,

Computer Architecture (Lab). Week 4

Vladislav, Artem, Hamza, Manuel

Innopolis University

vl.ostankovich@innopolis.ru
a.burmyakov@innopolis.ru
m.rodriguez.osuna@innopolis.university
h.salem@innopolis.university

September 12, 2019

Topic of the lab

- Combinational Logic on FPGA Board

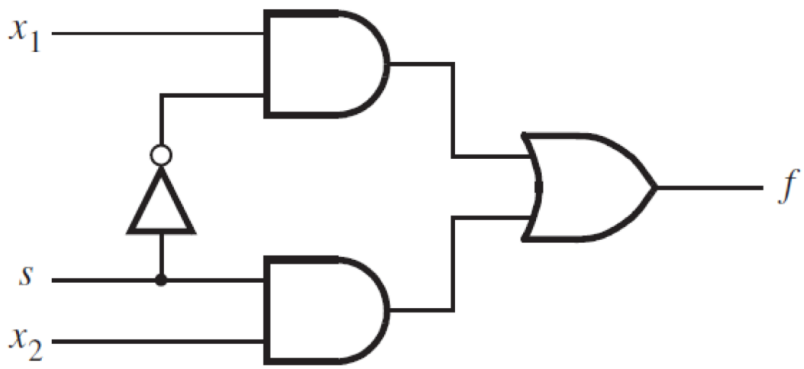
Verilog HDL

- Verilog, standardized as IEEE 1364, is a hardware description language (HDL) used to model electronic systems. It is most commonly used in the design and verification of digital circuits at the register-transfer level of abstraction. It is also used in the verification of analog circuits and mixed-signal circuits, as well as in the design of genetic circuits.

Basics of Verilog HDL

- The main element in the Verilog HDL is module.
- Every module describes a specific device or its part.
- The name of module is written after the “module” word.
- The name of module must be the same as the filename.

Example 1 (1)

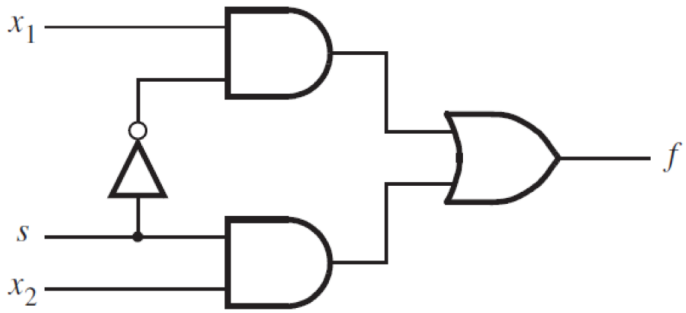



```
module example(x1,x2,s,f);
    input x1,x2,s;
    output f;

    not (k, s);
    and (g, k, x1);
    and (h, s, x2);
    or (f, g, h);
endmodule
```

Example 1 (2)

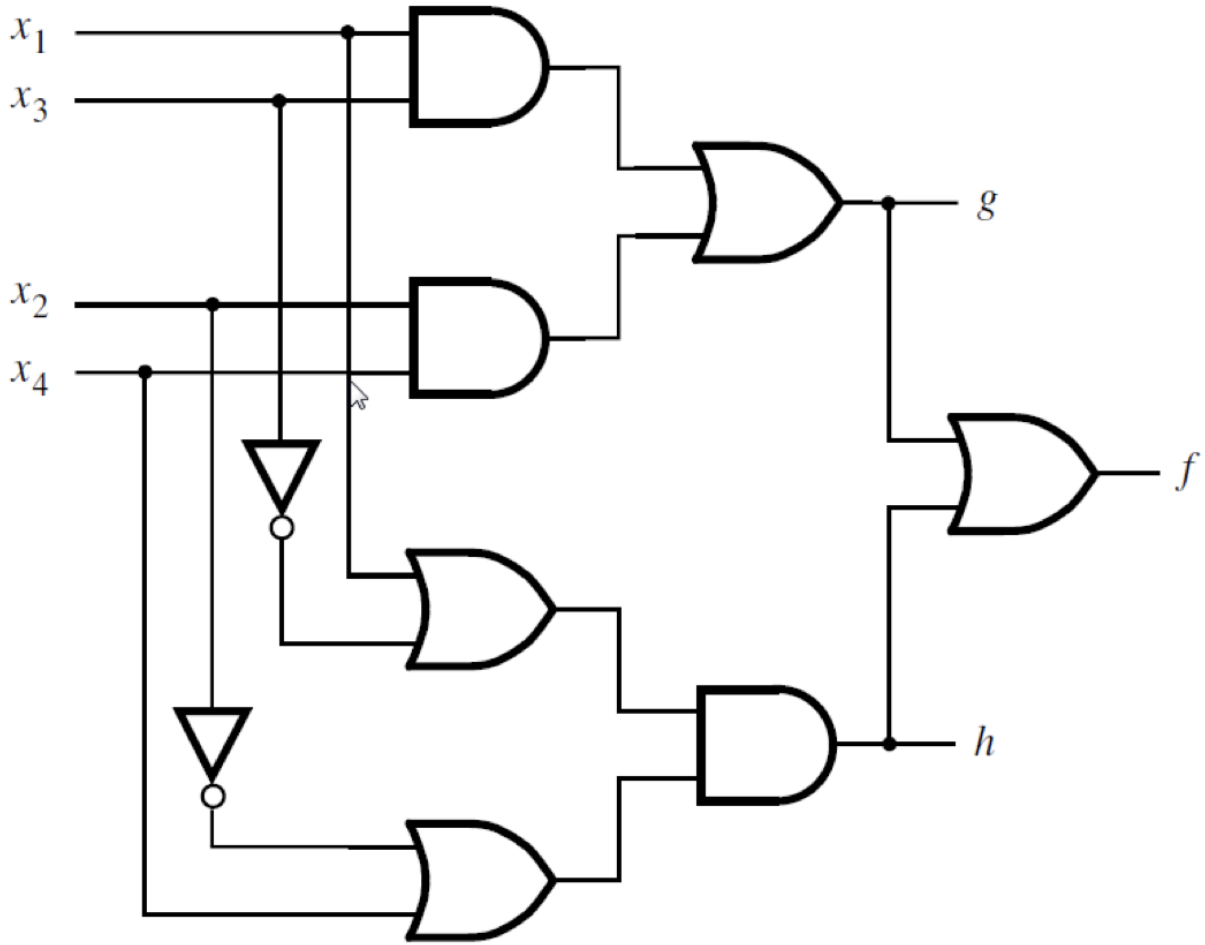
- Shorten the code




```
module example(x1,x2,s,f);
    input x1,x2,s;
    output f;

    assign f=(~s & x1)| (s & x2);
endmodule
```

Example 2 (1)



Example 2 (2)

```
module example2 (x1, x2, x3, x4, f, g, h);
        input x1, x2, x3, x4;
        output f, g, h;

        and (z1, x1, x3);
        and (z2, x2, x4);
        or (g, z1, z2);
        or (z3, x1, ~x3);
        or (z4, ~x2, x4);
        and (h, z3, z4);
        or (f, g, h);

endmodule
```

Example 2 (3)

- Shorten the code

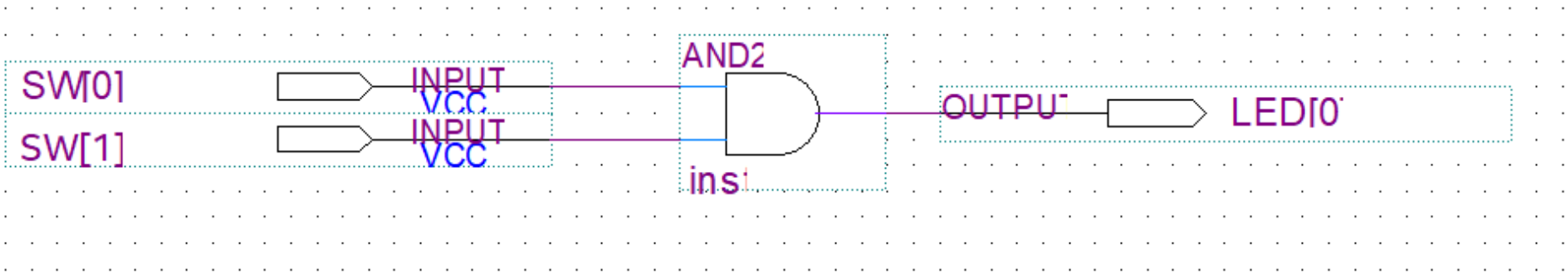
```
module example4 (x1, x2, x3, x4, f, g, h);
        input x1, x2, x3, x4;
        output f, g, h;

        assign g = (x1 & x3) | (x2 & x4);
        assign h = (x1 | ~x3) & (~x2 | x4);
        assign f = g | h;

endmodule
```

Exercise 1

- Repeat Exercise 1 from previous lab using Verilog HDL. Create new project with the same configuration as before, then click File → New → Verilog HDL File. Do not forget to compile, check RTL Viewer and assign pins and compile one more time before simulation on FPGA



Exercise 1: Solution

```
module ex1 (sw0, sw1, led0);
    input sw0, sw1;
    output led0;
    and (led0, sw0, sw1);
endmodule
```

Exercise 2

- Write a program on Verilog which will print message with your date of birth on six 7-segment displays (for MAX 10) when the first push-button is pressed. For Cyclone IV one 7-segment display should show any number, the rest displays should be disabled. When the second push-button is pressed make all user-defined LEDs with even numbers turn on.

Exercise 2: Solution (1)

```
module lab3
(
        input  [1:0] KEY, //keys
        output [9:0] LED, //leds for MAX 10
        //[11:0] for Cyclone IV
        output [47:0] HEX //displays
);
genvar i;
generate
        for (i=0; i<9; i=i+2) //for MAX 10
        //i<11 for Cyclone IV
        begin: gen
                assign LED[i]=~KEY[1];
        end
endgenerate
```

Exercise 2: Solution (2)

```
genvar j;
generate
        for (j=0; j<6; j=j+1)
        begin: gen2
                case(j)
                0: assign HEX[4]=~KEY[0],
                          HEX[5]=~KEY[0],
                          HEX[7]=~KEY[0];
                1: assign HEX[4+j*8]=~KEY[0],
                          HEX[7+j*8]=~KEY[0];
                2: assign HEX[4+j*8]=~KEY[0],
                          HEX[5+j*8]=~KEY[0];
                3: assign HEX[6+j*8]=~KEY[0],
                          HEX[7+j*8]=~KEY[0];
                4: assign HEX[6+j*8]=~KEY[0];
```

Exercise 2: Solution (3)

```
                4: assign HEX[6+j*8]=~KEY[0];
                5: assign HEX[2+j*8]=~KEY[0],
                       HEX[5+j*8]=~KEY[0],
                           HEX[7+j*8]=~KEY[0];
                endcase
        end
endgenerate
endmodule
```

Extra Exercise

- Divide into teams of several people and choose English/Russian word, which can be represented on six 7-segment displays. Opponent team should not know your word. Take into account that not each letter can be displayed. The length of the word should not exceed 6 letters.
- Implement code in Verilog which will create this word using switches. Each switch should turn on some segments of displays. Correct combination of turned on switches should show your secret word on display.
- In the end teams should swap FPGAs without disconnection and try to guess opponent's secret word.
- Let's play!

Acknowledgements

- This lab was created and maintained by Vitaly Romanov, Aidar Gabdullin, Munir Makhmutov, Ruzilya Mirgalimova, Muhammad Fahim, Vladislav Ostankovich, Alena Yuryeva and Artem Burmyakov