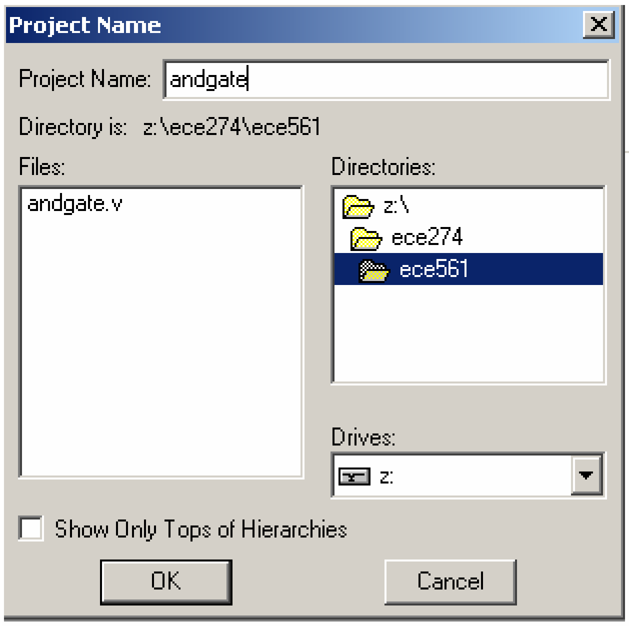
ECE 274 Tutorial for Lab1

This Course will use MAX+PLUS II software for Verilog design.

In the first lab, please generate a AND gate step by step follow the following instructions:

1: Open the MAX+PLUS II software and generate a new project – choose File – Project to generate project "andgate"



2: Open the text editor and input the Verilog code for AND gate:

```
module andgate(a,b,c);
input a, b;
output c;
reg c;

      always @(a or b) begin
            c = a & b;
      end

endmodule
```

andgate takes three arguments: two of them are inputs and the third is the output, which is specified in the header of the module.

'~' operator negates the input . '&' operator ANDs two inputs , '|' operator ORs two inputs, while '^' operator performs a logical XOR on two inputs. Hence, ~ (a | b) implements a NOR function where a and b are taken to be two inputs. Verilog uses C-style comments. '//' marks the beginning of a line of comments. Moreover, anything between /* and */ is also treated as a comment.

always is a statement that lets you define blocks of code, in which assignments will only happen as selected signals (specified in the header of the statement) change.

For example:

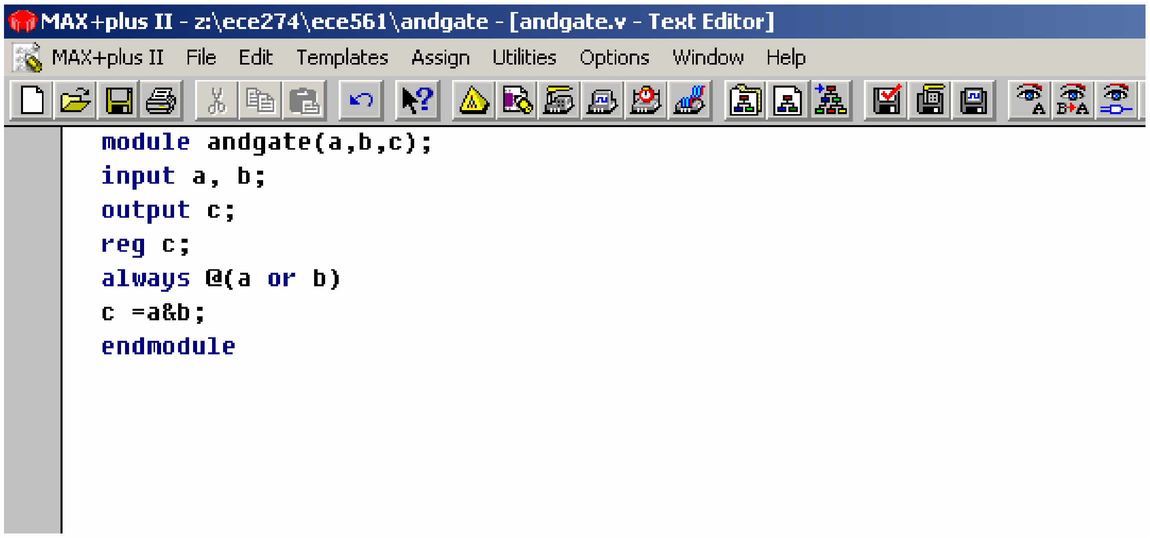
```
input a, d
reg out

always @(a or d) begin
      out = d;
end
```

sets out to d only if the value of a or the value of d changes. Notice a slightly different syntax (we’re using the word ‘or’ as opposed to the symbol ‘|’, and ‘=’ instead of ‘assign’).
Notice that all signals on the left-hand side of the ‘=’ sign inside ***always*** blocks must be declared as ***reg***, not wire.

If you need to perform more than one operation under the always block, you need to enclose the operations with begin and end.

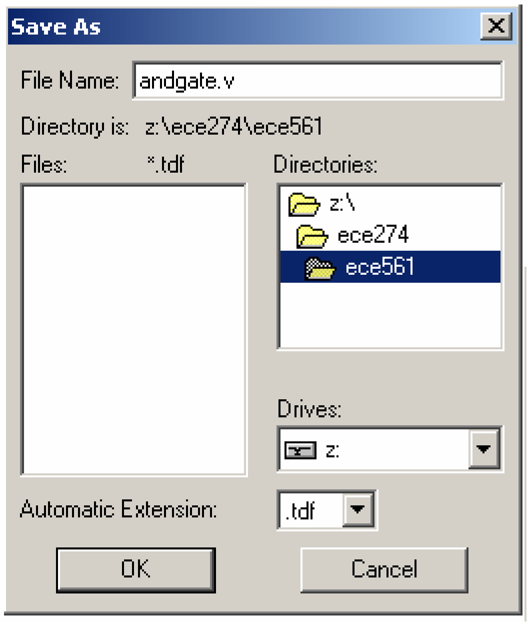
always is most often combined with if (and possibly else if and else) to create powerful sequential logic systems. Study the following example:

```
always @(a) begin
      if (a==2’b01) begin
            b = 2’b01;
      end
      else if (a>2’b01) begin
            b = a;
            c = 1;
      end
end
```

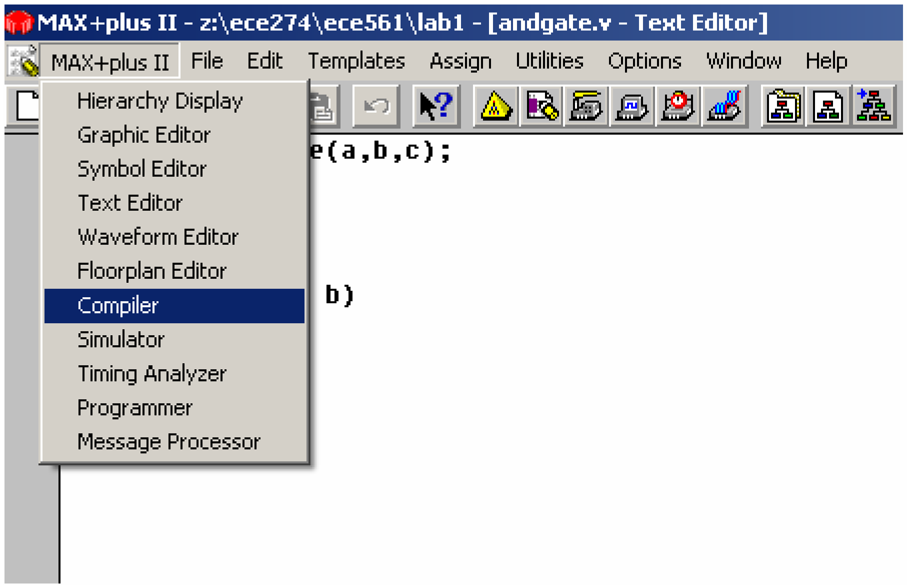

MAX+plus II - z:\ece274\ece561\andgate - [andgate.v - Text Editor]
MAX+plus II
File
Edit
Templates
Assign
Utilities
Options
Window
Help

```
module andgate(a,b,c);
input a, b;
output c;
reg c;
always @(a or b)
c =a&b;
endmodule
```

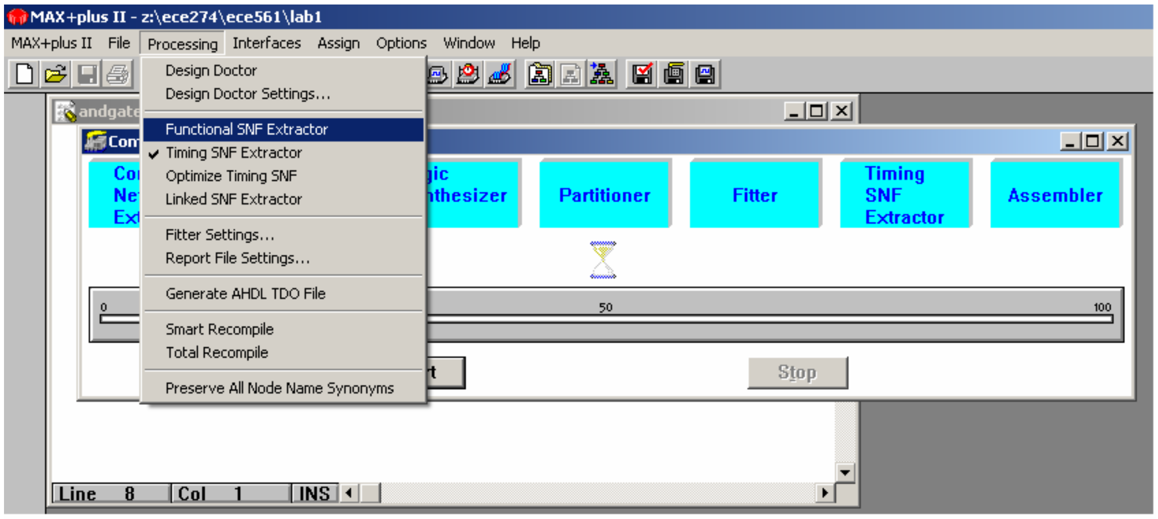
3. Save the Verilog code as .v file.
Please pay attention here; the file name must be the same as the module name. So this will be saved as andgate.v



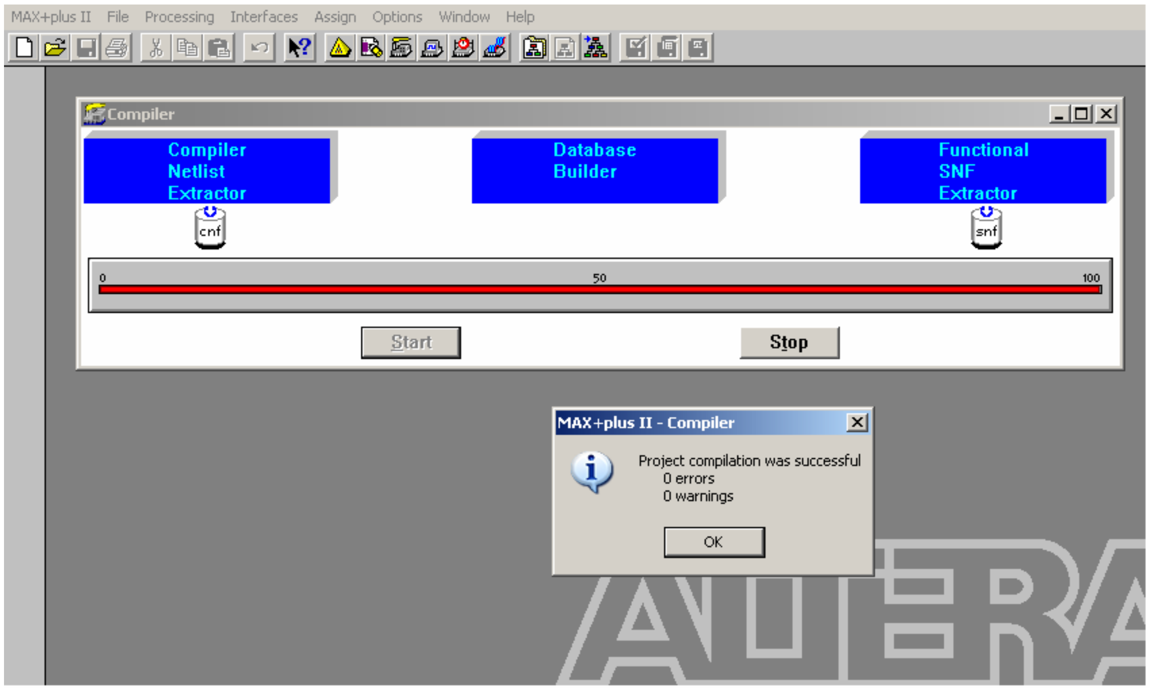
4. Use the compiler to compile the code:



Choose "Processing" – "Functional SNF Extractor"



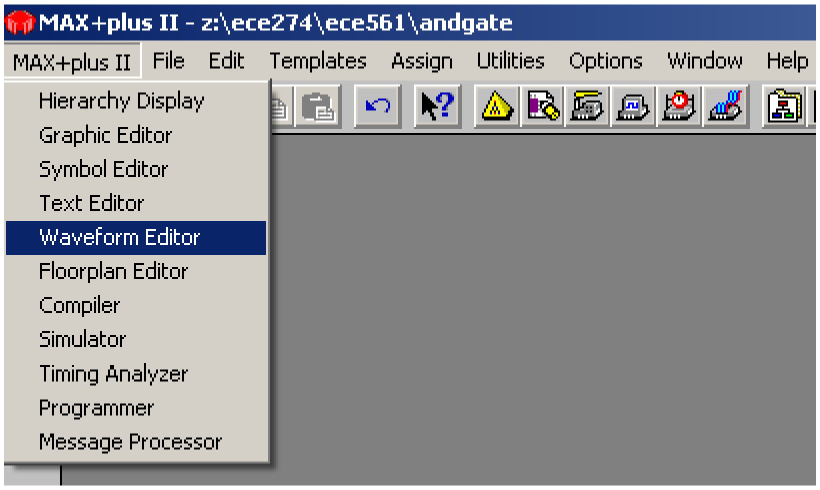
Click “Start” to run the compiler.



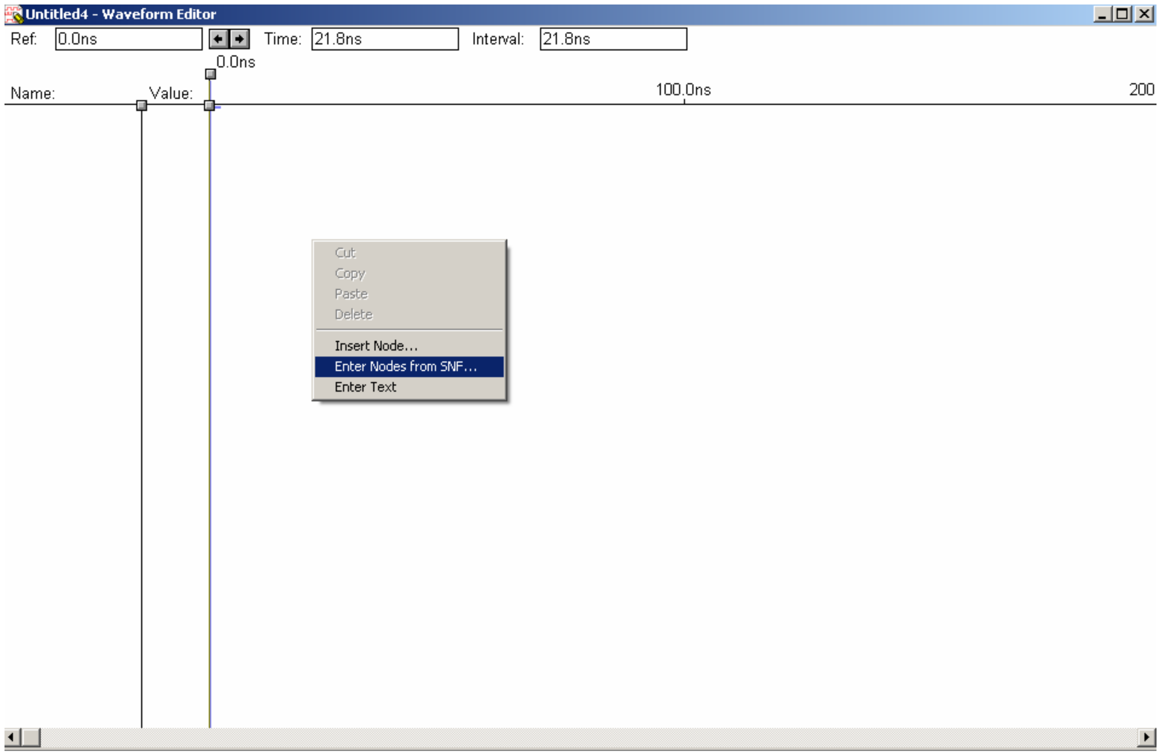
Project compilation was successful
0 errors
0 warnings

The compiler will show there are no errors if the code is OK.

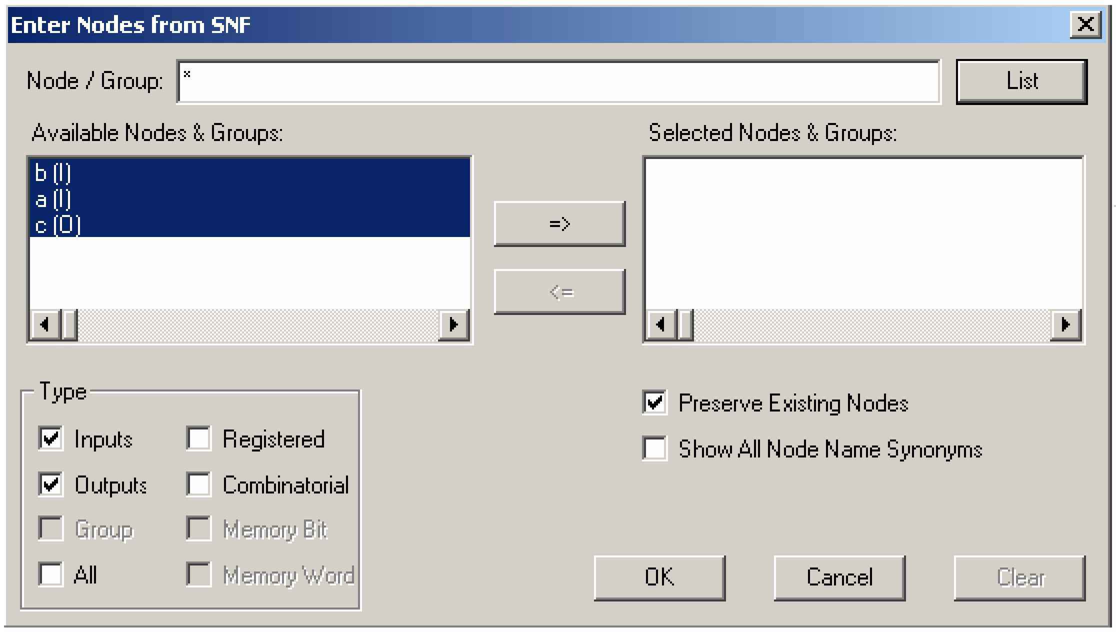
5. Simulation
First, choose the “Waveform Editor”.



Then choose "node" and select "Enter Nodes From SNF"



Click list.



Enter Nodes from SNF

Node / Group: *

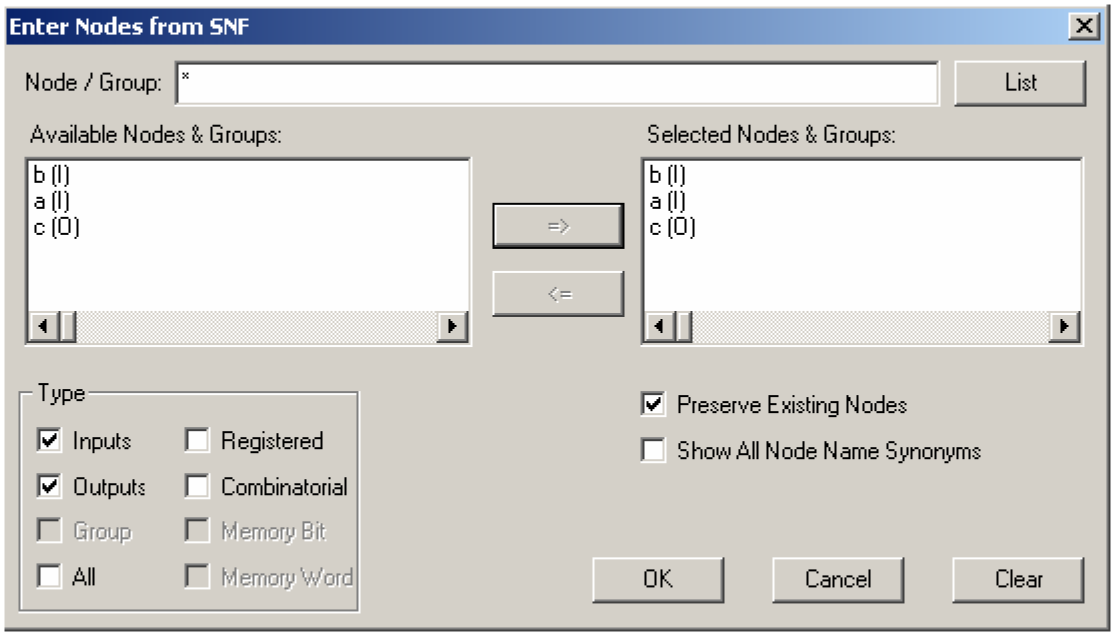
Available Nodes & Groups:
b (I)
a (I)
c (O)

Selected Nodes & Groups:

Type
[✓] Inputs
[ ] Registered
[✓] Outputs
[ ] Combinatorial
[ ] Group
[ ] Memory Bit
[ ] All
[ ] Memory Word

[✓] Preserve Existing Nodes
[ ] Show All Node Name Synonyms

Click “>>” button. All the input and output pins will be showed in the window “selected nodes & groups”.



Enter Nodes from SNF

Node / Group: *

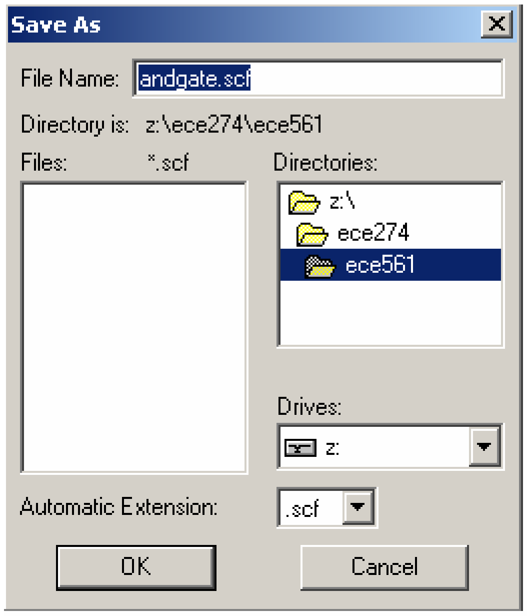
Available Nodes & Groups:
b (I)
a (I)
c (O)

Selected Nodes & Groups:
b (I)
a (I)
c (O)

Type
[✓] Inputs
[ ] Registered
[✓] Outputs
[ ] Combinatorial
[ ] Group
[ ] Memory Bit
[ ] All
[ ] Memory Word

[✓] Preserve Existing Nodes
[ ] Show All Node Name Synonyms

Save as “andgate.scf”



File Name: andgate.scf

Directory is: z:\ece274\ece561

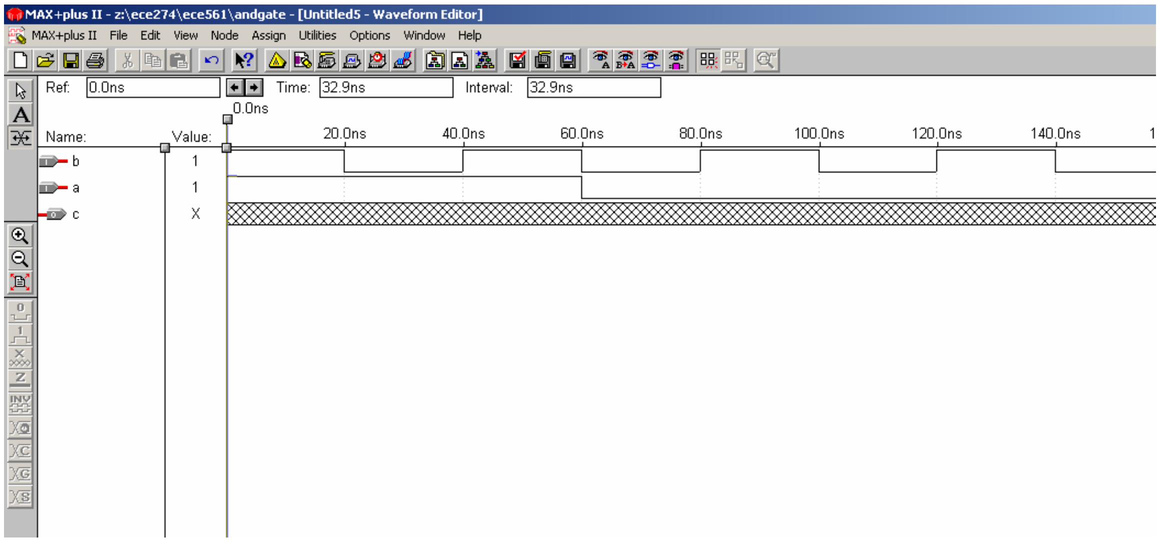
Files: *.scf

Directories: z:\ ece274 ece561

Drives: z:

Automatic Extension: .scf

Change the input signals *a* and *b* manually to “a=0, b=0”, “a=0, b=1”, “a=1, b=0”, “a=1, b=1”, 20 ns a period.

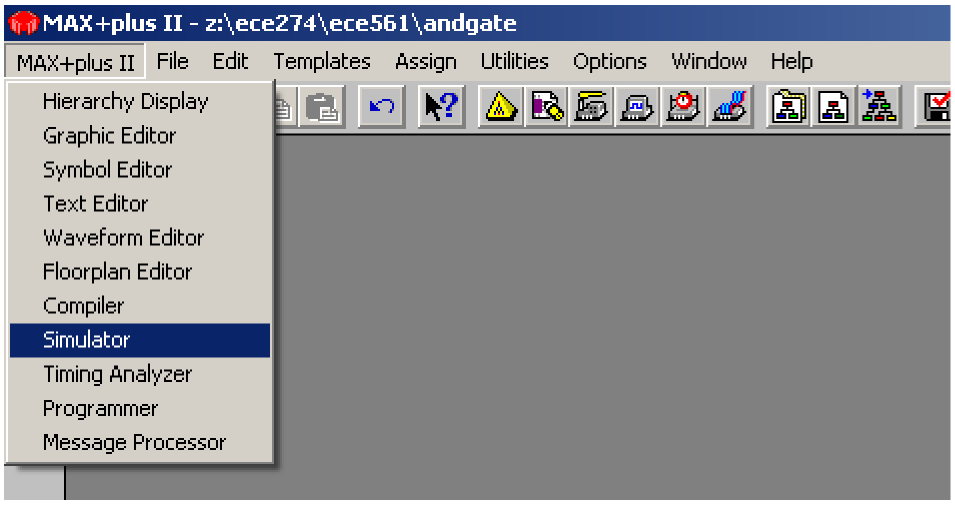

MAX+plus II File Edit View Node Assign Utilities Options Window Help
Name:
Value:
b
a
c
0.0ns
20.0ns
40.0ns
60.0ns
80.0ns
100.0ns
120.0ns
140.0ns

MAX+plus II - z:\ece274\ece561\andgate - [Untitled5 - Waveform Editor]

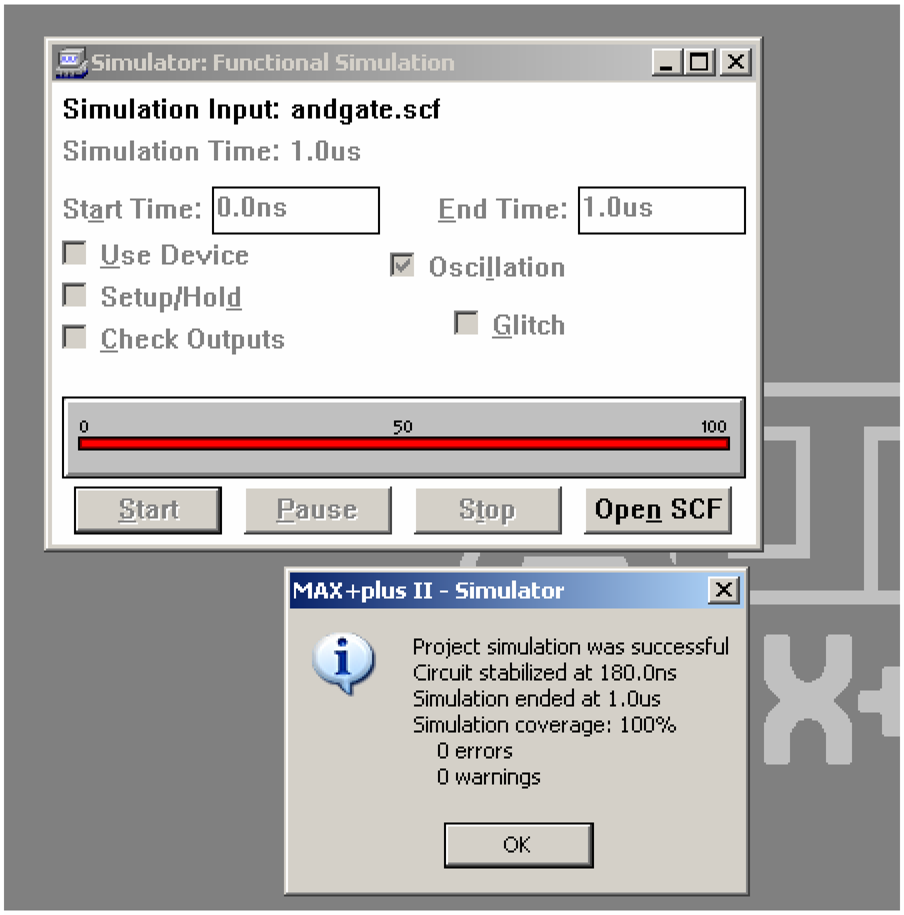
Ref: 0.0ns Time: 32.9ns Interval: 32.9ns

Name:	Value:
b	1
a	1
c	X

Choose Simulator



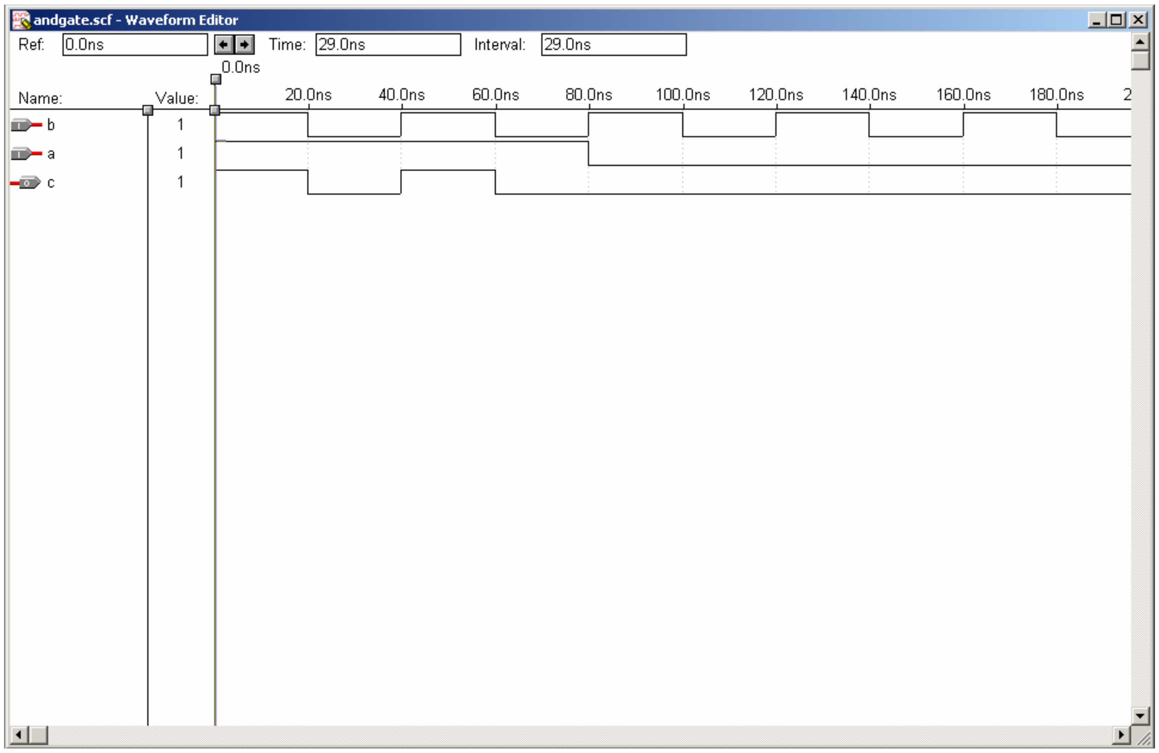
Set end time to 100ns and click start,



Simulation Input: andgate.scf
Simulation Time: 1.0us
Start Time: 0.0ns
End Time: 1.0us
Use Device
Setup/Hold
Check Outputs
Oscillation [✓]
Glitch

```
Project simulation was successful
Circuit stabilized at 180.0ns
Simulation ended at 1.0us
Simulation coverage: 100%
   0 errors
   0 warnings
```

Check the result waveform with the “and” truth table. And show the result to TA.



By doing the design of “andgate” you will become familiar with the software and Verilog language. Please design the OR and Inverter gate using Verilog by yourself. Compile the code and check the waveform. Show the result to TA.