# Introduction to Logic Design
# EEF205E

**Homework 1**

**Rüzgar Erik**
**040240783**

Istanbul Technical University
Faculty of Electrical and Electronics Engineering

# Part 1

1. **Convert the hexadecimal number $64CD_{16}$ to binary, to octal, and decimal.**
   **Solution:**

   **a.** to binary

   $$6_{16} = 0110_2$$
   $$4_{16} = 0100_2$$
   $$C_{16} = 1100_2$$
   $$D_{16} = 1101_2$$

   So, $64CD_{16} = 0110\ 0100\ 1100\ 1101_2$.

   **b.** to octal
   To convert to octal the binary number is divided into 3 digit groups from the LSB.

   $$000 \parallel 110 \parallel 010 \parallel 011 \parallel 001 \parallel 101$$

   $$000_2 = 0_8$$
   $$110_2 = 6_8$$
   $$010_2 = 2_8$$
   $$011_2 = 3_8$$
   $$001_2 = 1_8$$
   $$101_2 = 5_8$$

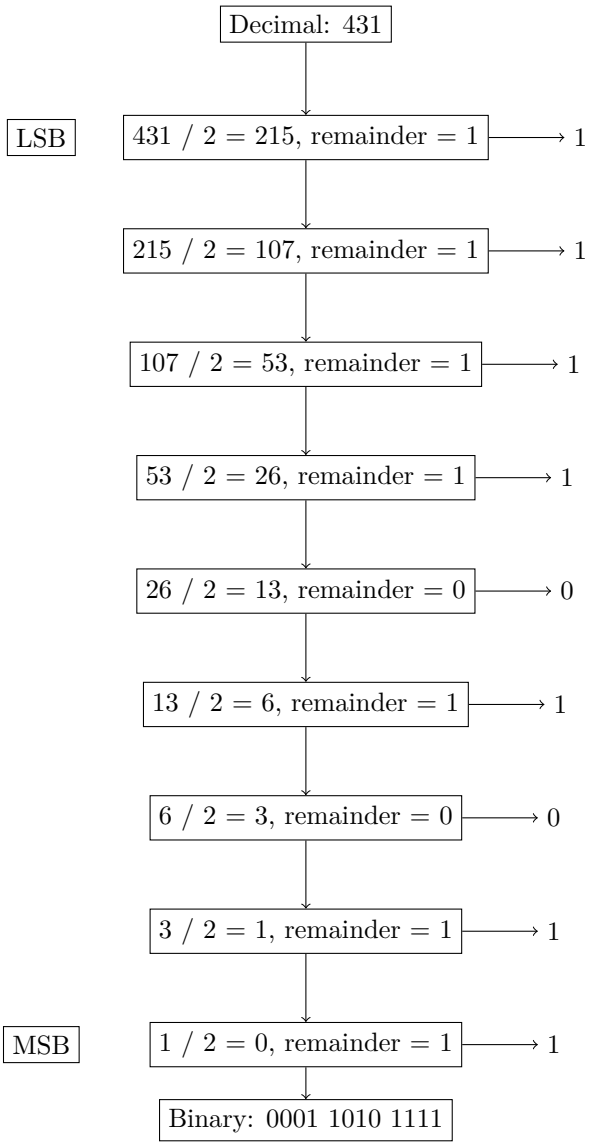
   So, $64CD_{16} = 62315_8$.

   **c.** to decimal

   $$\begin{aligned} C_{16} &= 12_{10} \\ D_{16} &= 13_{10} \\ 64CD_{16} &= (6 \times 16^3) + (4 \times 16^2) + (12 \times 16^1) + (13 \times 16^0) \\ &= (6 \times 4096) + (4 \times 256) + (12 \times 16) + (13 \times 1) \\ &= 24576 + 1024 + 192 + 13 \\ &= 25805 \end{aligned}$$

**2. Convert the decimal number 431 to binary, hexadecimal and octal**
**Solution:**

**a.** To binary




The binary representation of $431_{10}$ is $0001\ 1010\ 1111_2$

**b.** To hexadecimal

The binary representation of the number was calculated in the previous question as $0001\ 1010\ 1111_2$ to convert it to hexadecimal, the binary number is divided into groups that contains 4 digits starting from the LSB.

$$0001 \parallel 1010 \parallel 1111$$

The hexadecimal representations of binary numbers are calculated below:

$$0001_2 = 1_{16}$$
$$1010_2 = A_{16}$$
$$1111_2 = F_{16}$$

So, the hexadecimal representation is :

$$431_{10} = 0001\ 1010\ 1111_2 = 1AF_{16}$$

**c.** To octal

The binary representation of the number was calculated previously to convert the binary number to octal the digits are grouped into 3 digit groups starting from the LSB.

$$000 \parallel 110 \parallel 101 \parallel 111$$

The octal counterparts of the binary numbers are calculated below:

$$000_2 = 0_8$$
$$110_2 = 6_8$$
$$101_2 = 5_8$$
$$111_2 = 7_8$$

The leading zeros are omitted. So, the octal representation is calculated as:

$$431_{10} = 0001\ 1010\ 1111_2 = 657_8$$

**3. Express the following numbers in decimal:**

(a) $(10110.0101)_2$

(b) $(16.5)_{16}$

(c) $(26.24)_8$

(d) $(\text{DADA.B})_{16}$

(e) $(1010.1101)_2$

**Solutions:**

**a.** $(10110.0101)_2$ to decimal

$$\begin{aligned}(10110.0101)_2 &= (1\times 2^4)+(0\times 2^3)+(1\times 2^2)+(1\times 2^1)+(0\times 2^0)+(0\times 2^{-1})+(1\times 2^{-2})+(0\times 2^{-3})+(1\times 2^{-4})\\ &= 16+0+4+2+0+0+0.25+0+0.0625\\ &= 22.3125\end{aligned}$$

**b.** $(16.5)_{16}$ to decimal

$$\begin{aligned}(16.5)_{16} &= (1\times 16^1)+(6\times 16^0)+\left(5\times 16^{-1}\right)\\ &= 16+6+\frac{5}{16}\\ &= 16+6+0.3125\\ &= 22.3125\end{aligned}$$

**c.** $(26.24)_8$ to decimal

$$\begin{aligned}(26.24)_8 &= \left(2\times 8^1\right)+\left(6\times 8^0\right)+\left(2\times 8^{-1}\right)+\left(4\times 8^{-2}\right)\\ &= 16+6+0.25+0.0625\\ &= 22.3125\end{aligned}$$

**d.** $(\text{DADA.B})_{16}$ to decimal

To convert the hexadecimal number to decimal first the values of hexadecimal letters must be converted to decimal.

$$\begin{aligned}\text{A}_{16} &= 10_{10}\\ \text{B}_{16} &= 11_{10}\\ \text{D}_{16} &= 13_{10}\end{aligned}$$

$$\begin{aligned}\text{DADA.B}_{16} &= (13\times 16^3)+(10\times 16^2)+(13\times 16^1)+(10\times 16^0)+(11\times 16^{-1})\\ &= (13\times 4096)+(10\times 256)+(13\times 16)+(10\times 1)+(11\times 0.0625)\\ &= 53248+2560+208+10+0.6875\\ &= 56026.6875\end{aligned}$$

**e.** $(1010.1101)_2$ to decimal

$$\begin{aligned}(1010.1101)_2 &= (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) + (1 \times 2^{-1}) + (1 \times 2^{-2}) + (0 \times 2^{-3}) + (1 \times 2^{-4}) \\ &= 8 + 0 + 2 + 0 + 0.5 + 0.25 + 0 + 0.0625 \\ &= 10.8125\end{aligned}$$

**4. Convert the following binary numbers to hexadecimal and to decimal:**

**a.** 1.10010

**b.** 110.010. Explain why the decimal answer in (b) is 4 times that in (a).

**Solutions:**

**a.** 1.10010 to hexadecimal and decimal

**i.** To hexadecimal

To convert the binary number to hexadecimal the number is grouped into 4 digit groups starting from the decimal point.

$$0001 \parallel . \parallel 1001 \parallel 0000$$

The hexadecimal representations are calculated as follows:

$$\begin{aligned}0001_2 &= 1_{16} \\ 1001_2 &= 9_{16} \\ 0000_2 &= 0_{16}\end{aligned}$$

So the resukting hexadecimal number is:

$$(1.10010)_2 = (1.9)_{16}$$

**ii.** To decimal

$$\begin{aligned}(1.10010)_2 &= (1 \times 2^0) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (0 \times 2^{-3}) + (1 \times 2^{-4}) + (0 \times 2^{-5}) \\ &= 1 + 0.5 + 0 + 0 + 0.0625 + 0 \\ &= 1.5625\end{aligned}$$

**b.** 110.010 to hexadecimal and decimal

**i.** To hexadecimal

To convert the binary number to hexadecimal the number is grouped into 4 digit groups starting from the decimal point.

$$0110 \parallel . \parallel 0100$$

The hexadecimal representations are calculated as follows:

$$0110_2 = 6_{16}$$
$$0100_2 = 4_{16}$$

So the resukting hexadecimal number is:

$$(110.010)_2 = (6.4)_{16}$$

**ii.** To decimal

$$\begin{aligned}(110.010)_2 &= (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) + (0 \times 2^{-1}) + (1 \times 2^{-2}) + (0 \times 2^{-3}) \\ &= 4 + 2 + 0 + 0 + 0.25 + 0 \\ &= 6.25\end{aligned}$$

The decimal answer in (b) is 4 times that in (a) because the binary number in (b) is left shifted by 2 bits with moving the decimal point right by 2 bits. If the decimal point is moved right by n bits the resultant number is multiplied by $2^n$. If we consider a decimal number $AB.CD_2$ the decimal equivalent is calculated as:

$$\text{AB.CD}_2 = (A \times 2^1) + (B \times 2^0) + (C \times 2^{-1}) + (D \times 2^{-2})$$

If we left shift the number by 2 bits the number becomes ABCD and the decimal equivalent is calculated as:

$$\text{ABCD}_2 = (A \times 2^3) + (B \times 2^2) + (C \times 2^1) + (D \times 2^0)$$

By factoring out the resultant equation by $2^2$ we get:

$$\text{ABCD}_2 = 2^2 \times \big((A \times 2^1) + (B \times 2^0) + (C \times 2^{-1}) + (D \times 2^{-2})\big)$$

So, the decimal equivalent of the left shifted number is 4 times the decimal equivalent of the original number. Also it can be seen from the powers of 2 is increased by 2 in the left shifted number.

**5. Draw truth table of the Boolean function** $f(x, y, z) = x'y + z$

**Solution:**

Table 1: Truth Table for the Function $f(x, y, z) = x'y + z$

| $x$ | $y$ | $z$ | $f(x, y, z)$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

# Part 2

As per instructions, Xilinx ISE 14.7 webpack was installed using the Windows 10 version, the program runs on Oracle Virtual Box using Oracle Linux to mitigate compatibility issues. I have previously attempted to install the software to my computer runnnig Fedora 40 but the signal simulator tool Xilinx ISim was not running properly.

## Importing Files

From Ninova "`AND_gate.v`" and "`AND_gate_tb.vhd`" files were downloaded and imported into the project navigator using add source command. By further inspection the file "`AND_gate.v`" was found to be a Verilog file and the file "`AND_gate_tb.vhd`" was found to be a VHDL file for testing the Verilog file.

### AND_gate.v

The Verilog file "`AND_gate.v`" contains the following code:

Listing 1: AND_gate.v

```
1 module AND_gate(a,b,c);
2 input a,b;
3 output c;
4
5 assign c = a & b;
6
7 endmodule
```

The verilog file describes an AND gate with two inputs and one output. The module is named as `AND_gate` in the first line starting the module via the module keyword. The inputs are declared in line 2 as `a` and `b` and the output is declared in line 3 as `c`. In the line 5 the assign statement is used for assigning the output `c` to the result of and operation between the inputs. The final argument `endmodule` is used for declaring the end of the module.

### AND_gate_tb.vhd

The VHDL file "`AND_gate_tb.vhd`" contains the following code:

Listing 2: AND_gate_tb.vhd

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3
4  ENTITY AND_gate_tb IS
5  END AND_gate_tb;
6
7  ARCHITECTURE behavior OF AND_gate_tb IS
8
9      -- Component Declaration for the Unit Under Test (UUT)
10
11     COMPONENT AND_gate
12     PORT(
13          IN1 : IN  std_logic;
14          IN2 : IN  std_logic;
15          OUT1 : OUT  std_logic
16         );
17     END COMPONENT;
18
19
20    --Inputs
21    signal IN1 : std_logic := '0';
22    signal IN2 : std_logic := '0';
23
```

```
24    --Outputs
25     signal OUT1 : std_logic;
26
27  BEGIN
28
29    -- Instantiate the Unit Under Test (UUT)
30     uut: AND_gate PORT MAP (
31            IN1 => IN1,
32            IN2 => IN2,
33            OUT1 => OUT1
34          );
35
36       -- insert stimulus here
37       process
38       begin
39       wait for 10 ns;
40
41       IN1 <= '1';
42       IN2 <= '0';
43
44       wait for 10 ns;
45
46       IN1 <= '0';
47       IN2 <= '1';
48
49      wait for 10 ns;
50
51       IN1 <= '1';
52       IN2 <= '1';
53
54       wait for 10 ns;
55
56       IN1 <= '0';
57       IN2 <= '0';
58
59        wait;
60     end process;
61
62  END;
```

This file is a testbench described in VHDL which is used for testing and verifying the functionality of the Verilog file.

In the first line the IEEE library was imported and in the second line `STD_LOGIC_1164` package was made available. The `STD_LOGIC_1164` package contains the IEEE standard 1164 which includes the definitions of logic values used for signals.

In the 4th and 5th lines an ENTITY named `AND_gate_tb` is declared without any ports.

In the 7th line the ARCHITECTURE of the ENTITY is declared as `behavioral`.

Between the lines 11 and 17 the `AND_gate` component is declared with the ports `IN1`, `IN2` and `OUT1`.
The lines 21 and 22 initialize the to the value of `0`.

Between the lines 30 and 34 the unit under test is is instantiated with the name `uut` and the ports are connected to the signals `IN1` and `IN2` and the output is connected to the signal `OUT1`.

Process and begin statements are used for beginning a process block. A process block in VHDL is a sequential block that executes when the signals change.

The process simply begins by waiting 10 nanoseconds.

The following lines assign `IN1` and `IN2` 1 and 0 respectively.

After waiting for 10 nanoseconds again the values of `IN1` and `IN2` are changed to 0 and 1 respectively.

After another 10 nanoseconds the values of `IN1` and `IN2` are changed to 1 and 1 respectively.

Finally waiting after 10 nanoseconds the values of `IN1` and `IN2` are changed to 0 and 0 respectively.

The wait at the and of the process waits indefinitely.

## The Error In Code

The error in code is that in the testbench the signals were defined as `IN1`, `IN2` and `OUT1` but in the Verilog module `AND_gate` the signals were defined as `a`, `b` and `c`. The signals in the testbench must be defined as the same as the signals in the module. PORT MAP in the VHDL testbench or the Verilog module input, output names can be changed to correct the issues. I have changed the Verilog code to use `IN1` and `IN2` instead of `a` and `b` and `OUT1` instead of `c`.

The corrected code can be seen below:

Listing 3: Corrected AND_gate.v

```
1 module AND_gate(IN1,IN2,OUT1);
2 input IN1,IN2;
3 output OUT1;
4
5 assign OUT1 = IN1 & IN2;
6
7 endmodule
```

## Viewing the RTL Schematic

RTL is an abbreviation for Register Transfer Level. An RTL schematic graphically represents the digital circuit.

Using the Xilinx ISE the RTL schematic of the Verilog module `AND_gate` was viewed. The RTL schematics with different settings are shown below.

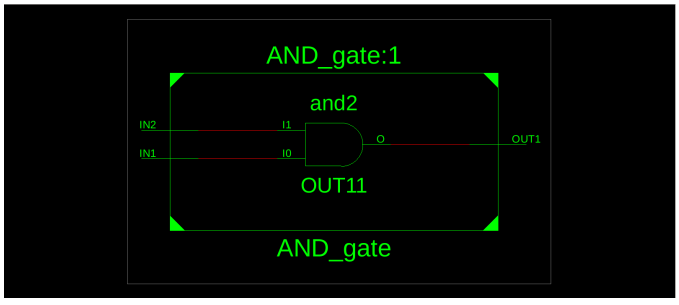



Figure 1: RTL Schematic of the Verilog Module AND_gate. Every option is selected from the visualisation menu.

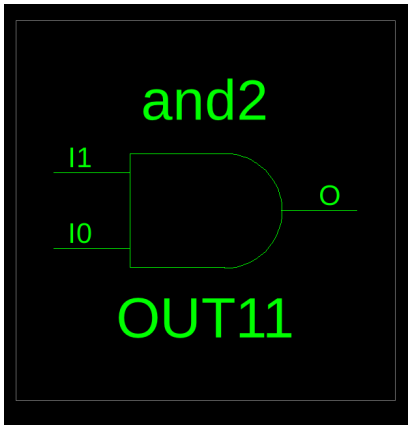



Figure 2: RTL Schematic of the Verilog Module AND_gate with only the option "AND_gate primitive" selected.

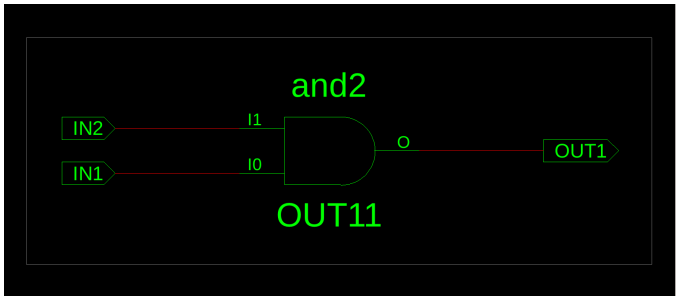



Figure 3: RTL Schematic of the Verilog Module AND_gate with "AND_gate with all signals" selected.

## Simulating the Design

The file `AND_gate_tb.vhd` was selected and "Simulate Behavioral Model" was run the results generated are shown below.

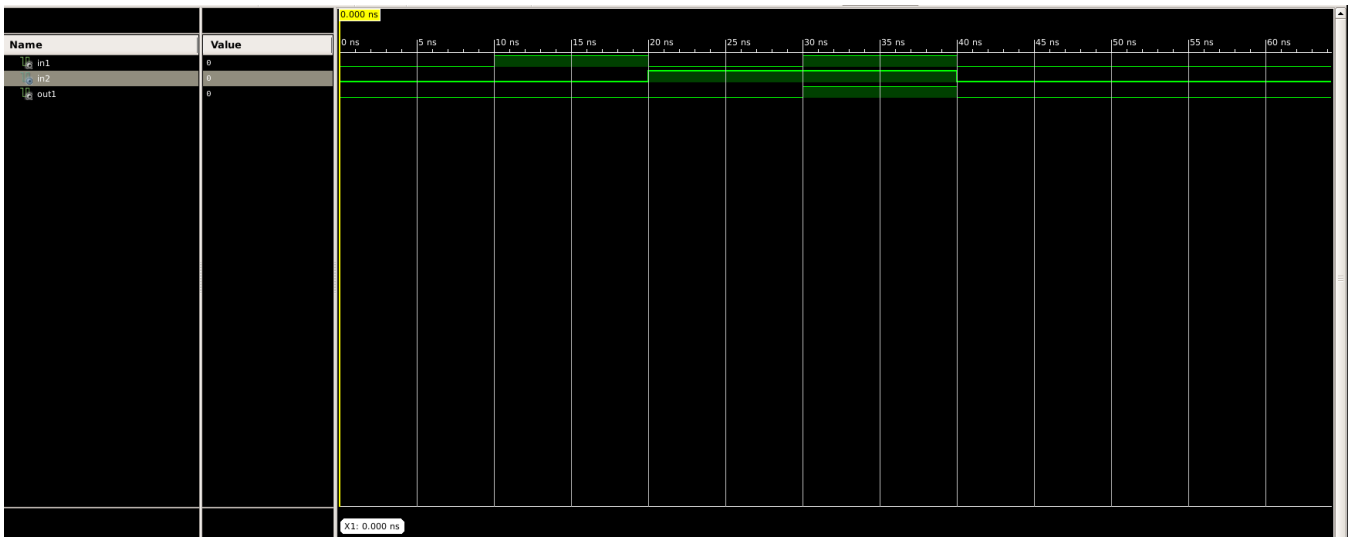



Figure 4: Simulation Results of the Testbench of the AND_gate Module at t=0.

In the VHDL testbench code the initialization of the signals `IN1` and `IN2` were set to 0. The expected output for an AND gate is 0 when both inputs are 0. The simulation between 0-10ns verifies this.

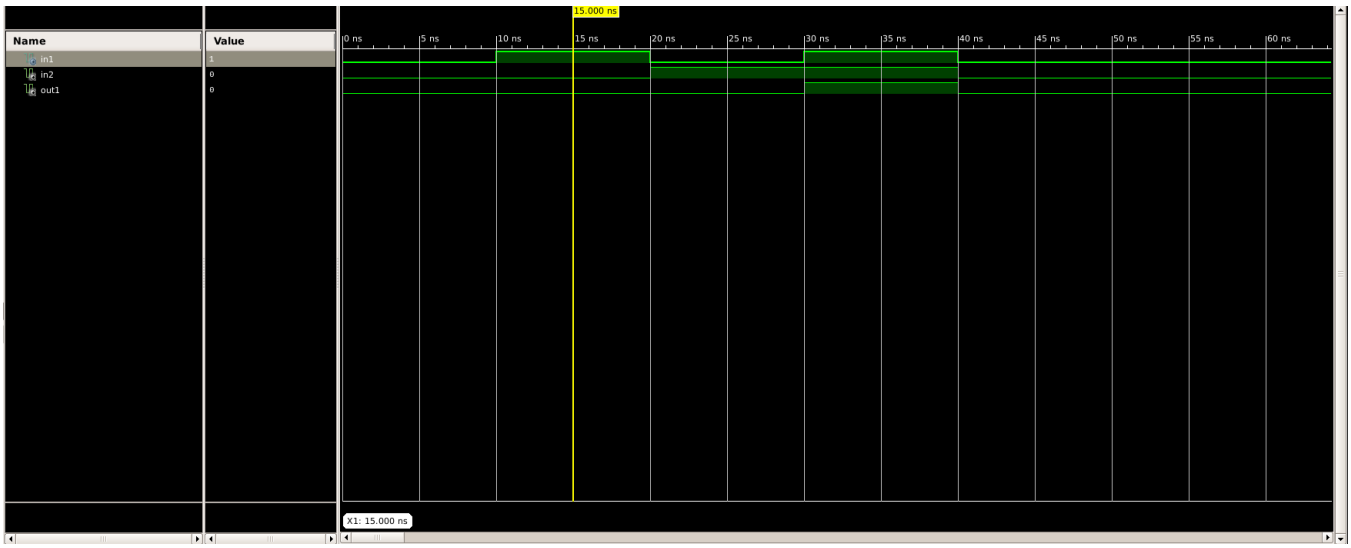



Figure 5: Simulation Results of the Testbench of the AND_gate Module at t=15ns.

In the VHDL testbench the code sets the signals `IN1` and `IN2` to 1 and 0 respectively. The expected output for an AND gate is 0 when one of the inputs is 0. The simulation between 10-20ns verifies this.

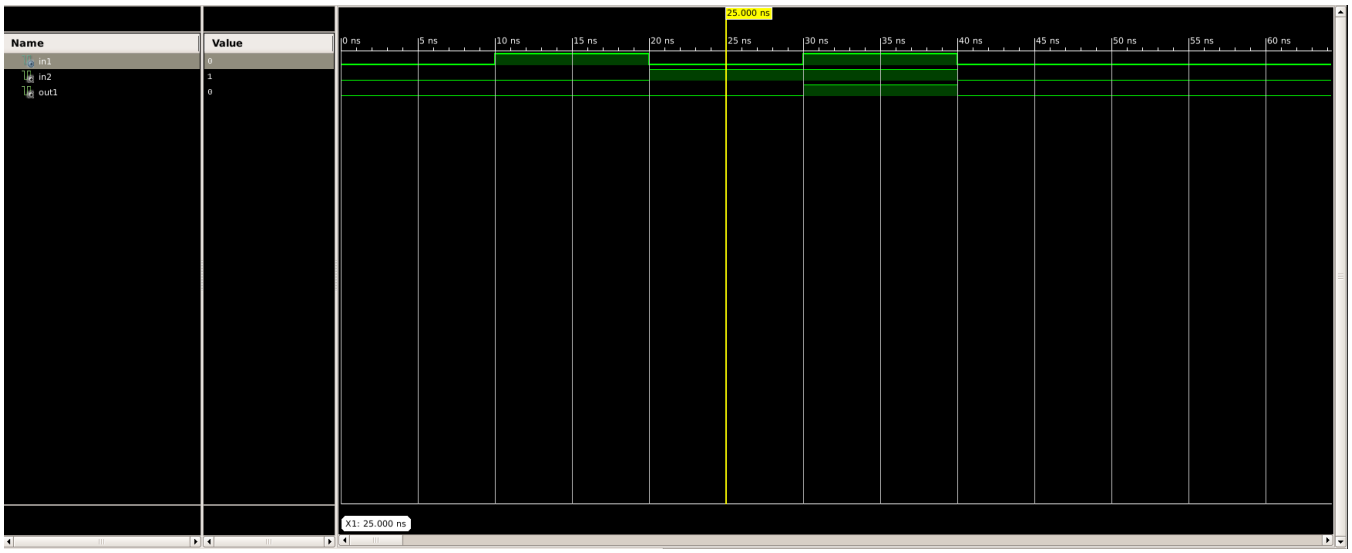



Figure 6: Simulation Results of the Testbench of the AND_gate Module at t=25ns.

In the VHDL testbench the code sets the signals `IN1` and `IN2` to 0 and 1 respectively. The expected output for an AND gate is 0 when one of the inputs is 0. The simulation between 20-30ns verifies this.


Figure 7: Simulation Results of the Testbench of the AND_gate Module at t=35ns.

In the VHDL testbench the code sets the signals `IN1` and `IN2` to 1. The expected output for an AND gate is 1 when two of the inputs are 1. The simulation between 30-40ns verifies this.

## Results

The simulation successfully verifies and tests the AND gate module described in the Verilog code the behavior of the AND gate is consistent with the expected behavior. The RTL schematics were viewed at different levels and included in the report also the simulation graph is included between 0-60ns.