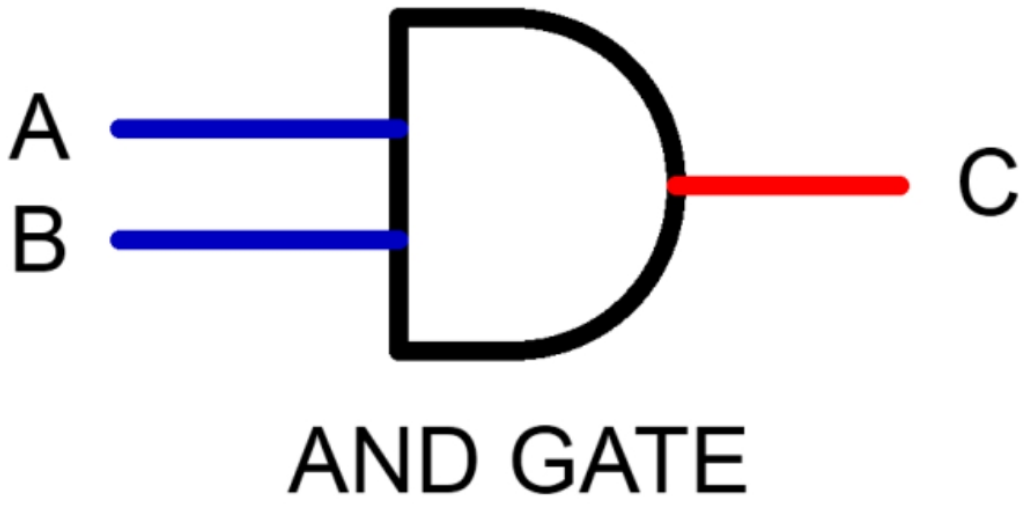
COL215 HW Assignment-0

AND GATE

Jahnabi Roy : 2022CS11094
Anamika : 2022CS11098

25th July, 2023



1 AIM

To design and implement And Gate which takes input from two switches and displays its output on an LED. .

2 MODULE IMPLEMENTED

2.1 AND Gate

We implemented this module to display the output of the boolean operation AND on the two inputs. The two inputs were taken from two slide switches on the Basys 3 board. We mapped V16 and V17 to the two inputs. The output was mapped to the LED at U16. The AND gate returns output bool(1) only when both the switches are ON (i.e. both the inputs are 1), hence the LED is turned ON only when both the inputs are 1. On all other inputs the LED stays OFF as the output returned by the AND gate is bool(0).

2.2 AND Gate Testbench

For the simulation of the AND gate, we made a test bench which gives different inputs at different time stamps and 'run simulation' command allows us to see the wave generated in the process. This is only for simulation process and observing the resultant waveform. We connect the test bench signals with the AND GATE source file with the input ports. The inputs are timed as :

'00' at 0 ns.

'10' at 20 ns.

'01' at 40 ns.

'11' at 60 ns.

The corresponding outputs received as seen in the waveform :

'0' at 0 ns.

'0' at 20 ns.

'0' at 40 ns.

'1' at 60 ns.

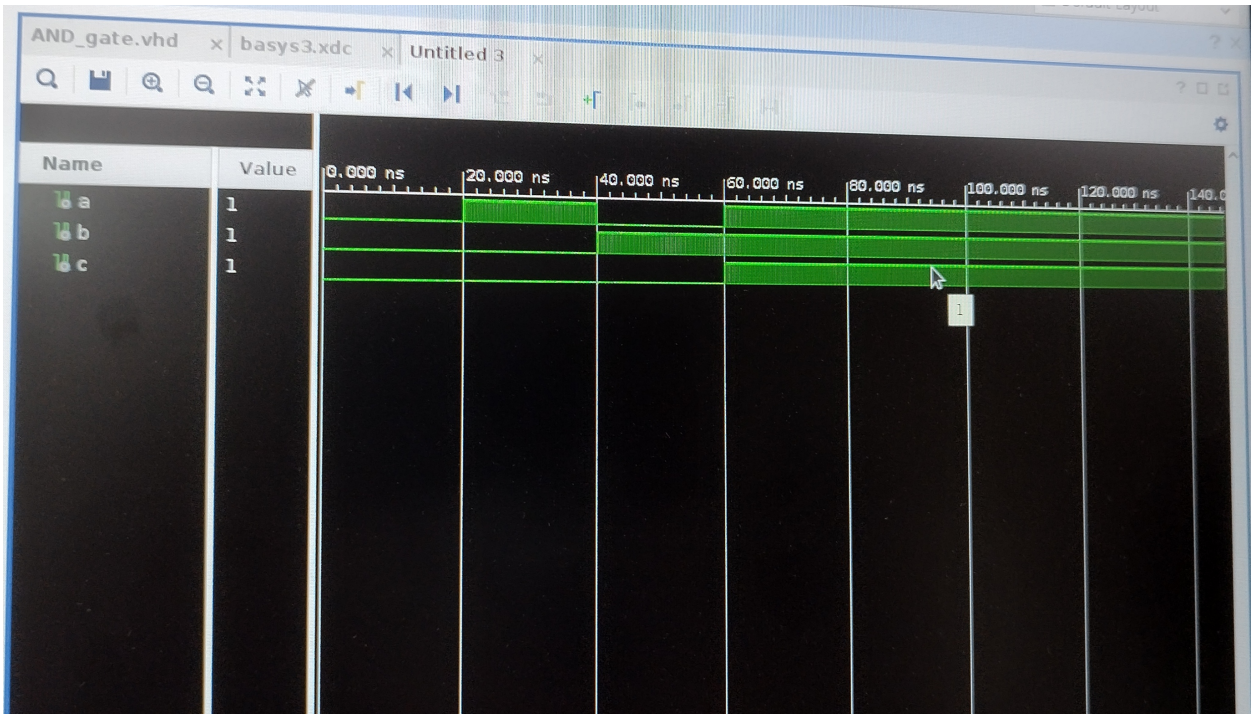
2.2.1 OUR APPROACH

- Created the Truth Table for AND Gate with 2 inputs, namely, a and b.
- Then referring to the assignment-0 pdf given, we wrote the VHDL code for AND Gate. Our inputs were 'a' and 'b' and output was 'c'.
- Then for simulation, we created the test-bench according to the pdf given.
- After running simulation, we disabled the test-bench file and added constraints to `basys3.xdc`, mapping input 'a' to port V17 and input 'b' to V16 and the output is mapped to LED U16.
- We then ran synthesis and then the output of the synthesis was used to run implementation.
- The output of the implementation was used to generate the bitstream.
  We then demonstrated the functioning to our TA.

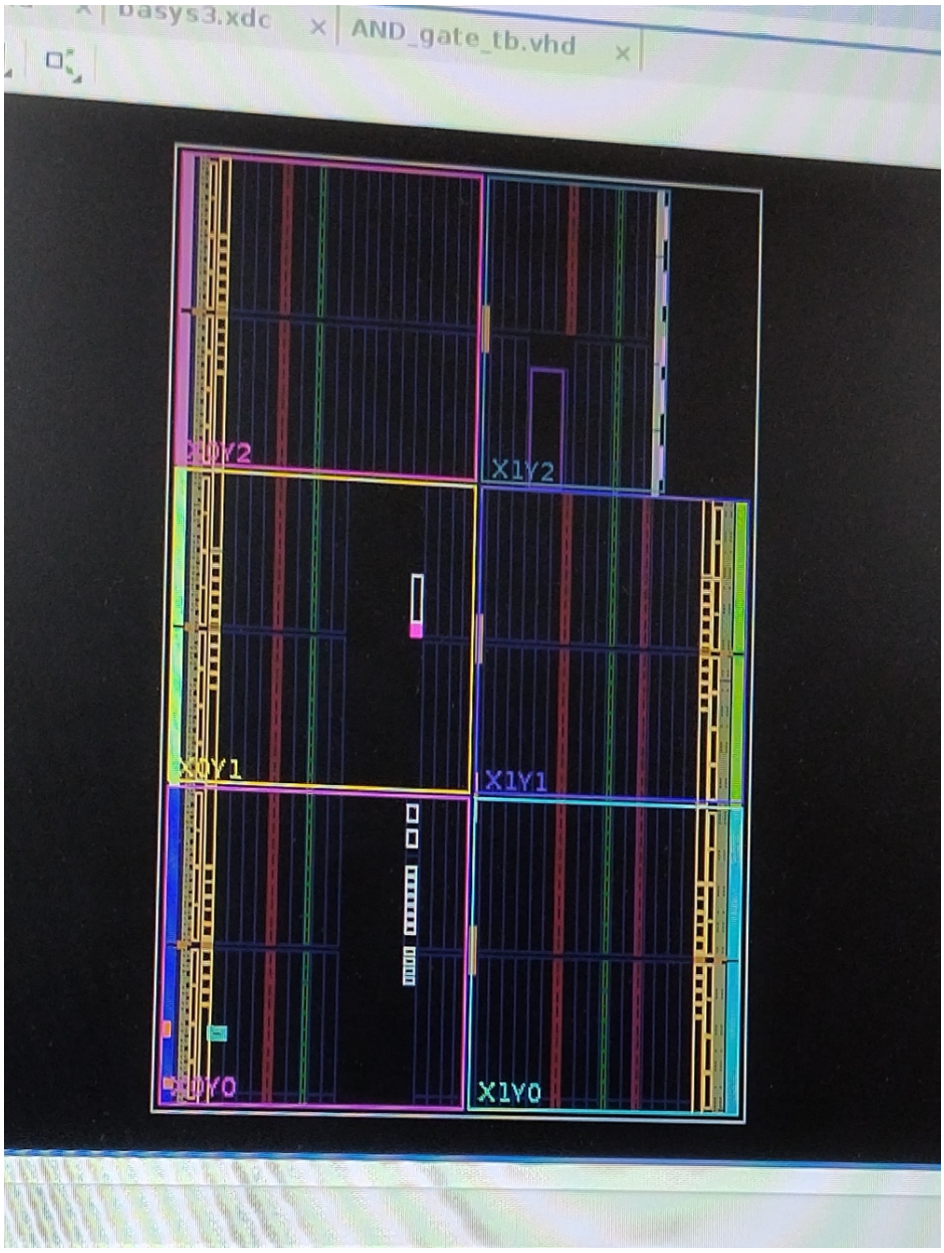
Truth Table for inputs (a,b) and corresponding output 'c' for AND Gate.

a	*b*	*c*
0	0	0
1	0	0
0	1	0
1	1	1

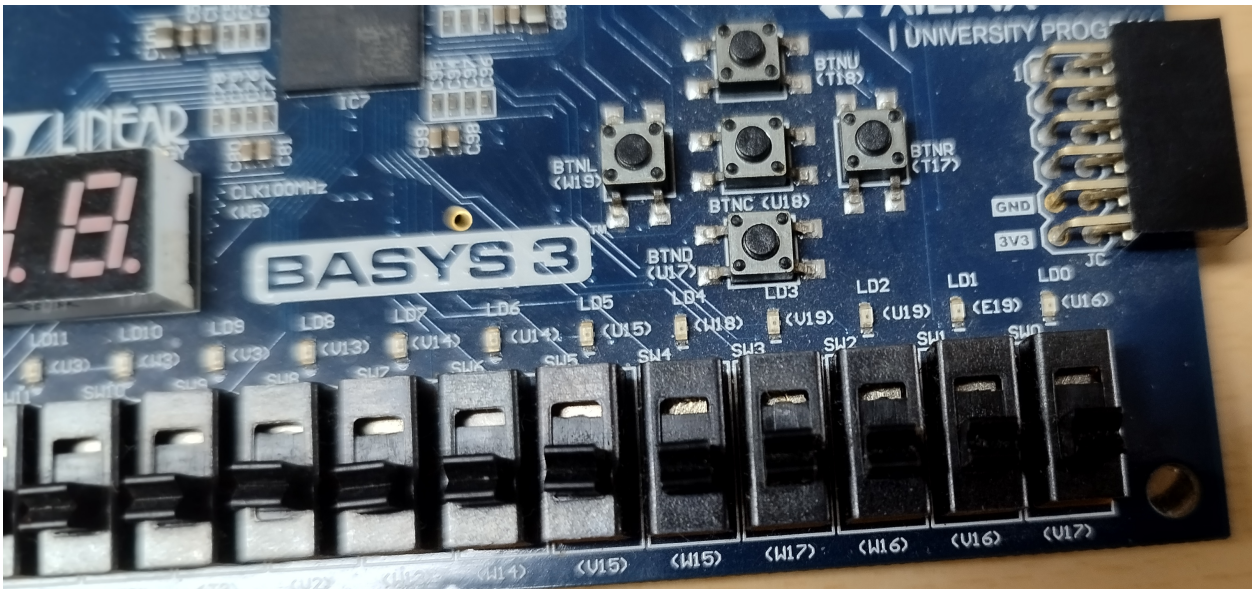
3 OUTPUT DISPLAY AND WAVEFORM



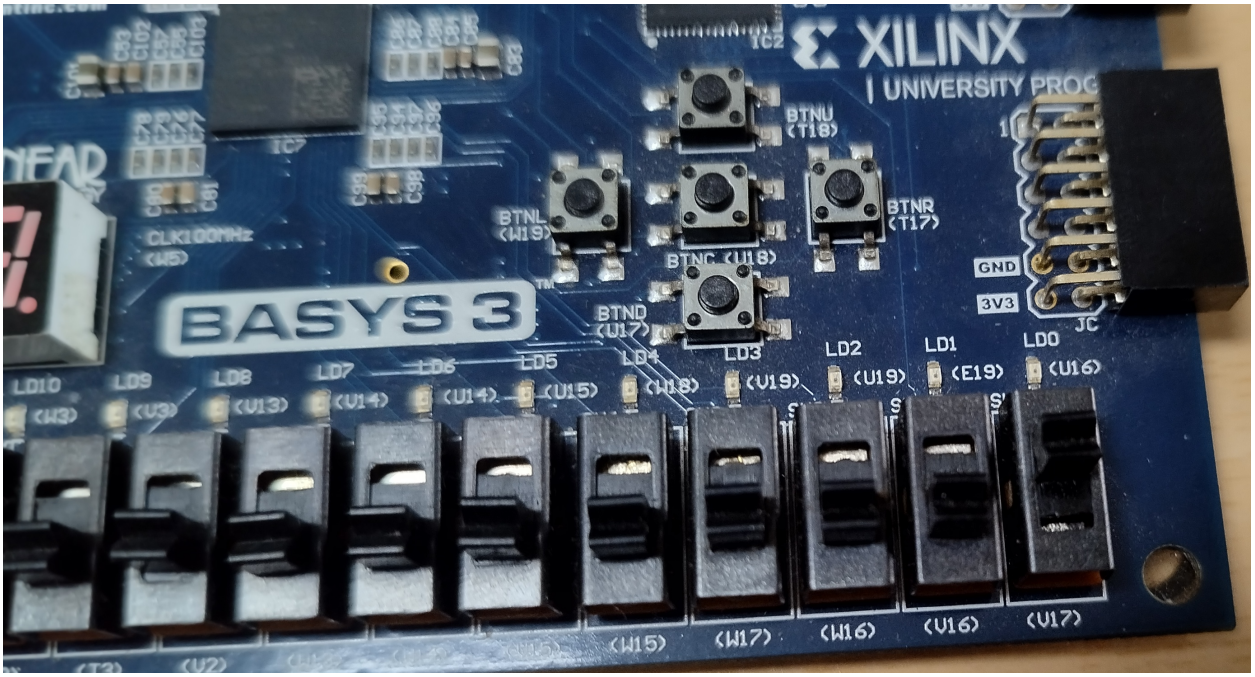
Waveform obtained by 'RUN SIMULATION'



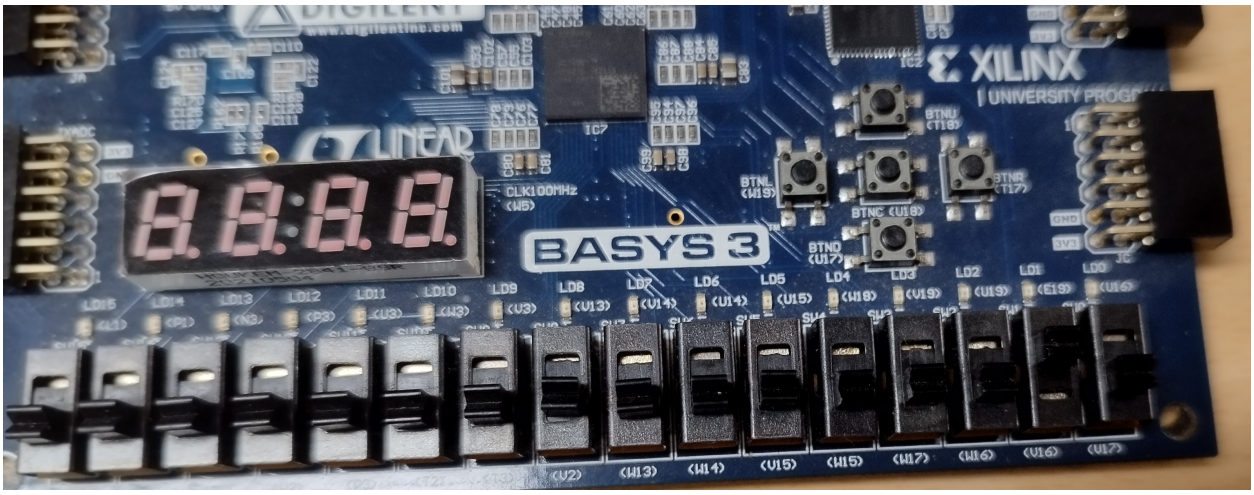
Implemented Design of AND Gate



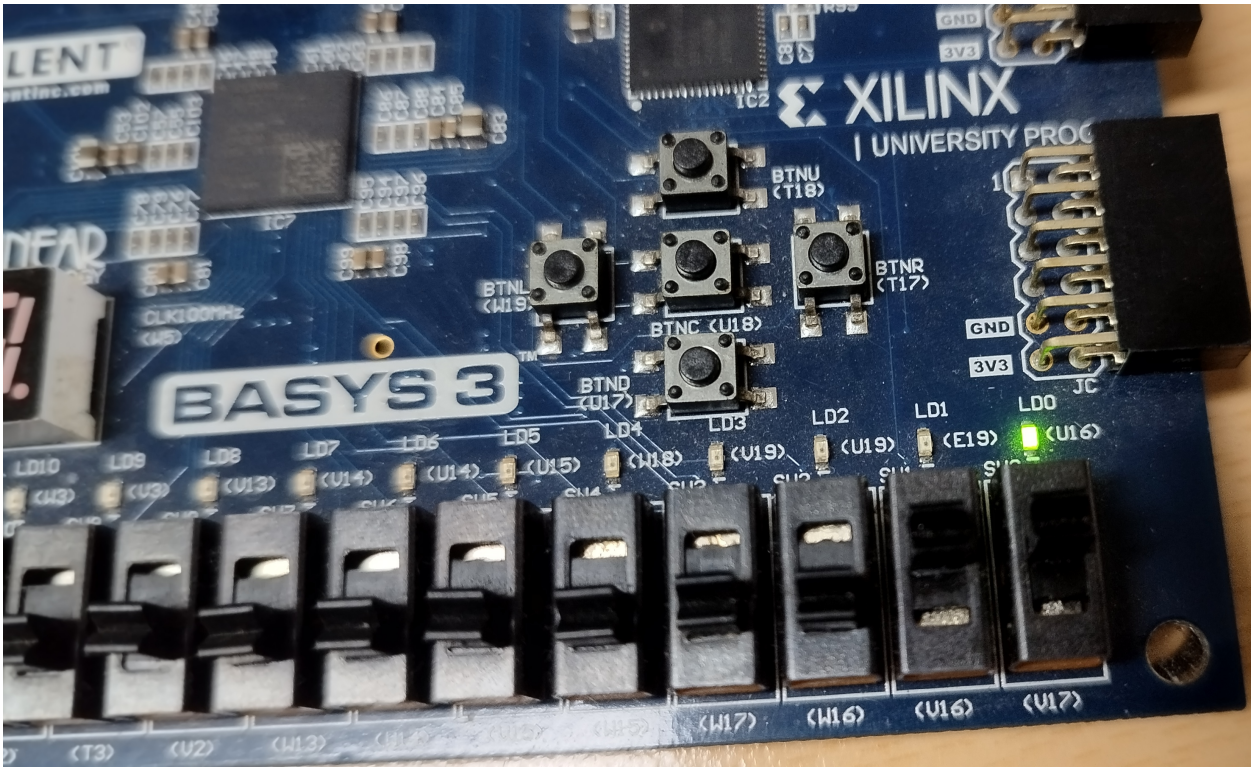
INPUT : '00', OUTPUT : '0'



INPUT : '01', OUTPUT : '0'

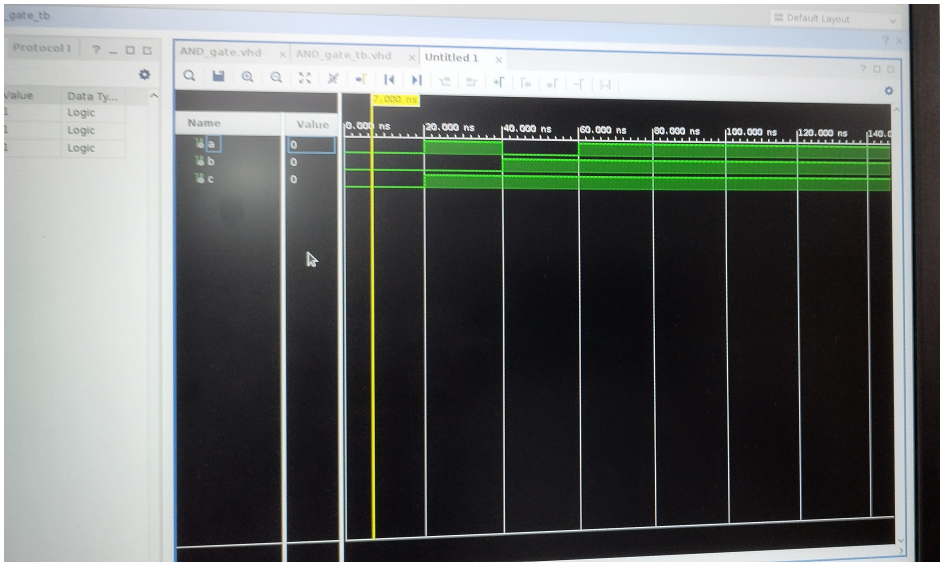


INPUT : '10', OUTPUT : '0'

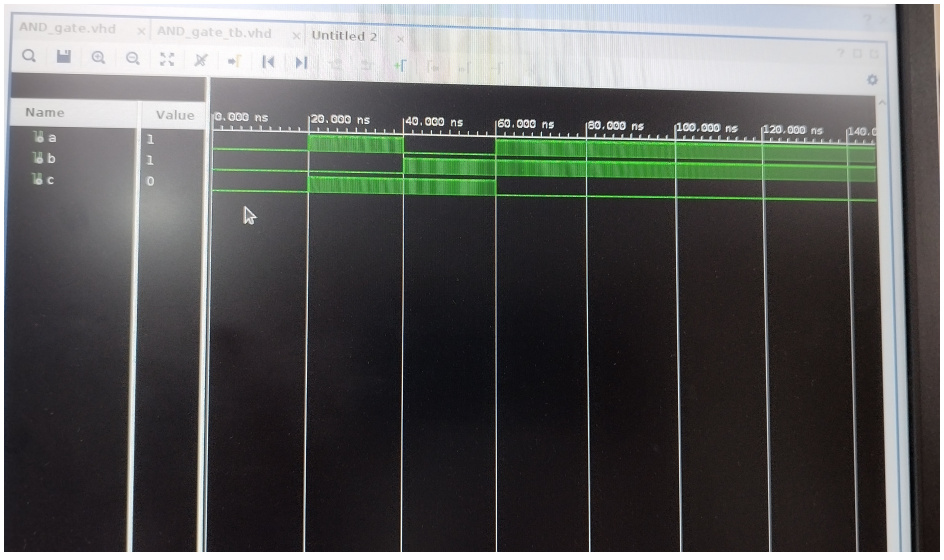


INPUT : '11', OUTPUT : '1'

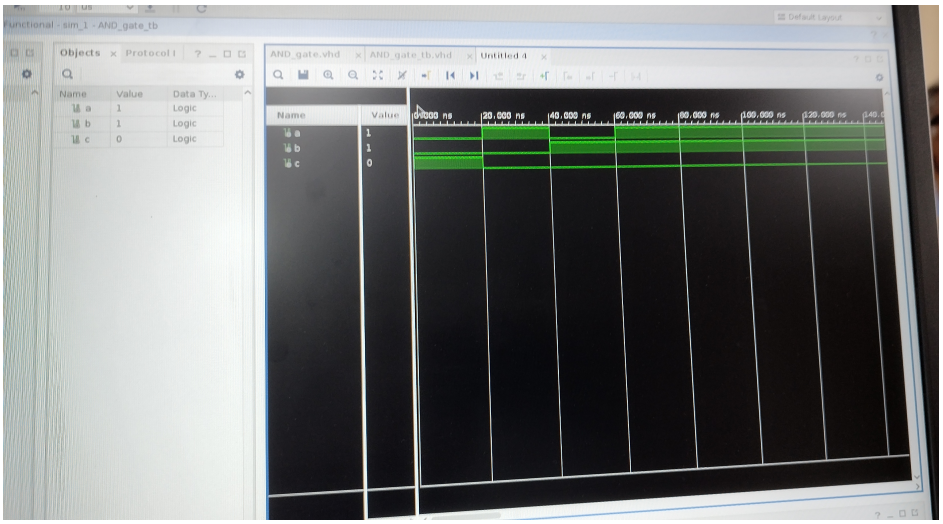
4 SIMULATION OF OTHER GATES



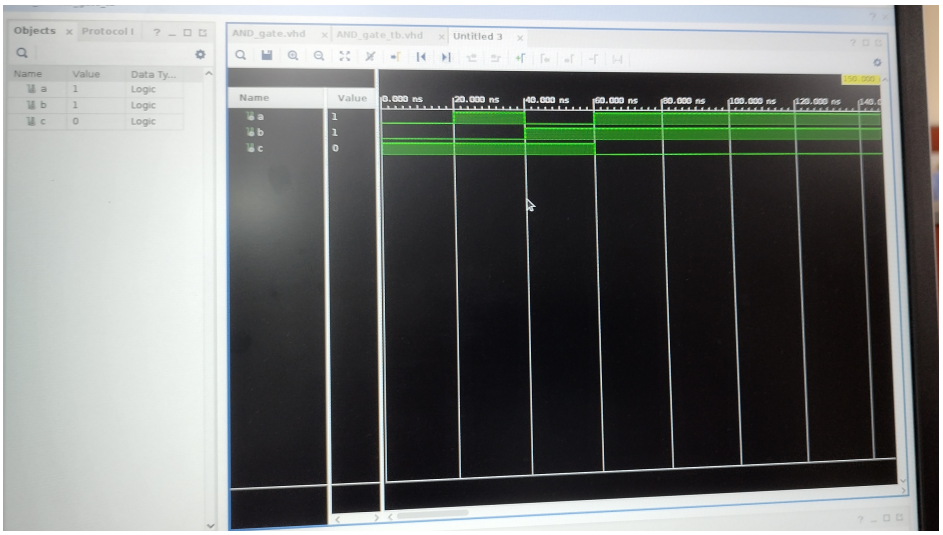
OR GATE WAVEFORM



XOR GATE WAVEFORM

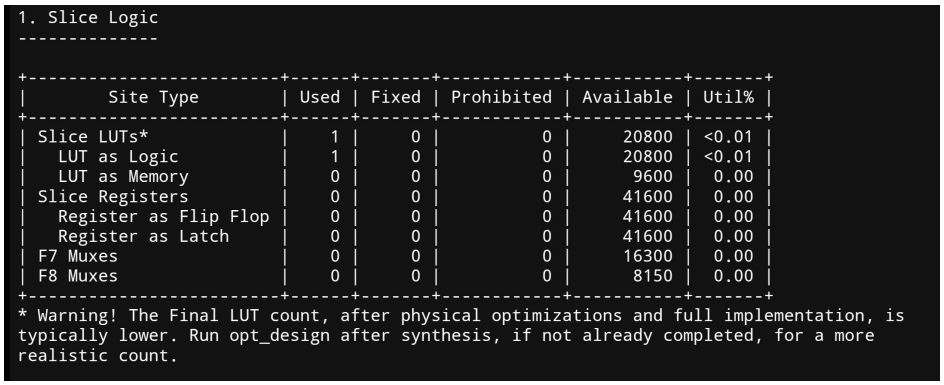


NOR GATE WAVEFORM



NAND GATE WAVEFORM

5 ATTACHMENTS AND REFERENCES



1. Slice Logic

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	1	0	0	20800	<0.01
LUT as Logic	1	0	0	20800	<0.01
LUT as Memory	0	0	0	9600	0.00
Slice Registers	0	0	0	41600	0.00
Register as Flip Flop	0	0	0	41600	0.00
Register as Latch	0	0	0	41600	0.00
F7 Muxes	0	0	0	16300	0.00
F8 Muxes	0	0	0	8150	0.00

* Warning! The Final LUT count, after physical optimizations and full implementation, is typically lower. Run opt_design after synthesis, if not already completed, for a more realistic count.

LUTs

The submission of the assignment consists of six files - the constraint file `basys3.xdc`, the VHDL code file `AND_gate.vhd`, the VHDL code for test bench `AND_gate_tb.vhd`, the `AND_gate.bit` file generated after running synthesis and implementing the synthesized design, `AND_gate_utilization_synth.rpt` where we can see the LUTs utilised, and the `AND_gate.vds` file.

For reference, we used the references given to us in the assignment.