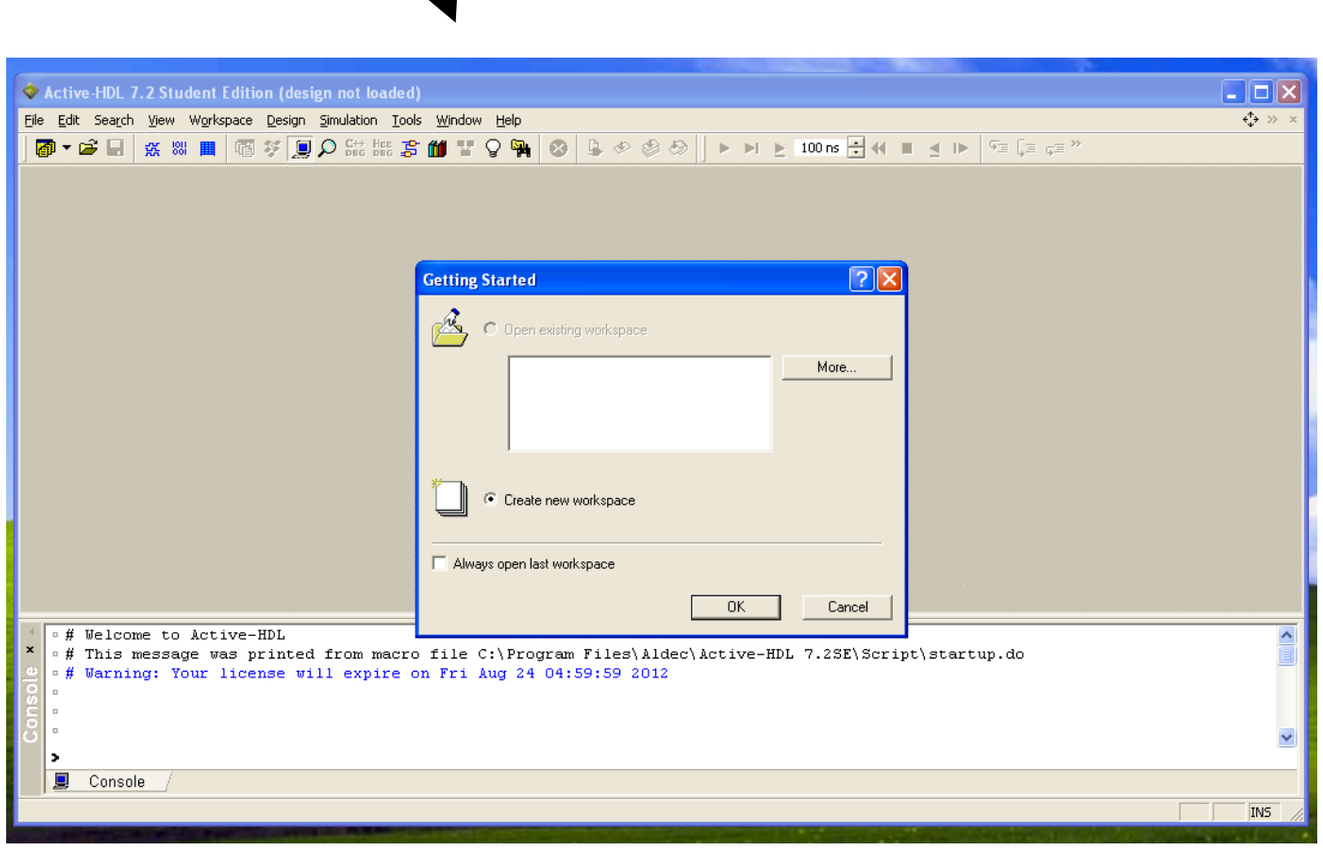
CPE 462
VHDL: Simulation and Synthesis

Topic #03 - c) Introduction to Aldec Active HDL development environment

Introduction to Active HDL

- Active HDL is a neat HDL simulation program
- It is fast, small, easy to use and straight to the point
- Latest Xilinx student development environment was 6 GB!
- We will use Active HDL as our development environment
- Install the software and request a license (painless process)

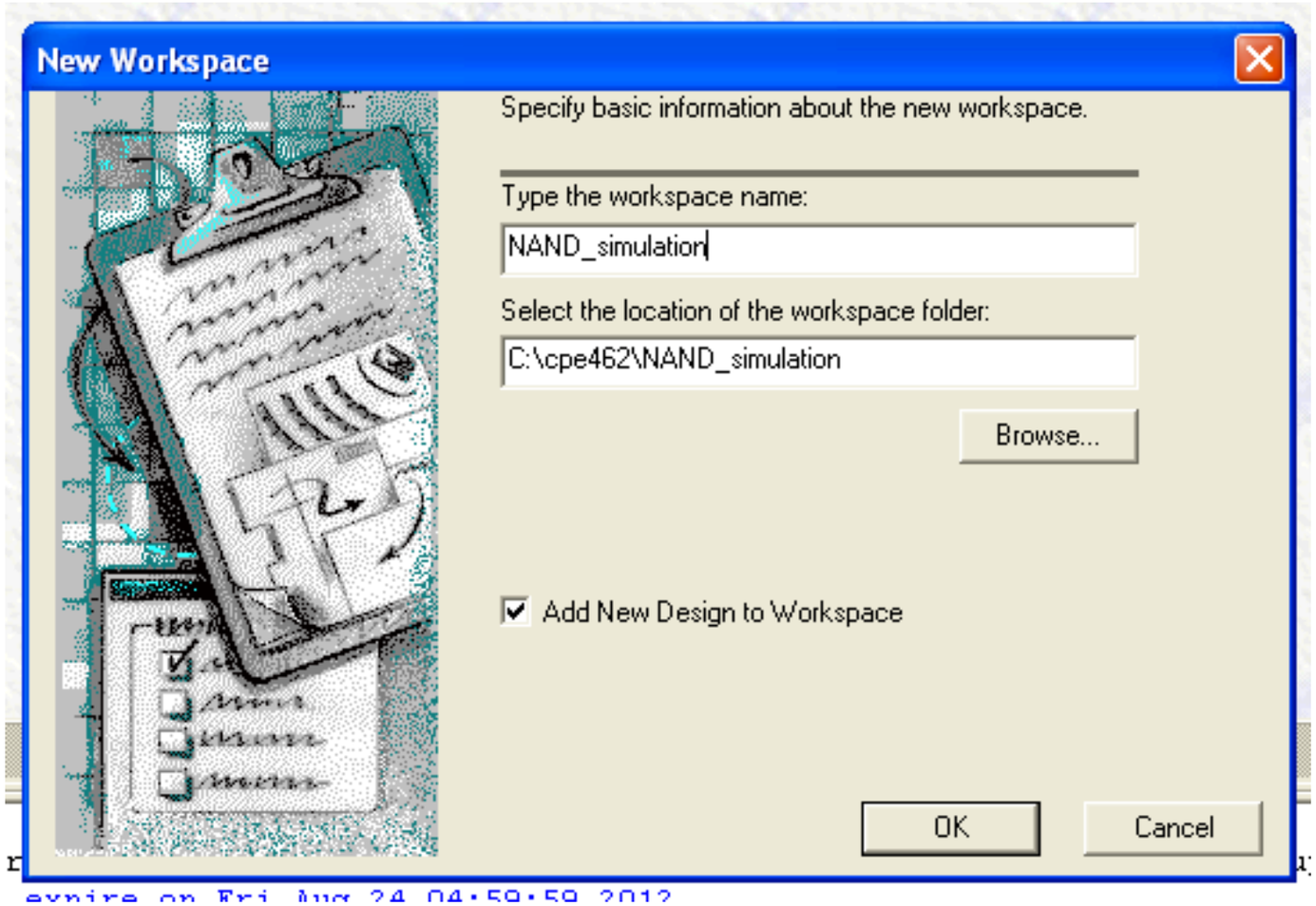
This is what you will see when you start the program...



Simulating a NAND with Active HDL

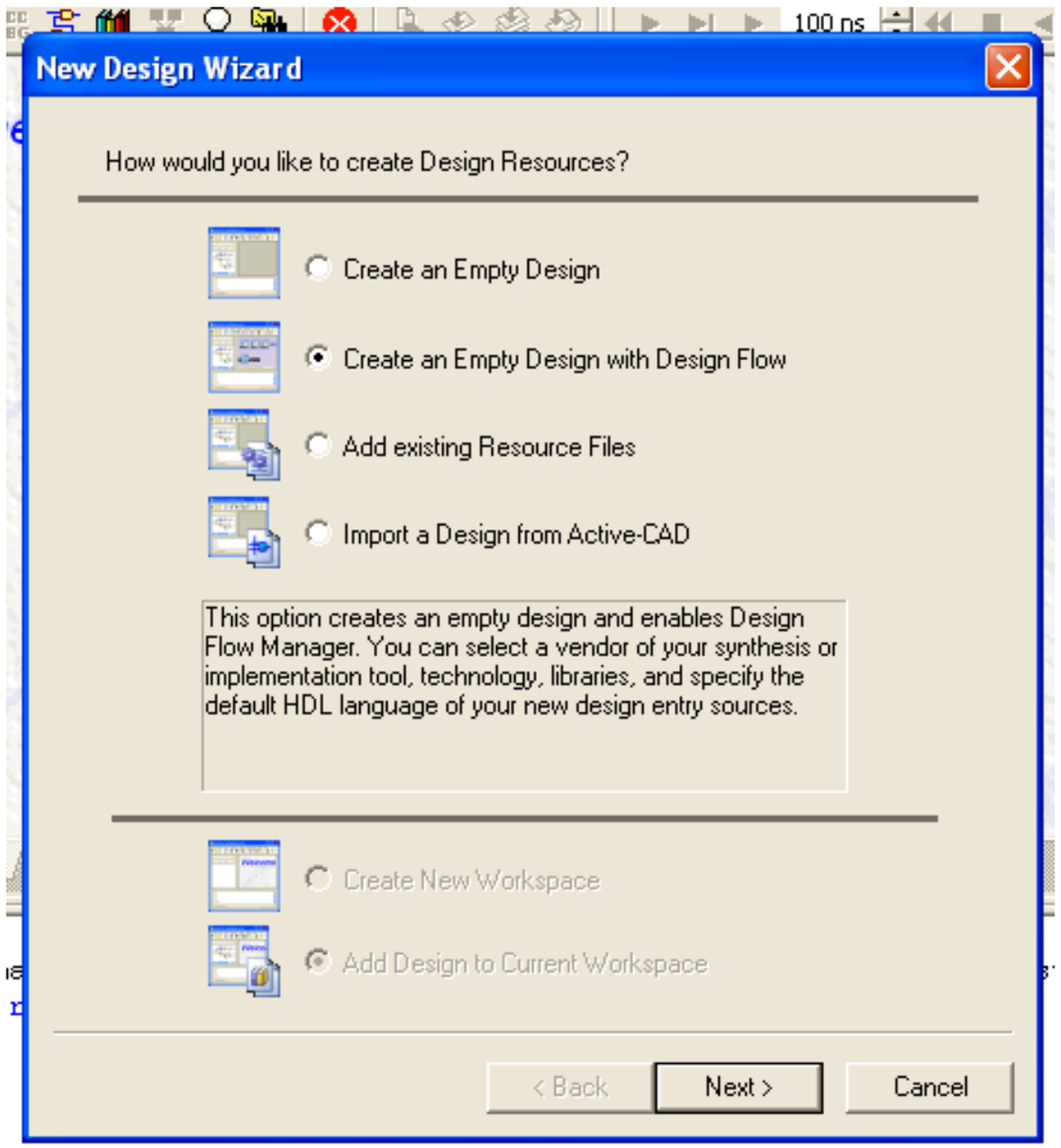
First steps with the wizard

- Create a new workspace and call it NAND_simulation



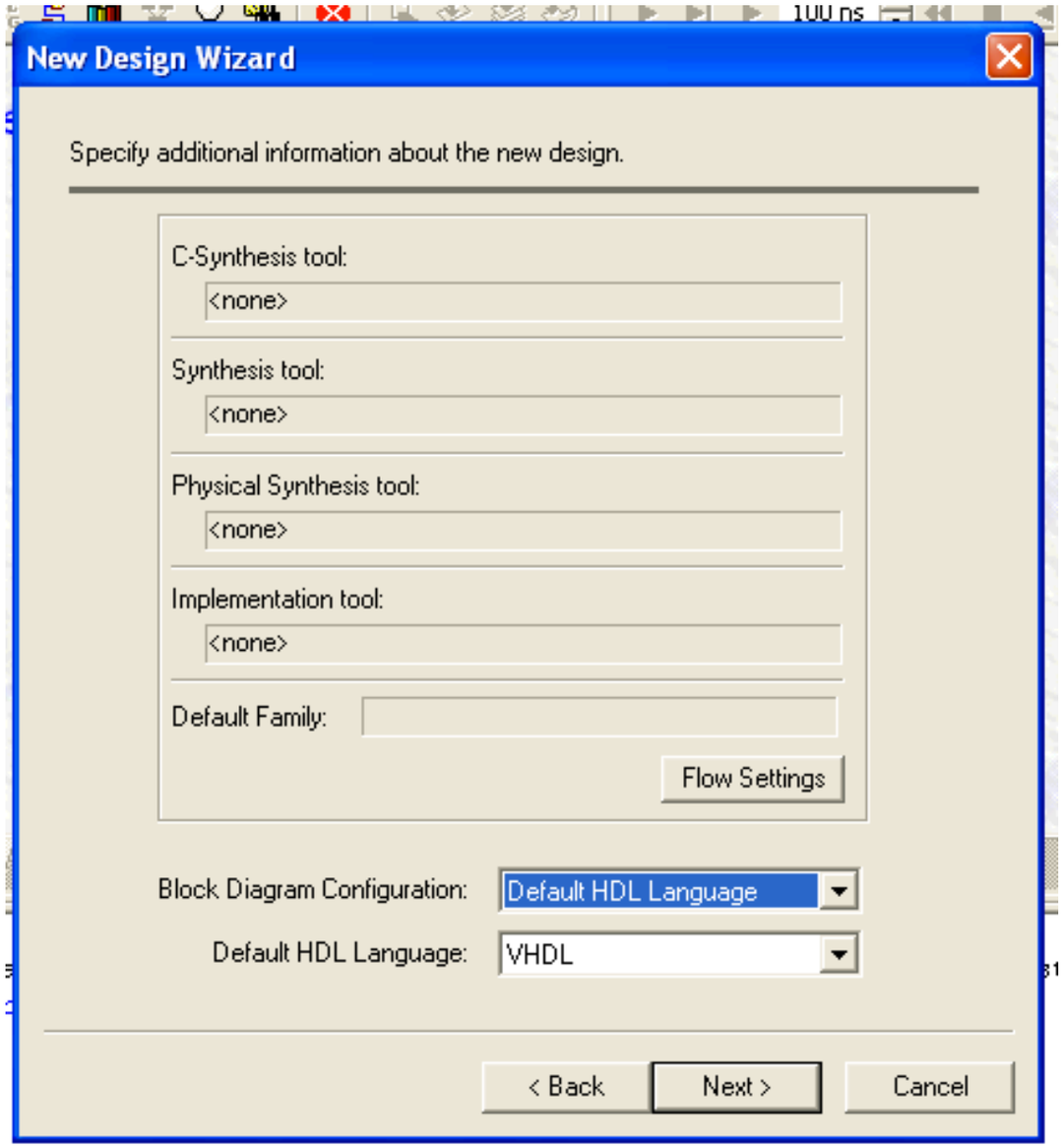
New design wizard 1/3

- *Create an Empty Design with Design Flow*



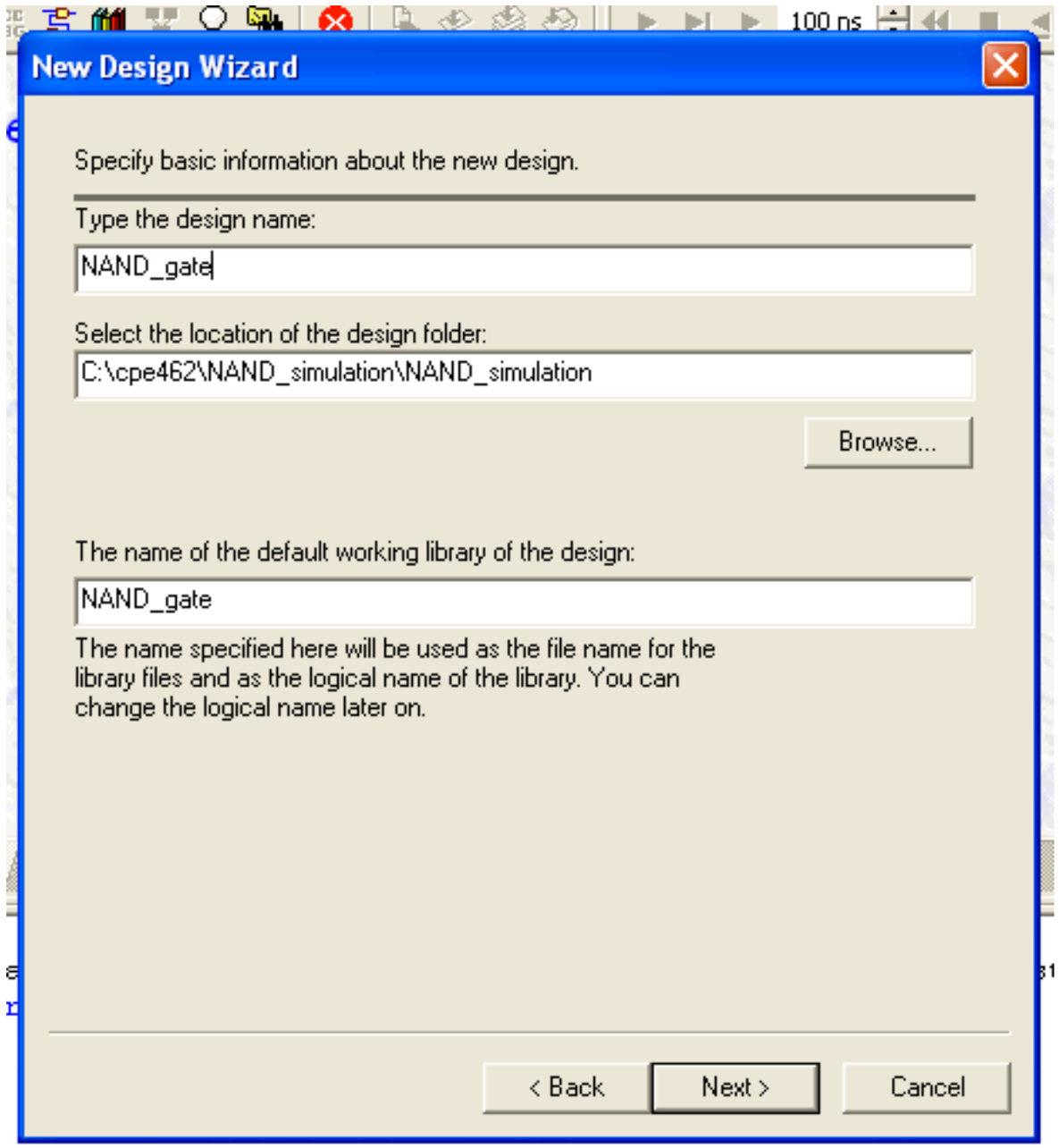
New design wizard 2/3

- Make sure VHDL is the default language
- If we have time we will introduce other synthesis methods towards the end of the class



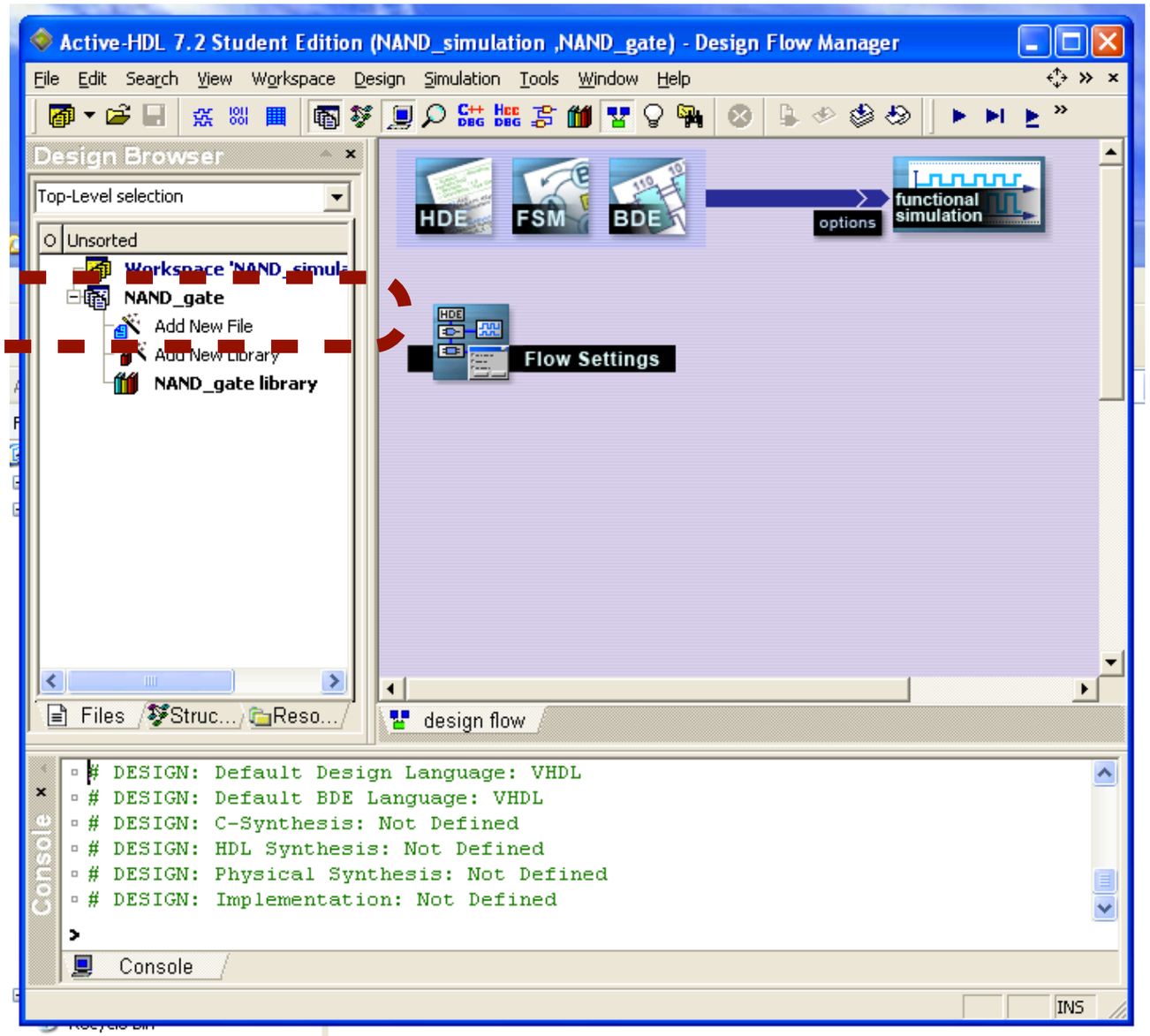
New design wizard 3/3

- Pick a design name
- NAND_gate works fine

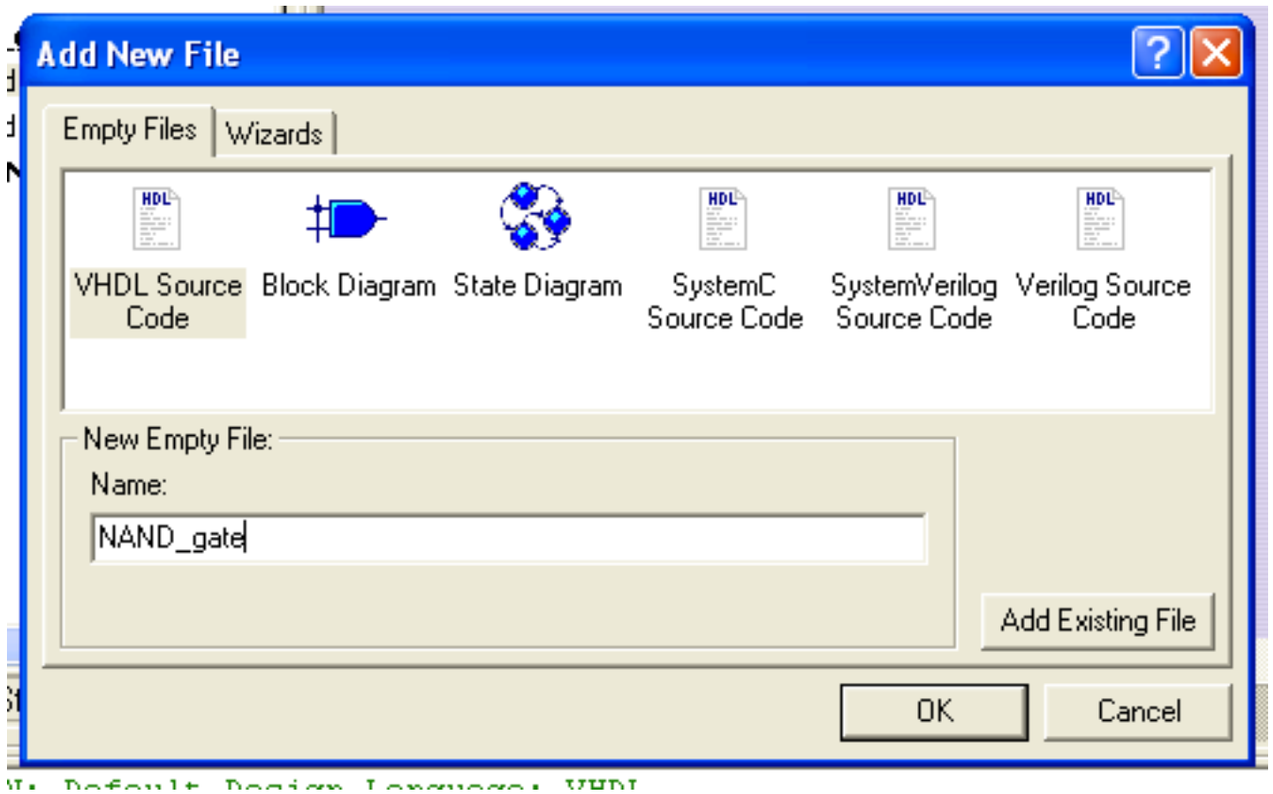


Main window

- Once you click finish, you will be greeted with this screen
- Double click on *Add New File*



Adding a new source file



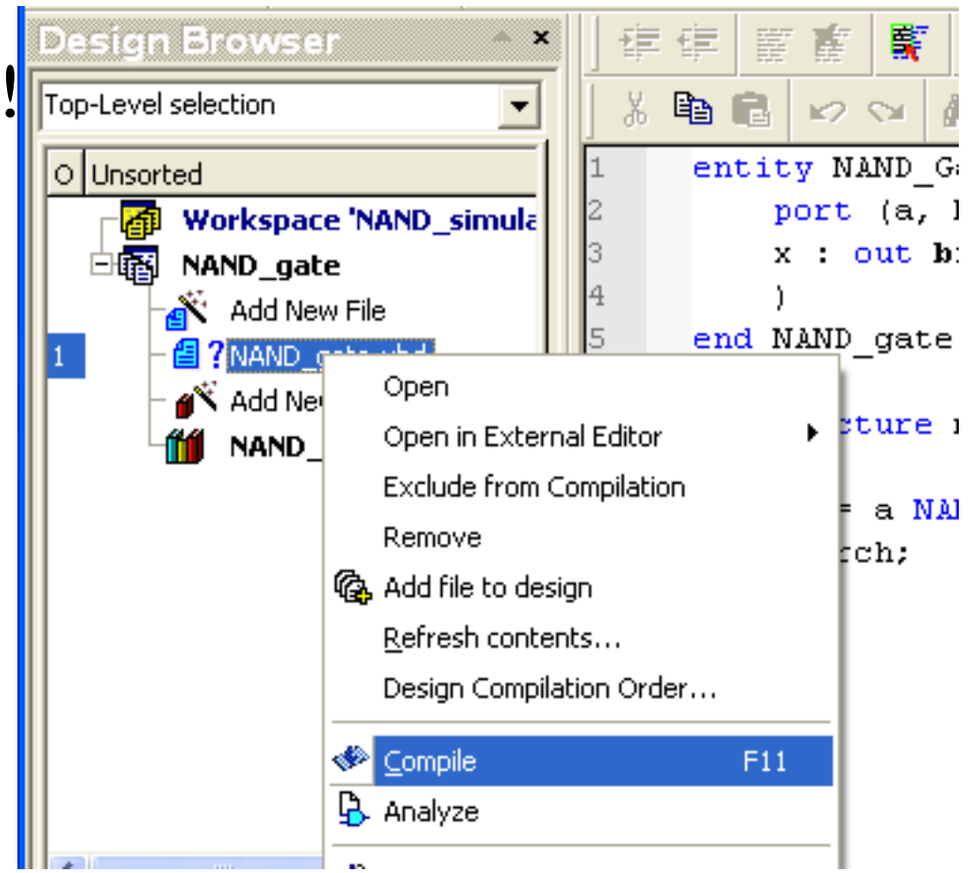

```
entity NAND_Gate is
    port (a, b : in bit;
    x : out bit);
end NAND_gate;

architecture myarch of NAND_gate is
begin
    x <= a NAND b;
end myarch;
```

- Select *VHDL code* and give it a name
- This is the code we need to add to the project

Add in the NAND code

Something dubious in this line!

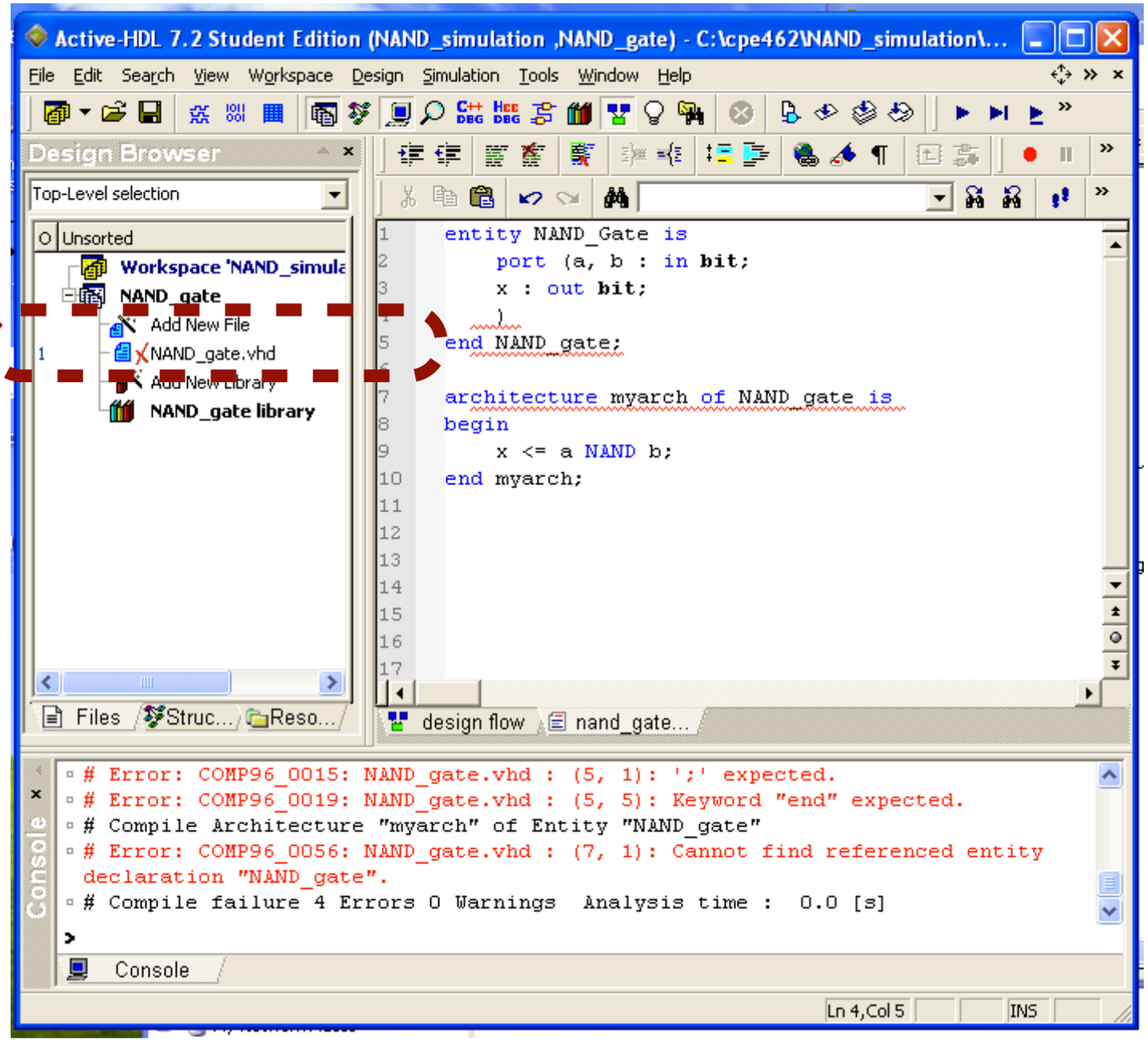
```
1   entity NAND_Gate is
2       port (a, b : in bit;
3       x : out bit;
4       )
5   end NAND_gate;
6
7   architecture myarch of NAND_gate is
8   begin
9       x <= a NAND b;
10  end myarch;
```



- Type in the standard NAND operation code
- Compile the code to see if there are any errors

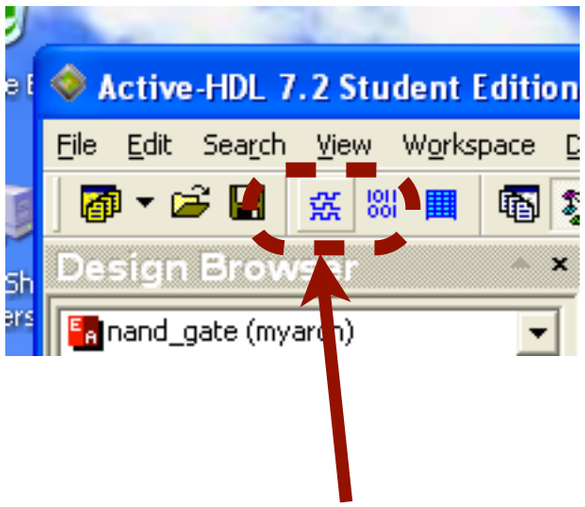
Syntax errors are like this

- Misplaced semi-colon will generate a syntax error
- And a cross will also show up next to the file



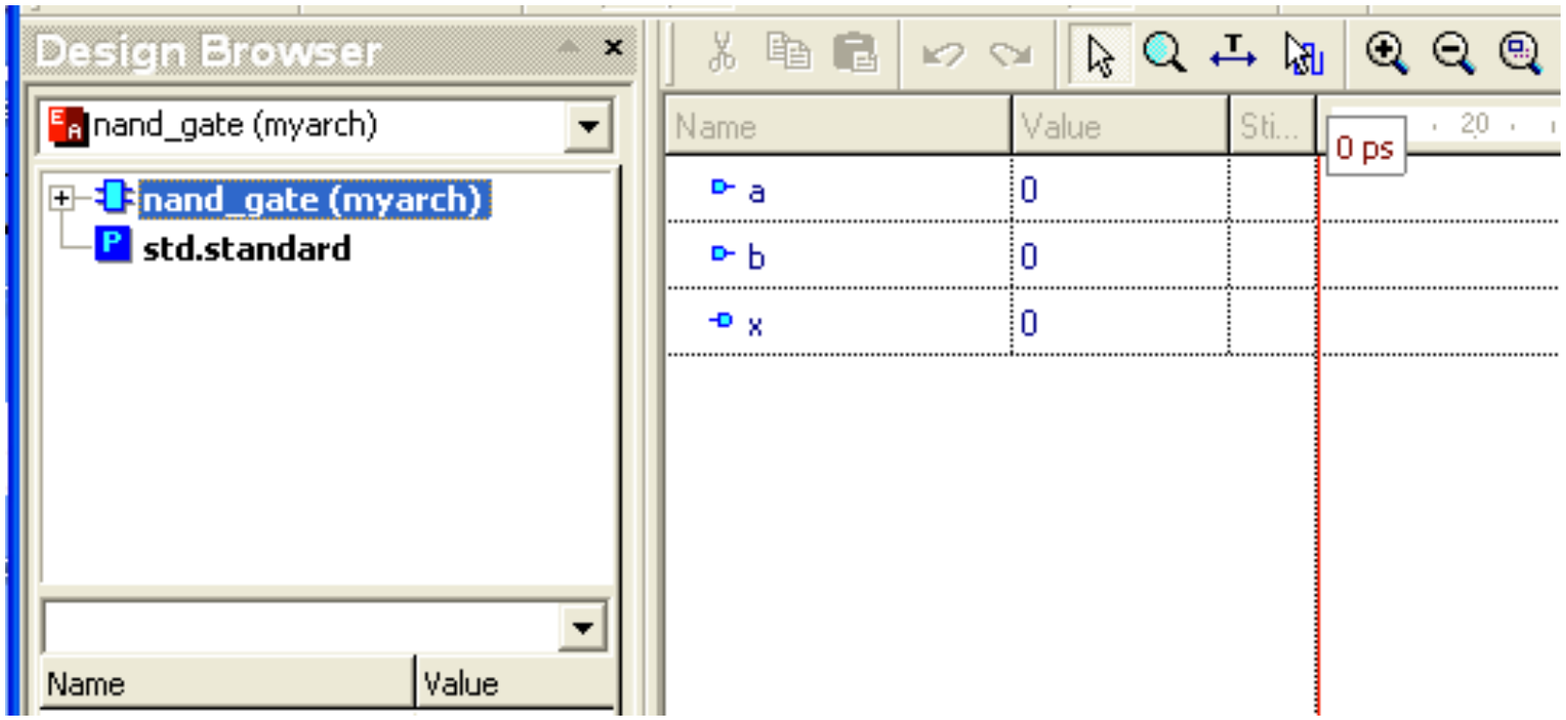
Let's start simulation

- Click on *Initialize Simulation*



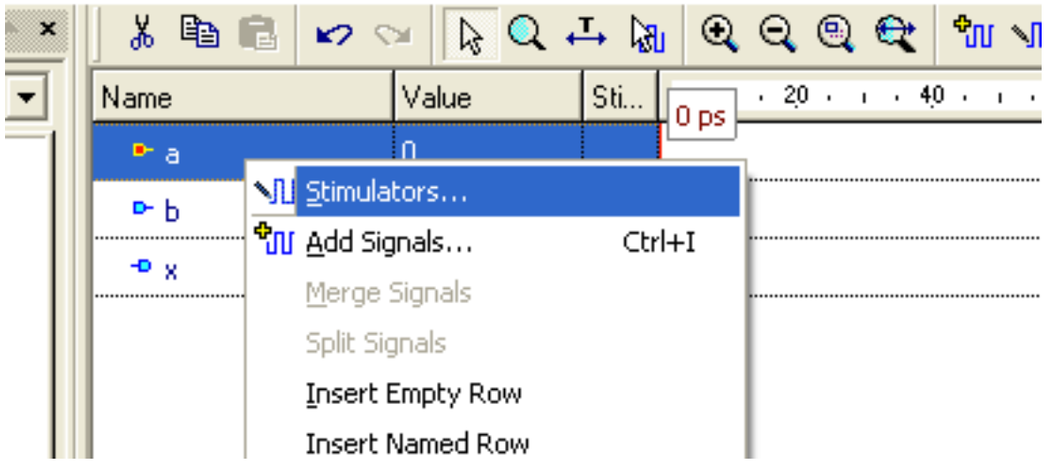
- Then click on the *Create a New Waveform*

Obtaining the I/O port names



- Click on *nand_gate (myarch)* and drag it to the empty area to the right
- This is what it will show...

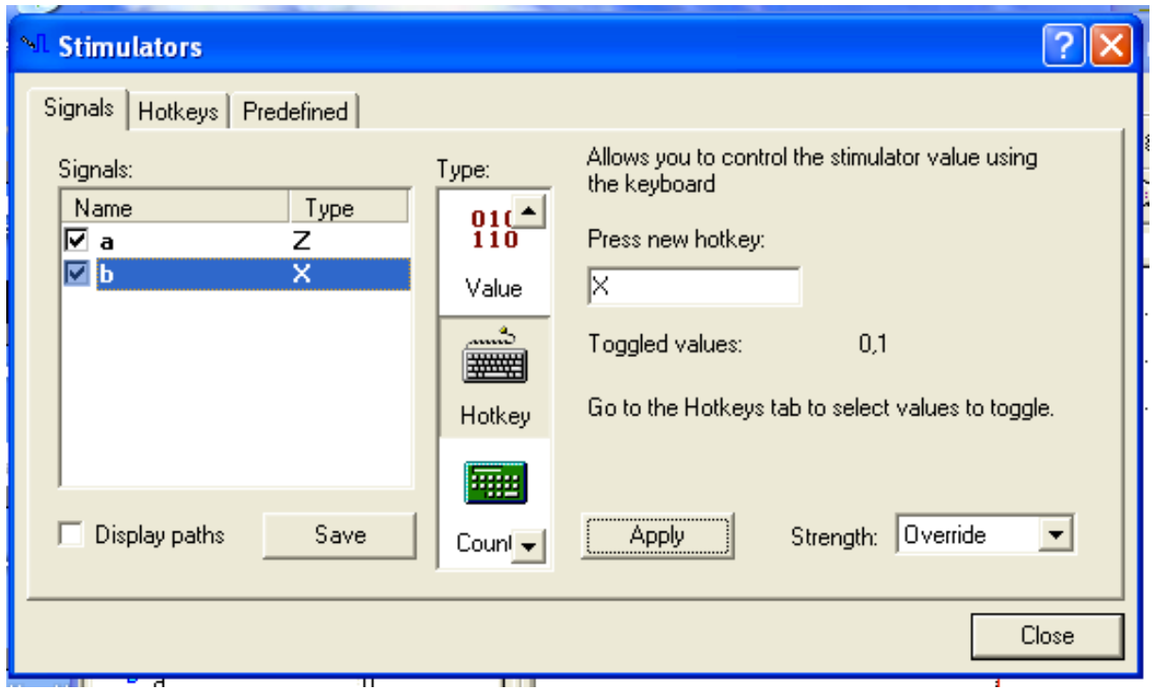
Adding stimulus signals 1/2

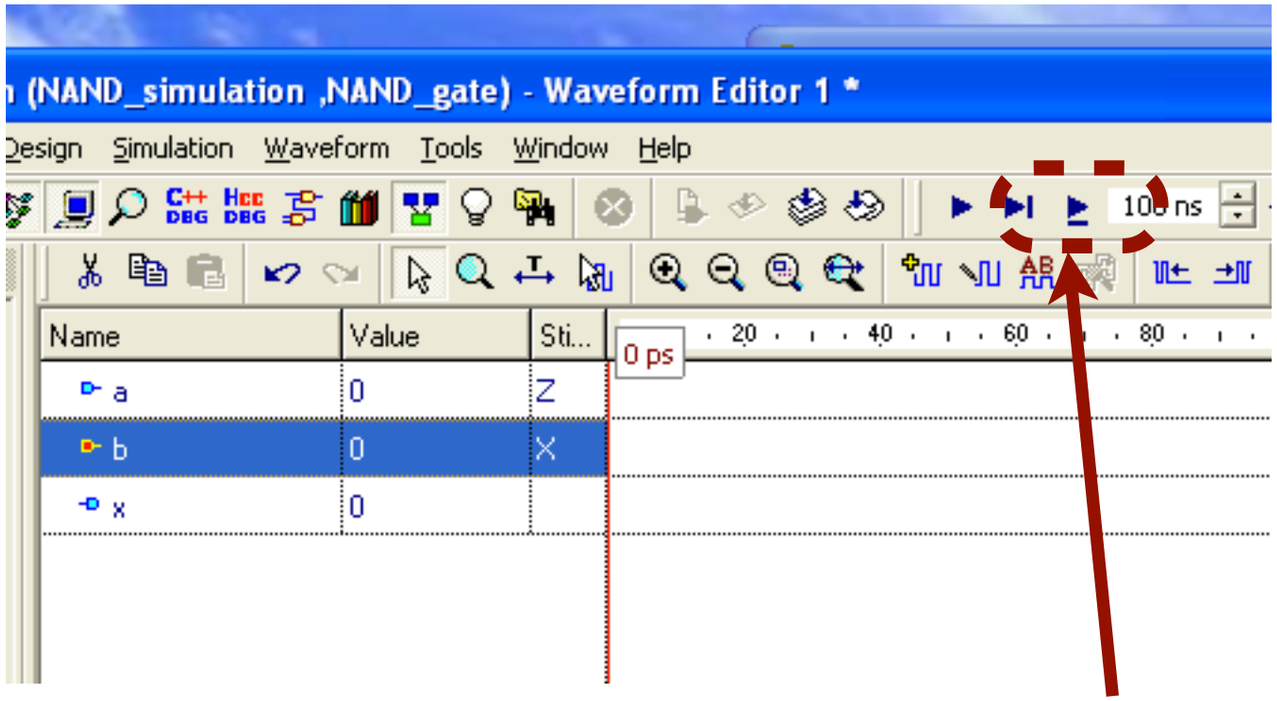


- Right-click on signal **a**
- Select the *Simulators* option

- Scroll the *Type* all the way down and select *Hotkey*
- Assign **Z** as the hotkey
- During simulation, pressing **Z** will switch the signal **a**

Adding stimulus signals 2/2

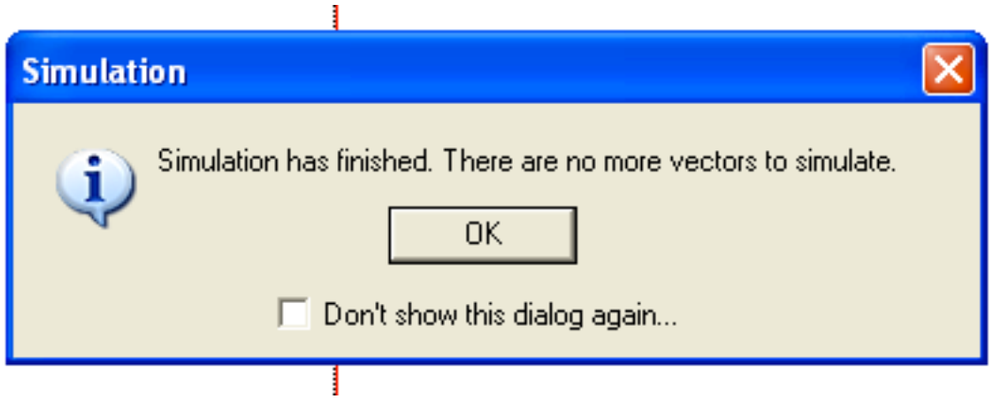


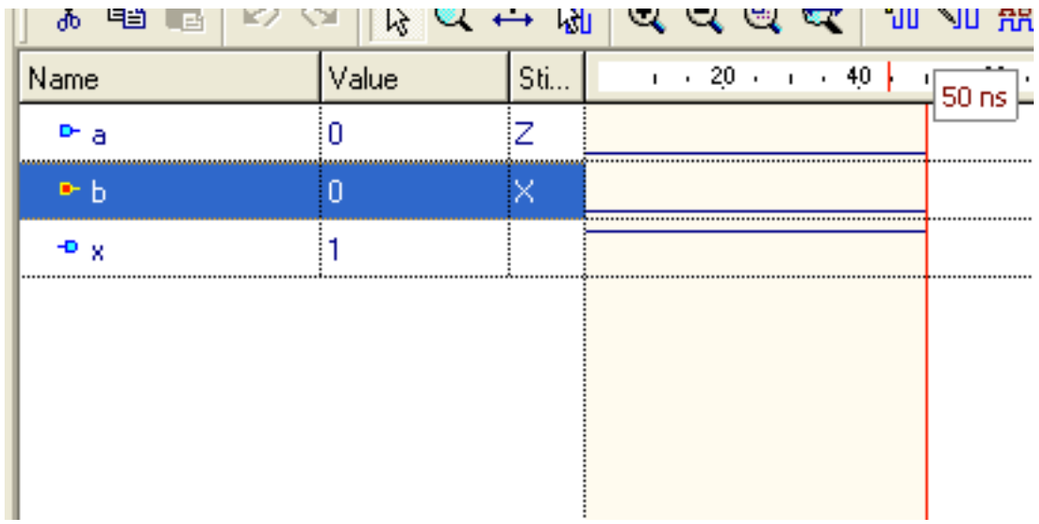


- After assigning the hotkey X to b

- Right now a and b are both logic-0
- Click on the *run for*
- This will move the simulation whatever number is on the next field

Simulation



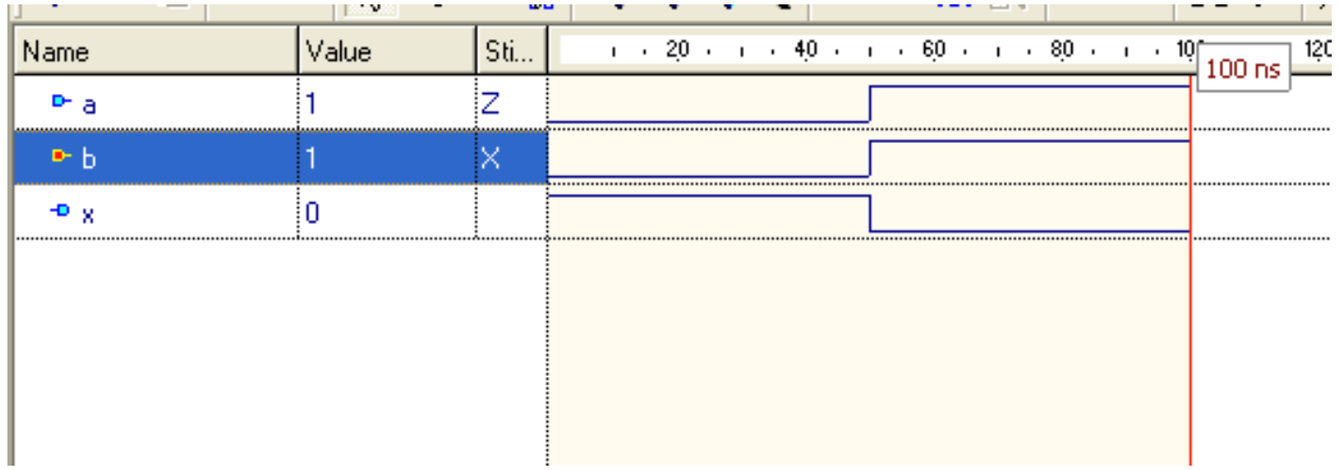


- This message will show up since we are going through a manual process

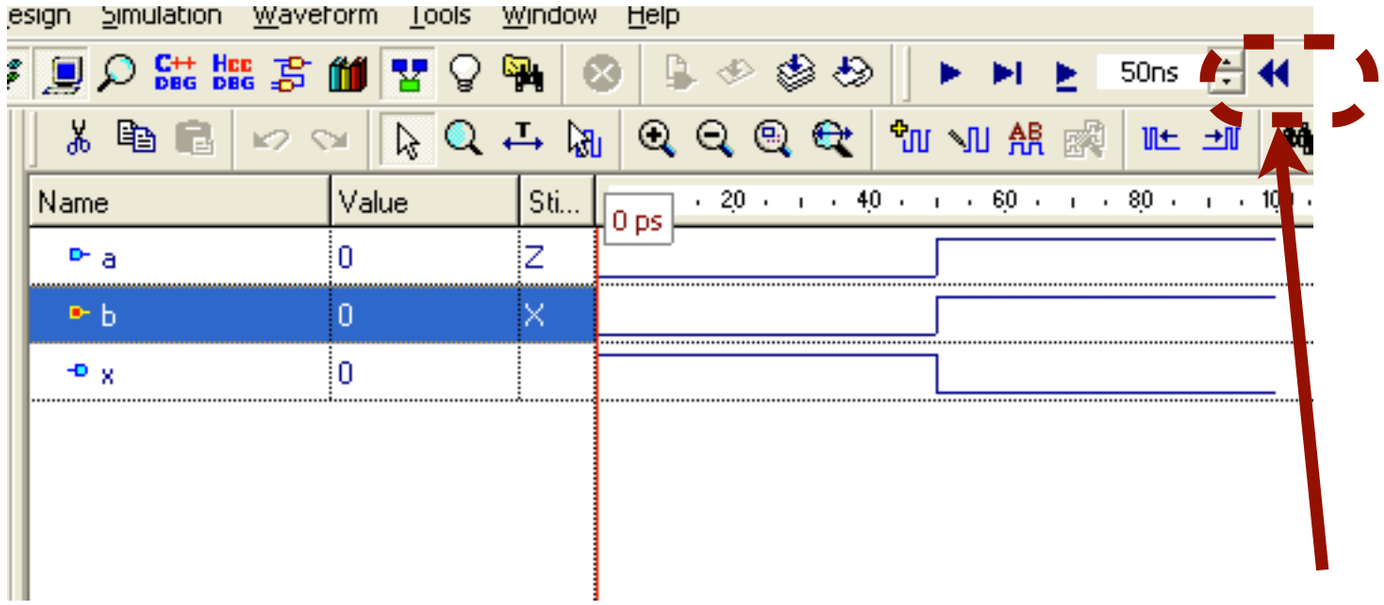
- As expected the output is logic-1

Changing simulation values

- Pressing **Z** and **X** will switch the values of **a** and **b**
- As expected the output is a logic-0

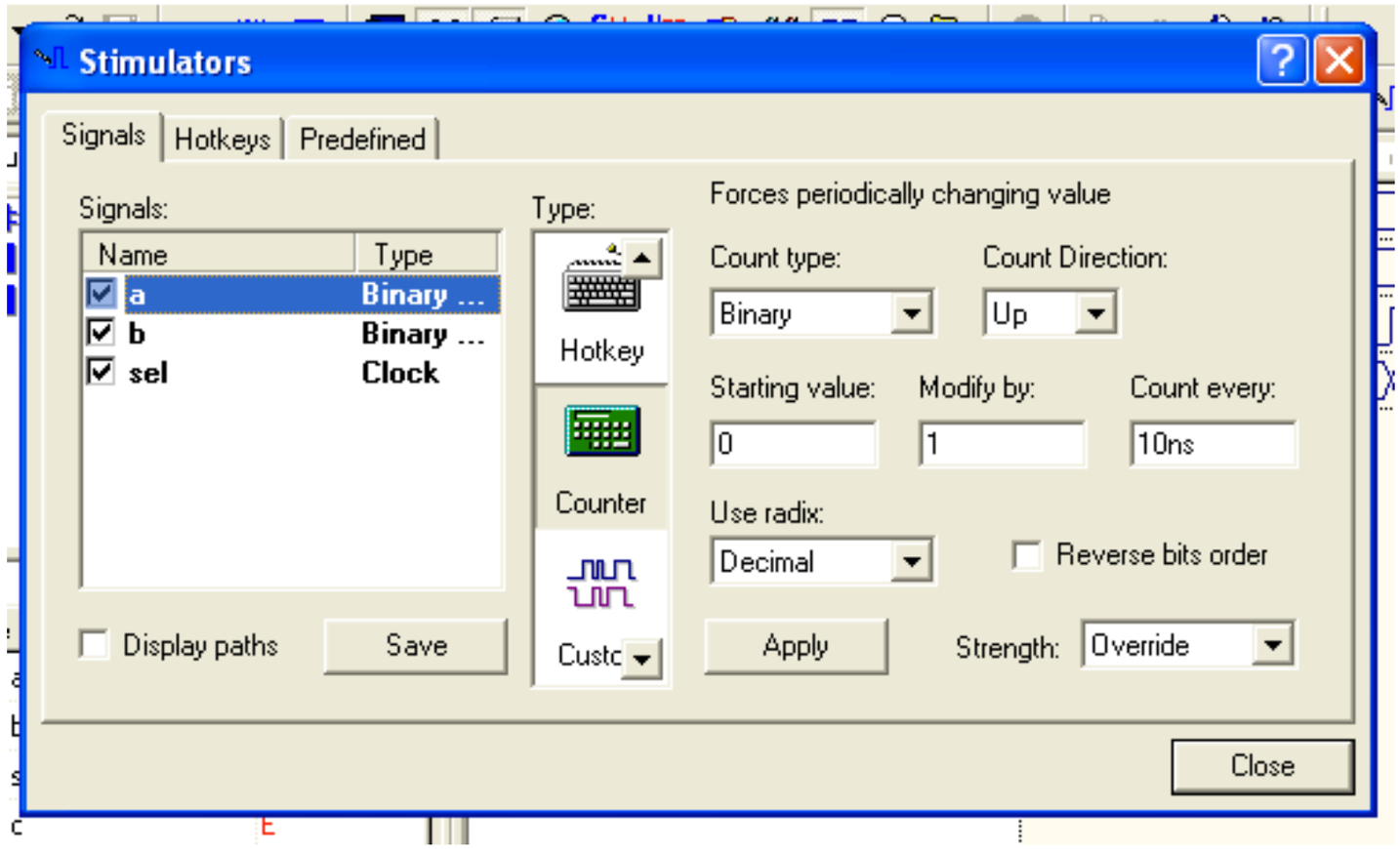


- To restart the simulation, which will bring the marker to *0ps* click on the appropriate button



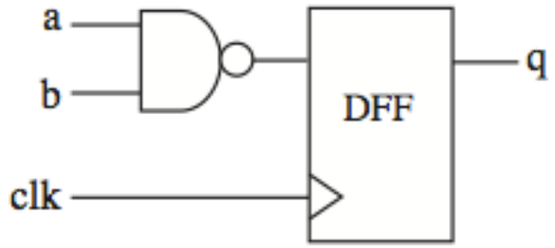
Another neat Active HDL stimulus: counter

- If you have a bus with many input lines, you can use the counter stimulus to cycle through all logic.



Practice Exercises

Exercise #1- NAND + DFF

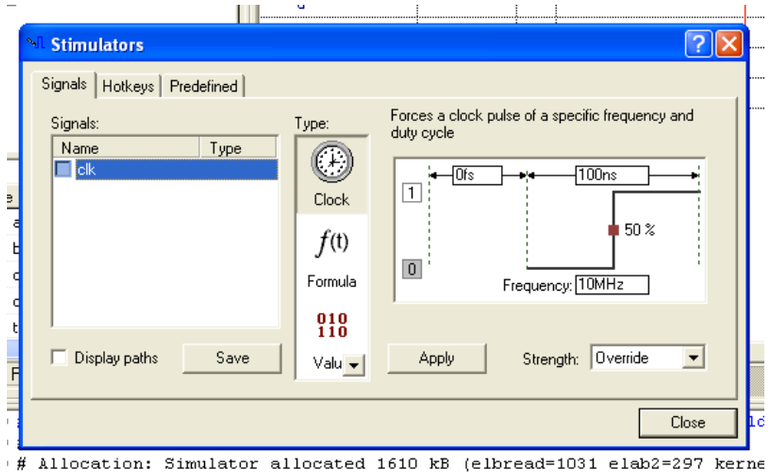



```
entity example is
    port (a,b,clk : in bit;
    q: out bit);
end example;


architecture example of example is
signal temp : bit;

begin
  temp <= a NAND b;
  process (clk)
    begin
    if (clk'event and clk='1') then q<=temp;
    end if;
  end process;
end example;
```

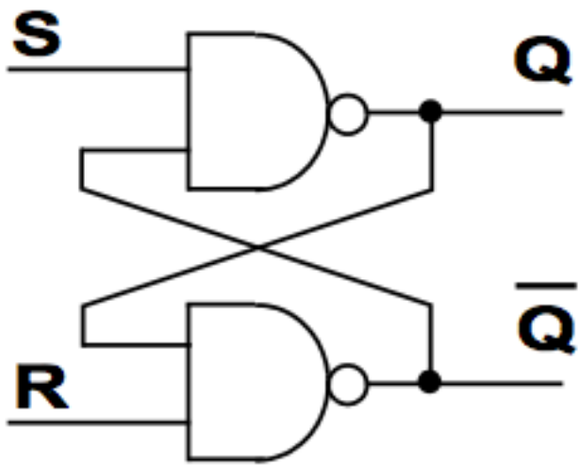
- Re-implement this code but find a way to remove *q<=temp*

- Display the wave-form. Use the "*Clock function*" as a stimulus for the clock signal and "hotkey" for the other inputs

- Set the clock period to 20ns



Exercise #2- Set Reset (SR) latch



S	R	Q	$\overline{Q}$	Function
0	0	1-?	1-?	Indeterminate State
0	1	1	0	Set
1	0	0	1	Reset
1	1	Q^+	$\overline{Q}^+$	Storage State

- Implement this SR latch
- Hint: Make sure you appropriately configure the signal modes
- Simulate the circuit and make sure it works as expected