Laboratory 1: Tools

More information on GitHub Tomáš Frýza

My GitHub

Description of laboratory

Operator	Description
<=	Value assignment
and	Logical AND
nand	Logical AND with negated output
or	Logical OR
nor	Logical OR with negated output
not	Nagation

Operator	Description
`xor`	Exclusive OR
`xnor`	Exclusive OR with negated output
`-- comment`	Comments

```
\begin{align*}
    f(c,b,a) =&~ \overline{b}\,a + \overline{c}\,\overline{b}\\
    f(c,b,a)_{\textup{NAND}} =\overline{\overline{a\cdot \overline{b}}\cdot \over
    f(c,b,a)_{\textup{NOR}} =\overline{b+ \overline{a}}+\overline{b+c}\
\end{align*}
```

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

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

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

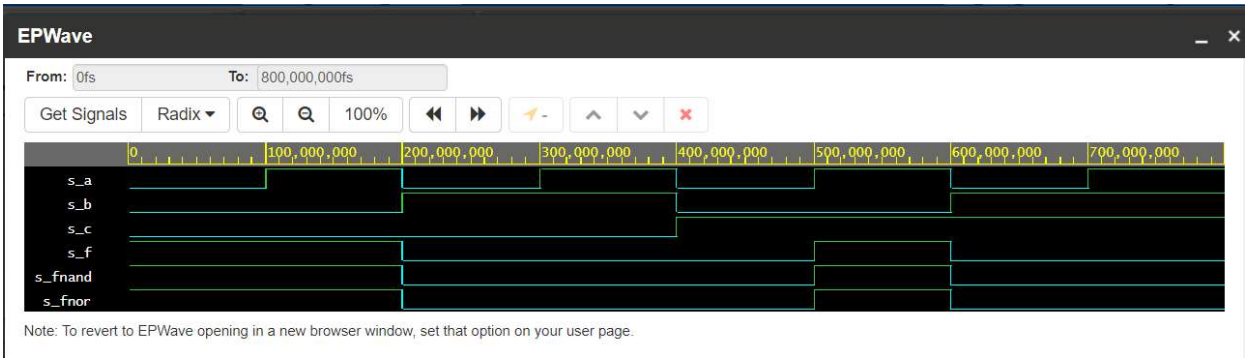
c	b	a	f(c,b,a)	fNAND	fNOR
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	0	0

```
entity gates is
    port(
        a_i    : in  std_logic;         -- Data input
        b_i    : in  std_logic;         -- Data input
        c_i    : in  std_logic;         -- Data input
        f_o  : out std_logic;         -- normal output function
        fNAND_o : out std_logic;         -- NAND output function
        fNOR_o : out std_logic          -- NOR output function
    );
end entity gates;

--------------------------------------------------------------------------
```

```
-- Architecture body for basic gates
-------------------------------------------------------------------------
architecture dataflow of gates is
begin
    f_o  <= (not (b_i) and a_i) or( not (c_i) and not(b_i));
    fNAND_o <= not (not(a_i and not(b_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;
```

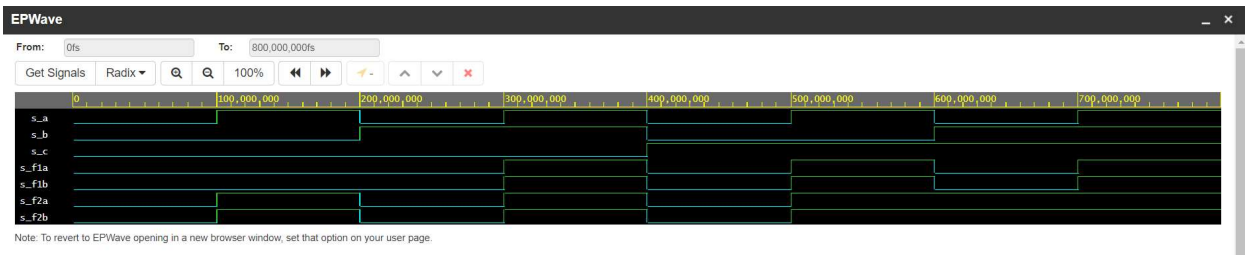


My Example

```
entity gates is
    port(
        a_i    : in  std_logic;         -- Data input
        b_i    : in  std_logic;         -- Data input
        c_i    : in  std_logic;         -- Data input
        f_o1a  : out std_logic;         -- output function 1
        f_o1b : out std_logic;         -- output function 1
        f_o2a : out std_logic;         -- output function 2
        f_o2b : out std_logic         -- output function 2
    );
end entity gates;

-------------------------------------------------------------------------
-- Architecture body for basic gates
-------------------------------------------------------------------------
architecture dataflow of gates is
begin
    f_o1a  <= (a_i and b_i) or (a_i and c_i);
    f_o1b  <= a_i and (b_i or c_i);
    f_o2a  <= (a_i or b_i) and (a_i or c_i);
    f_o2b  <= a_i or (b_i and c_i);

end architecture dataflow;
```


EPWave
From:
0fs
To:
800,000,000fs
Get Signals
Radix
100%
0
100,000,000
200,000,000
300,000,000
400,000,000
500,000,000
600,000,000
700,000,000
s_a
s_b
s_c
s_f1a
s_f1b
s_f2a
s_f2b
Note: To revert to EPWave opening in a new browser window, set that option on your user page.

My Example