Nemecxpetr / **Digital-electronics-1**

Code Issues Pull requests Actions Projects Wiki Security

main

Digital-electronics-1 / Labs / 01-gates / **README.md**

Nemecxpetr Update README.md History

1 contributor

Raw Blame

```
103 lines (69 sloc) | 3.18 KB
```

01 - Logic gates

Lab assignment

- ✓ 1. Submit the GitHub link to your Digital-electronics-1 repository.
- ✓ 2. Verification of De Morgan's laws of function f(c,b,a). Submit:

  - Listing of VHDL code design.vhd,
  - Screenshot with simulated time waveforms,
  - Link to your public EDA Playground example.

- ✓ 3. Verification of Distributive laws. Submit:

  - Listing of VHDL code design.vhd,
  - Screenshot with simulated time waveforms,
  - Link to your public EDA Playground example.

> ☐ Prepare all tasks in your README file Digital-electronics-1/Labs/01-gates/README.md, export/print it to PDF, use BUT e-learning web page and

> submit a single PDF file. The deadline for submitting the task is the day before the next laboratory exercise.

1 Submitting GitHub repository

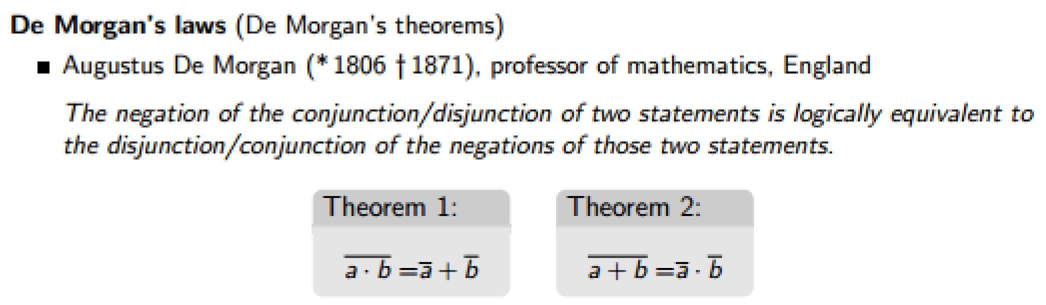
My GitHub repository.

2 Verification of De Morgan's laws of function f(c,b,a)

Parts:

- Listing of VHDL code `design.vhd` ,
- Screenshot with simulated time waveforms,
- Link to your public EDA Playground example.

  Function is defined as follows. We also derived the only NAND and NOR form of function using *De Morgan's laws*.



De Morgan's laws (De Morgan's theorems)

- Augustus De Morgan (*1806 †1871), professor of mathematics, England

The negation of the conjunction/disjunction of two statements is logically equivalent to the disjunction/conjunction of the negations of those two statements.

Theorem 1: $\overline{a\cdot b} = \overline{a} + \overline{b}$

Theorem 2: $\overline{a+b} = \overline{a}\cdot\overline{b}$

Derived functions:

$$f(c,b,a) = \overline{b}\,a + \overline{c}\,\overline{b}$$

$$f(c,b,a)_{\mathrm{NAND}} = \overline{\overline{\overline{b}\,a}\;\overline{\overline{c}\,\overline{b}}}$$

$$f(c,b,a)_{\mathrm{NOR}} = \overline{\overline{\overline{\overline{b}}+\overline{a}}+\overline{\overline{\overline{c}}+\overline{\overline{b}}}} = \overline{b+\overline{a}} + \overline{c+b}$$

> Equations were generated by Online LaTeX Equation Editor.

Code from EDU Playground:

> Excerpt from `design.vhd` :

```
architecture dataflow of gates is
begin
```

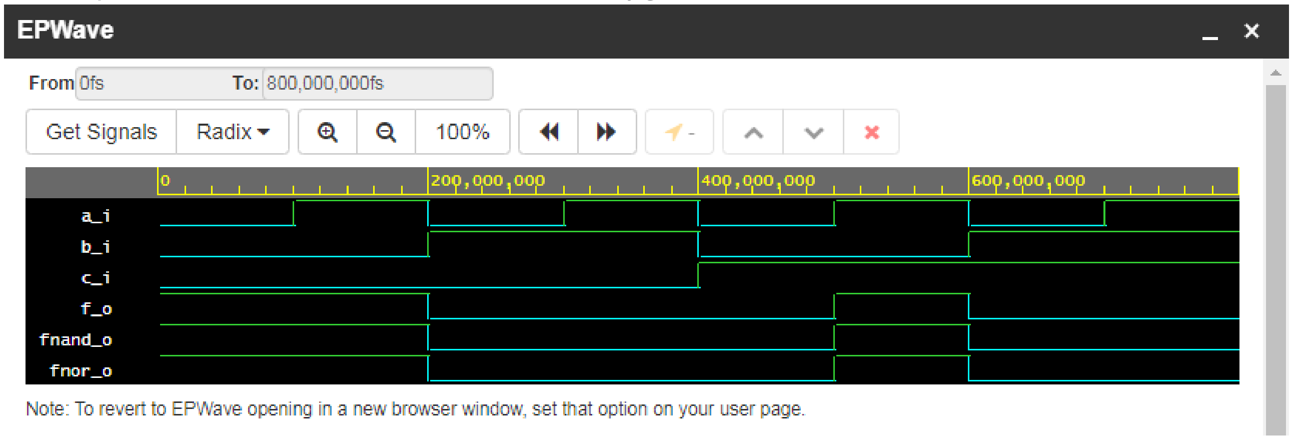
```
        f_o  <= (not b_i and a_i) or (not b_i and not c_i);
        fnand_o <= not(not((not b_i) and a_i) and not(not c_i and not b_i));
        fnor_o <= not(b_i or not a_i) or not(c_i or b_i);

    end architecture dataflow;
```

Here are the simulation results in table:

c	b	a	f(c,b,a)	f(c,b,a)nand	f(c,b,a)nor
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	0	0

And a photo of simulation results in EDA Playground:



Note: To revert to EPWave opening in a new browser window, set that option on your user page.

We succesfully verified De Morgan's laws with function f(c,b,a).

3 Verification of Distributive laws

Parts:

- Listing of VHDL code `design.vhd`,
- Screenshot with simulated time waveforms,

- Link to your public EDA Playground example.

Distributive laws

$$x \cdot y + x \cdot z = x \cdot (y + z)$$
$$(x + y) \cdot (x + z) = x + (y \cdot z)$$

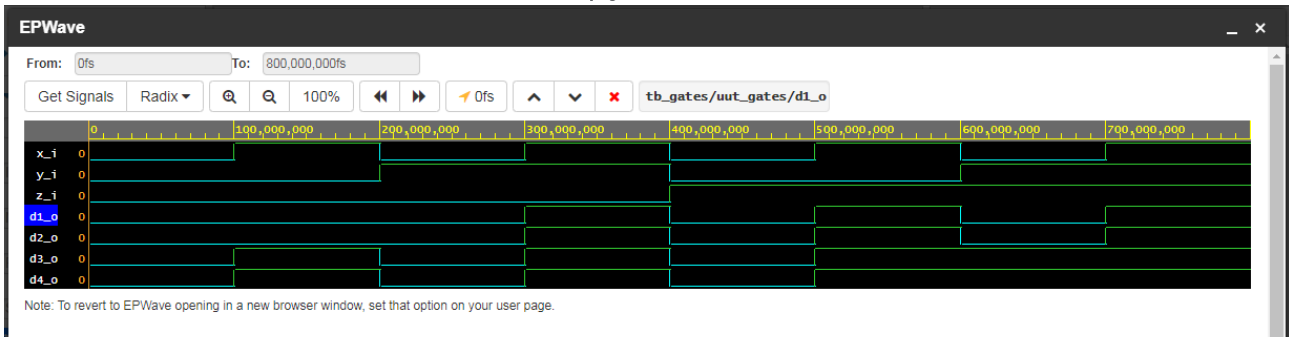
Code from EDU Playground

> Excerpt from `design.vhd`:

```
architecture dataflow of gates is
begin
    d1_o  <= (x_i and y_i) or (x_i and z_i);
    d2_o  <= x_i and (y_i or z_i);
    d3_o  <= (x_i or y_i) and (x_i or z_i);
    d4_o  <= x_i or (y_i and z_i);

end architecture dataflow;
```

Photo of the simulation results in EDA Playground:



Note: To revert to EPWave opening in a new browser window, set that option on your user page.

From the simulation it is apparent that *d1* and *d2* are equal as well as *d3* and *d4* which **verifies distributive laws**.