# Digital Design Laboratory

Dr. Oniga István
University of Debrecen, Faculty of Informatics

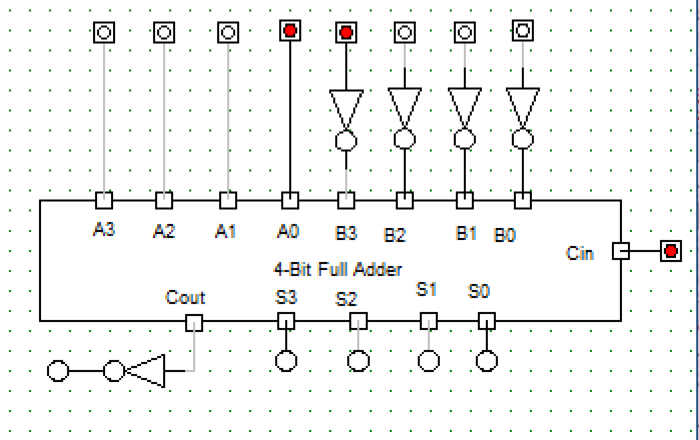
This work was supported by the construction EFOP-3.4.3-16-2016-00021.
The project was supported by the European Union, co-financed by the European Social Fund.

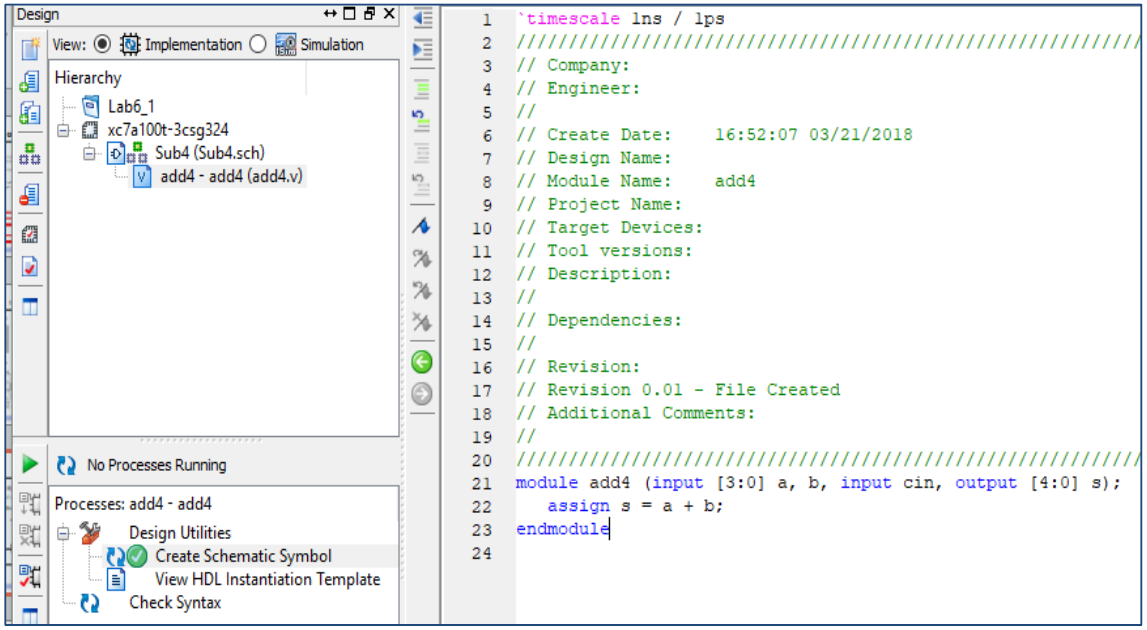
## 6. Laboratory assignments

- Arithmetical and Logical Unit
  - 4-bits adders/subtractors
  - 1-bit ALU
  - 4-bits ALU description using VERILOG
  - 4-bits ALU results display on 7 segment display

# Lab6_1a: 4-bits subtractor

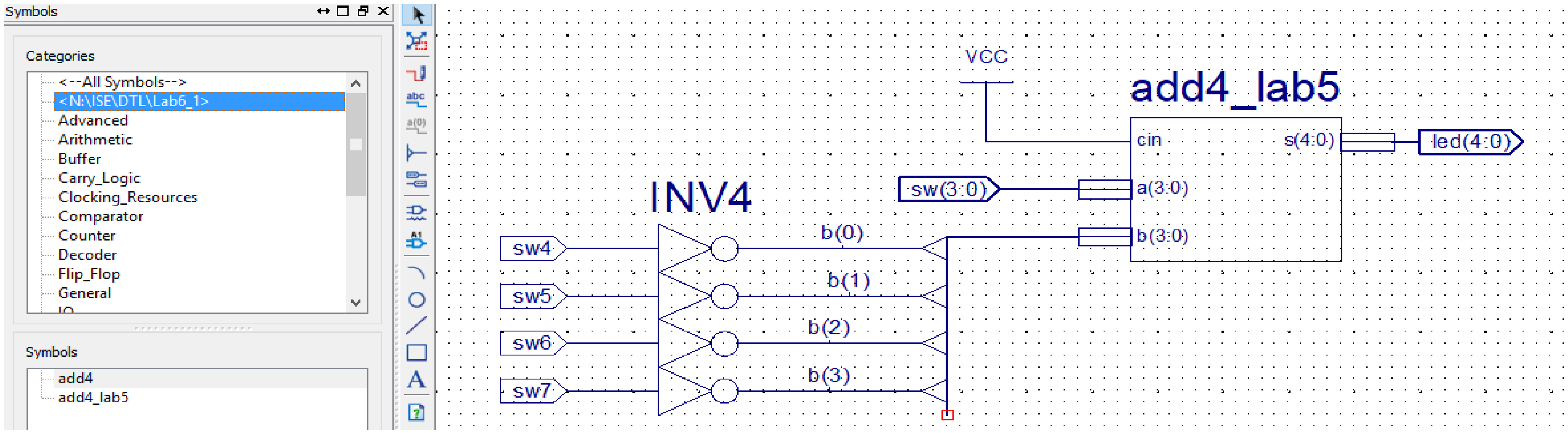
- Create a new HDL project (Lab6_1)
- Add (Add Copy of Source) the Verilog source add4.v file created for assignment Lab5_5b. Add a new input named cin to this source.
- In the Processes window double click on Create Schematic Symbol
- Add a new Schematic source file subb4.v.
- In the schematic editor in Categories window choose the actual folder
- In Symbols window find the previously created add4 symbol. Add this symbol to page and finish the schematic as it can be seen in the next page.





```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    16:52:07 03/21/2018
// Design Name:
// Module Name:    add4
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////
module add4 (input [3:0] a, b, input cin, output [4:0] s);
   assign s = a + b;
endmodule
```

# Lab6_1a: 4-bits subtractor



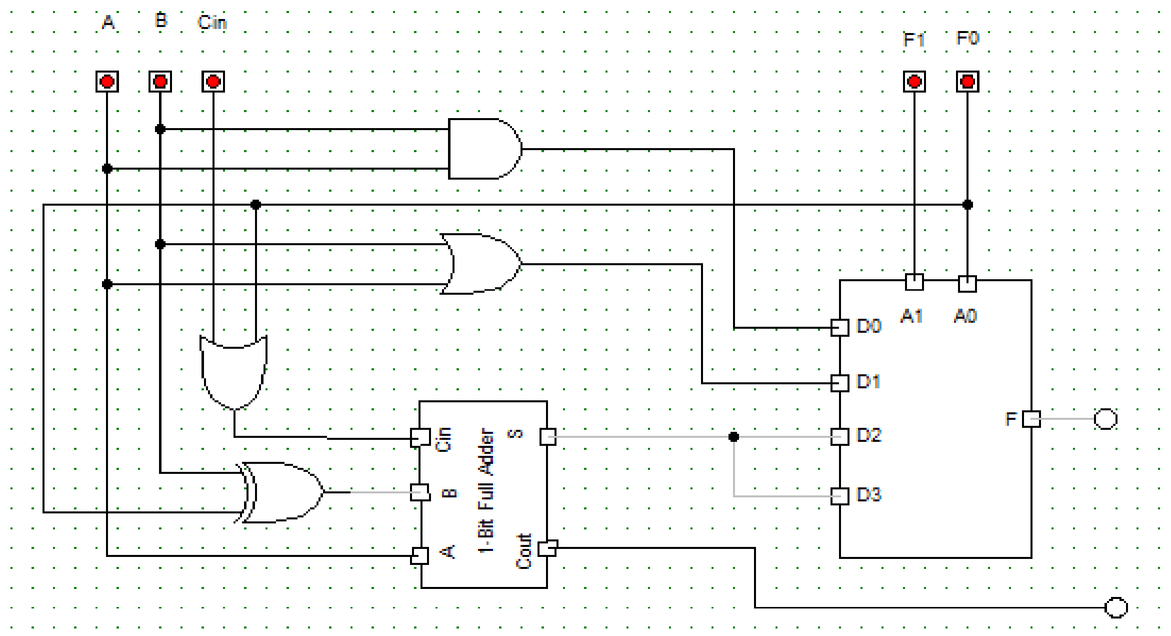
- Add and adapt the Nexysx.ucf file. Note the difference between the specification of a bus: sw(3:0) and wire sw4, sw5, sw6, sw7 in the ucf file
- Generate the configuration file, download to board and test

```
• NET "sw<0>"    LOC=J15 | IOSTANDARD=LVCMOS33; #IO_L24N_T3_RS0_15
• NET "sw<1>"    LOC=L16 | IOSTANDARD=LVCMOS33; #IO_L3N_T0_DQS_EMCCLK_14
• NET "sw<2>"    LOC=M13 | IOSTANDARD=LVCMOS33; #IO_L6N_T0_D08_VREF_14
• NET "sw<3>"    LOC=R15 | IOSTANDARD=LVCMOS33; #IO_L13N_T2_MRCC_14
• NET "sw4"   LOC=R17 | IOSTANDARD=LVCMOS33; #IO_L12N_T1_MRCC_14
• NET "sw5"   LOC=T18 | IOSTANDARD=LVCMOS33; #IO_L7N_T1_D10_14
• NET "sw6"   LOC=U18 | IOSTANDARD=LVCMOS33; #IO_L17N_T2_A13_D29_14
• NET "sw7"   LOC=R13 | IOSTANDARD=LVCMOS33; #IO_L5N_T0_D07_14
• NET "led<0>"    LOC=H17 | IOSTANDARD=LVCMOS33; #IO_L18P_T2_A24_15
• NET "led<1>"    LOC=K15 | IOSTANDARD=LVCMOS33; #IO_L24P_T3_RS1_15
• NET "led<2>"    LOC=J13 | IOSTANDARD=LVCMOS33; #IO_L17N_T2_A25_15
• NET "led<3>"    LOC=N14 | IOSTANDARD=LVCMOS33; #IO_L8P_T1_D11_14
• NET "led<4>"    LOC=R18 | IOSTANDARD=LVCMOS33; #IO_L7P_T1_D09_14
```

# Lab6_2: 1 bit ALU

- Create a new project (Lab6_2)
- Add (Add Copy of Source) the add1_full .v  source file created for a Lab5_4.
- In Processes window duble click on Create Schematic Symbol

- Add a new Schematic file (Sub4)
- In schematic editor in the Categories window choose the current work folder
- In the Symbols window find the symbol for add4 and add it to the schematic page and finish the following schematic.
- Ad to project the Verilog description of the multiplexer used in Lab4_3c, and create a schematic symbol from this. Add it to schematic page and connect it as in the next figure.



# Lab6_2: 1 bits ALU

- Add and adapt the Nexysx.ucf file.
- Generate the configuration file, download to board and test. (Fill the next tables)

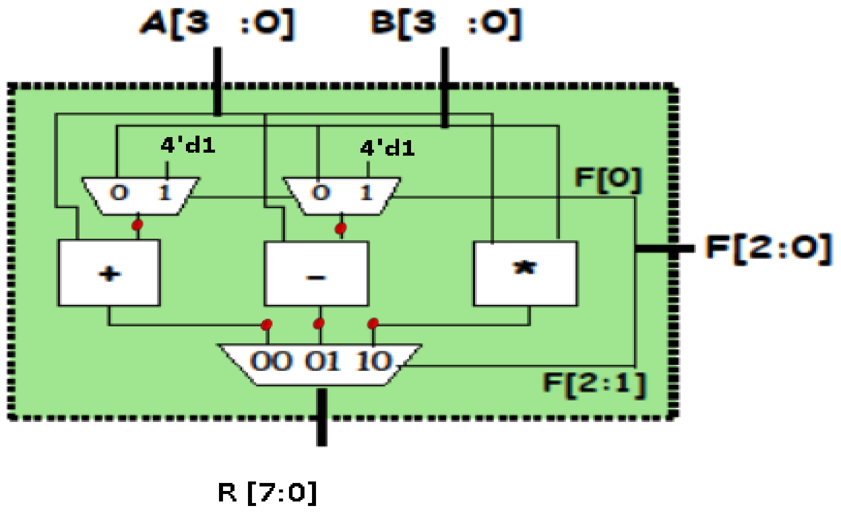
## Logical operations

| F1 | F0 | A | B | Cin | F |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | x | |
| 0 | 0 | 0 | 1 | x | |
| 0 | 0 | 1 | 0 | x | |
| 0 | 0 | 1 | 1 | x | |
| 0 | 1 | 0 | 0 | x | |
| 0 | 1 | 0 | 1 | x | |
| 0 | 1 | 1 | 0 | x | |
| 0 | 1 | 1 | 1 | x | |
| 1 | 0 | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

## Arithmetical operations

| F1 | F0 | A | B | Cin | F | Cout |
|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | | |
| 1 | 0 | 0 | 1 | 0 | | |
| 1 | 0 | 1 | 0 | 0 | | |
| 1 | 0 | 1 | 1 | 0 | | |
| 1 | 0 | 0 | 0 | 1 | | |
| 1 | 0 | 0 | 1 | 1 | | |
| 1 | 0 | 1 | 0 | 1 | | |
| 1 | 0 | 1 | 1 | 1 | | |
| 1 | 1 | 0 | 0 | 0 | | |
| 1 | 1 | 0 | 1 | 0 | | |
| 1 | 1 | 1 | 0 | 0 | | |
| 1 | 1 | 1 | 1 | 0 | | |
| 1 | 1 | 0 | 0 | 1 | | |
| 1 | 1 | 0 | 1 | 1 | | |
| 1 | 1 | 1 | 0 | 1 | | |
| 1 | 1 | 1 | 1 | 1 | | |

# Lab6_3: 4-bits ALU



| F2 | F1 | F0 | Function |
|---|---|---|---|
| 0 | 0 | 0 | A + B |
| 0 | 0 | 1 | A + 1 |
| 0 | 1 | 0 | A - B |
| 0 | 1 | 1 | A - 1 |
| 1 | 0 | X | A * B |

- 4 and 8 bits wires

- Create a new projectLab6_3
- Add a new „Verilog” sorce file (alu_top). This module will connect the ALU modules

```
module alu_top(input [3:0] a, b, input [2:0] f, output [7:0] r ); wire
[3:0] addmux_out, submux_out;
wire [7:0] add_out, sub_out, mul_out;
        mux2_4 adder_mux(b, 4'd1, f[0], addmux_out);
        mux2_4 sub_mux(b, 4'd1, f[0], submux_out);
        add4 our_adder(a, addmux_out, add_out);
        sub4 our_subtracer(a, submux_out, sub_out);
        mul4 our_multiplier(a,b,mul_out);
        mux3_8 output_mux(add_out, sub_out, mul_out, f[2:1], r);
endmodule
```

- Add a new „Verilog” source file (alu4_modules)
- Replace the content created automatic with the modules description on the next page.

## Modules description

```
module mux2_4(input [3:0] i0, i1, input sel, output [7:0] out);
         assign out = sel ? i1 : i0;
endmodule

module mux3_8(input [7:0] i0, i1, i2, input [1:0] sel, output reg [7:0] out);
     always @(i0 or i1 or i2 or sel)
         begin
              case (sel)
                   2'b00: out = i0;
                   2'b01: out = i1;
                   2'b10: out = i2;
                   default: out = 8'bx;
              endcase
         end
endmodule

module add4(input [3:0] i0, i1, output [7:0] sum);
         assign sum=i0+i1;
endmodule

module sub4(input [3:0] i0, i1, output [7:0] diff);
         assign diff=i0-i1;
endmodule

module mul4(input [3:0] i0, i1, output [7:0] prod);
         assign prod=i0*i1;
endmodule
```

# ALU implementation and test

- Add and adapt the Nexysx.ucf file (F[2:0] <=> sw [15:13]; a [3:0] <=> sw [3:0], b [3:0] <=> sw [7:4], r [7:0] <=> led [7:0])
- Generate the configuration file, download to board and test. (Fill the next tables)
- Using sw [7:0] set the following input operands:
  - a = 3, b = 2 and using sw [15:13] set the 5 possible operations. In each case fill in the table the result.

| F2 sw[15] | F1 sw[14] | F0 sw[13] | r[7] | r[6] | r[5] | r[4] | r[3] | r[2] | r[1] | r[0] |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | | | | | | | | |
| 0 | 0 | 1 | | | | | | | | |
| 0 | 1 | 0 | | | | | | | | |
| 0 | 1 | 1 | | | | | | | | |
| 1 | 0 | X | | | | | | | | |

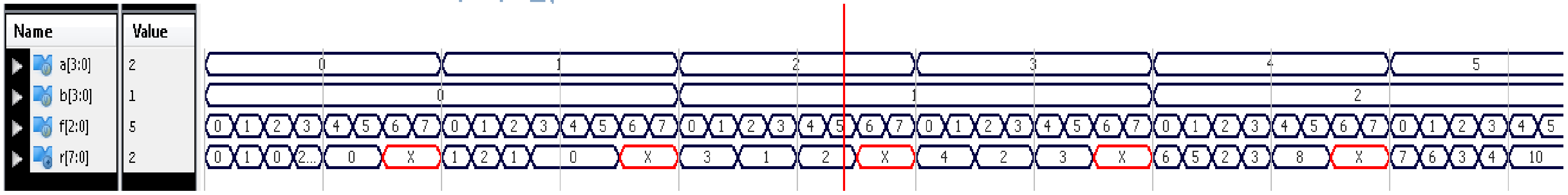
- Test again using new operands: a = 10, b = 12. The result of subtraction is correct?
- How much will the highest result of multiplication be?

```
NET "a<0>" LOC=J15 | IOSTANDARD=LVCMOS33;
NET "a<1>" LOC=L16 | IOSTANDARD=LVCMOS33;
NET "a<2>" LOC=M13 | IOSTANDARD=LVCMOS33;
NET "a<3>" LOC=R15 | IOSTANDARD=LVCMOS33;

NET "b<0>" LOC=R17 | IOSTANDARD=LVCMOS33;
NET "b<1>" LOC=T18 | IOSTANDARD=LVCMOS33;
NET "b<2>" LOC=U18 | IOSTANDARD=LVCMOS33;
NET "b<3>" LOC=R13 | IOSTANDARD=LVCMOS33;

NET "f<0>" LOC=U12 | IOSTANDARD=LVCMOS33;
NET "f<1>" LOC=U11 | IOSTANDARD=LVCMOS33;
NET "f<2>" LOC=V10 | IOSTANDARD=LVCMOS33;
## LEDs
NET "r<0>" LOC=H17 | IOSTANDARD=LVCMOS33;
NET "r<1>" LOC=K15 | IOSTANDARD=LVCMOS33;
NET "r<2>" LOC=J13 | IOSTANDARD=LVCMOS33;
NET "r<3>" LOC=N14 | IOSTANDARD=LVCMOS33;
NET "r<4>" LOC=R18 | IOSTANDARD=LVCMOS33;
NET "r<5>" LOC=V17 | IOSTANDARD=LVCMOS33;
NET "r<6>" LOC=U17 | IOSTANDARD=LVCMOS33;
NET "r<7>" LOC=U16 | IOSTANDARD=LVCMOS33;
```

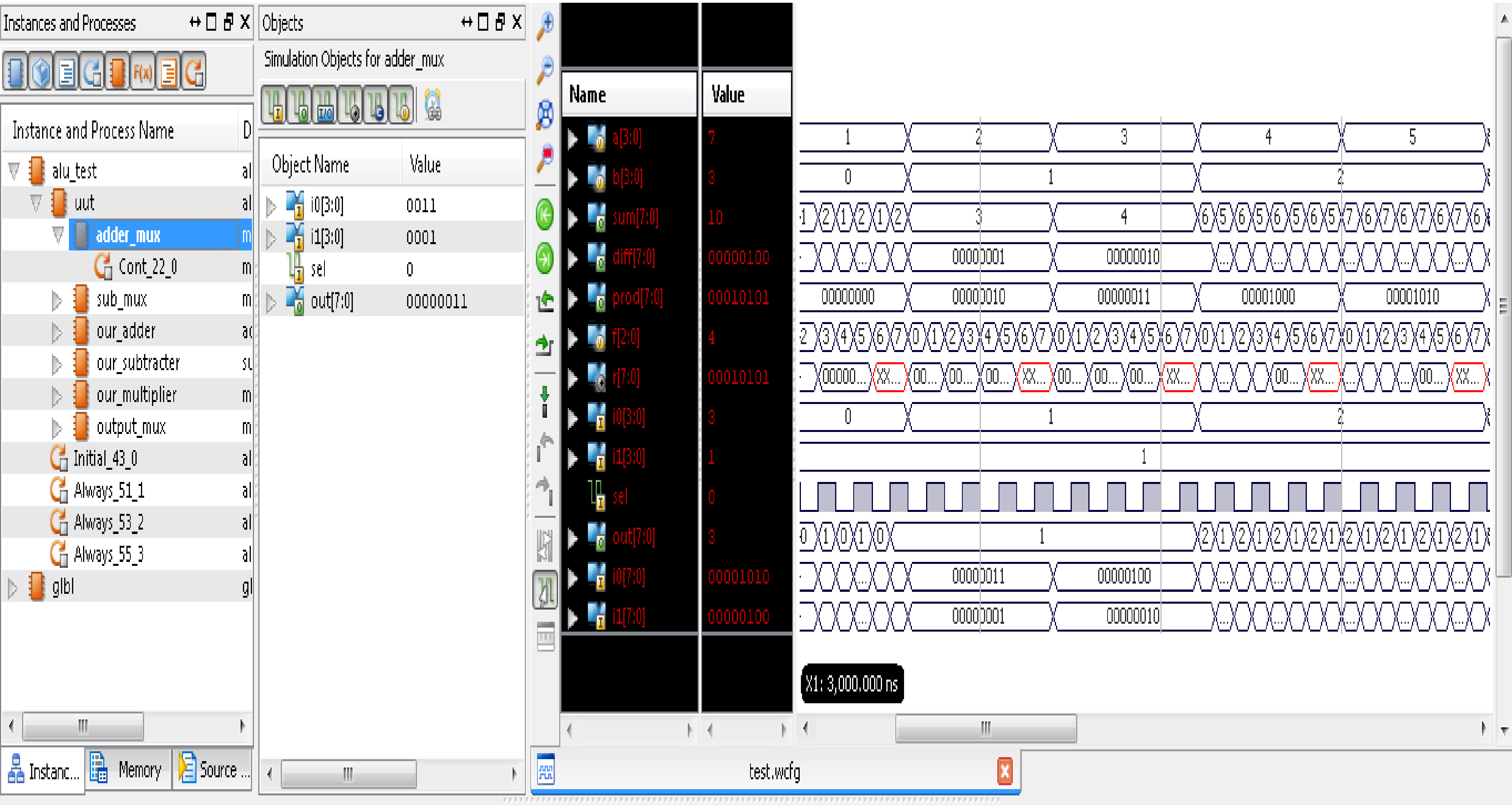
# Lab6_3b: 4-bits ALU – simulation (not mandatory)

## ALU test modul description

```
initial begin
        a = 0;
        b = 0;
        f = 0;
end

always # 4000
        a= a+1;
always # 800
        b= b+1;
always # 50
        f= f+1;
```



# Checking the status of internal wires



Instances and Processes

| Instance and Process Name | D |
|---|---|
| alu_test | al |
| uut | al |
| adder_mux | m |
| Cont_22_0 | m |
| sub_mux | m |
| our_adder | ac |
| our_subtracter | su |
| our_multiplier | m |
| output_mux | m |
| Initial_43_0 | al |
| Always_51_1 | al |
| Always_53_2 | al |
| Always_55_3 | al |
| glbl | gl |

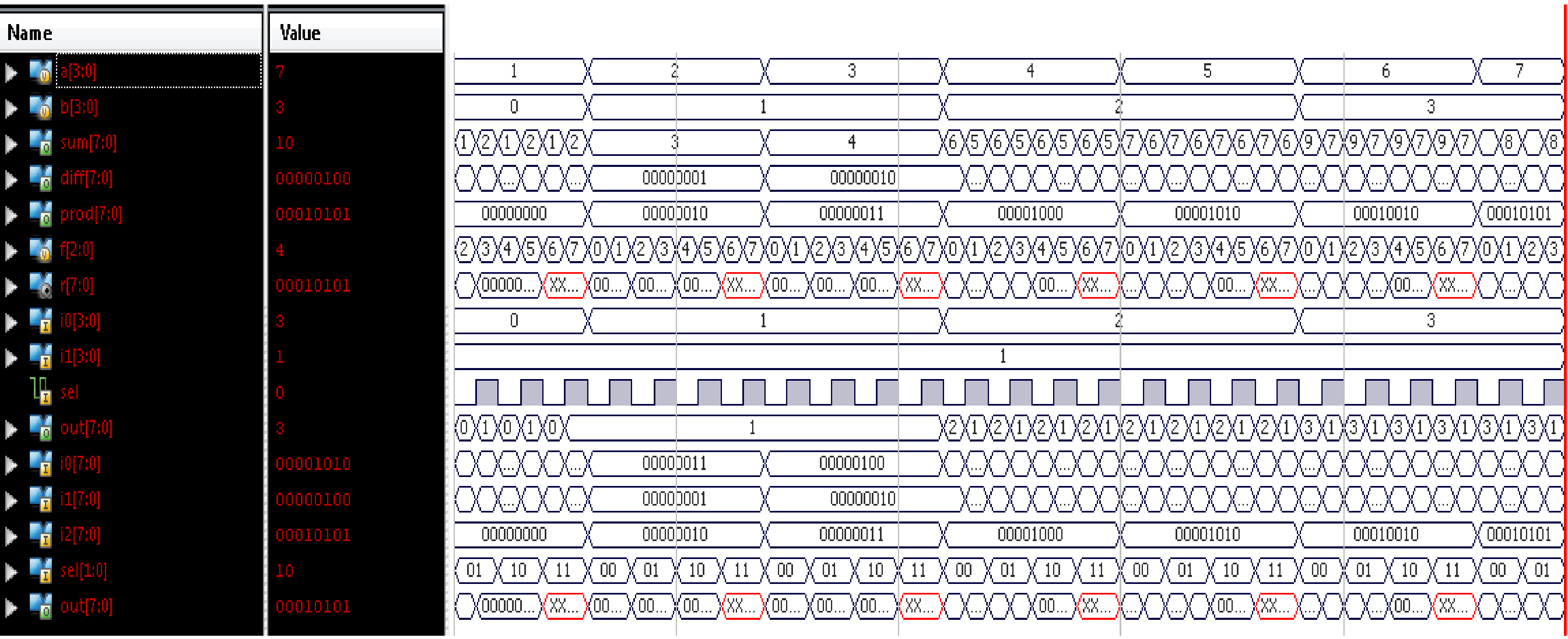
Objects

Simulation Objects for adder_mux

| Object Name | Value |
|---|---|
| i0[3:0] | 0011 |
| i1[3:0] | 0001 |
| sel | 0 |
| out[7:0] | 00000011 |

| Name | Value |
|---|---|
| a[3:0] | 7 |
| b[3:0] | 3 |
| sum[7:0] | 10 |
| diff[7:0] | 00000100 |
| prod[7:0] | 00010101 |
| f[2:0] | 4 |
| r[7:0] | 00010101 |
| i0[3:0] | 3 |
| i1[3:0] | 1 |
| sel | 0 |
| out[7:0] | 3 |
| i0[7:0] | 00001010 |
| i1[7:0] | 00000100 |

# Checking the status of internal wires



| Name | Value |
|---|---|
| a[3:0] | 7 |
| b[3:0] | 3 |
| sum[7:0] | 10 |
| diff[7:0] | 00000100 |
| prod[7:0] | 00010101 |
| f[2:0] | 4 |
| r[7:0] | 00010101 |
| i0[3:0] | 3 |
| i1[3:0] | 1 |
| sel | 0 |
| out[7:0] | 3 |
| i0[7:0] | 00001010 |
| i1[7:0] | 00000100 |
| i2[7:0] | 00010101 |
| sel[1:0] | 10 |
| out[7:0] | 00010101 |

# Lab6_4:

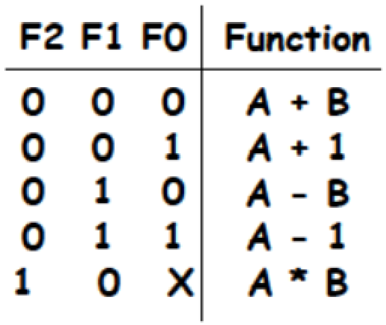
## 4-bits ALU result display on 7segments display

- Create a new project Lab6_4
- Add (Add copy of source) all files from Lab6_3 (alu_top, alu4_modules, Nexysx.ucf).
- Add a copy of the file created hex7seg on a previous lab (Lab3_3).
- Add a new „Verilog" source file (alu4_top). This will connect the alu_top  and hex7seg modules.

```
module alu4_top( input [3:0] a, input [3:0] b, input [2:0] f,
          output [6:0] a_to_g, output [7:0] an, output dp);
wire [7:0] r1;

assign an = 8'b11111110; // 1 digit on, 7 digits off
assign dp = 1; // dp off

hex7seg D4 (.x(r1[3:0]),.a_to_g(a_to_g));
alu_top D1 (.a(a), .b(b), .f(f), .r(r1));

endmodule
```

# ALU implementation and test

- Add and adapt the Nexysx.ucf file (F[2:0] <=> sw [15:13]; a [3:0] <=> sw [3:0], b [3:0] <=> sw [7:4], a_to_g [6:0] <=> a_to_g [6:0], an[7:0] <=> an[6:0], dp <=> dp)
- Generate the configuration file, download to board and test.
- Using sw [7:0] set the following operands:
  - a = 3, b = 2
- Using sw [15:13] set the 5 possible operations. In each case observe the correctness of the result.



| F2 | F1 | F0 | Function |
|---|---|---|---|
| 0 | 0 | 0 | A + B |
| 0 | 0 | 1 | A + 1 |
| 0 | 1 | 0 | A - B |
| 0 | 1 | 1 | A - 1 |
| 1 | 0 | X | A * B |

| F2 sw[15] | F1 sw[14] | F0 sw[13] | |
|---|---|---|---|
| 0 | 0 | 0 | |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | X | |