Week 4

Announcement

- Office Hours changed

  Office hours :

  -Monday 10-11

  - Tuesday 11-12

  - ~~Friday 12-1~~

Behavioral Modeling --- Continue

- **Conditional Statements**
- Syntax

  if (*expression)*

  then clause

  else

  else clause
- Note:
  - Else clause is optional
- Conditional Operator
  - Can be used in place of If…Then…Else
  - Syntax: *Condition ? True Expression : False Expression ;*

- Case
  - Syntax
- case (*expression)*

  alternative 1 : *statement(s) 1 ;*

  alternative 2 : *statement(s) 2 ;*

  …

  …

  alternative *n : statement(s) n ;*

  default : *default statement(s) ;*

  endcase
  - Default
    - Implemented when none of the alternatives are true
  - Variations
    - Replace *case with casex or casez for comparing don't cares or high impedance states*

Operators

Operation Type	*Symbol*	*Operation*
Arithmetic	*	Multiplication
	/	Division
	+	Addition
	-	Subtraction
	%	Modulus
Logical	!	Negation
	&&	AND
	||	OR
Relational	>	Greater Than
	<	Less Than
	<=	Greater Than or Equal To
	>=	Less Than or Equal To
Equality	==	Equality
	!=	Inequality
	===	Case Equality
	!==	Case Inequality

Samir Palnitkar, *Verilog HDL A Guide to Digital Design and Synthesis*, Prentice Hall, Inc., 4th Edition, Table 6-1, pp. 92-93, 1996

Operators

Operation Type	*Symbol*	*Operation*
Bitwise	~	Bitwise Negation
Operation Type	&	Bitwise AND
	|	Bitwise OR
	^	Bitwise XOR
	~^ *or* ^~	Bitwise XNOR
Reduction	&	Reduction AND
	~&	Reduction NAND
	|	Reduction OR
	~|	Reduction NOR
	^	Reduction XOR
	~^ *or* ^~	Reduction XNOR
Shift	>>	Right Shift
	<<	Left Shift
Concatenation	{ }	Concatentation
	{ { } }	Replication
Conditional	?:	Conditional

Samir Palnitkar, *Verilog HDL A Guide to Digital Design and Synthesis*, Prentice Hall, Inc., 4th Edition, Table 6-1, pp. 92-93, 1996

Operator Precedence

Operators	*Symbols*	*Precedence*
Unary	+ - ! ~	Highest
Multiply, Divide, Modulus	* / %	
Add, Subtract	+	
Shift	-	
Relational	%	
Equality	!	
Reduction	&, ~&	
	^, ~^	
	|, ~|	
Logical	&&	
	||	
Conditional	?:	Lowest

Samir Palnitkar, *Verilog HDL A Guide to Digital Design and Synthesis*, Prentice Hall, Inc., 4th Edition, Table 6-4, pp. 101-102, 1996

- **Numbers**
  - Syntax
    - Sized
      - *Size'Format Number*
      - *Size*
        - » Number of digits
      - *Format*
        - » h (Hexadecimal)
        - » d (Decimal)
        - » o (Octal)
        - » b (Binary)
      - Number
        - » Number specified
    - Unsized
      - *'Format Number*
    - Examples
      - 4'h a729
      - 'd 62923
      - 8'b 1101zzzz
      - 16'h x

Loops

- While
  - Syntax

    while (*condition)*
    begin
        statement(s);
    end
- For
  - Syntax

    for (*initial condition; termination condition; control variable change)*
    begin
        statement(s);
    end
- Repeat
  - Repeats a given number of times based on the number, variable, or signal value given
  - Syntax

    repeat (*number, variable, or signal value)*
    begin
        statement(s);
    end

Loops

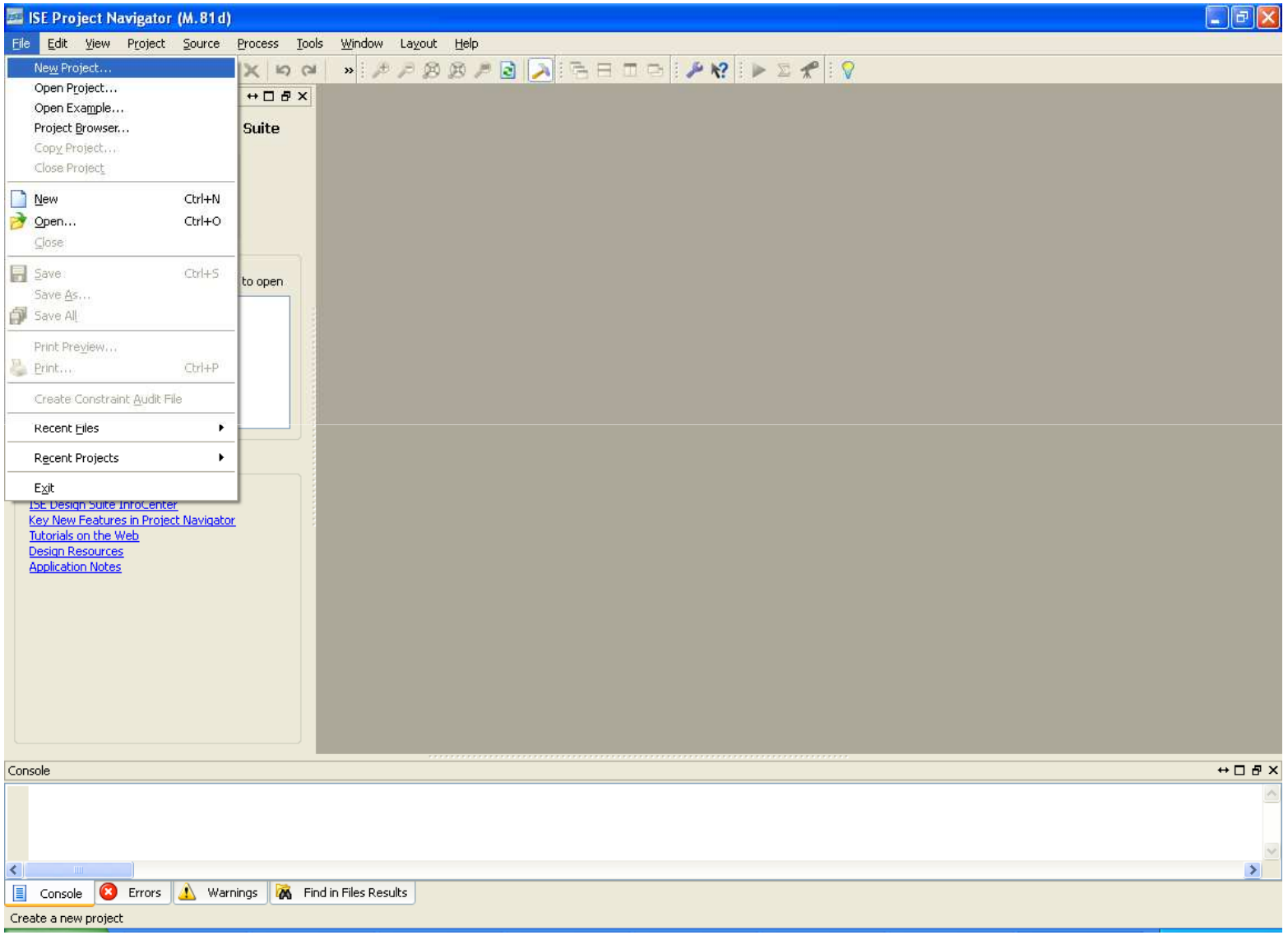
- Forever
  - Statement executes forever
    - Can be disabled by the keyword disable
  - Syntax
    - forever statement;
  - Example
    - forever #10 clk=~clk;

- Concatenation Operator
  - {…,…}
  - Allows several wires to be combined together into a single multi-bit wire.

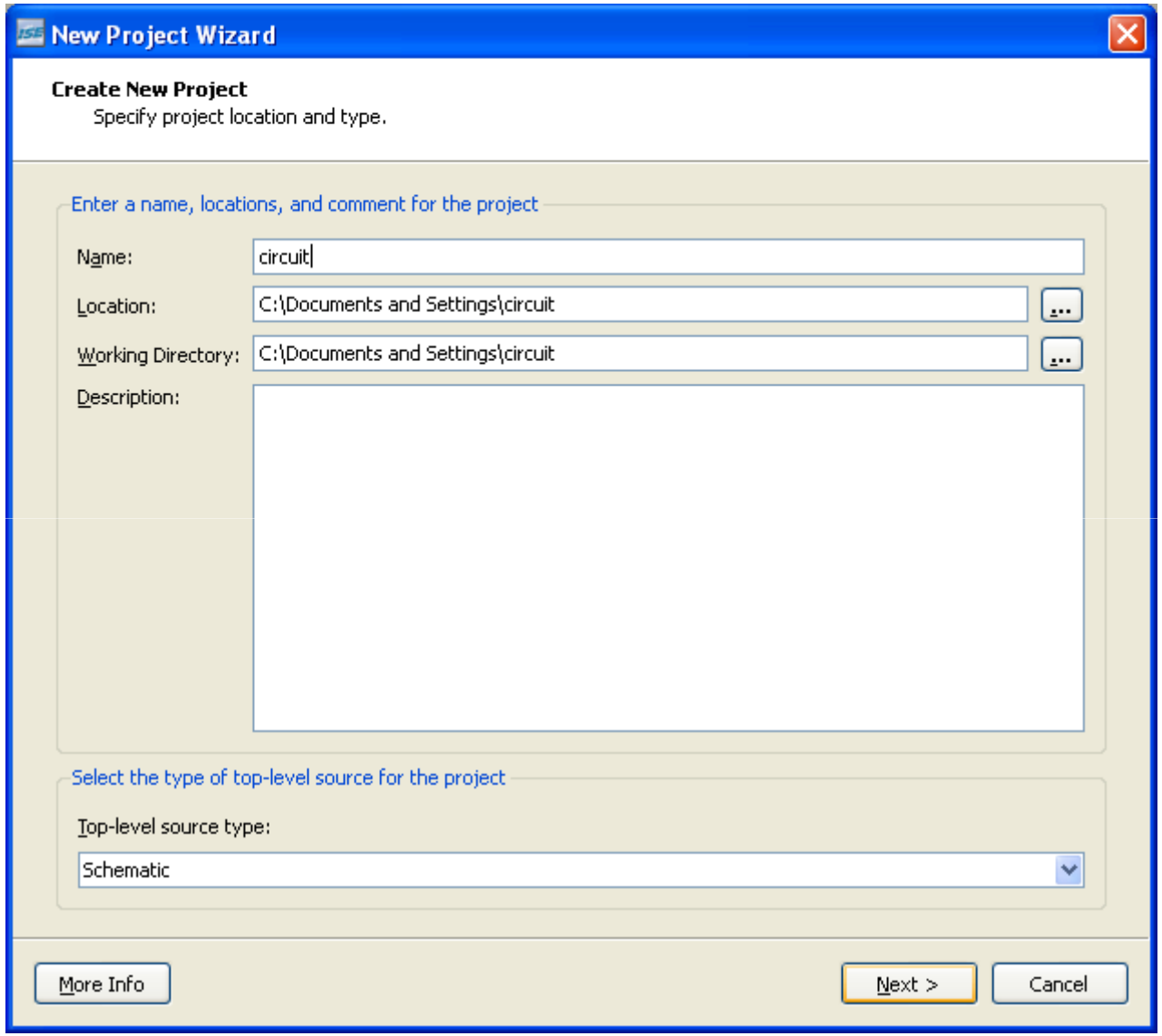
- **In the next slides we will implement a full example to program the fpga.**

- **Try to follow these step to create your first project.**

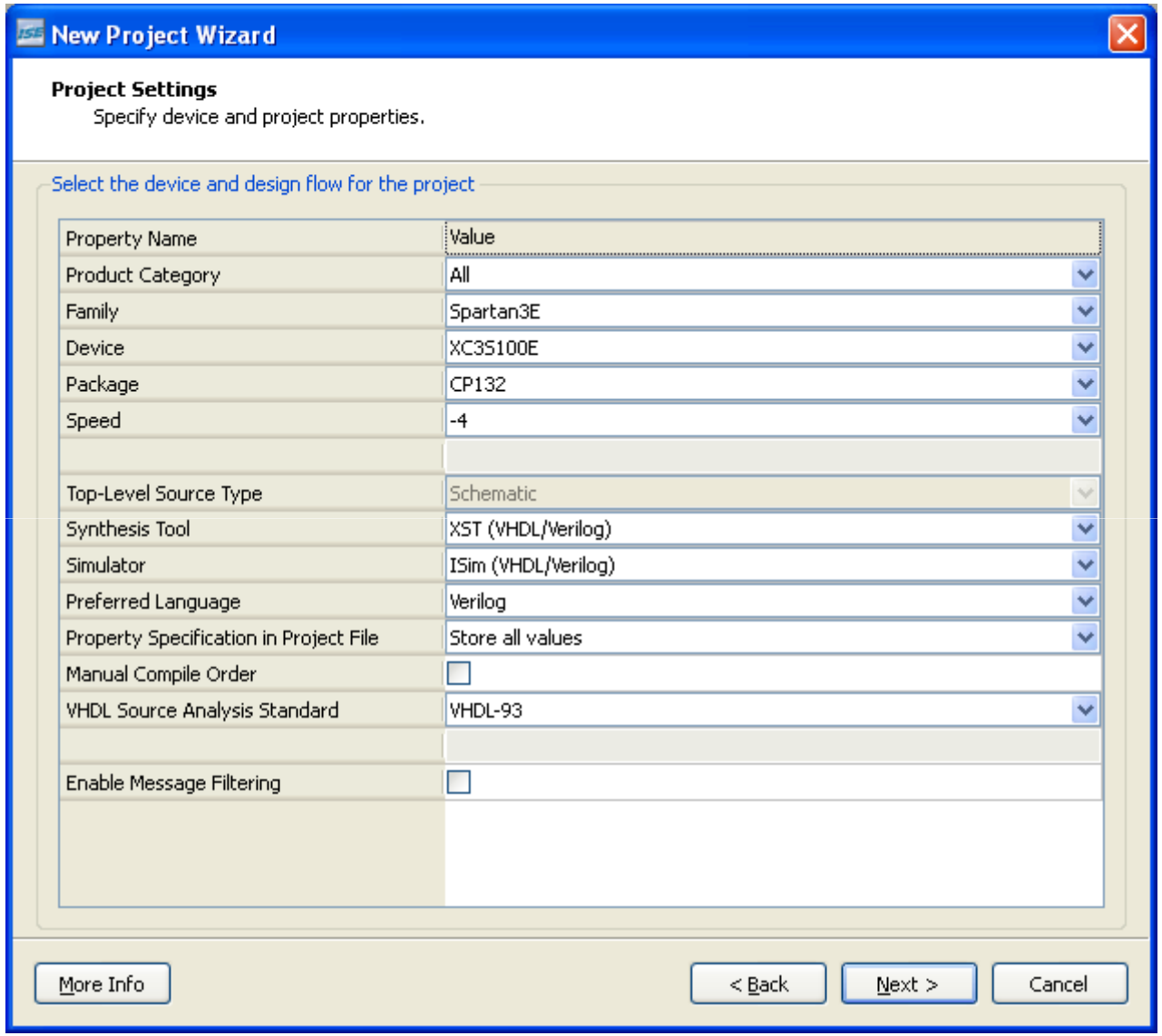
- Open the ISE project Navigator



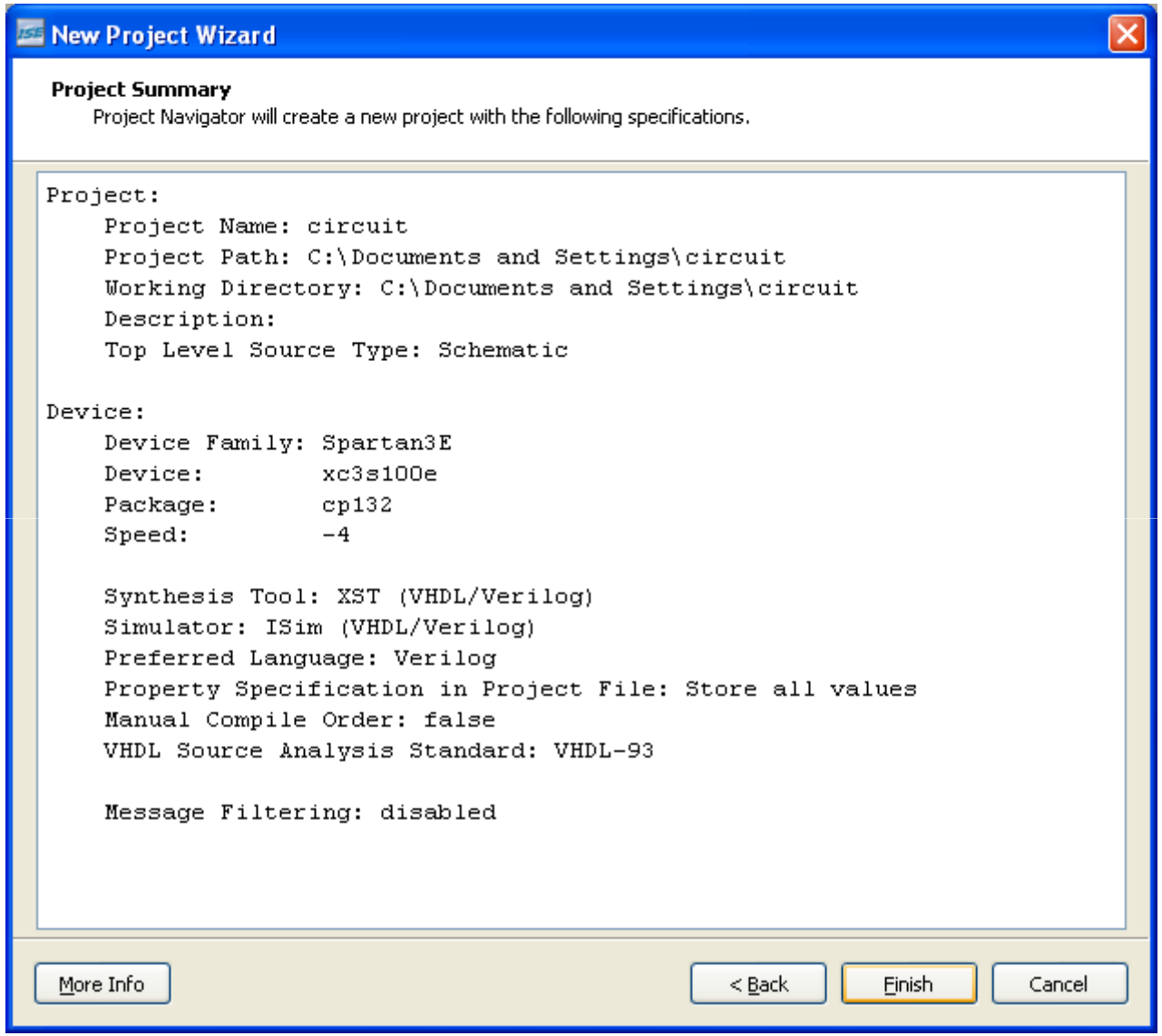
- Type the project name



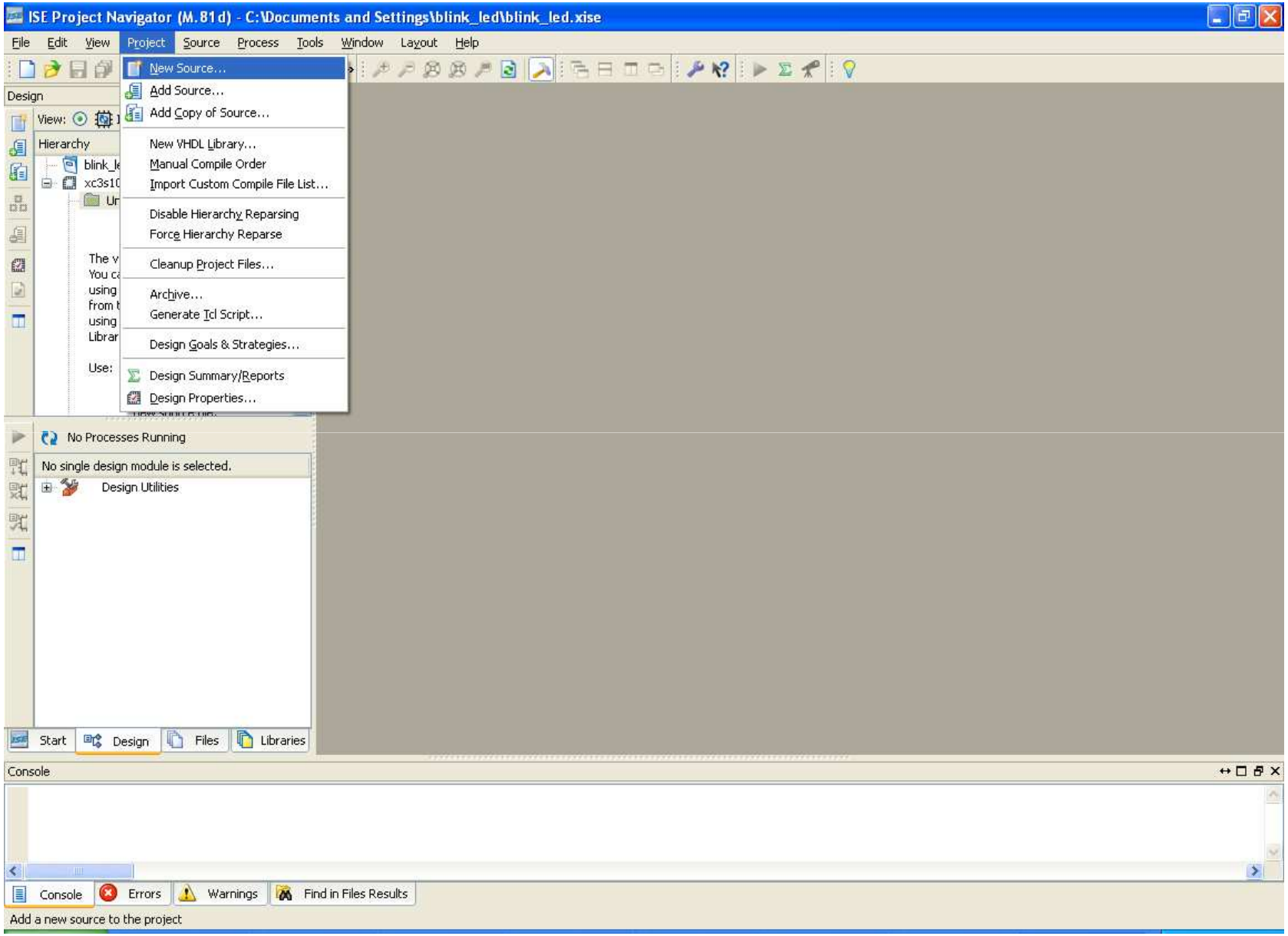
- Select these options then click next



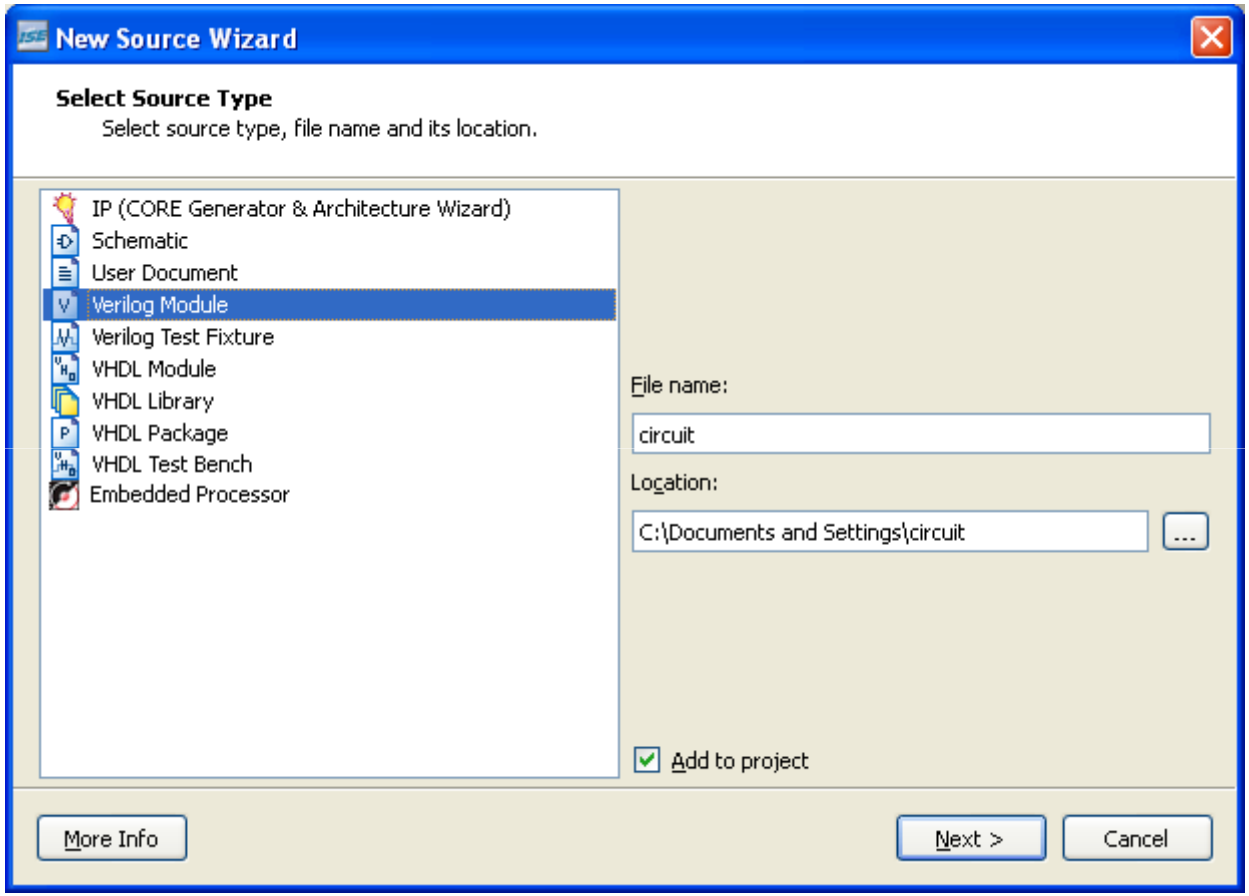
- click finish



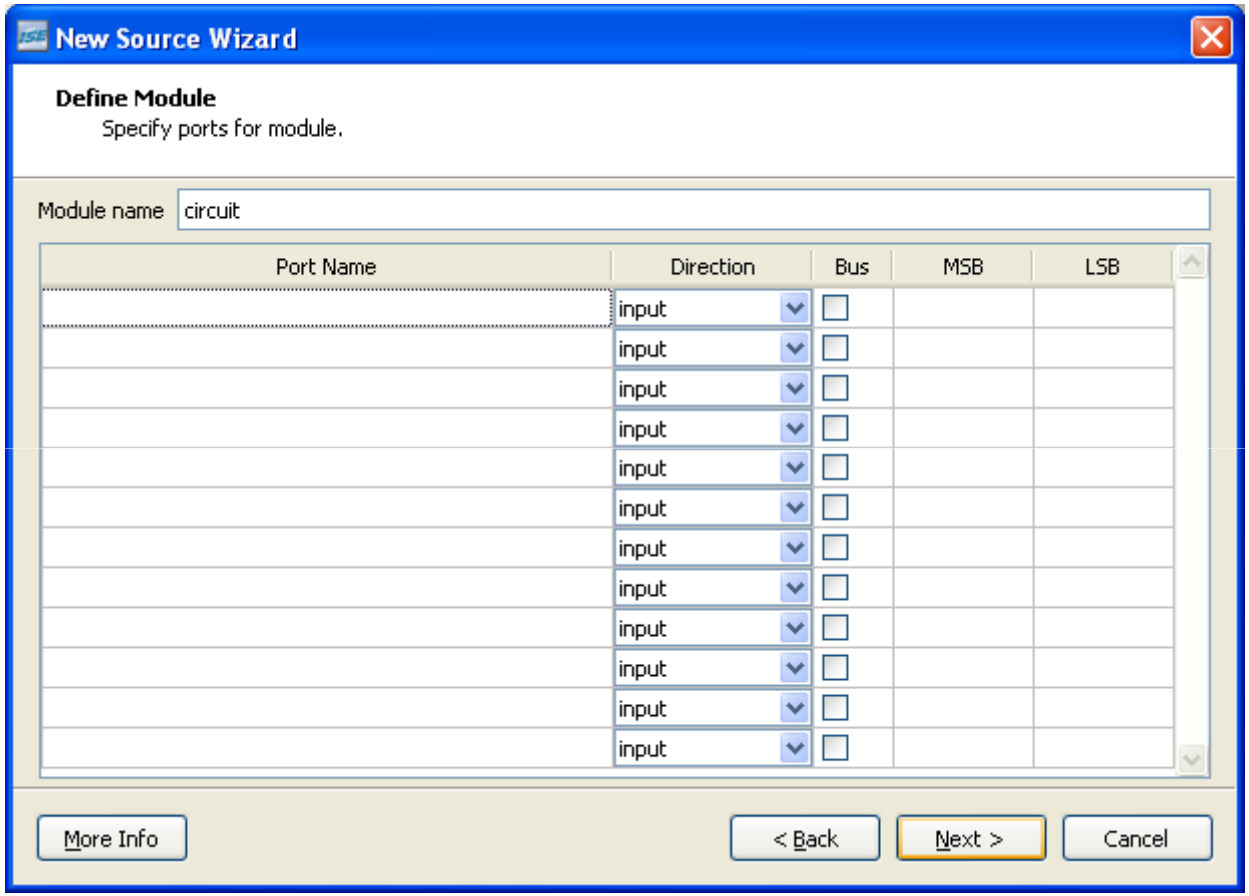
- Select Project →new source



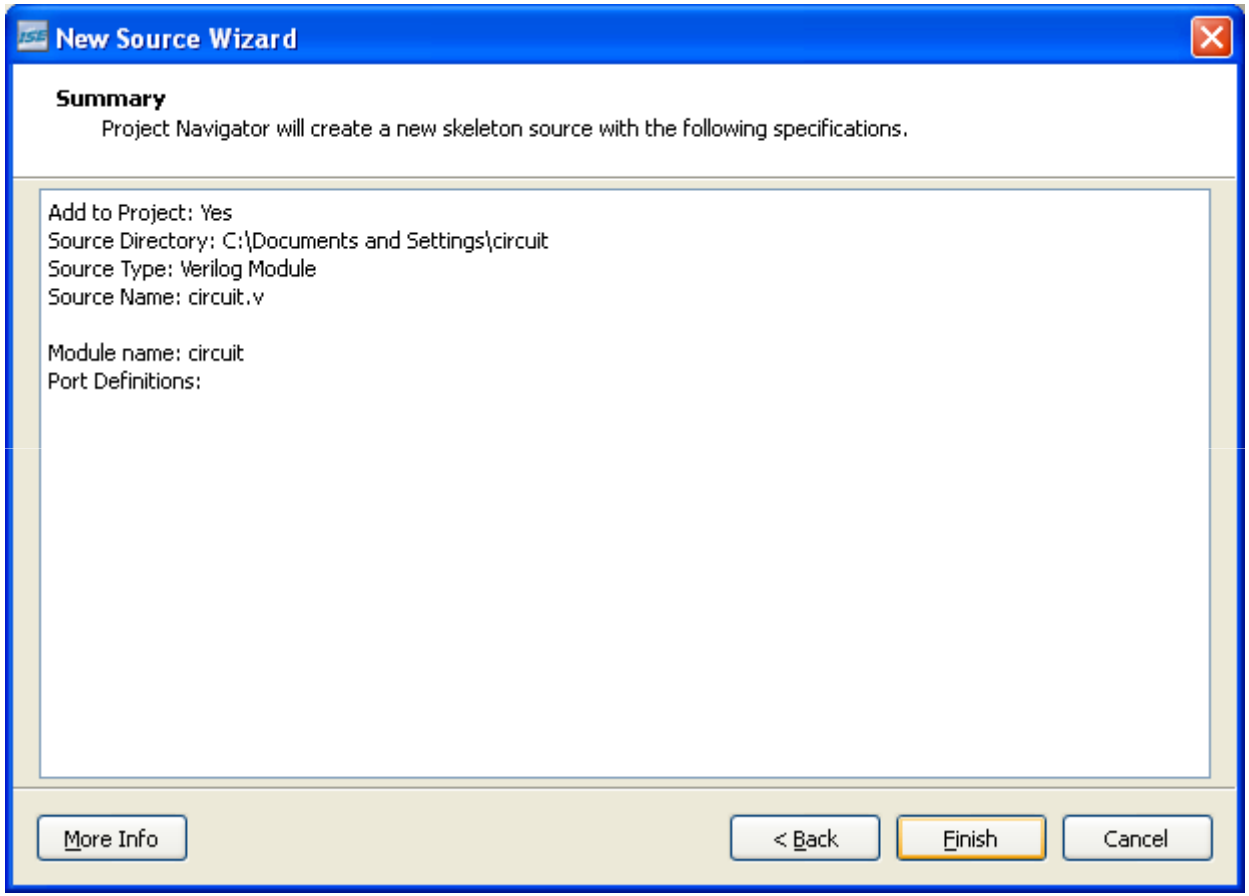
- Select Verilog module and write the file name, then click next



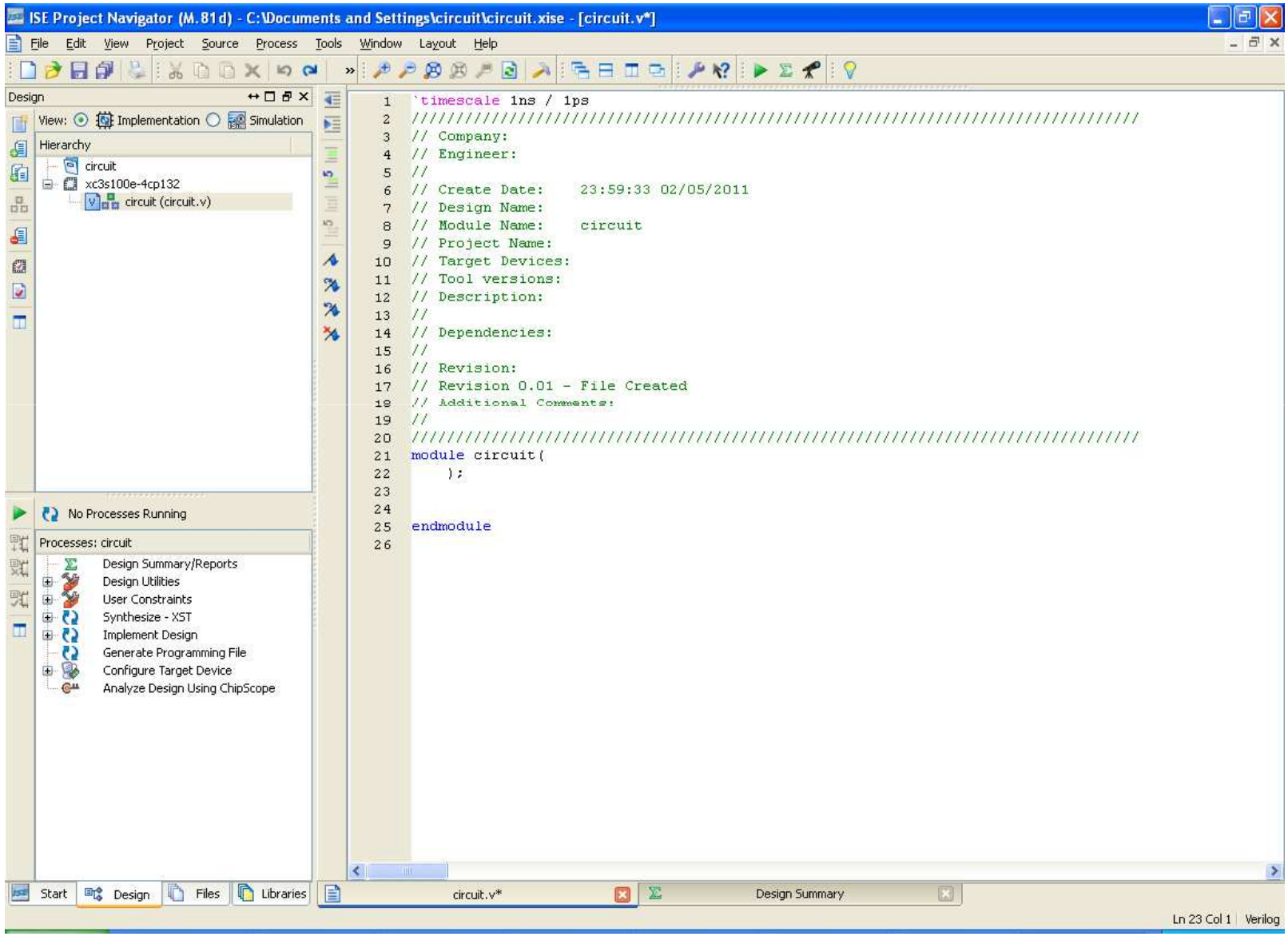
- click next


New Source Wizard
Define Module
Specify ports for module.
Module name
circuit
Port Name
Direction
Bus
MSB
LSB
input
More Info
< Back
Next >
Cancel

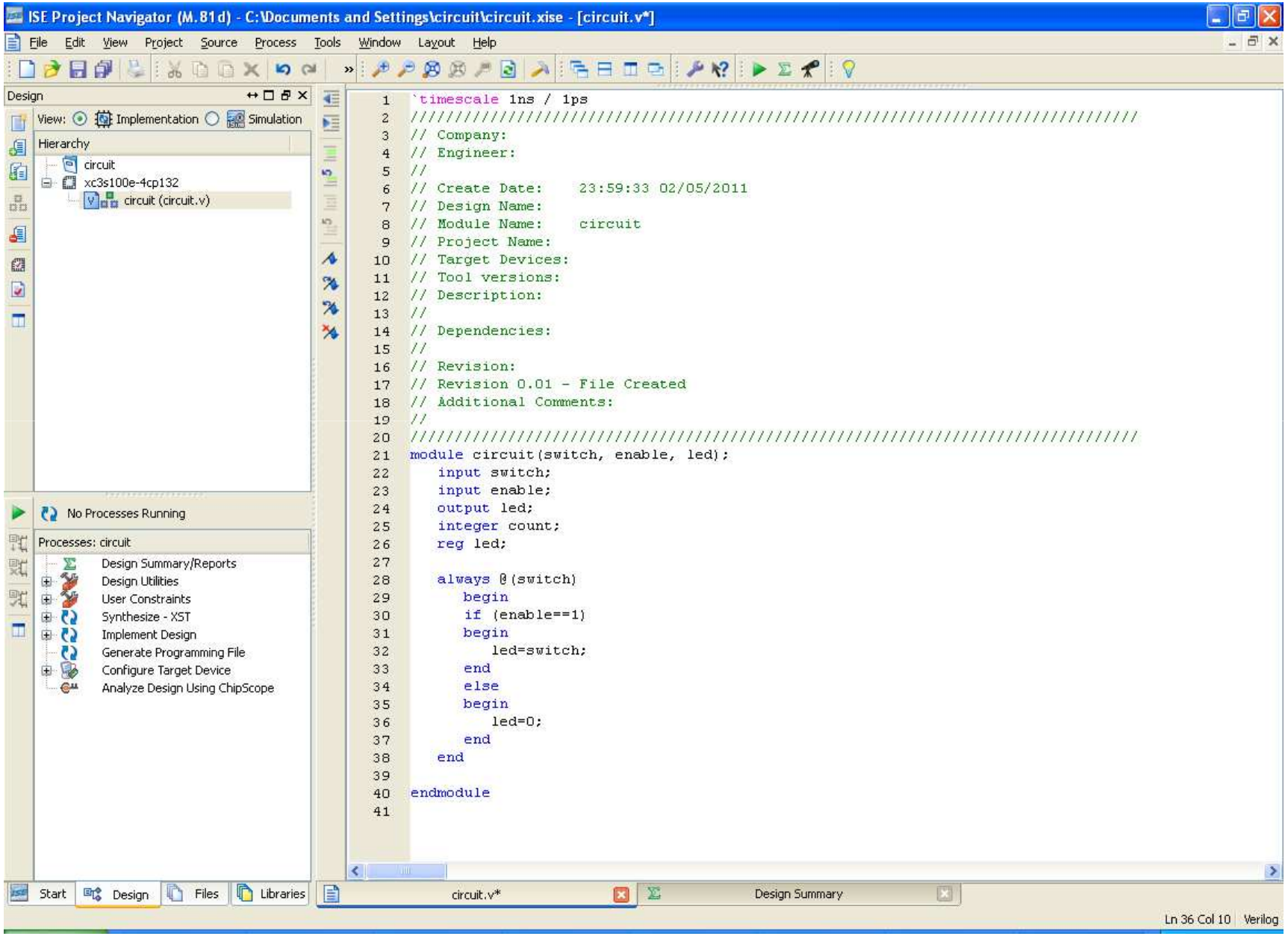
- Click Finish



- The file will be created and it will look like as below



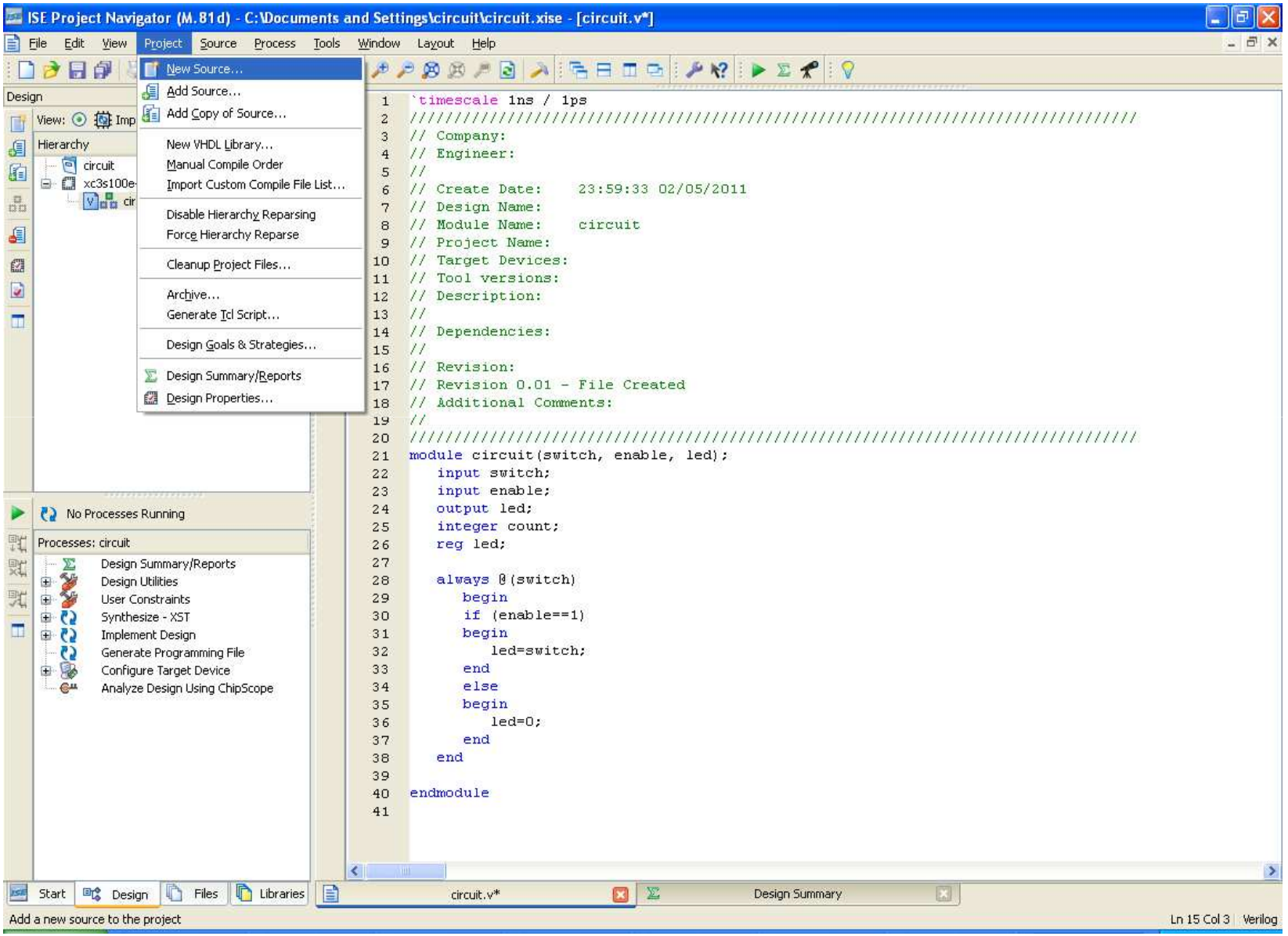
- Add your code shown



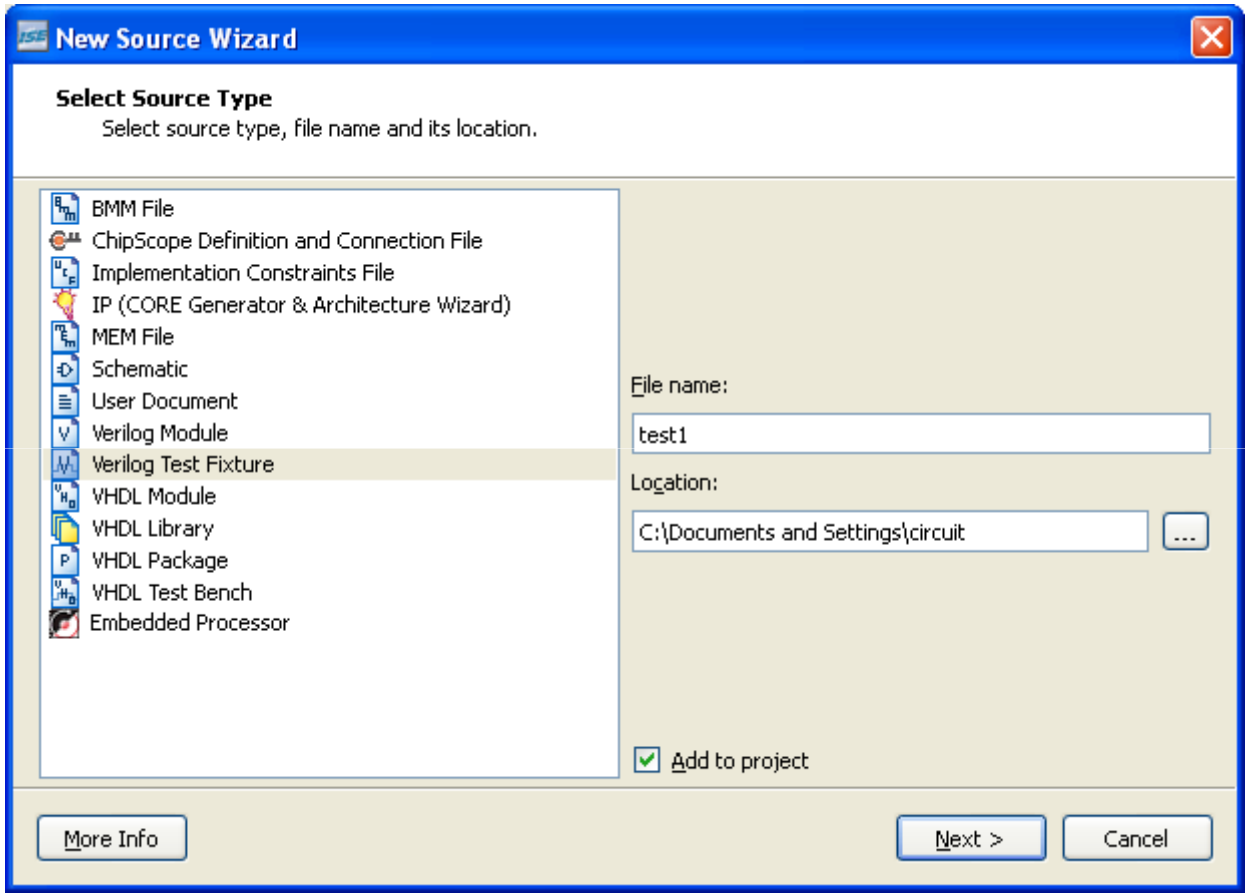
- You can copy the code from here and paste it in your project

```
module circuit(switch, enable, led);
        input switch;
        input enable;
        output led;
        integer count;
        reg led;

        always @(switch)
                begin
                if (enable==1)
                begin
                        led=switch;
                end
                else
                begin
                        led=0;
                end
        end

endmodule
```

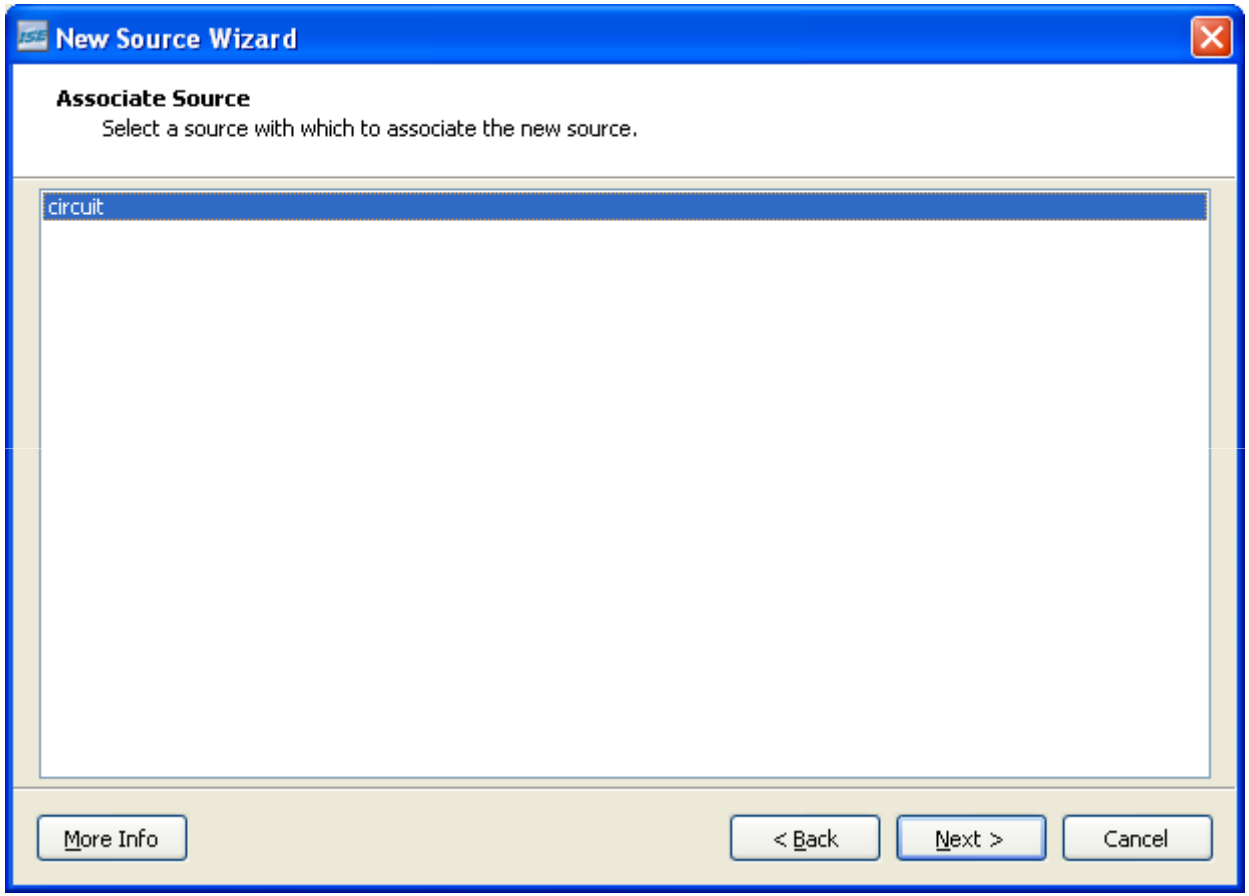
- To simulate the project select new source



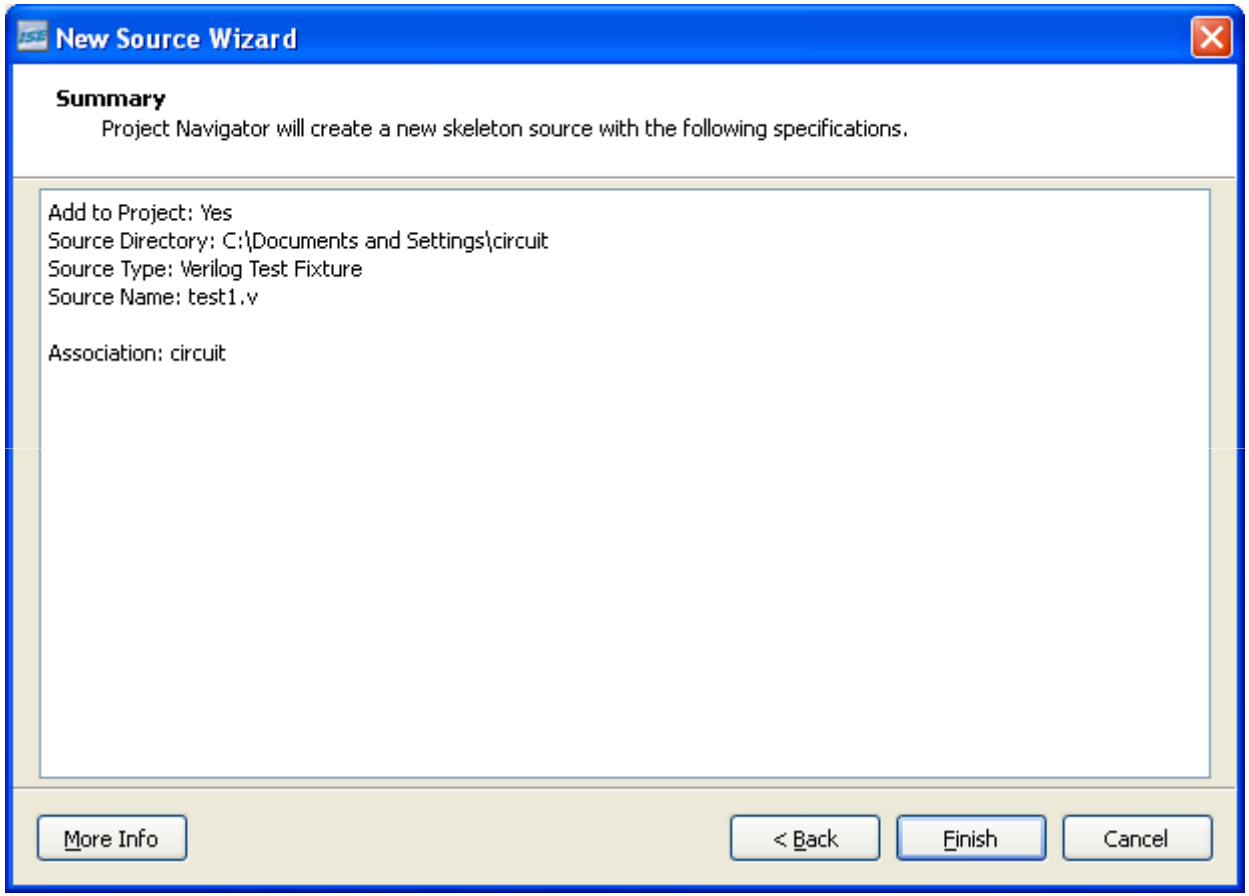
- Select Verilog Test Fixture and write the file name then click next



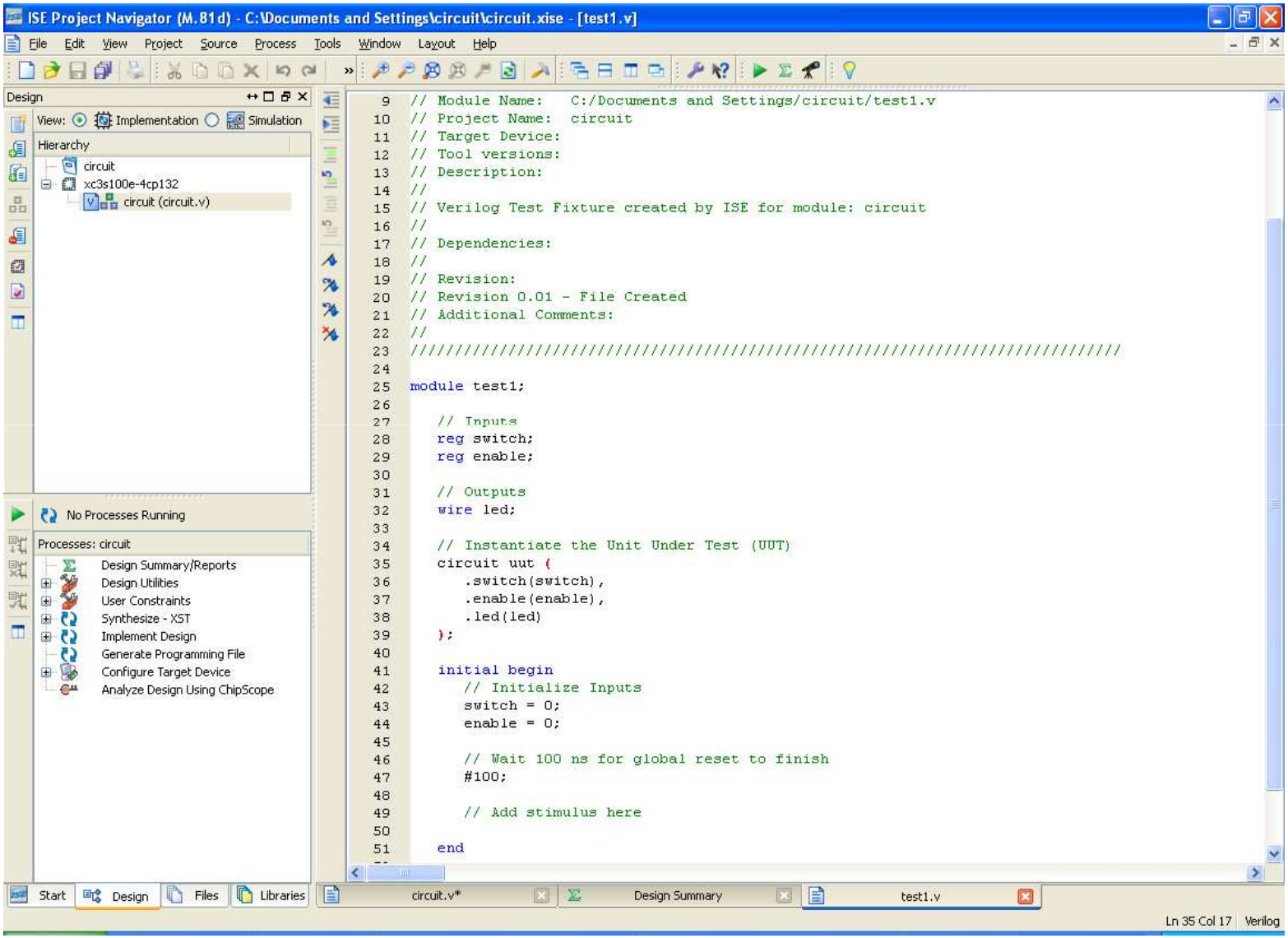
- Select the circuit module then click next



- click Finish to create the test file

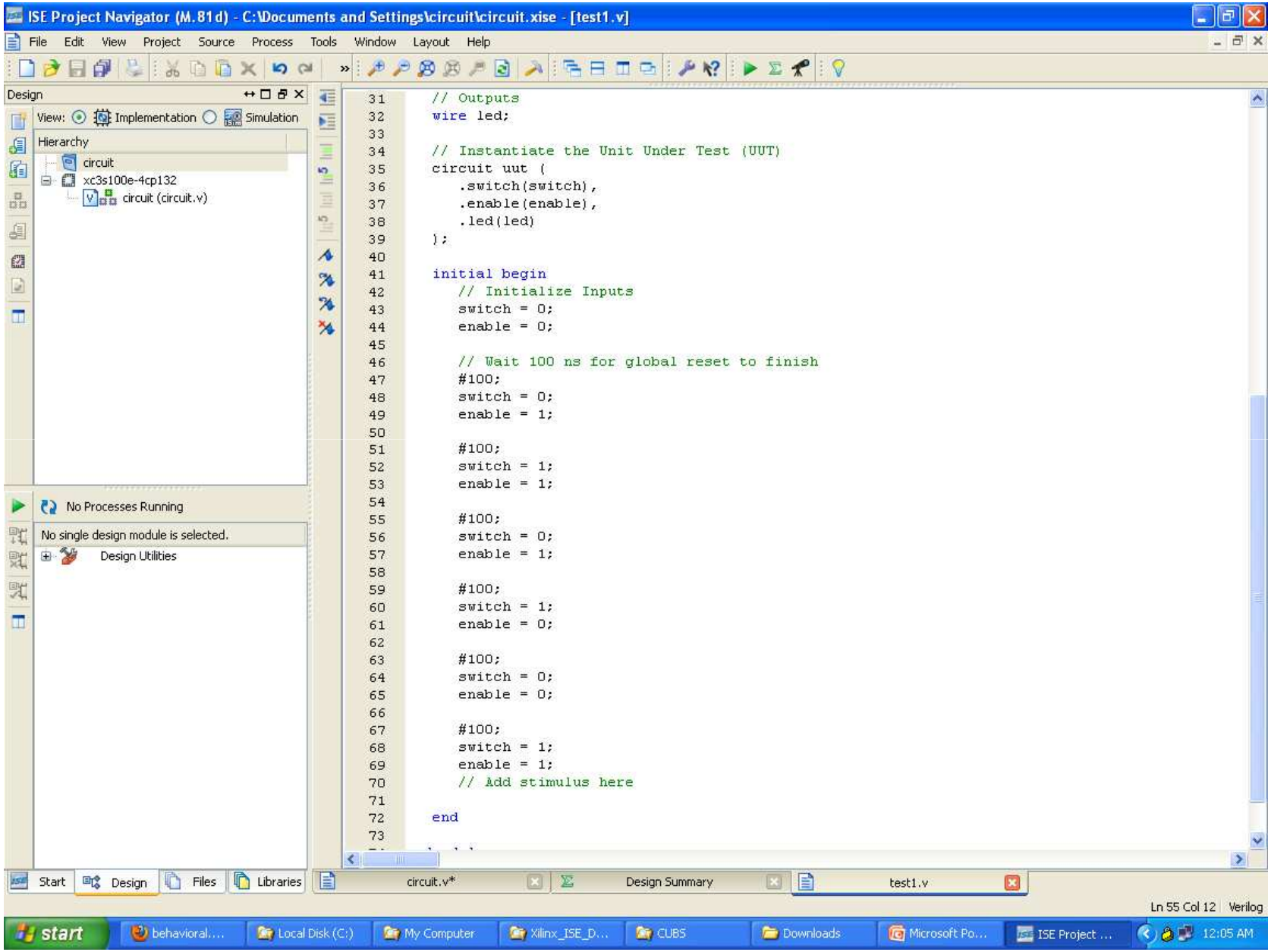


- This is how it will look like



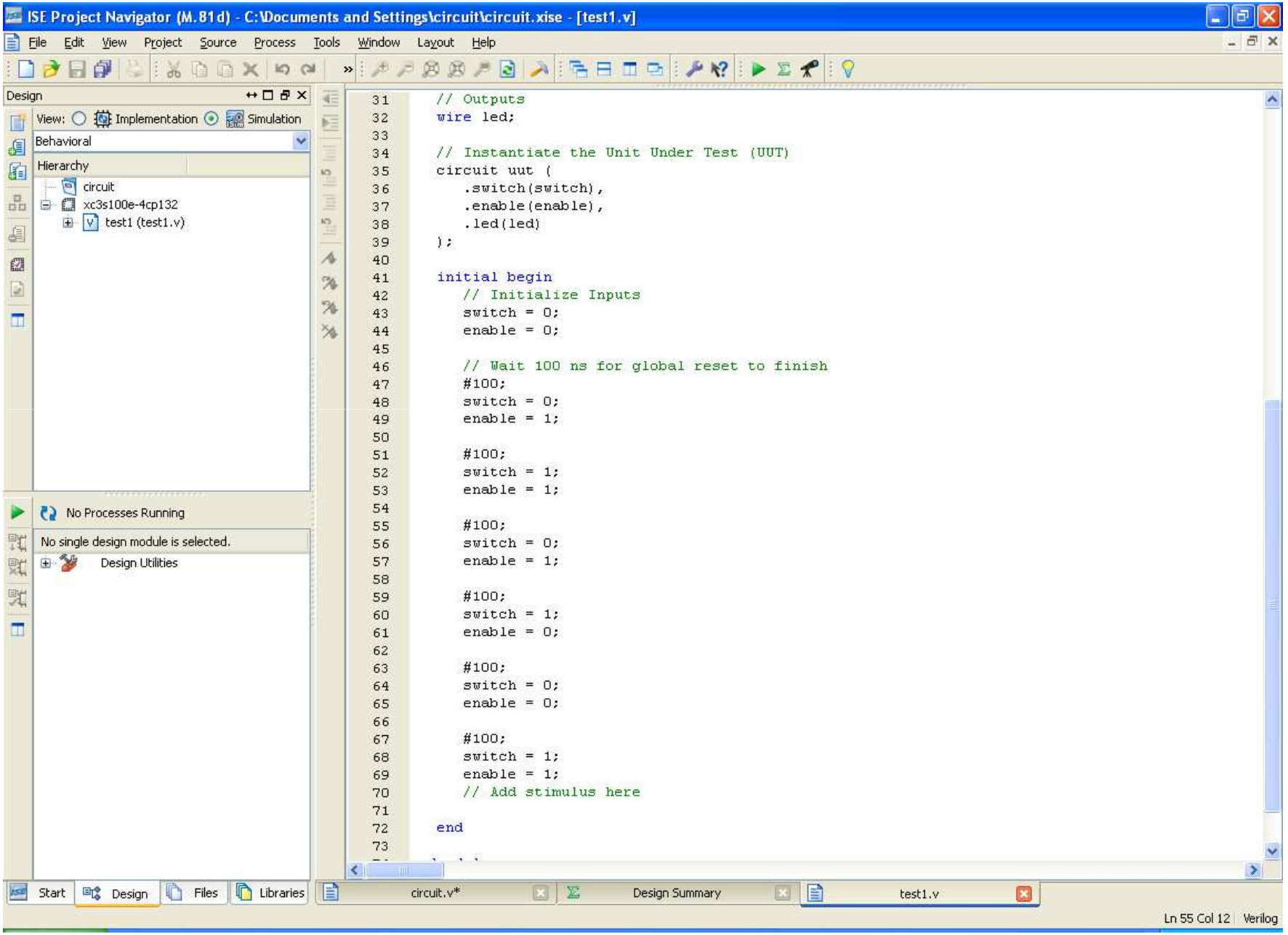

```
// Module Name:   C:/Documents and Settings/circuit/test1.v
// Project Name:  circuit
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: circuit
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////

module test1;

   // Inputs
   reg switch;
   reg enable;

   // Outputs
   wire led;

   // Instantiate the Unit Under Test (UUT)
   circuit uut (
      .switch(switch),
      .enable(enable),
      .led(led)
   );

   initial begin
      // Initialize Inputs
      switch = 0;
      enable = 0;

      // Wait 100 ns for global reset to finish
      #100;

      // Add stimulus here

   end
```

- Add your simulation code to the test file

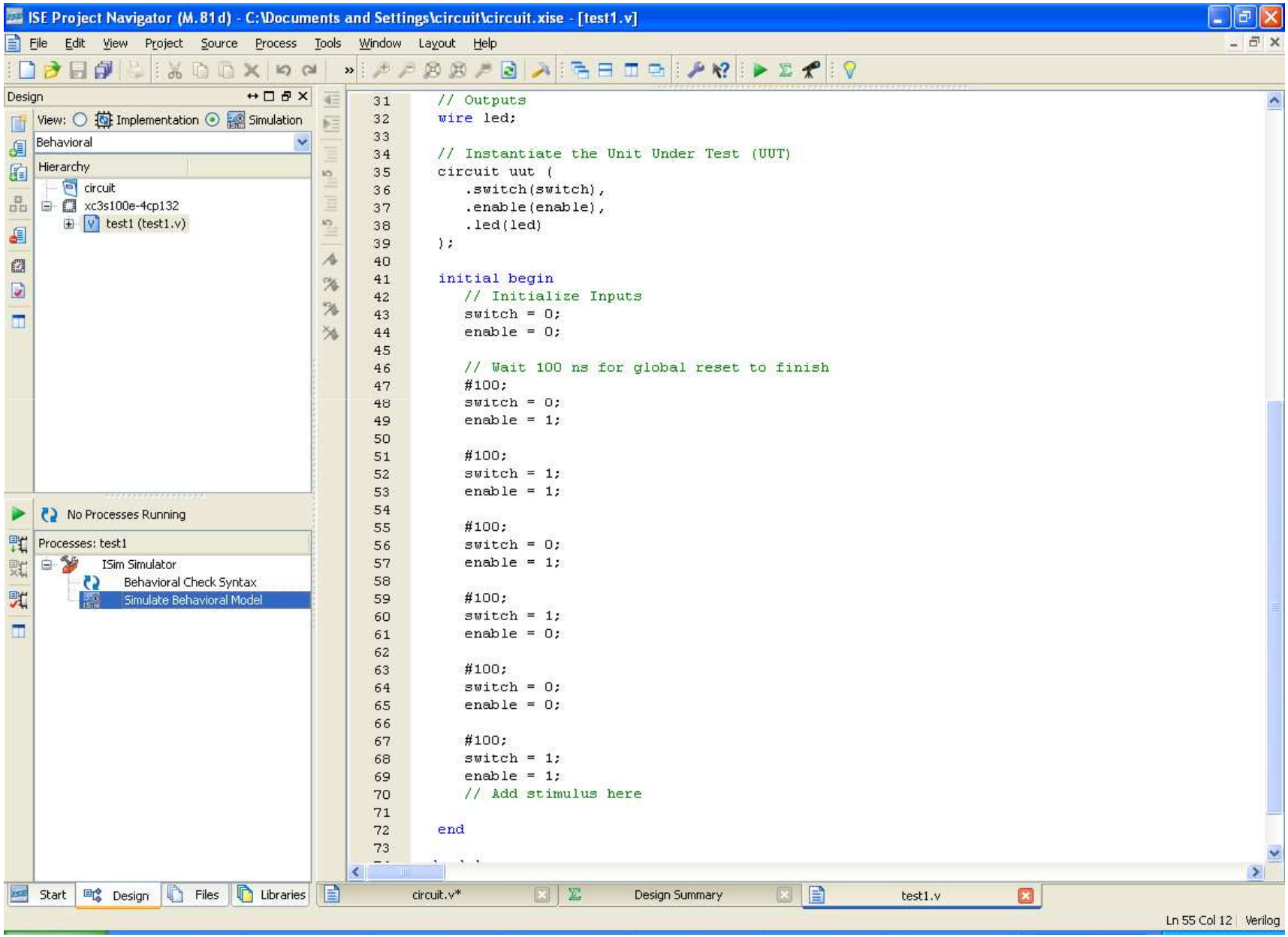



```
31    // Outputs
32    wire led;
33
34    // Instantiate the Unit Under Test (UUT)
35    circuit uut (
36       .switch(switch),
37       .enable(enable),
38       .led(led)
39    );
40
41    initial begin
42       // Initialize Inputs
43       switch = 0;
44       enable = 0;
45
46       // Wait 100 ns for global reset to finish
47       #100;
48       switch = 0;
49       enable = 1;
50
51       #100;
52       switch = 1;
53       enable = 1;
54
55       #100;
56       switch = 0;
57       enable = 1;
58
59       #100;
60       switch = 1;
61       enable = 0;
62
63       #100;
64       switch = 0;
65       enable = 0;
66
67       #100;
68       switch = 1;
69       enable = 1;
70       // Add stimulus here
71
72    end
73
```

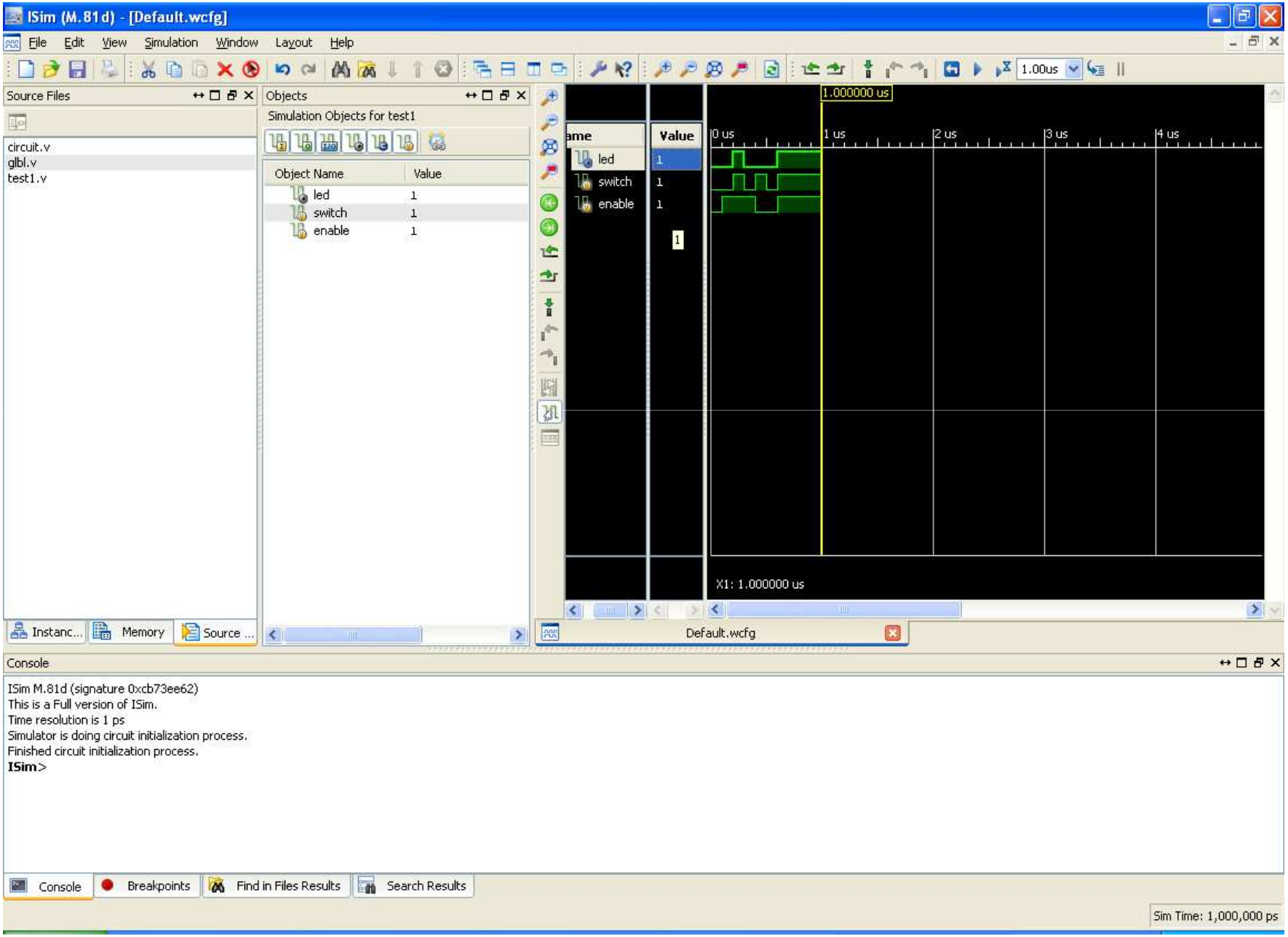
- Select the simulation option from Design tab



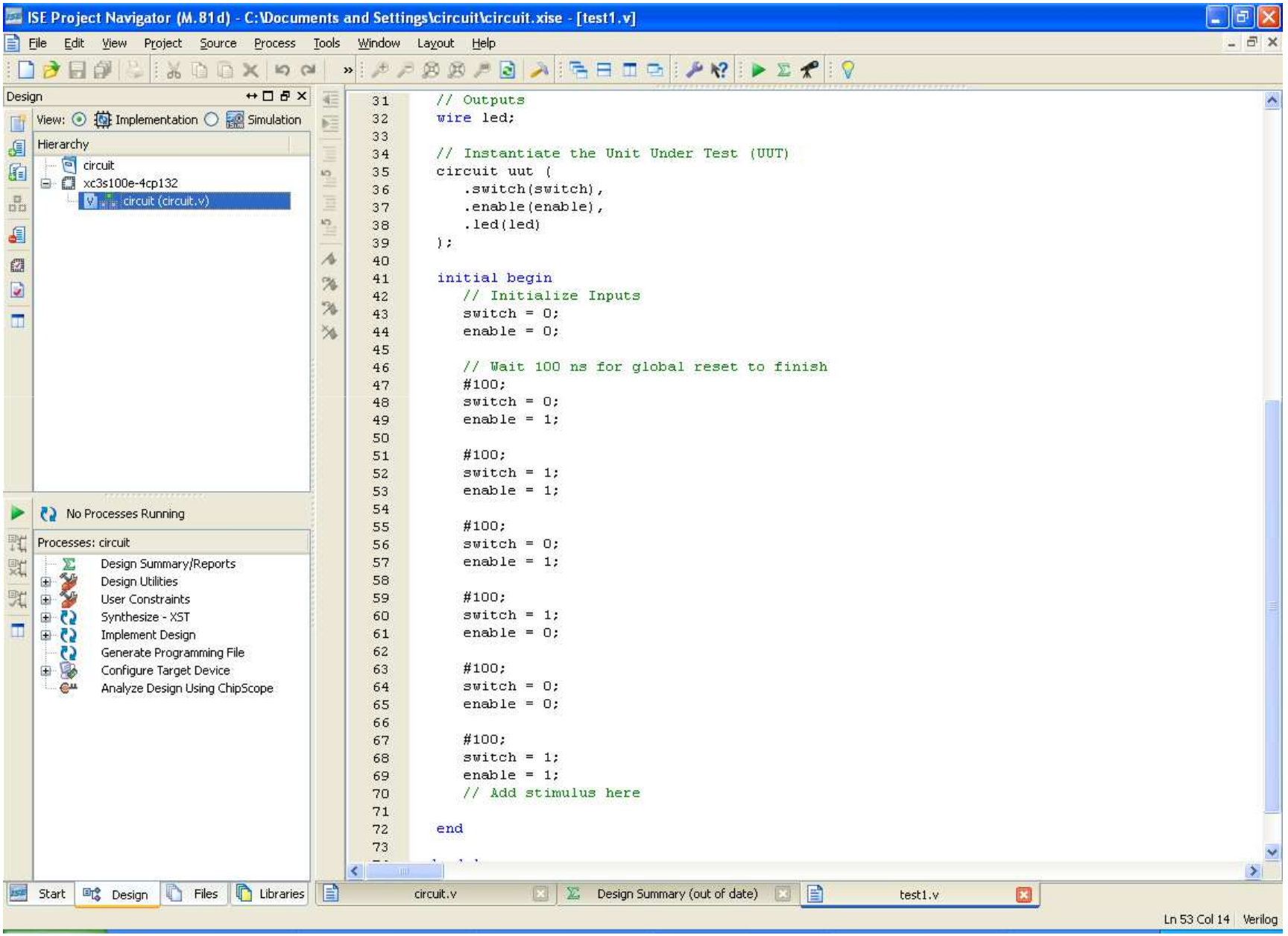
- Select test1.v file and D-Click Simulate Behavioral Model to open ISIM simulator



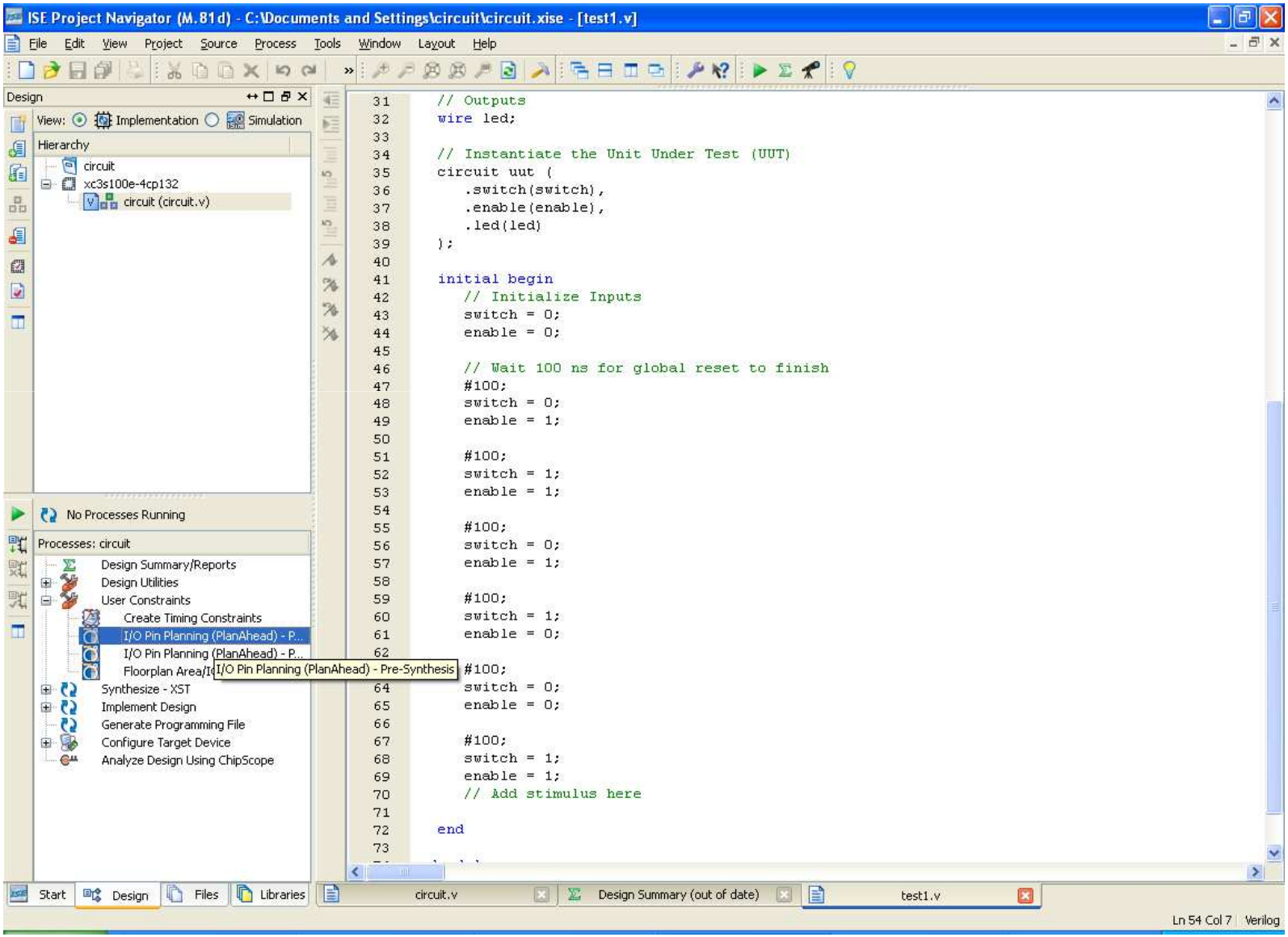
- ISIM application, use zoom in, out to view the results



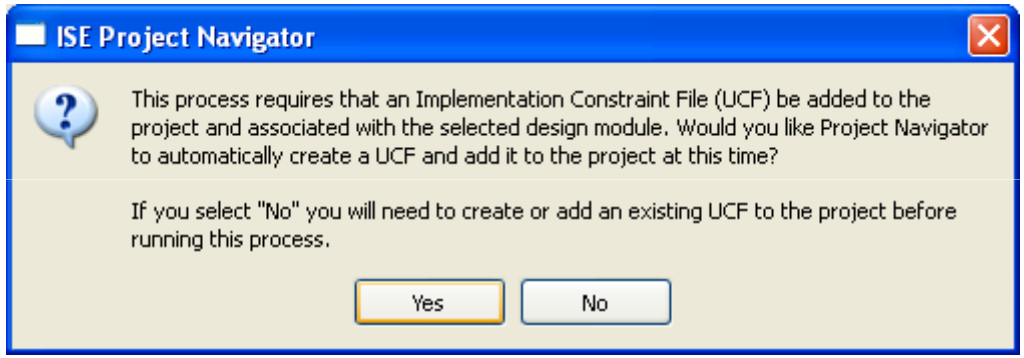
- To implement the code on the fpga board go to implementation mode



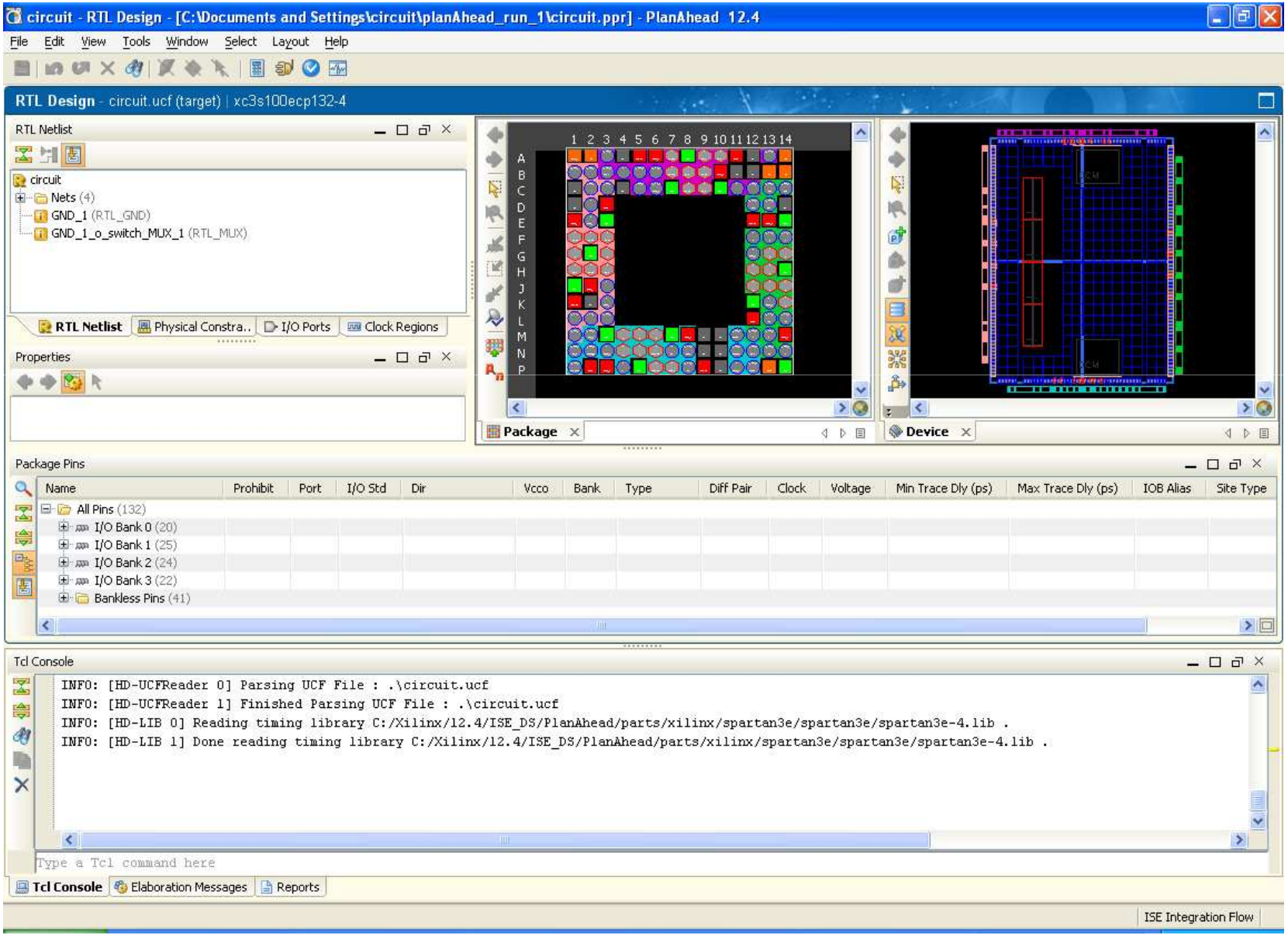
- Select circuit.v file and select I/O planning (PlanAhead-Presynthesis) to open PlanAhead application to assign the fpga I/O ports to the code I/O



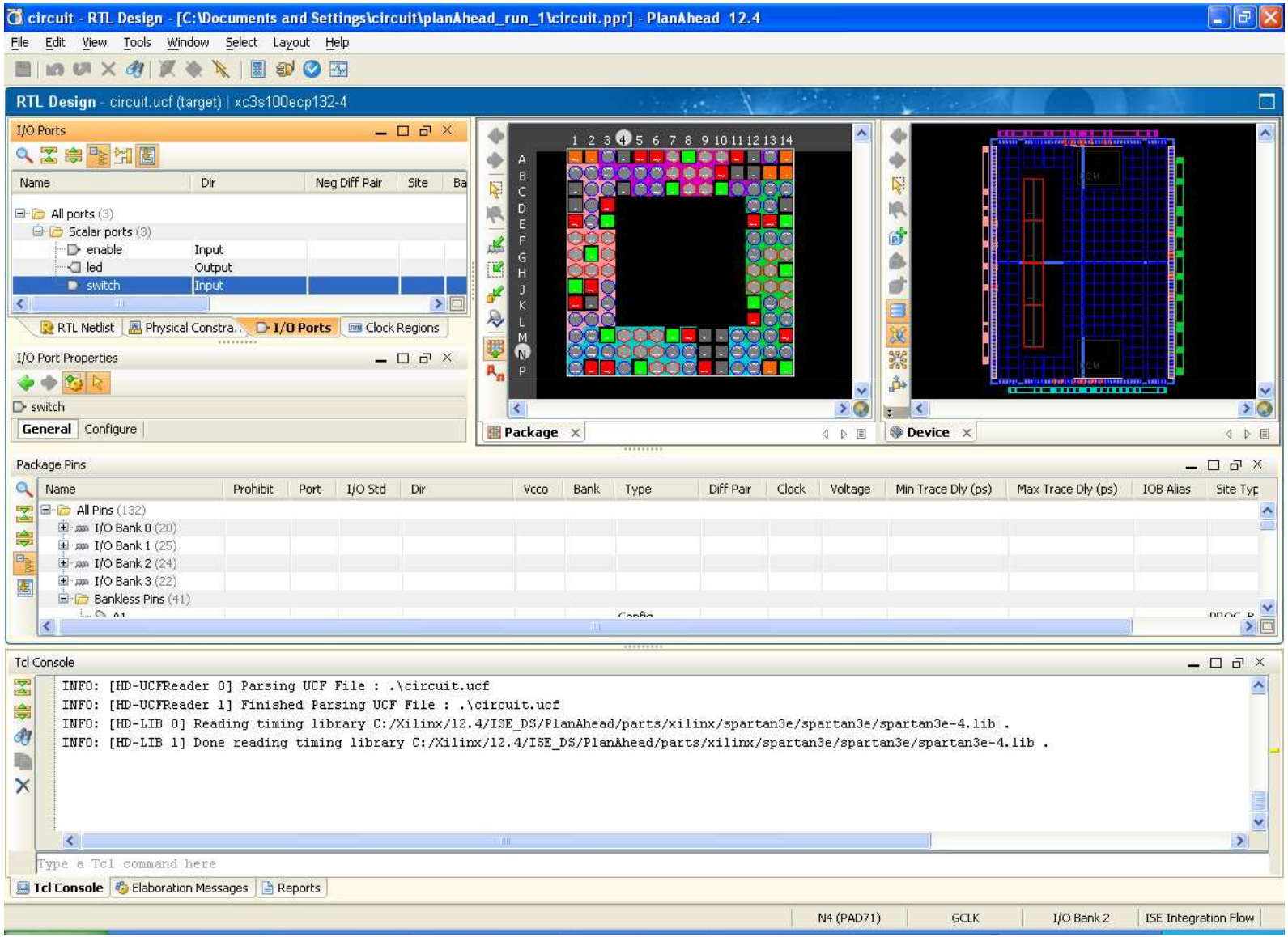
- Click yes to create UCF file and open Planahead application



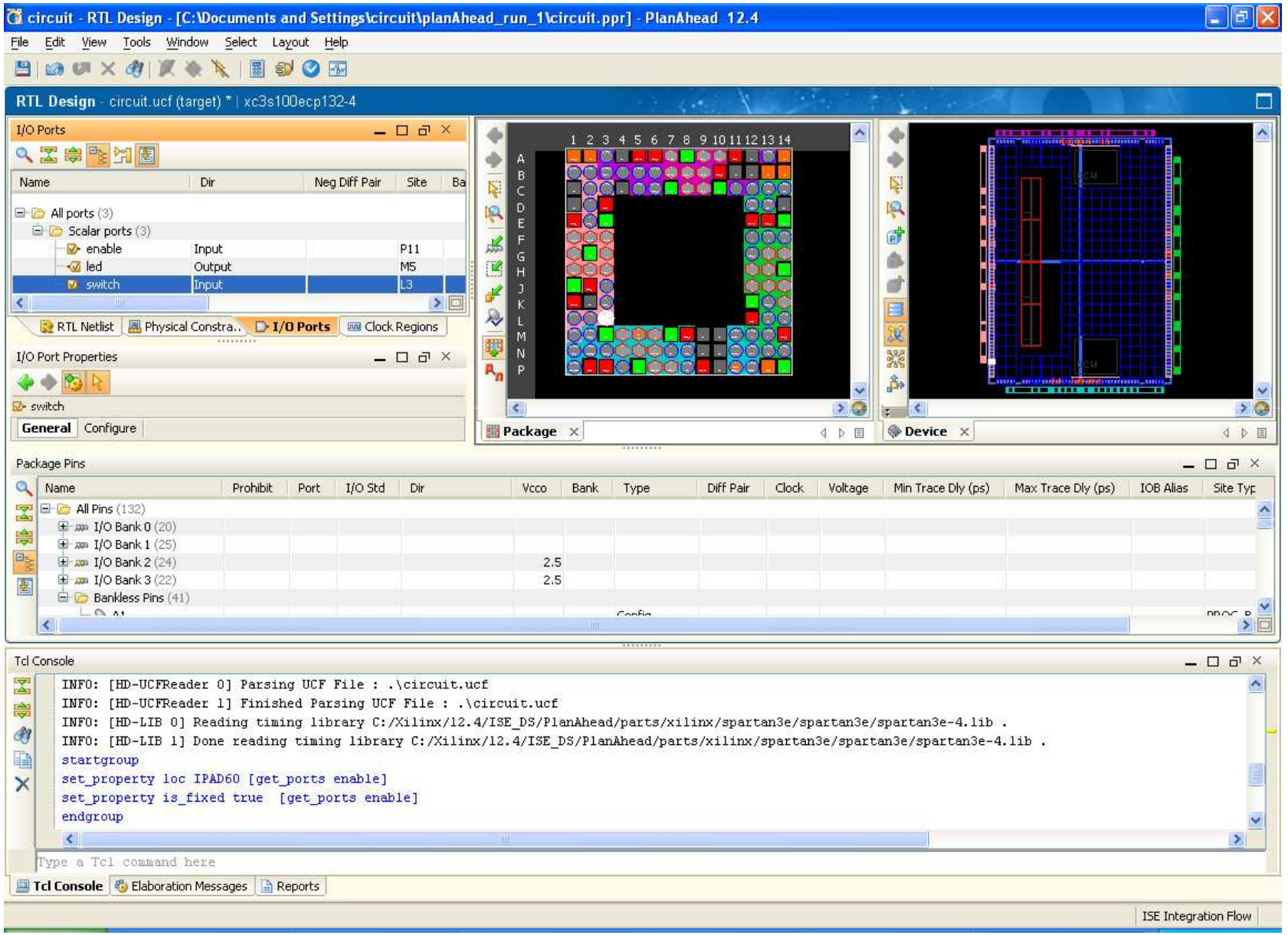
- Planahead Application after its opened



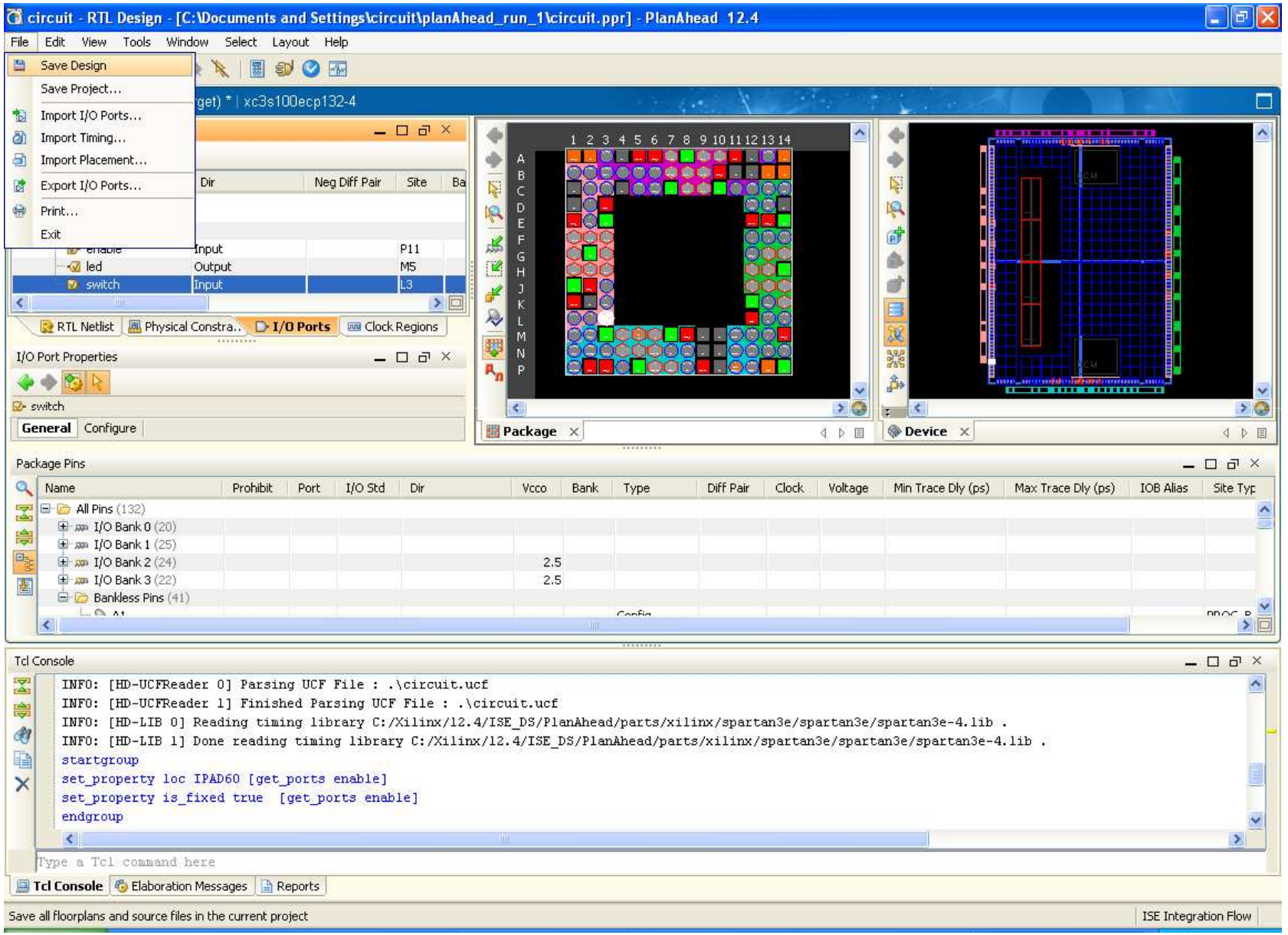
-Select I/O ports, drag and drop each of them to Pin in the Package figure, refer to http://www.digilentinc.com/Data/Products/BASYS2/Basys2_rm.pdf to find the pin definition



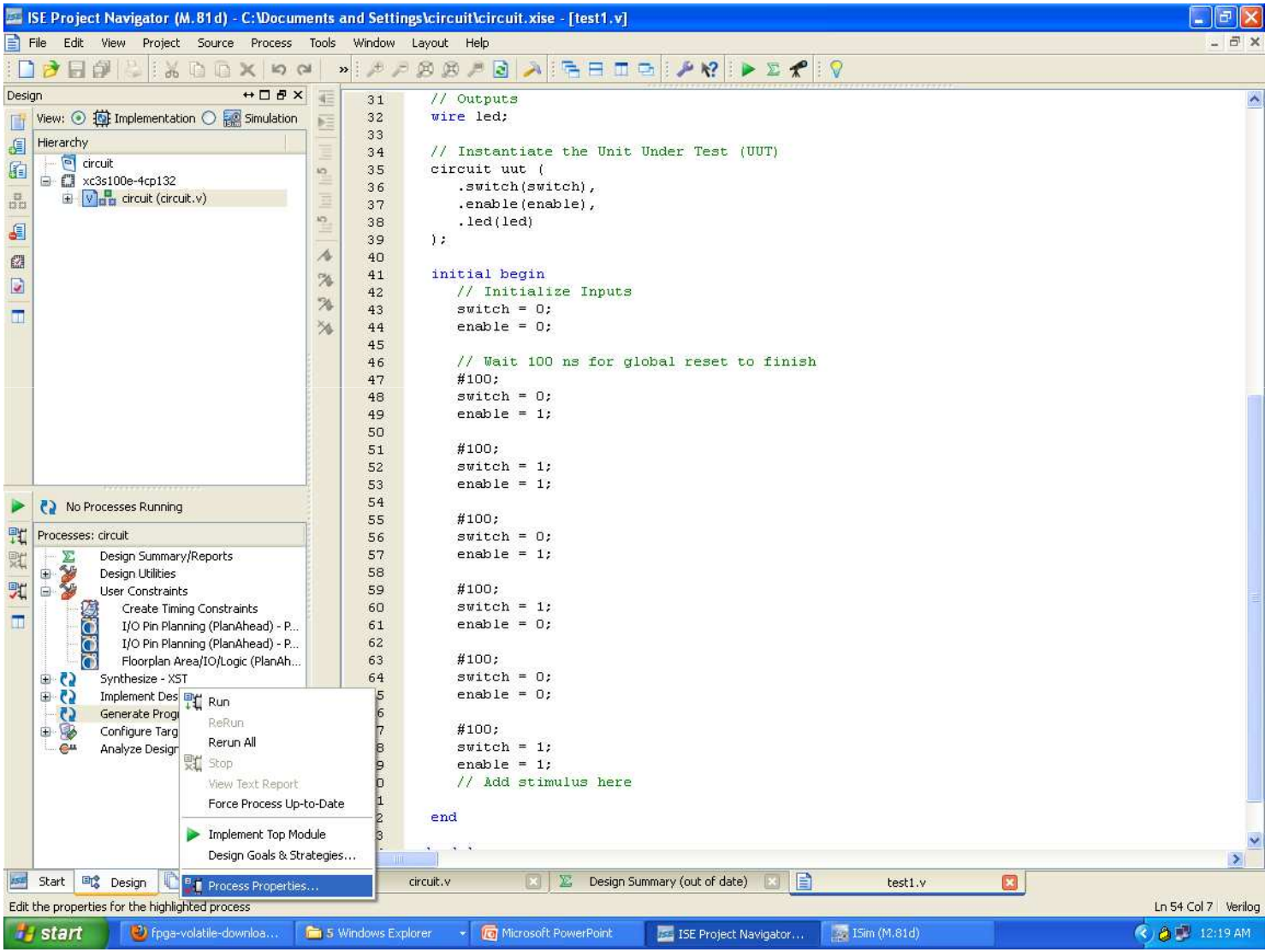
- After pin assignment



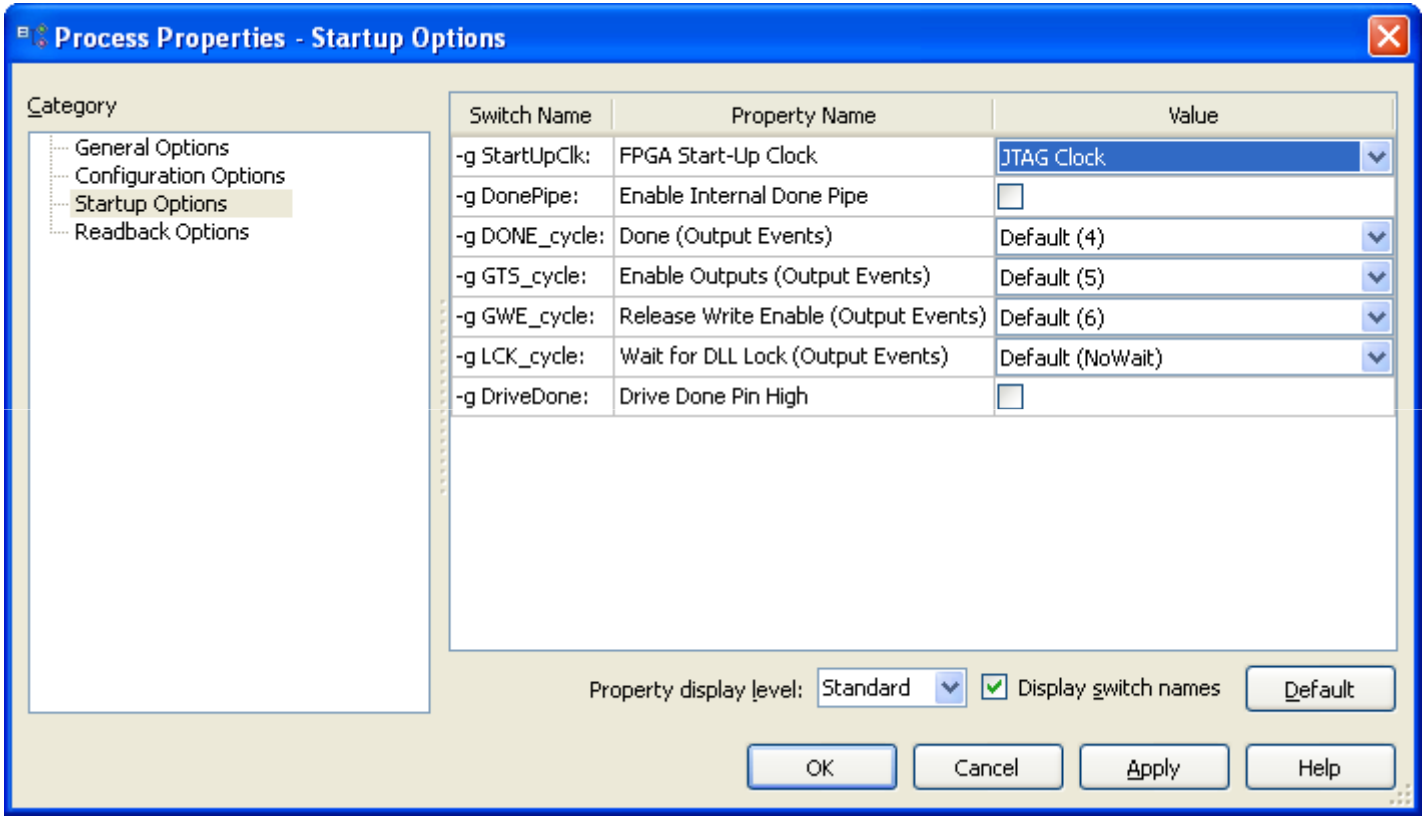
- Click save design and exit



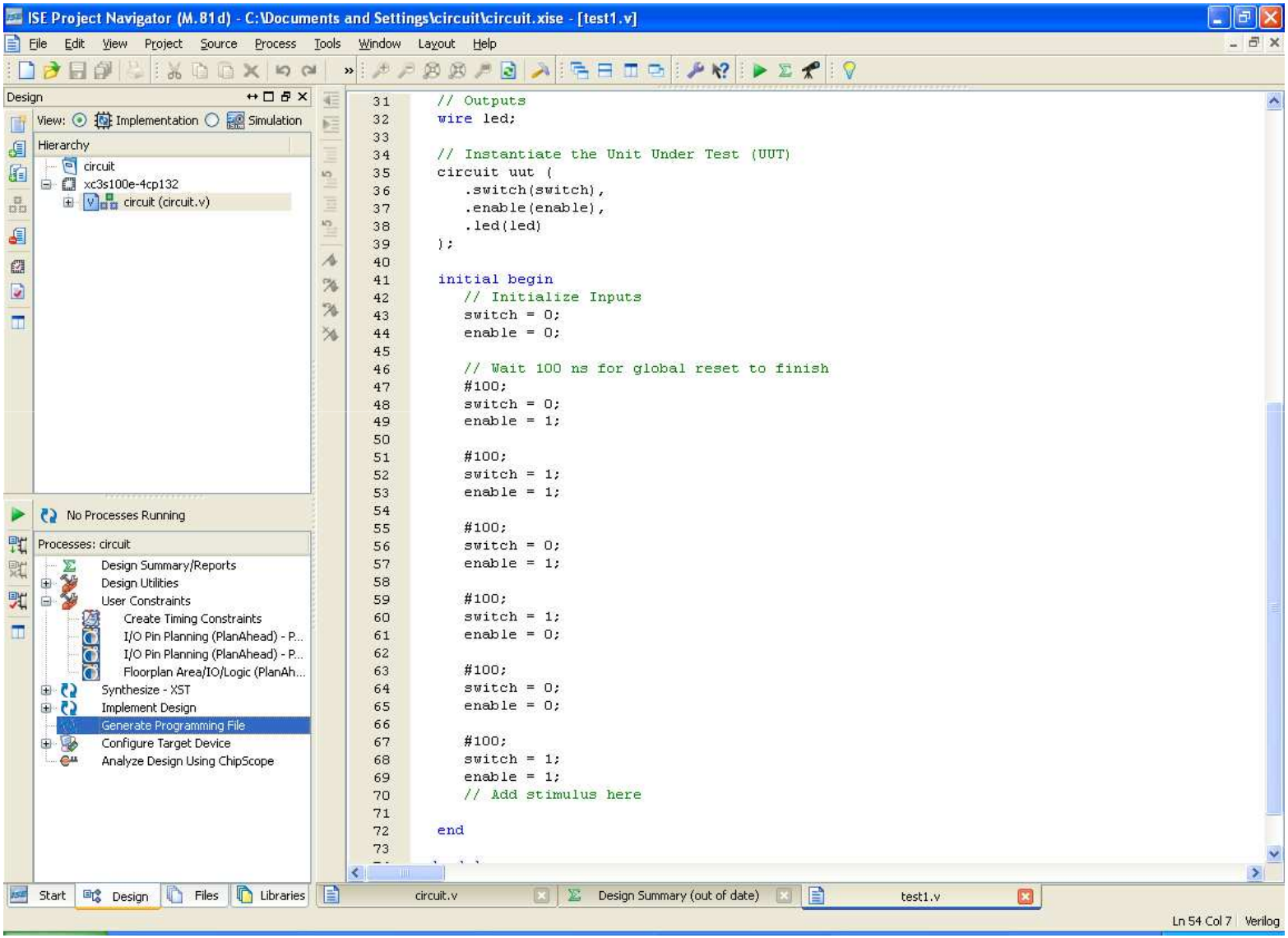
- To Prepare the .bit File to Program the FPGA,
- Set the FPGA Startup Clock to JTAG Clock, Right click on Generate Programming File in the Processes Window, select *Process Properties*



Change *Startup Options to JTAG clock* as shown, then press OK

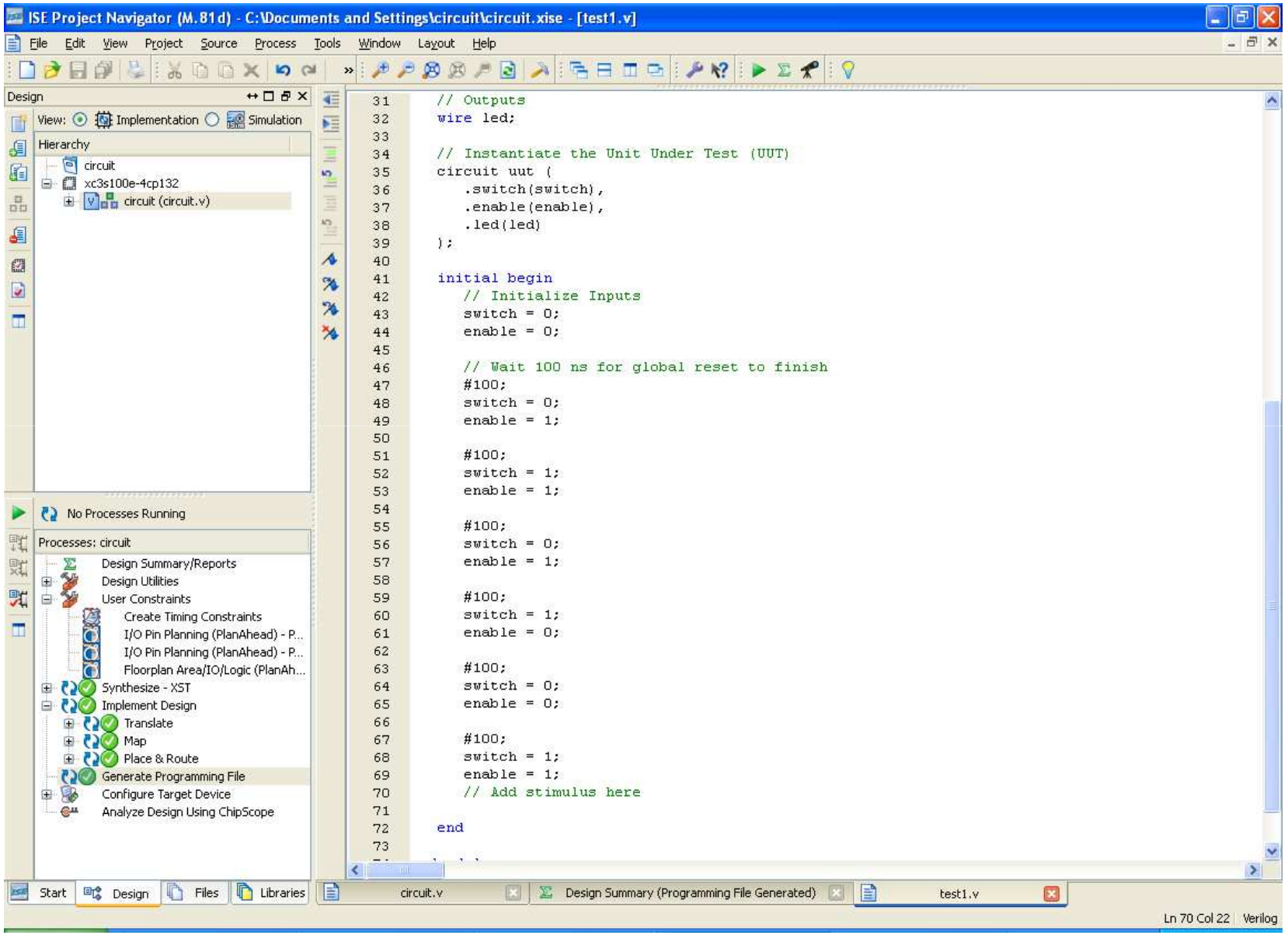


Create the *.bit file,* Double click on *Generate Programming File in the Processes Window*

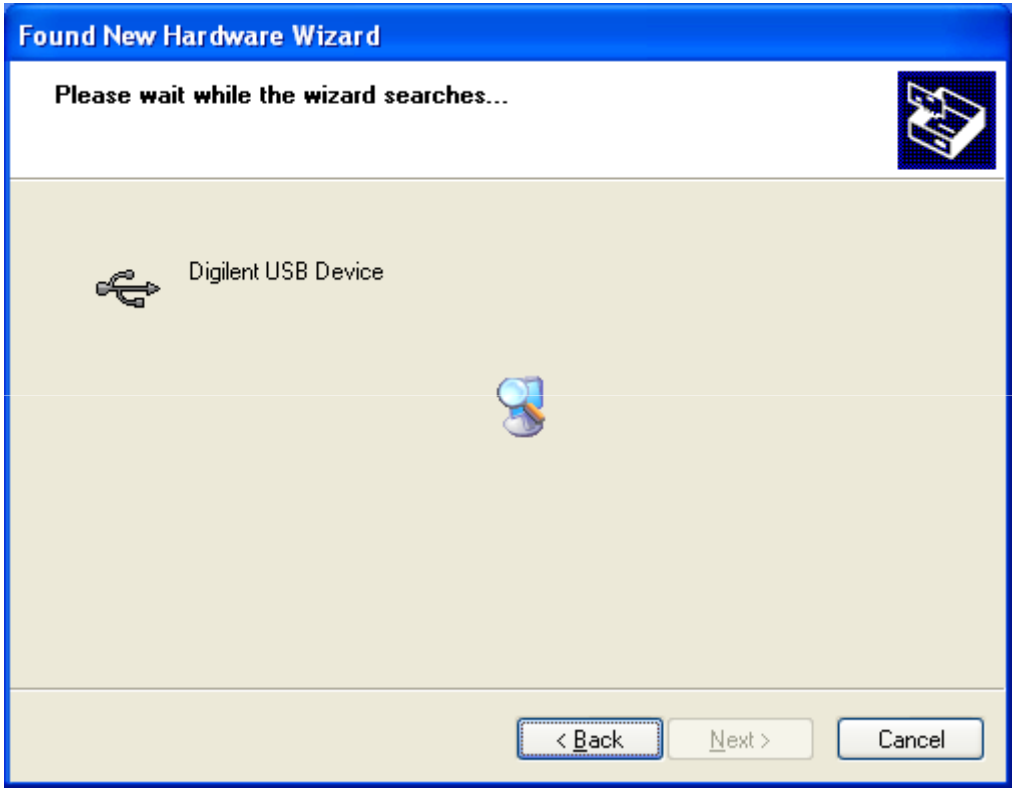



```
31    // Outputs
32    wire led;
33
34    // Instantiate the Unit Under Test (UUT)
35    circuit uut (
36       .switch(switch),
37       .enable(enable),
38       .led(led)
39    );
40
41    initial begin
42       // Initialize Inputs
43       switch = 0;
44       enable = 0;
45
46       // Wait 100 ns for global reset to finish
47       #100;
48       switch = 0;
49       enable = 1;
50
51       #100;
52       switch = 1;
53       enable = 1;
54
55       #100;
56       switch = 0;
57       enable = 1;
58
59       #100;
60       switch = 1;
61       enable = 0;
62
63       #100;
64       switch = 0;
65       enable = 0;
66
67       #100;
68       switch = 1;
69       enable = 1;
70       // Add stimulus here
71
72    end
73
```

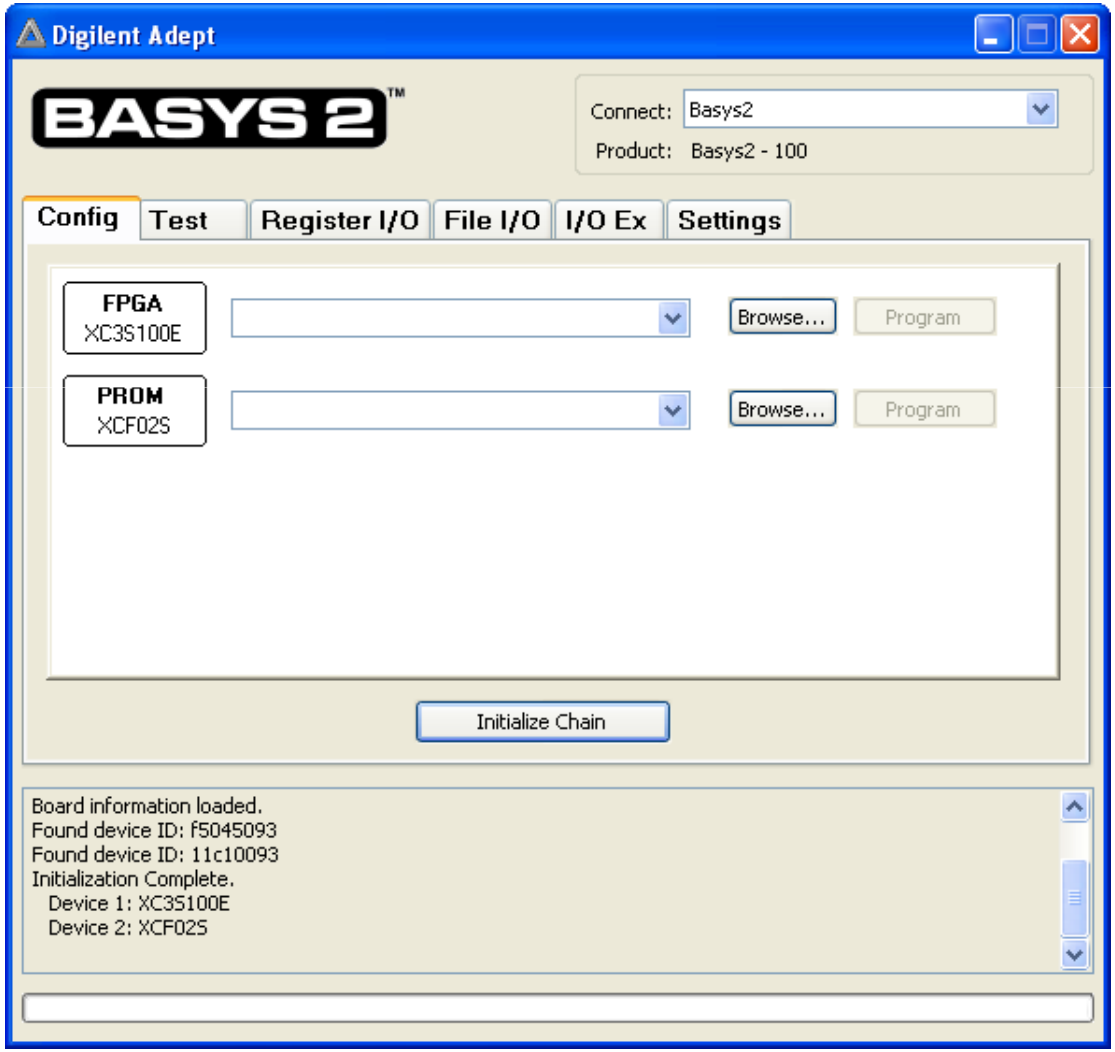
Make sure that every thing is compiling and building



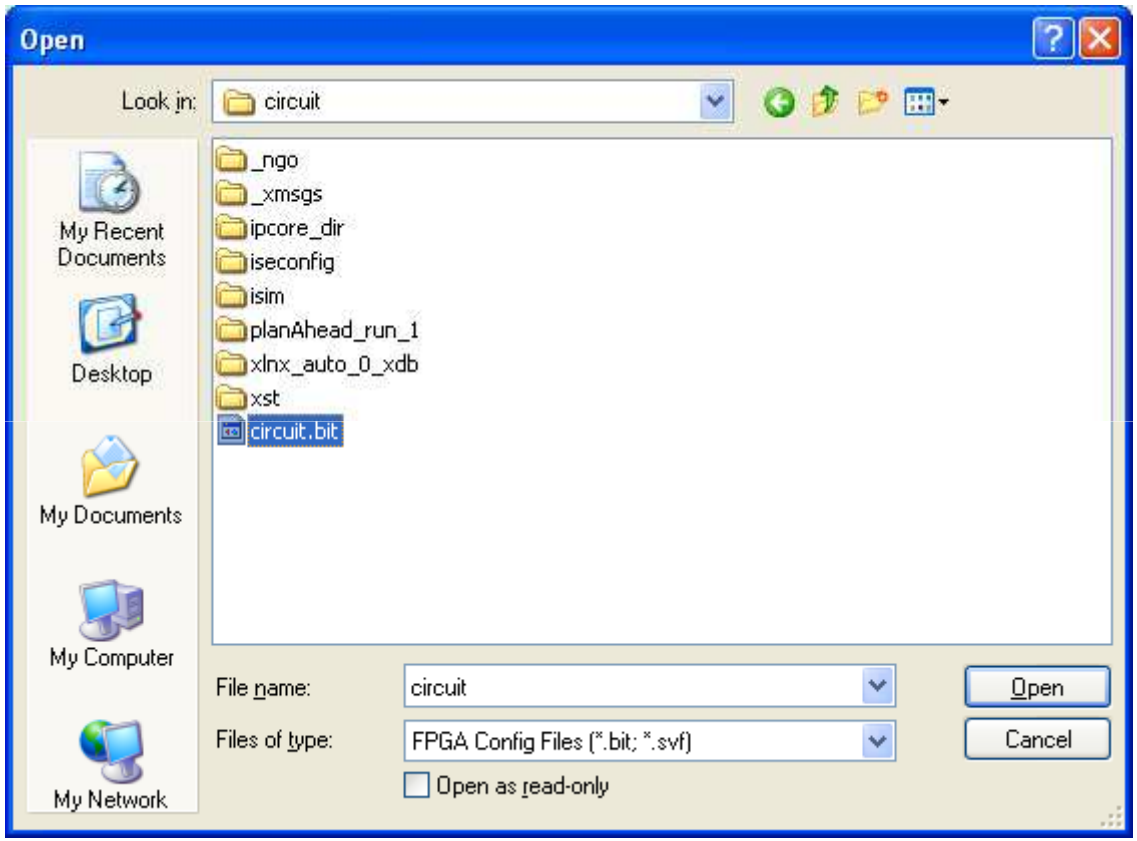
Connect Basys2 to your computer and make sure its correctly configured

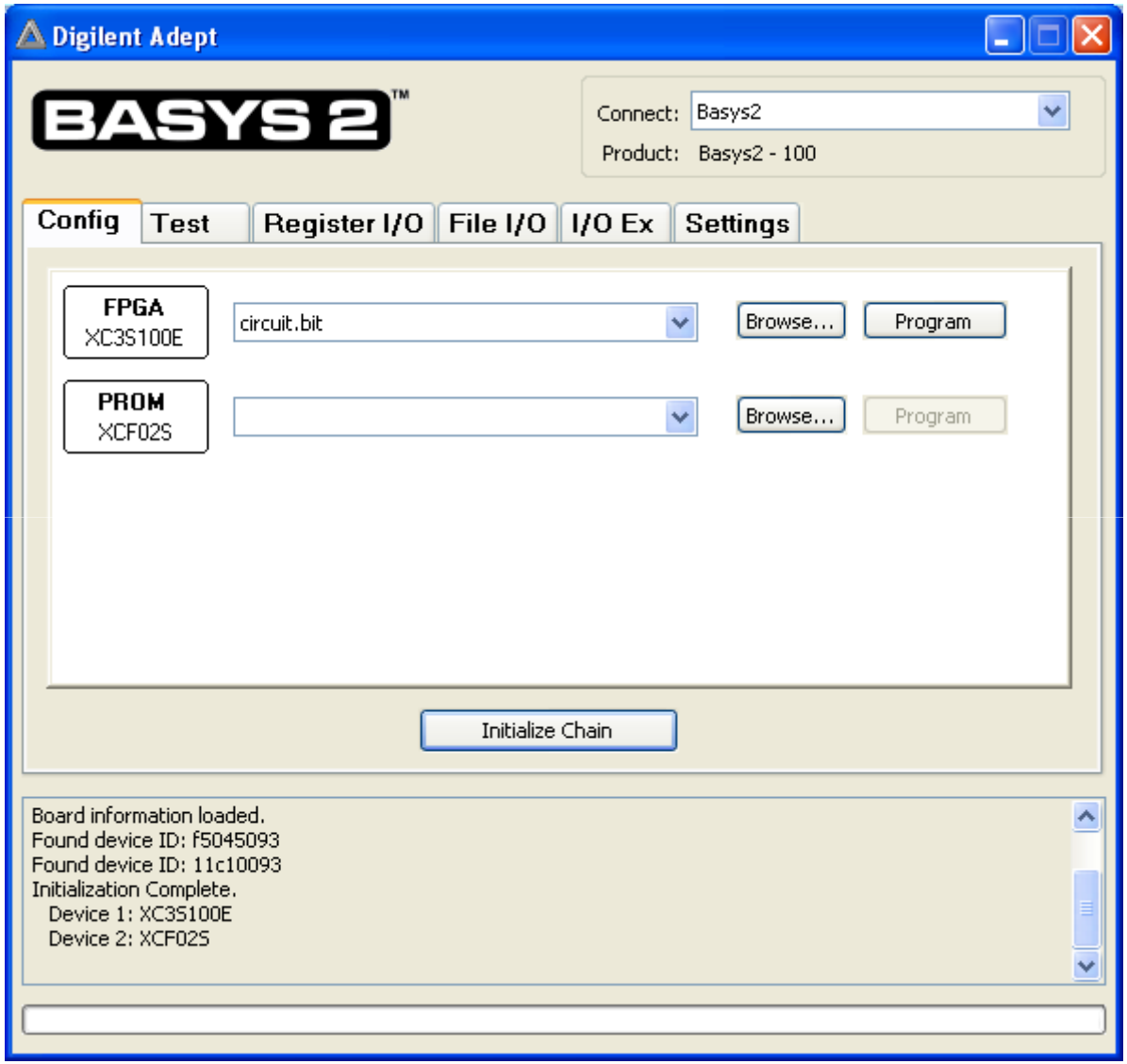


Open *Adept2.1 (Downloaded from Digilent's Website)*
http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,66,69&Prod=ADEPT

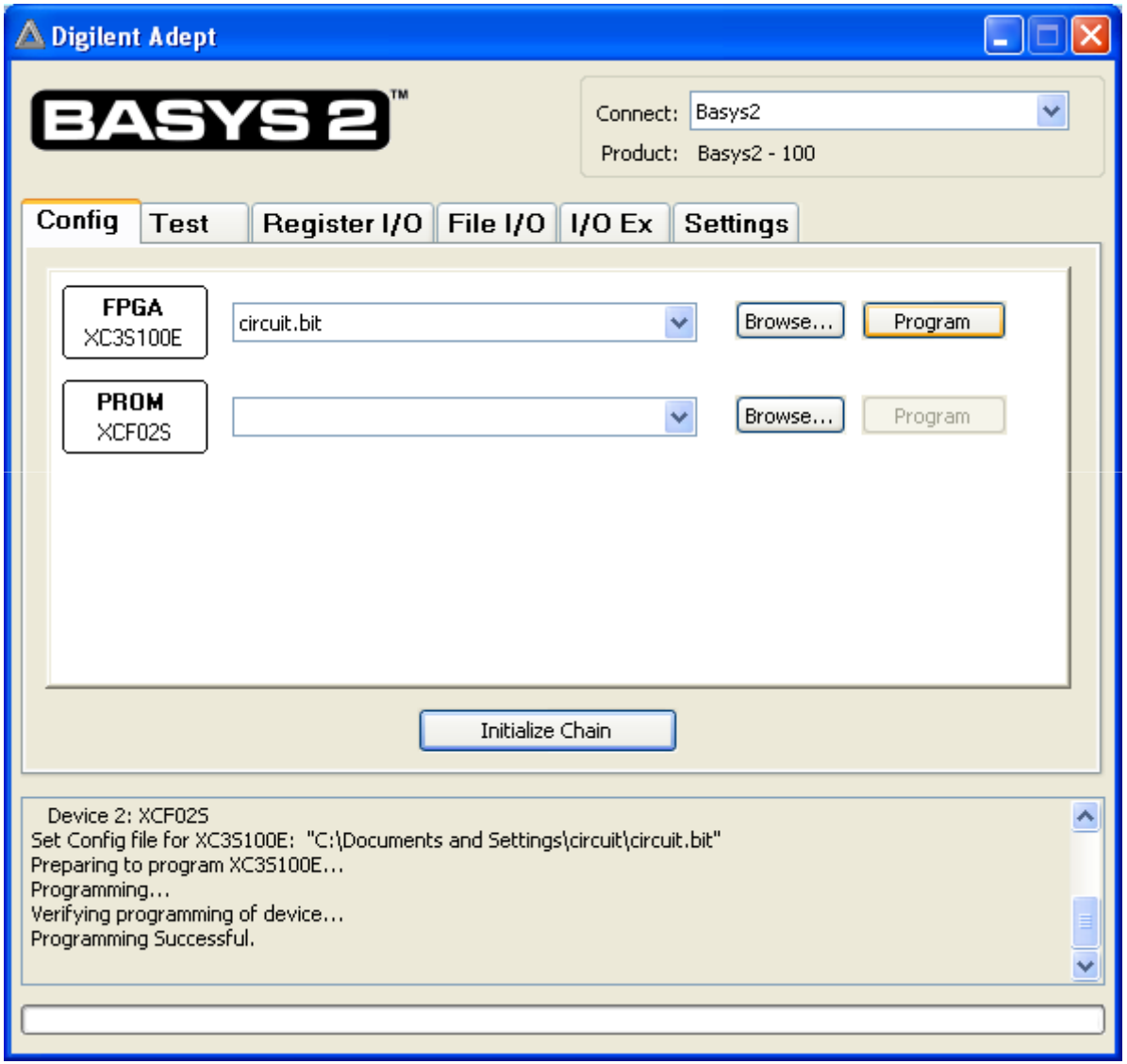


Click on Browse (next to FPGA Option) an select the bit file from the project Directory




Digilent Adept
BASYS 2™
Connect: Basys2
Product: Basys2 - 100
Config
Test
Register I/O
File I/O
I/O Ex
Settings
FPGA
XC3S100E
circuit.bit
Browse...
Program
PROM
XCF02S
Browse...
Program
Initialize Chain
Board information loaded.
Found device ID: f5045093
Found device ID: 11c10093
Initialization Complete.
Device 1: XC3S100E
Device 2: XCF02S

Make sure the basys2 is on, click on the program, and test your fpga



Another example

- The following circuit will blink LED #0 when enabled
  - Period = 2 seconds
  - Enable = Switch #0
    - When Switch #0 is off, LED #0 is off

Pin Assignments

- Clk
  - B8
- Enable (SW0)
  - P11
- LED (LD0)
  - M5

```
module blinking_led(clk, led, enable);
input clk;
output led;
input enable;
reg led;
integer count;

always @(posedge clk)
begin
        if (enable==0)
        begin
                count=0;
                led=0;
        end
        else
        begin
                if (count > 50000000)
                begin
                                count=0;
                                led=~led;
                end
        count = count+1;
        end
end
endmodule
```

Basys Board Resources

Basys board documentation and resources can be found at the Digilent Website. Click on http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,400,790&Prod=BASYS2 to go right to these resources.

Quick Links

Reference Manual
http://www.digilentinc.com/Data/Products/BASYS2/Basys2_rm.pdf
Schematics
http://www.digilentinc.com/Data/Products/BASYS2/Basys2_sch.pdf

Adept Programming Suite

The Adept Programming Suite is required to the Basys2 board using the USB cable. It can be downloaded by clicking http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,66,69&Prod=ADEPT

Required Software

- Xilinx ISE WebPack
  - Project Navigator (Main workspace for verilog)
  - ISIM (Simulator)
  - PlanAhead (to assign Pins)

- iVerliog (Simulator)

- Diligent
  - configuration and print the implementation of the fpga board

References

- Michael D. Ciletti, *Advanced Digital Design with the Verilog HDL, Pearson Education, Inc.*
- (Prentice Hall), 2003
- Donald E. Thomas and Philip R. Moorby, *The Verilog Hardware Description Language,*
- Kluwer Academic Publishers, 1998
- Samir Palnitkar, *Verilog HDL A Guide to Digital Design and Synthesis, Prentice Hall, Inc., 4th*
- Edition, 1996
- David R. Smith and Paul D. Franzon, *Verilog Styles of Digital Systems, Prentice Hall, Inc.,*
- 2000
- *Digilent Basys 2 Board Reference Manual, Digilent, Inc., May 25, 2009*
- *Digilent BASYS 2 System Board Schematics, Digilent, Inc., December 12, 2008*