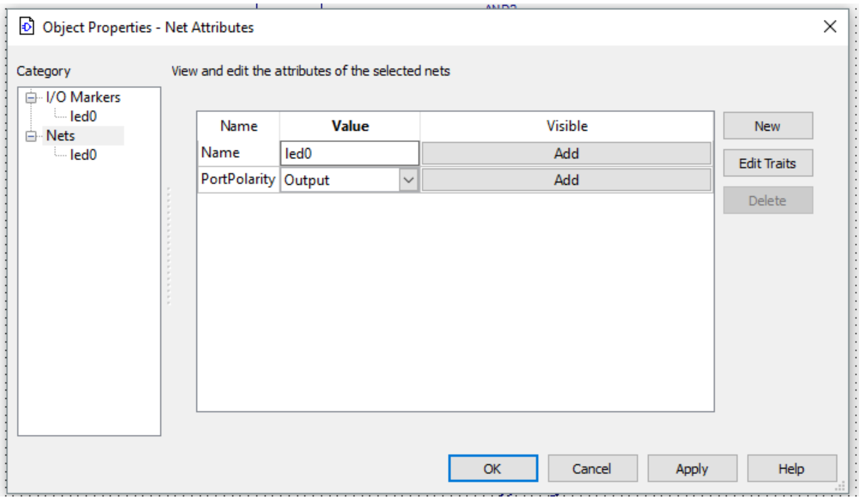


Object Properties - I/O Marker Traits

View and edit the traits of the selected I/O Markers

Port polarity	Output
I/O marker orientation	Right
I/O marker font size	28

You can select the net in the list at the left to see the attributes of the net that are associated with the selected I/O Markers



Object Properties - Net Attributes

View and edit the attributes of the selected nets

Name	Value	Visible
Name	led0	Add
PortPolarity	Output	Add

Constraints file

- Specify what physical pins on the FPGA will be connected to I/O ports from the design (I/O markers)

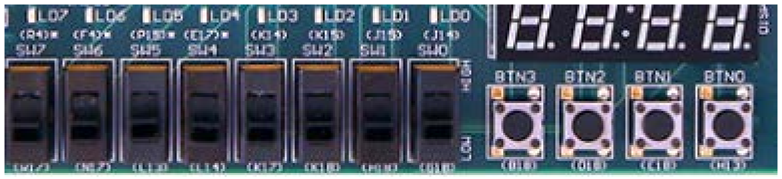
- **Project / New Source - > Implementation Constraint File**-, name first.ucf
- **Next/Finish**

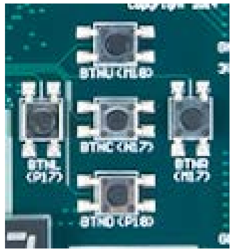
NEXYS 2 board

```
NET “sw0"   LOC = “B18"   ;
NET “sw1"   LOC = “D18"   ;
NET "led0"   LOC = “J14"   ;
NET "led1"   LOC = “J15"   ;
NET "led2"   LOC = “K15"   ;
NET "led3"   LOC = “K14"   ;
NET "led4"   LOC = “E17"   ;
NET "led5"   LOC = “p15"   ;
NET "led6"   LOC = “F4"   ;
```

NEXYS 4DDR board

```
NET “sw0"        LOC=J15 | IOSTANDARD=LVCMOS33;
NET “sw1"        LOC=L16 | IOSTANDARD=LVCMOS33;
# LEDs
NET "led0"       LOC=H17 | IOSTANDARD=LVCMOS33;
NET "led1"       LOC=K15 | IOSTANDARD=LVCMOS33;
NET "led2"       LOC=J13 | IOSTANDARD=LVCMOS33;
NET "led3"       LOC=N14 | IOSTANDARD=LVCMOS33;
NET "led4"       LOC=R18 | IOSTANDARD=LVCMOS33;
NET "led5"       LOC=V17 | IOSTANDARD=LVCMOS33;
NET "led6"       LOC=U17 | IOSTANDARD=LVCMOS33;
```

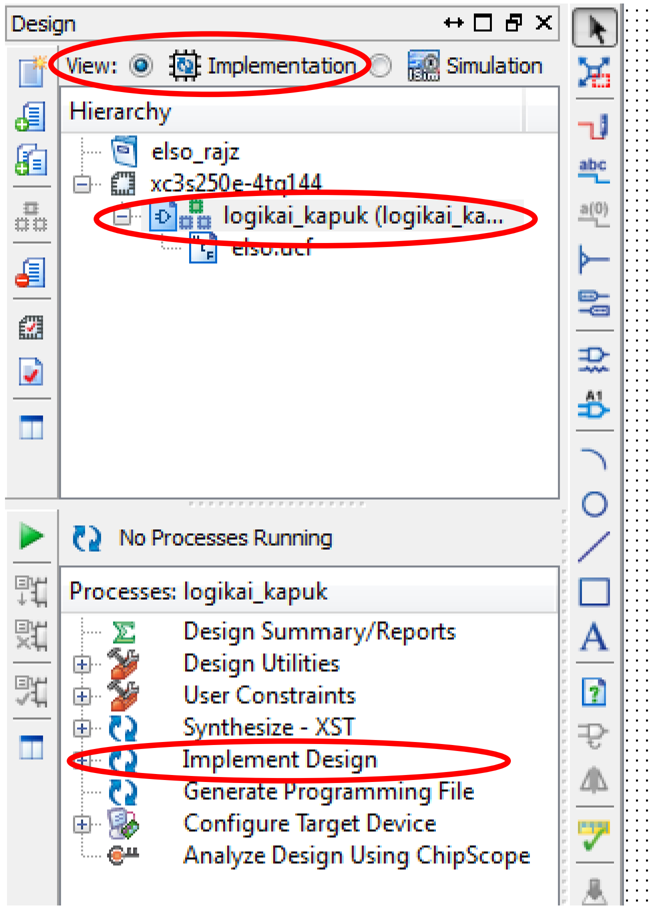




https://blog.digilentinc.com/the-constraints-file-also-known-as-magical-moving-stairs/

Project implementation

- **Implement Design,**
  - **View** →*implementation*
  - **Hierarchy** window→ select top level file
  - **Processes** ablak → Implement Design
- **Bit file generation**



Design

View: ● Implementation ○ Simulation

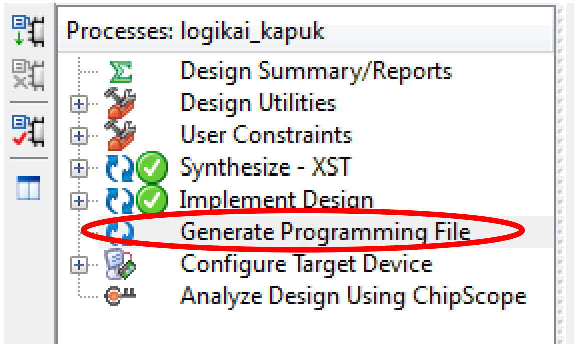
Hierarchy

elso_rajz
xc3s250e-4tq144
logikai_kapuk (logikai_ka...
elso.ucf

No Processes Running

Processes: logikai_kapuk

Design Summary/Reports
Design Utilities
User Constraints
Synthesize - XST
Implement Design
Generate Programming File
Configure Target Device
Analyze Design Using ChipScope



Processes: logikai_kapuk

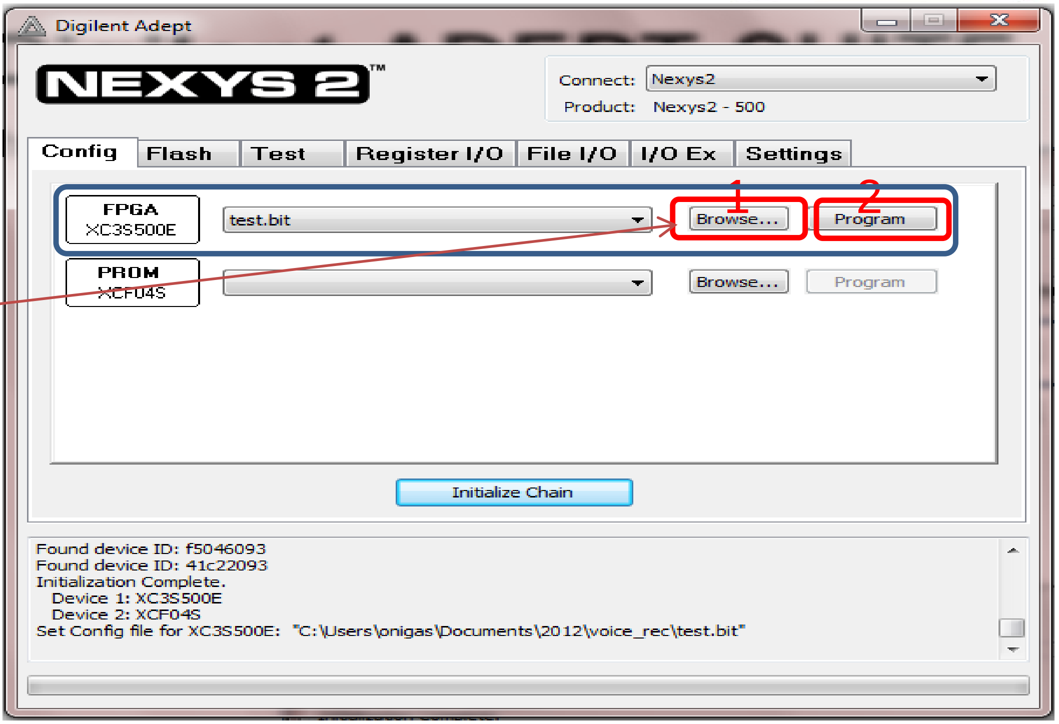
Design Summary/Reports
Design Utilities
User Constraints
Synthesize - XST
Implement Design
Generate Programming File
Configure Target Device
Analyze Design Using ChipScope

Configuration

- Final step in order to program the board. We will use the *.bit* file generated in the previous step.
  1. Using Impact program (part of ISE)
  2. Using Digilent Adept Suite

  https://www.digilentinc.com/Products/Detail.cfm?NavPath=2,66,828&Prod=ADEPT2

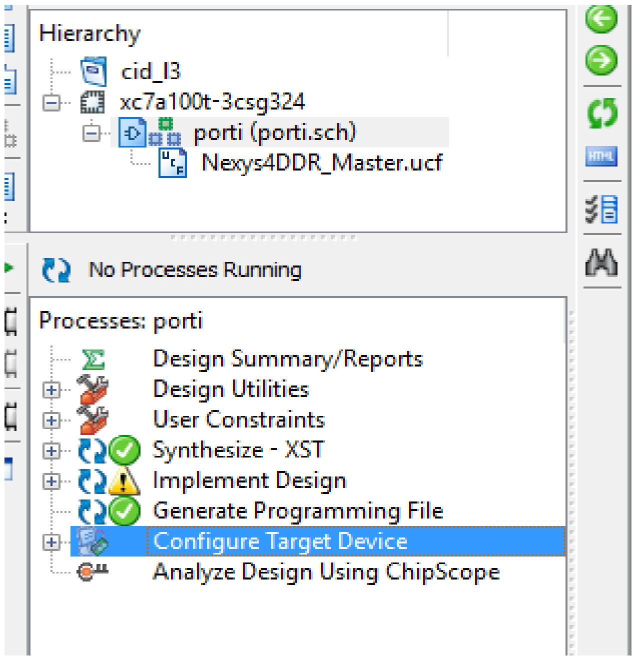
FPGA configuration: generated .bit (file)

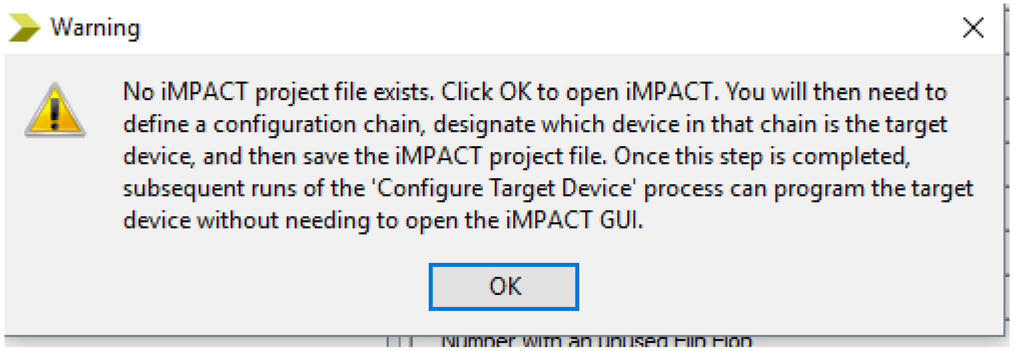


```
Found device ID: f5046093
Found device ID: 41c22093
Initialization Complete.
  Device 1: XC3S500E
  Device 2: XCF04S
Set Config file for XC3S500E:  "C:\Users\onigas\Documents\2012\voice_rec\test.bit"
```

Configuration using Impact

1. Configure Target Devices
2. OK

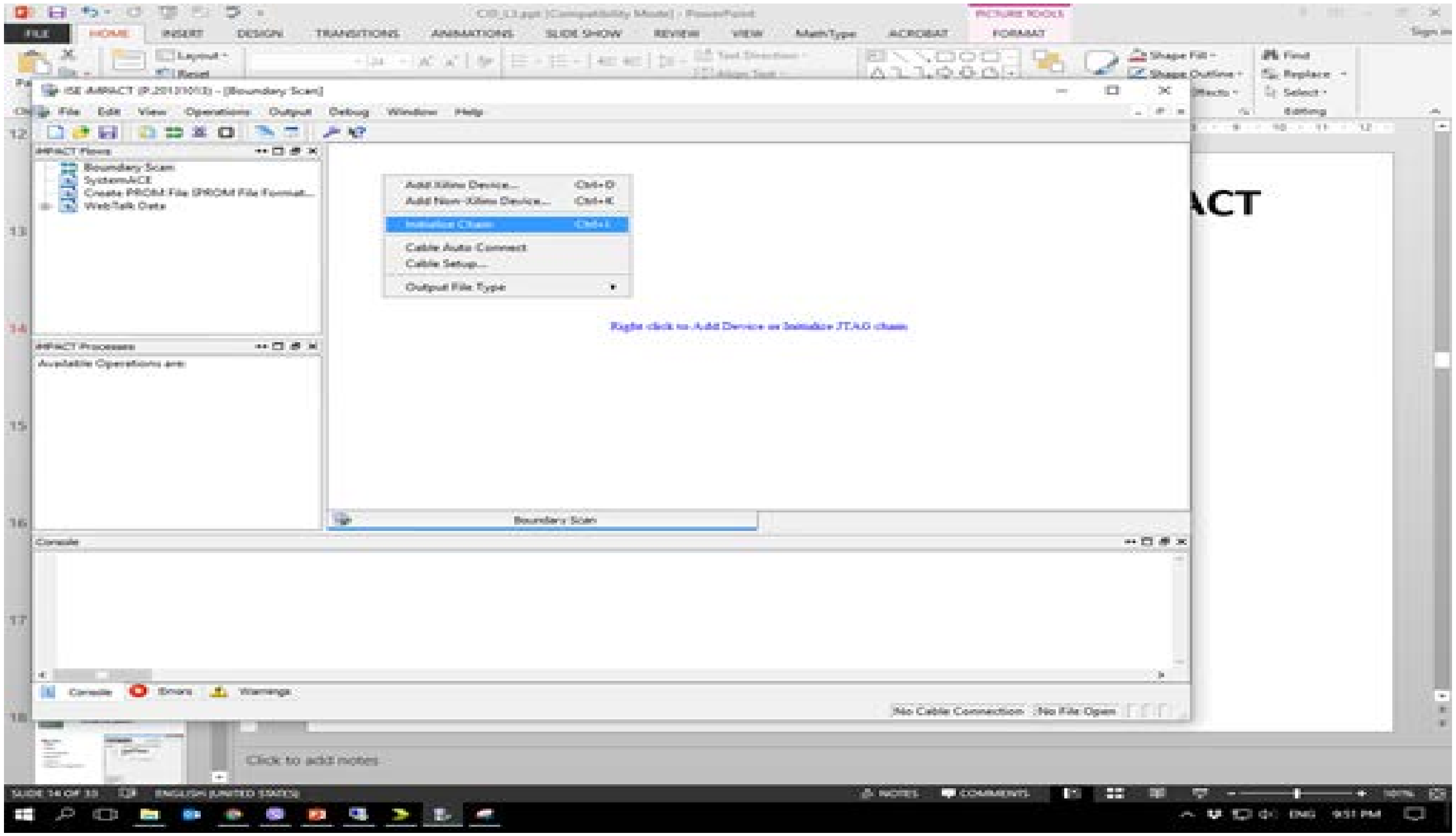




No iMPACT project file exists. Click OK to open iMPACT. You will then need to define a configuration chain, designate which device in that chain is the target device, and then save the iMPACT project file. Once this step is completed, subsequent runs of the 'Configure Target Device' process can program the target device without needing to open the iMPACT GUI.

Configuration using Impact

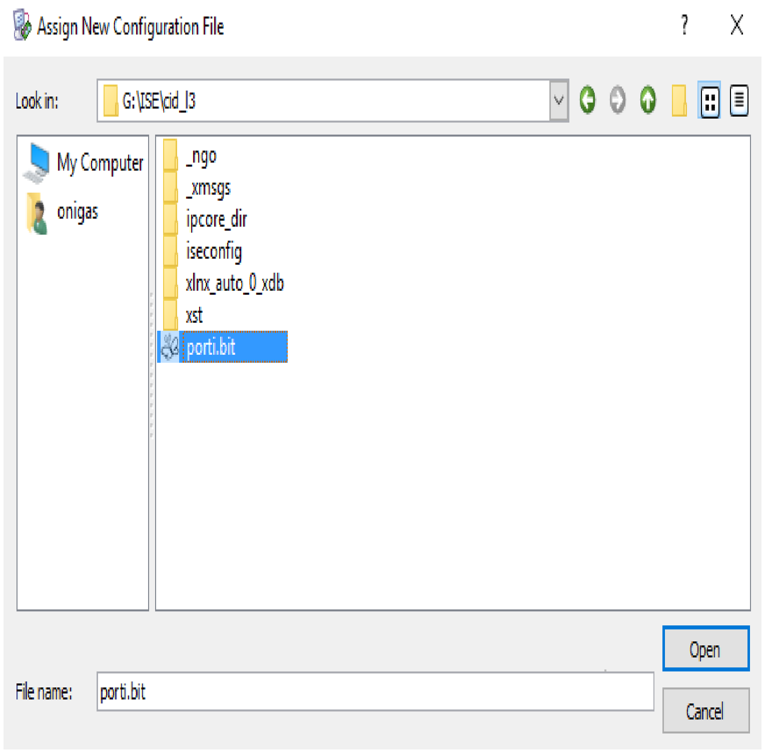
3. Boundary Scan (double click)
4. Right click on Boundary scan window
5. Initialize chain

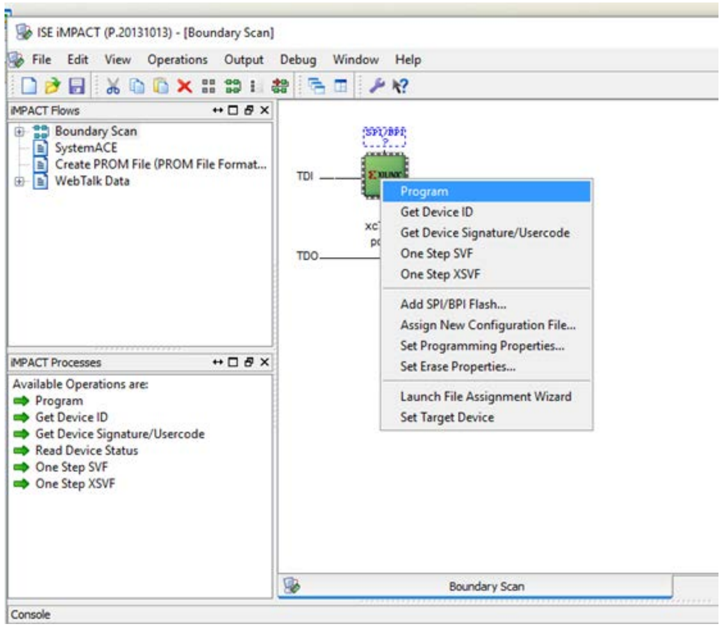

ISE iMPACT (P.20131013) - [Boundary Scan]
File Edit View Operations Output Debug Window Help
iMPACT Flows
Boundary Scan
SystemACE
Create PROM File (PROM File Formatter)
WebTalk Data
iMPACT Processes
Available Operations are:
Add Xilinx Device... Ctrl+D
Add Non-Xilinx Device... Ctrl+K
Initialize Chain Ctrl+I
Cable Auto Connect
Cable Setup...
Output File Type
Boundary Scan
Console
Console
Errors
Warnings
No Cable Connection
No File Open

Right click to Add Device or Initialize JTAG chain

Configuration using Impact

6. Assign new configuration file
7. Open, No, Ok
8. Right click on green icon, Program




ISE iMPACT (P.20131013) - [Boundary Scan]
File Edit View Operations Output Debug Window Help
iMPACT Flows
Boundary Scan
SystemACE
Create PROM File (PROM File Format...
WebTalk Data
iMPACT Processes
Available Operations are:
Program
Get Device ID
Get Device Signature/Usercode
Read Device Status
One Step SVF
One Step XSVF
TDI
TDO
Program
Get Device ID
Get Device Signature/Usercode
One Step SVF
One Step XSVF
Add SPI/BPI Flash...
Assign New Configuration File...
Set Programming Properties...
Set Erase Properties...
Launch File Assignment Wizard
Set Target Device
Boundary Scan
Console

Lab1_1Results

- Pushing sw0 and sw1 generate all inputs shown in next table and fill the table with led's state.

sw0	sw1	led0 NOT	led1 AND	led2 OR	led3 NAND	led4 NOR	led5 XOR	led6 XNOR
0	0							
0	1							
1	0							
1	1							

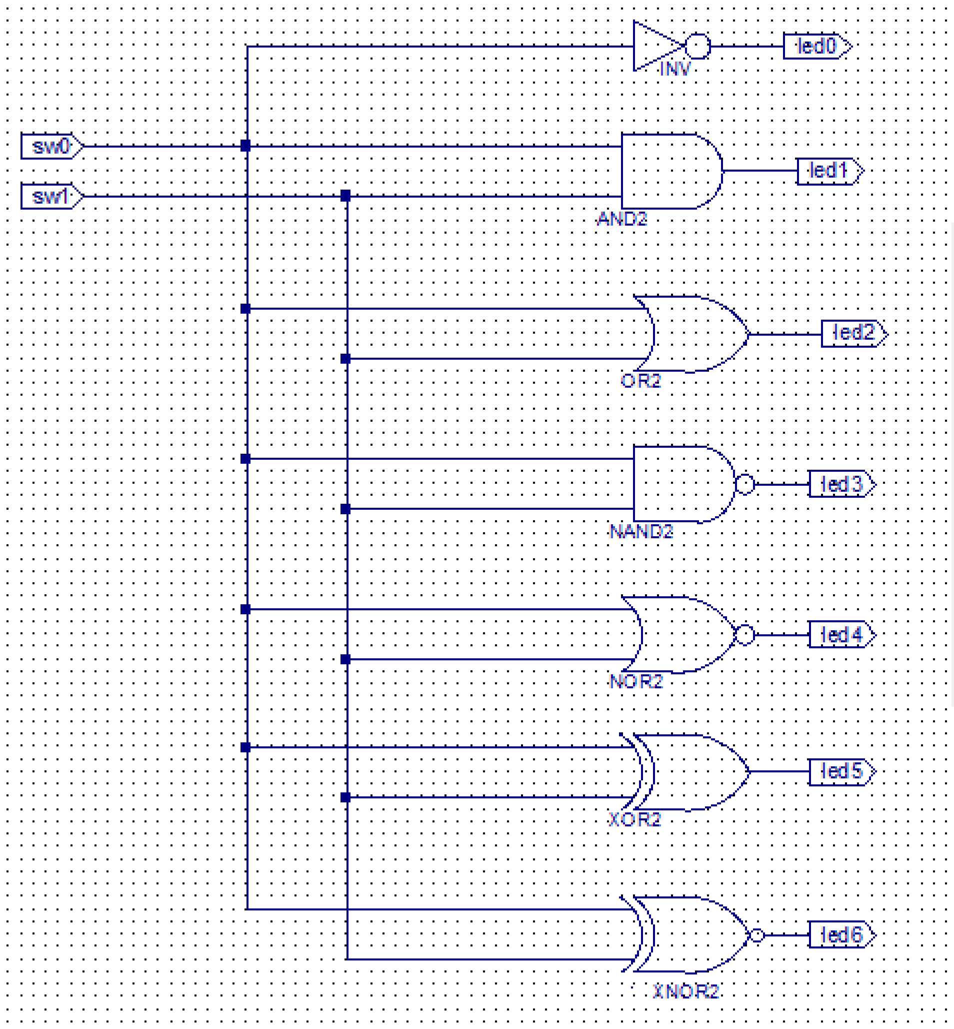
Lab1_2 assignment:

2 variable logic function implementation

- HDL design entry -

New task

- **Logic gates implementation using Verilog**



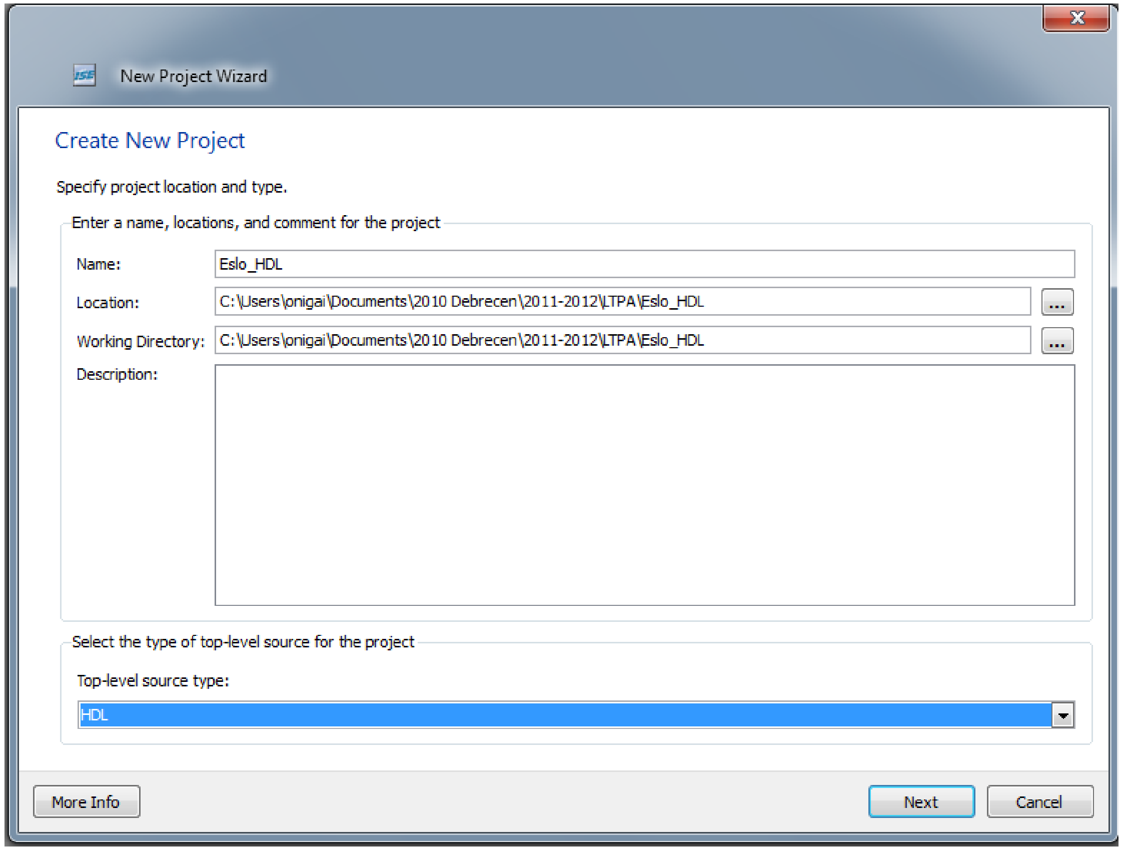
```
module elsoHDL(
    input sw0,
    input sw1,
    output led0, led1, led2, led3, led4, led5, led6

    );
assign led0 = ~ sw0;
assign led1 = sw0 & sw1;
assign led2 = sw0 | sw1;
assign led3 = ~ (sw0 & sw1);
assign led4 = ~ (sw0 | sw1);
assign led5 = sw0 ^ sw1;
assign led6 = sw0 ~^ sw1;

endmodule
```

New project

- Start -> Programs\Xilinx ISE Design Suite 14.7\ISE Design Tools\Project Navigator.
- New project (*File→New Project*) – name it firstHDL
- Top level souce HDL type!



Create New Project

Specify project location and type.

Enter a name, locations, and comment for the project

Name: Eslo_HDL

Location: C:\Users\onigai\Documents\2010 Debrecen\2011-2012\LTPA\Eslo_HDL

Working Directory: C:\Users\onigai\Documents\2010 Debrecen\2011-2012\LTPA\Eslo_HDL

Description:

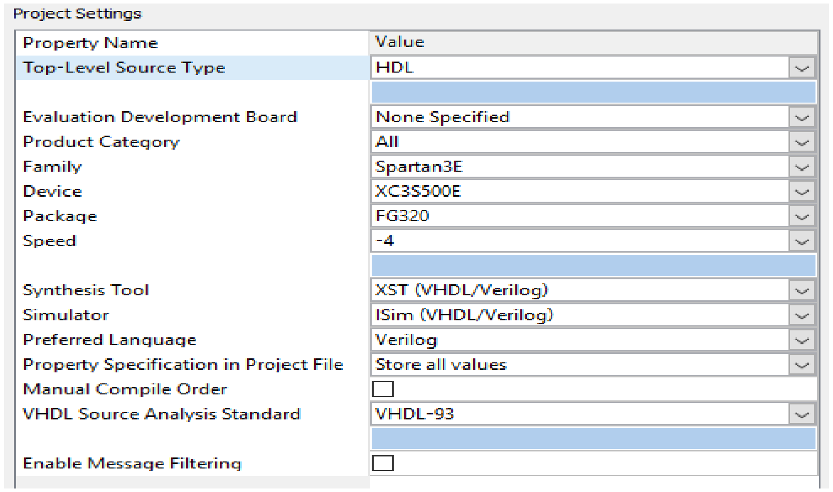
Select the type of top-level source for the project

Top-level source type: HDL

Project settings

- **Next -> Device Properties -> Value :**

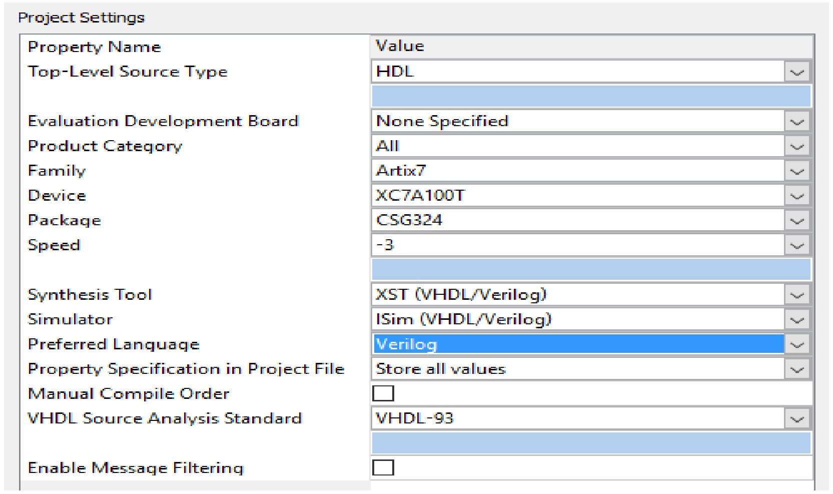
NEXYS 2 board
Device Family: Spartan3E
Device: xc3s500E
Package: FG320
Speed Grade: -4
Synthesis Tool: XST (VHDL/Verilog)
Simulator: ISim (VHDL/Verilog)
Preffered Language: Verilog



Project Settings

Property Name	Value
Top-Level Source Type	HDL
Evaluation Development Board	None Specified
Product Category	All
Family	Spartan3E
Device	XC3S500E
Package	FG320
Speed	-4
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	Verilog
Property Specification in Project File	Store all values
Manual Compile Order	☐
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	☐

NEXYS 4 DDR board
Family: Artix7
Device: XC7A100T
Package: CSG324
Speed Grade: -3
Synthesis Tool: XST (VHDL/Verilog)
Simulator: ISim (VHDL/Verilog)
Preffered Language: Verilog



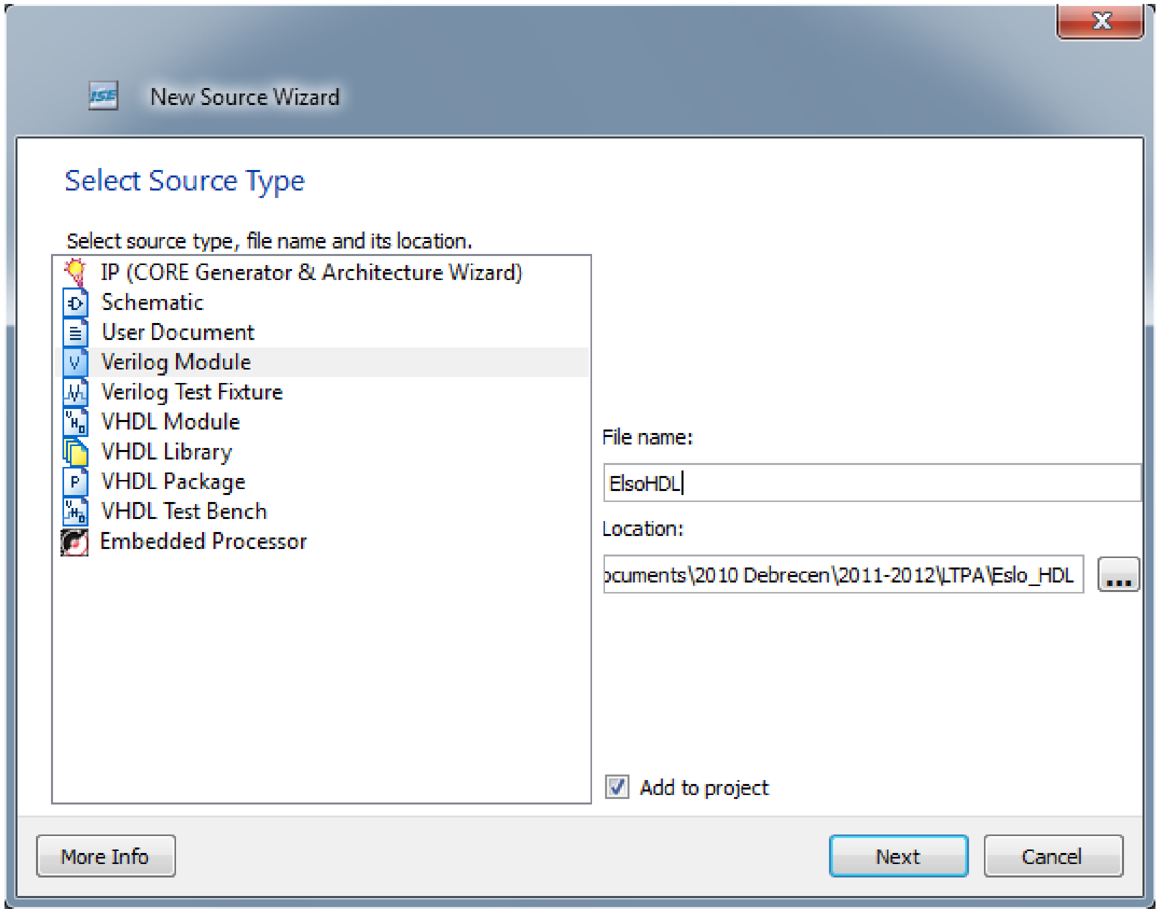
Project Settings

Property Name	Value
Top-Level Source Type	HDL
Evaluation Development Board	None Specified
Product Category	All
Family	Artix7
Device	XC7A100T
Package	CSG324
Speed	-3
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	Verilog
Property Specification in Project File	Store all values
Manual Compile Order	☐
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	☐

- **Next, Finish.**

Adding source file(HDL)

- *Project→New Source..!*
- Type **Verilog Module**, name firstHDL!
- If necessary (*Project→Add Source...*)


New Source Wizard

Select Source Type

Select source type, file name and its location.

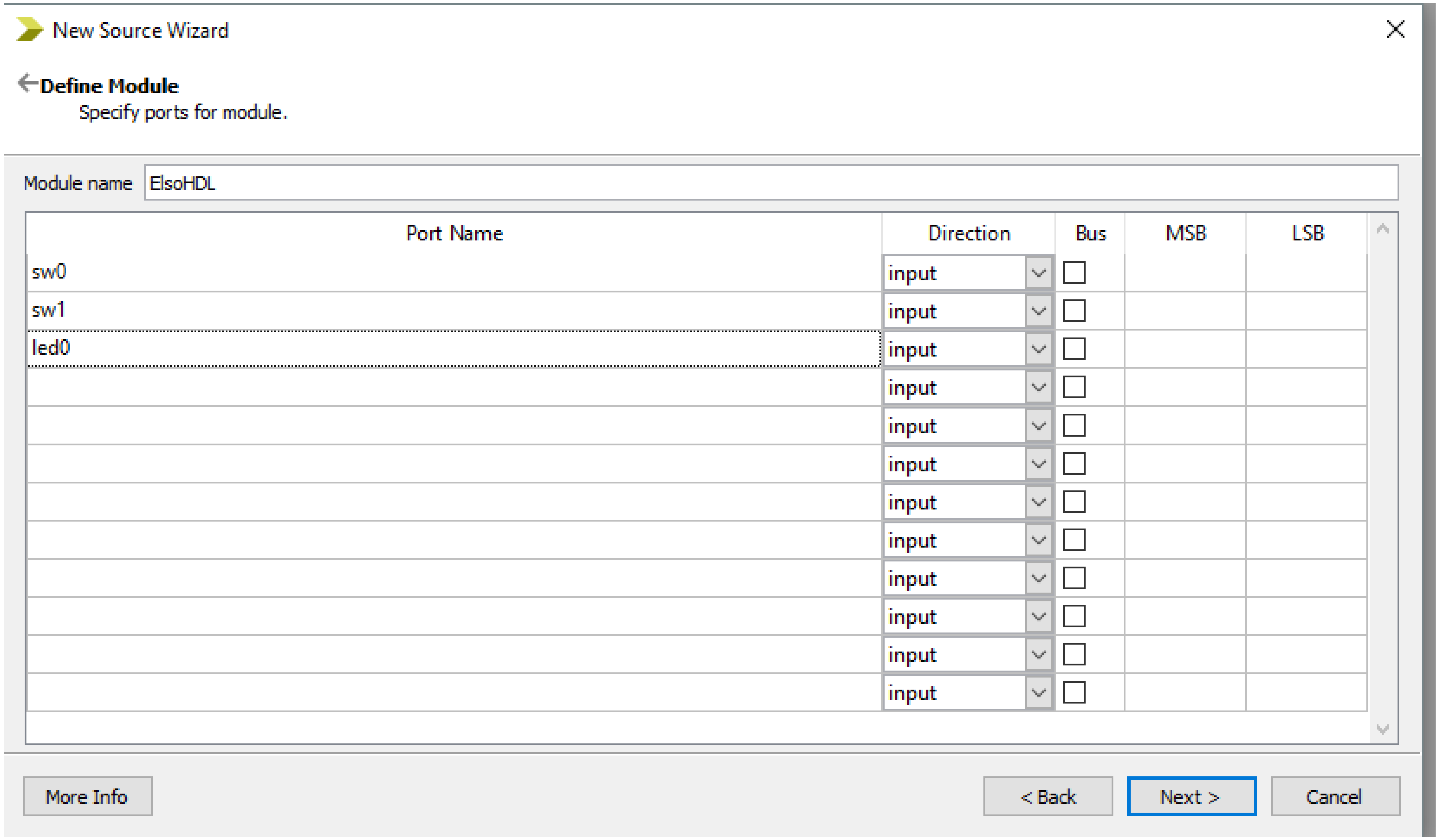
- IP (CORE Generator & Architecture Wizard)
- Schematic
- User Document
- Verilog Module
- Verilog Test Fixture
- VHDL Module
- VHDL Library
- VHDL Package
- VHDL Test Bench
- Embedded Processor

File name: ElsoHDL

Location: ocuments\2010 Debrecen\2011-2012\LTPA\Eslo_HDL

Add to project [✓]

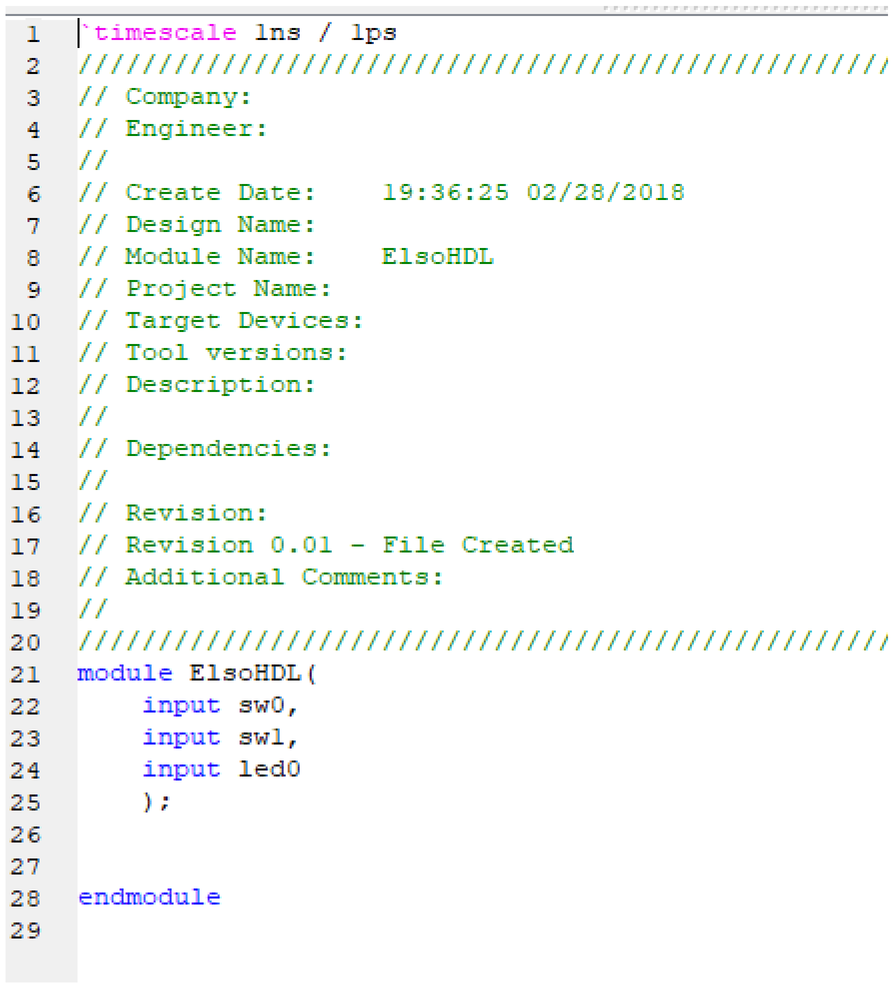
Defining input-output ports



Module name: ElsoHDL

Port Name	Direction	Bus	MSB	LSB
sw0	input	☐		
sw1	input	☐		
led0	input	☐		
	input	☐		
	input	☐		
	input	☐		
	input	☐		
	input	☐		
	input	☐		
	input	☐		
	input	☐		
	input	☐		

Generated HDL file

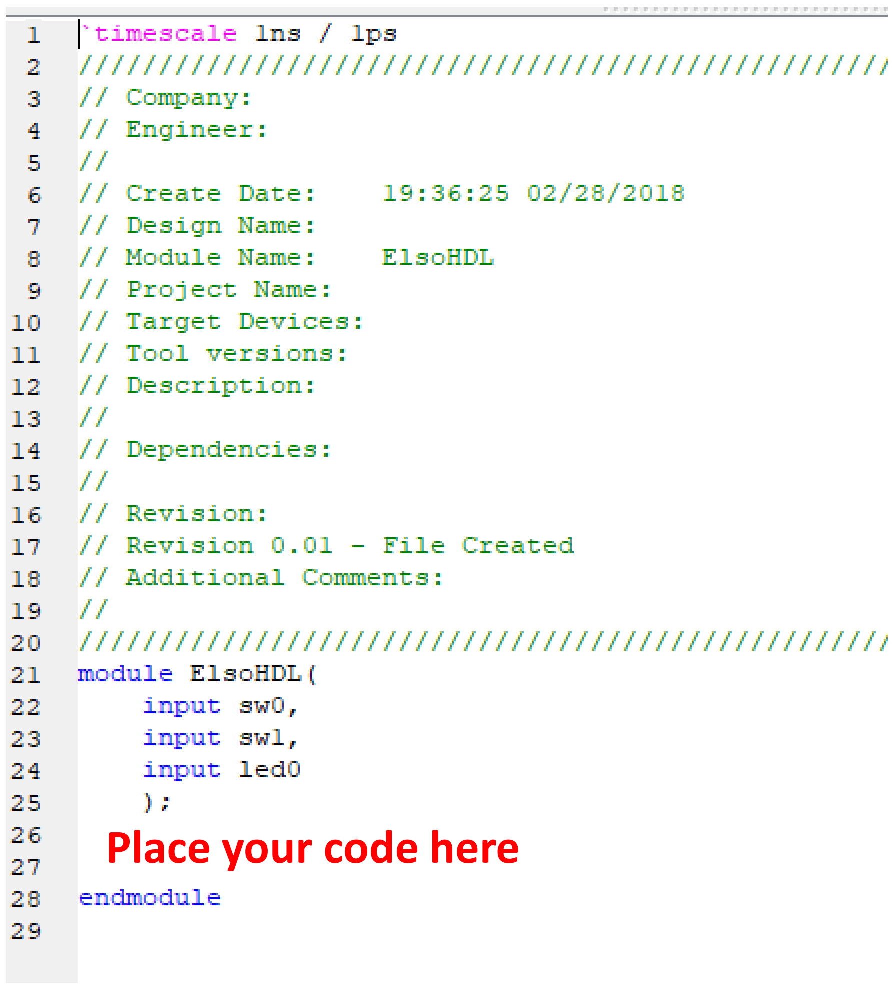


```
`timescale 1ns / 1ps
////////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    19:36:25 02/28/2018
// Design Name:
// Module Name:    ElsoHDL
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////////
module ElsoHDL(
    input sw0,
    input sw1,
    input led0
    );


endmodule

```

Generated HDL file



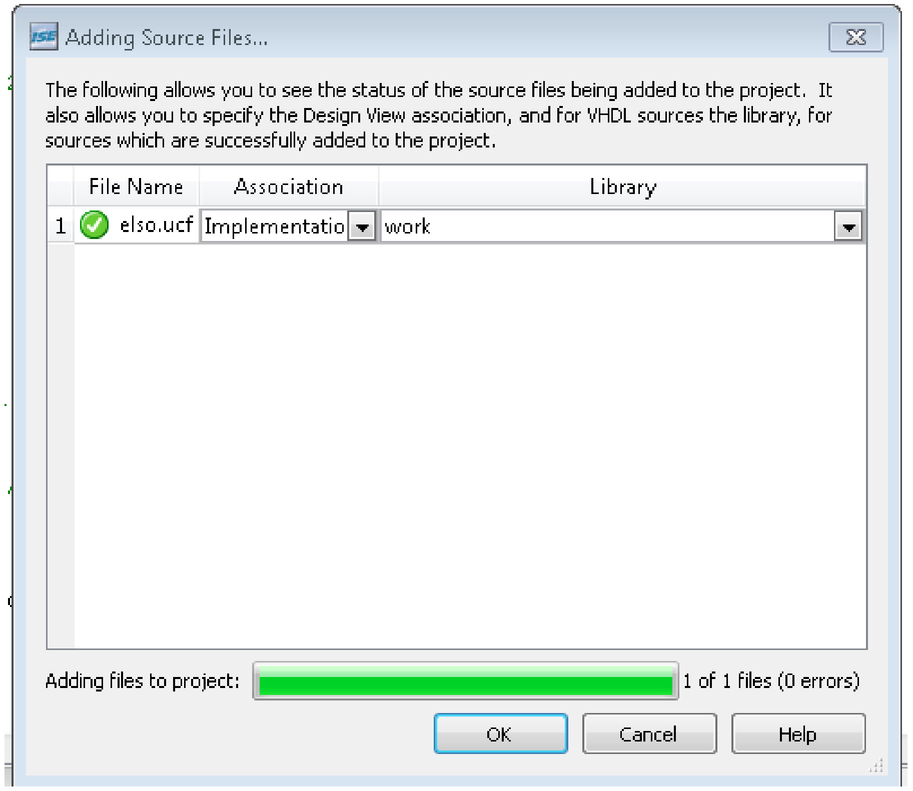
```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    19:36:25 02/28/2018
// Design Name:
// Module Name:    ElsoHDL
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module ElsoHDL(
    input sw0,
    input sw1,
    input led0
    );
```

Place your code here

```
endmodule
```

Constraint

- Choose **Project / Add Copy of Source** first.ucf (created in last project).
- Press **OK Sources** window will show first.*ucf* file.

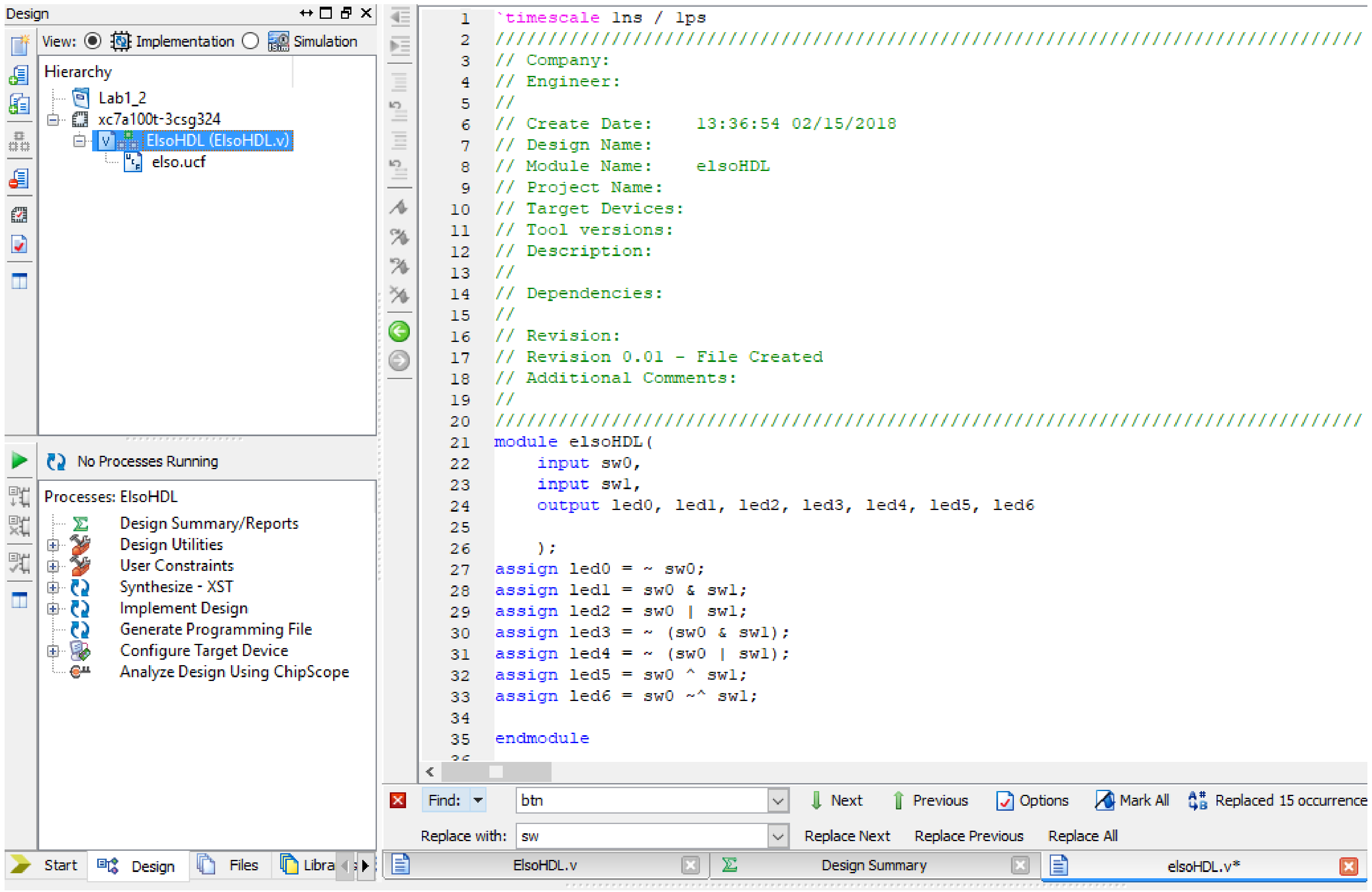

Adding Source Files...

The following allows you to see the status of the source files being added to the project. It also allows you to specify the Design View association, and for VHDL sources the library, for sources which are successfully added to the project.

	File Name	Association	Library
1	✓ elso.ucf	Implementatio	work

Adding files to project: 1 of 1 files (0 errors)

Constraints



Design

View: Implementation Simulation

Hierarchy

- Lab1_2
- xc7a100t-3csg324
  - ElsoHDL (ElsoHDL.v)
    - elso.ucf

No Processes Running

Processes: ElsoHDL

- Design Summary/Reports
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
- Generate Programming File
- Configure Target Device
- Analyze Design Using ChipScope

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    13:36:54 02/15/2018
// Design Name:
// Module Name:    elsoHDL
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module elsoHDL(
    input sw0,
    input sw1,
    output led0, led1, led2, led3, led4, led5, led6

    );
assign led0 = ~ sw0;
assign led1 = sw0 & sw1;
assign led2 = sw0 | sw1;
assign led3 = ~ (sw0 & sw1);
assign led4 = ~ (sw0 | sw1);
assign led5 = sw0 ^ sw1;
assign led6 = sw0 ~^ sw1;

endmodule
```

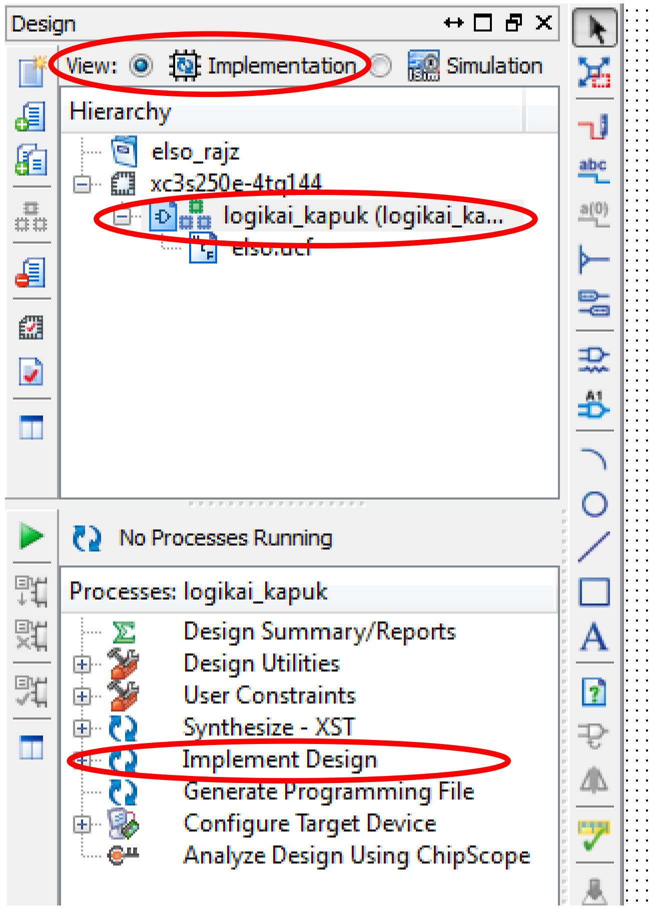
Find: btn

Replace with: sw

Replaced 15 occurrence

Project implementation

- **Implement Design,**
  - **View** →*implementation*
  - **Hierarchy** window→ select top level file
  - **Processes** ablak → Implement Design
- **Bit file generation**



Design

View: Implementation Simulation

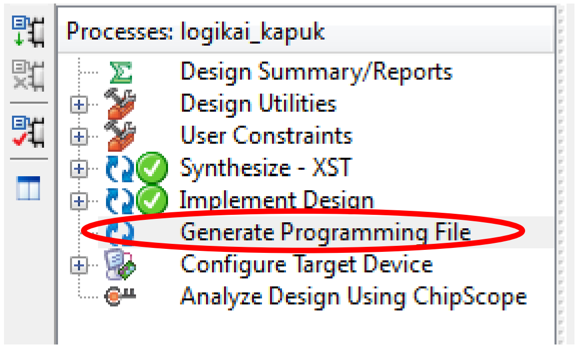
Hierarchy

elso_rajz
xc3s250e-4tq144
logikai_kapuk (logikai_ka...
elso.ucf

No Processes Running

Processes: logikai_kapuk

Design Summary/Reports
Design Utilities
User Constraints
Synthesize - XST
Implement Design
Generate Programming File
Configure Target Device
Analyze Design Using ChipScope



Processes: logikai_kapuk

Design Summary/Reports
Design Utilities
User Constraints
Synthesize - XST
Implement Design
Generate Programming File
Configure Target Device
Analyze Design Using ChipScope

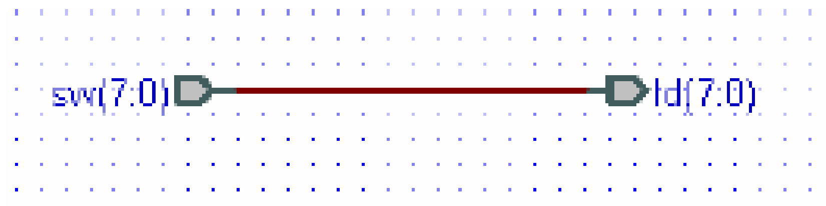
Lab1_2 Results

- Pushing sw0 and sw1 generate all inputs shown in next table and fill the table with led's state.

sw0	sw1	led0 NOT	led1 AND	led2 OR	led3 NAND	led4 NOR	led5 XOR	led6 XNOR
0	0							
0	1							
1	0							
1	1							

Lab1_3 assignment:

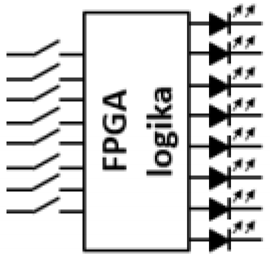
command 8 LEDs using 8 switches on the board
- HDL design entry-



Lab1_3

- The 8 inputs sw [7:0] (switches) and 8 outputs ld [7:0] (leds) cpuld be considered as individual bits as in previous projects or as vectors.

- Individual bits:



```
module sw2led(
      input sw0, sw1, sw2, sw3, sw4, sw5, sw6, sw7,
      output ld0, ld1, ld2, ld3, ld4, ld5, ld6, ld7
        );
assign ld0 = sw0;
assign ld1 = sw1;
assign ld2 = sw2;
assign ld3 = sw3;
assign ld4 = sw4;
assign ld5 = sw5;
assign ld6 = sw6;
assign ld7 = sw7;

endmodule
```

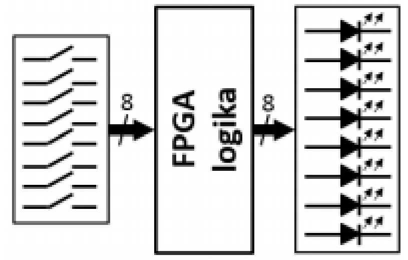
<u>**NEXYS 2 board**</u>

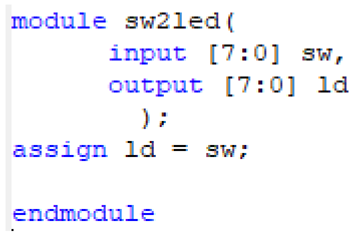
```
# 8 switches, from left to right
NET "sw7"   LOC = “R17";
NET "sw6"   LOC = “N17";
NET "sw5"   LOC = “L13";
NET "sw4"   LOC = “L14";
NET "sw3"   LOC = “K17";
NET "sw2"   LOC = “K18";
NET "sw1"   LOC = “H18";
NET "sw0"   LOC = “G18";

# 8 LEDs, from left to right
NET "ld7"   LOC = “R4";
NET "ld6"   LOC = “F4";
NET "ld5"   LOC = “P15";
NET "ld4"   LOC = “E17";
NET "ld3"   LOC = “K14";
NET "ld2"   LOC = “K15";
NET "ld1"   LOC = “J15";
NET "ld0"   LOC = “J14";
```

Lab1_3

- Bit vector description:





```
module sw2led(
        input  [7:0] sw,
        output  [7:0] ld
           );
assign ld = sw;

endmodule
```

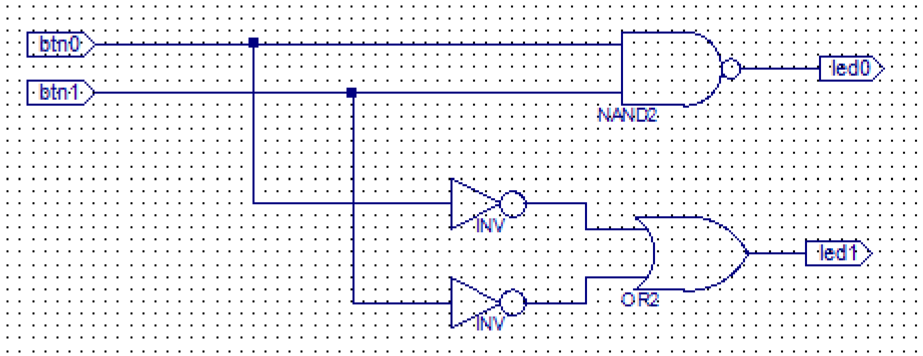
NEXYS 2 board

```
# 8 switches, from left to right
NET "sw<7>"   LOC = “R17";
NET "sw<6>"   LOC = “N17";
NET "sw<5>"   LOC = “L13";
NET "sw<4>"   LOC = “L14";
NET "sw<3>"   LOC = “K17";
NET "sw<2>"   LOC = “K18";
NET "sw<1>"   LOC = “H18";
NET "sw<0>"   LOC = “G18";

# 8 LEDs, from left to right
NET "ld<7>"   LOC = “R4";
NET "ld<6>"   LOC = “F4";
NET "ld<5>"   LOC = “P15";
NET "ld<4>"   LOC = “E17";
NET "ld<3>"   LOC = “K14";
NET "ld<2>"   LOC = “K15";
NET "ld<1>"   LOC = “J15";
NET "ld<0>"   LOC = “J14";
```

„Schematic” extra task*

Design using schematic entry and implement the circuit on following circuit

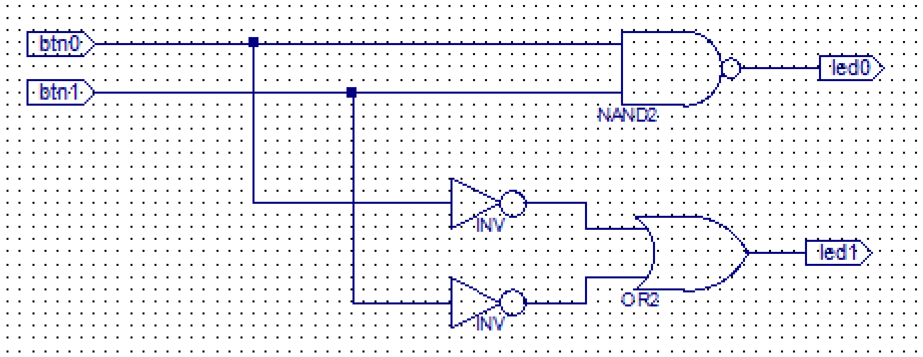


btn0	btn1	led0=(AB)’	led1=A’+B’
0	0		
0	1		
1	0		
1	1		

*** only for interested students**

„HDL” extra task*

Design using HDL entry and implement the circuit on following circuit



btn0	btn1	led0=(AB)’	led1=A’+B’
0	0		
0	1		
1	0		
1	1		

*** only for interested students**