Lab 1: Combinational circuit design

1. Objectives:

(1) Learn about using Verilog to design combination circuits.

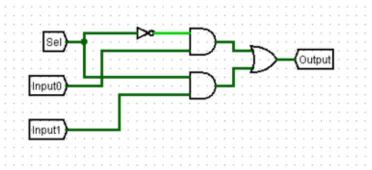
2. Requirements

(1) Paste your design code, RTL schematic, and simulation results to the report.

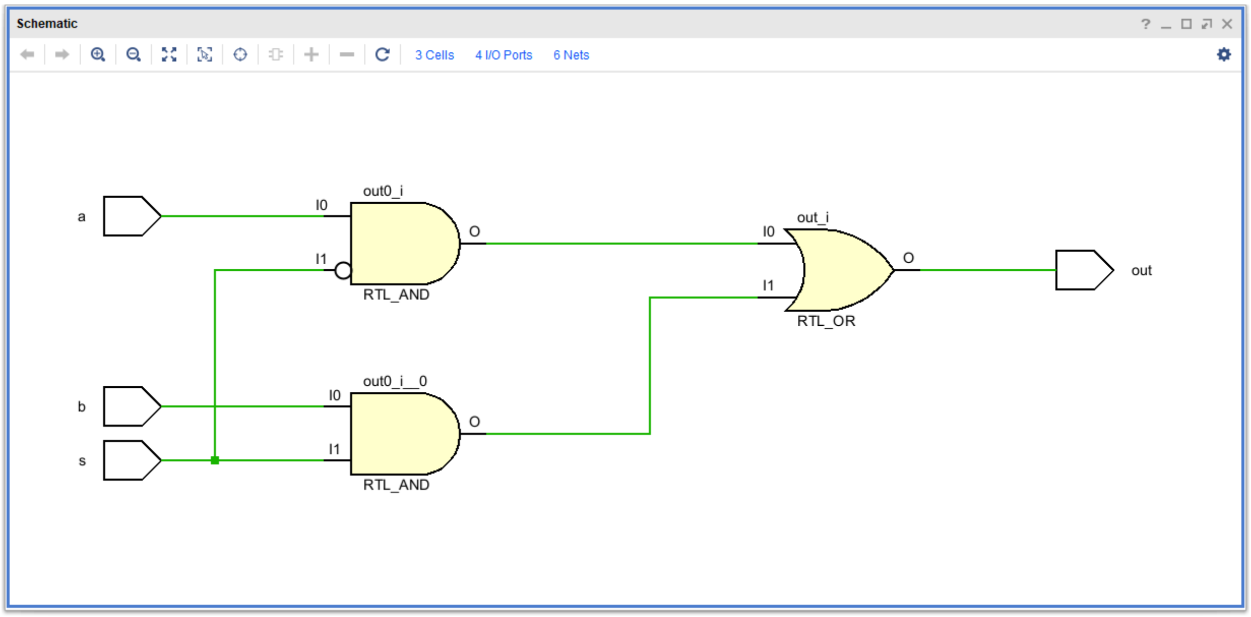
3. Tasks:

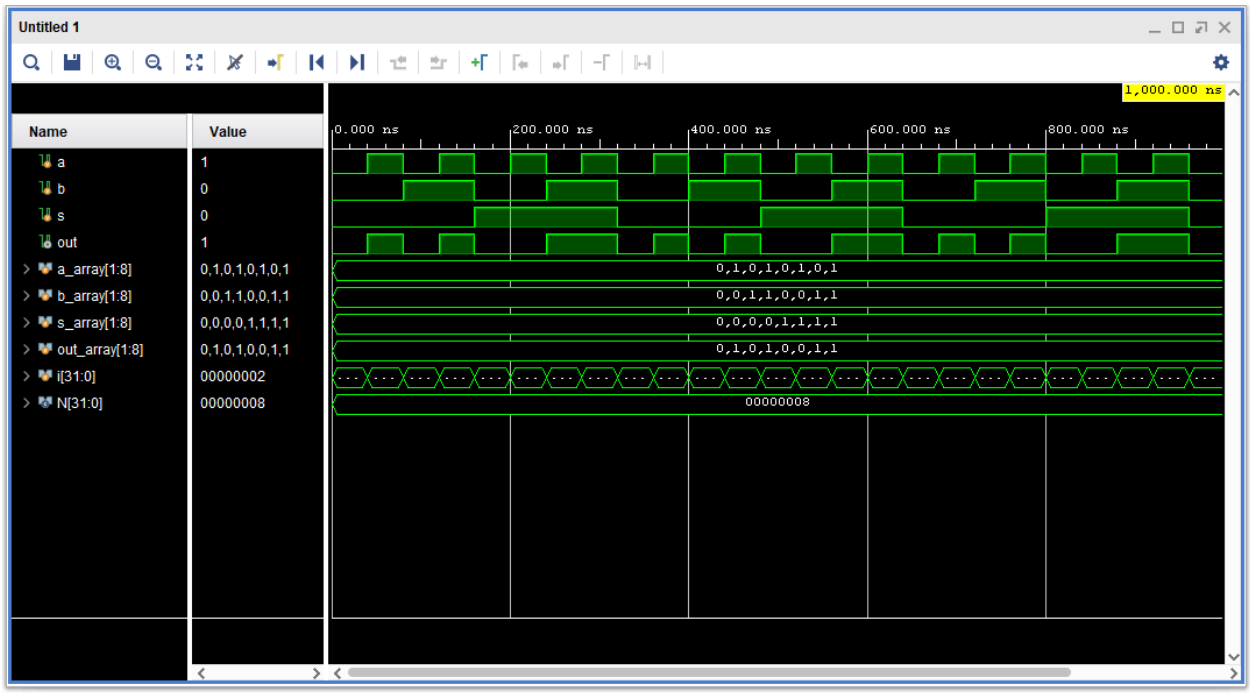
3.1 Design a 2-channel 1-bit multiplexer using dataflow modeling methods (15 points)

Verilog code, RTL schematic, and simulation result:



```
module mux_2c_1b_dataflow(out, s, a, b);

    output out;           // mux output
    input  s, a, b;       // mux inputs

    assign out = (a & (~ s)) | (b & s);

endmodule
```





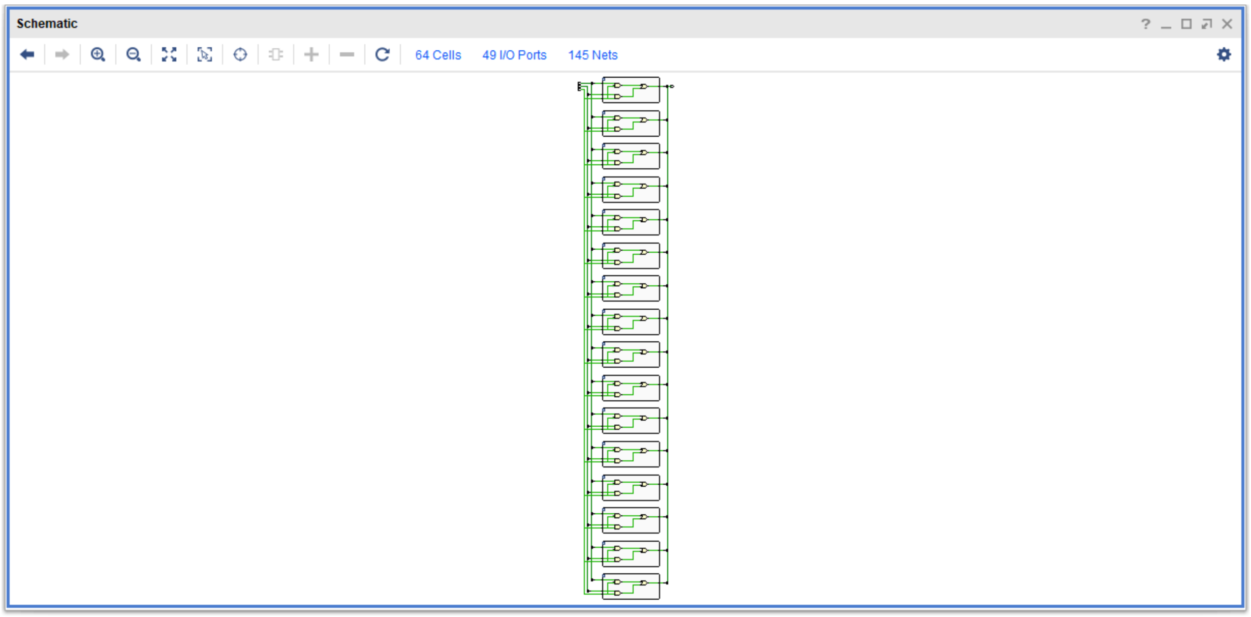
Name	Value
a	1
b	0
s	0
out	1
a_array[1:8]	0,1,0,1,0,1,0,1
b_array[1:8]	0,0,1,1,0,0,1,1
s_array[1:8]	0,0,0,0,1,1,1,1
out_array[1:8]	0,1,0,1,0,0,1,1
i[31:0]	00000002
N[31:0]	00000008

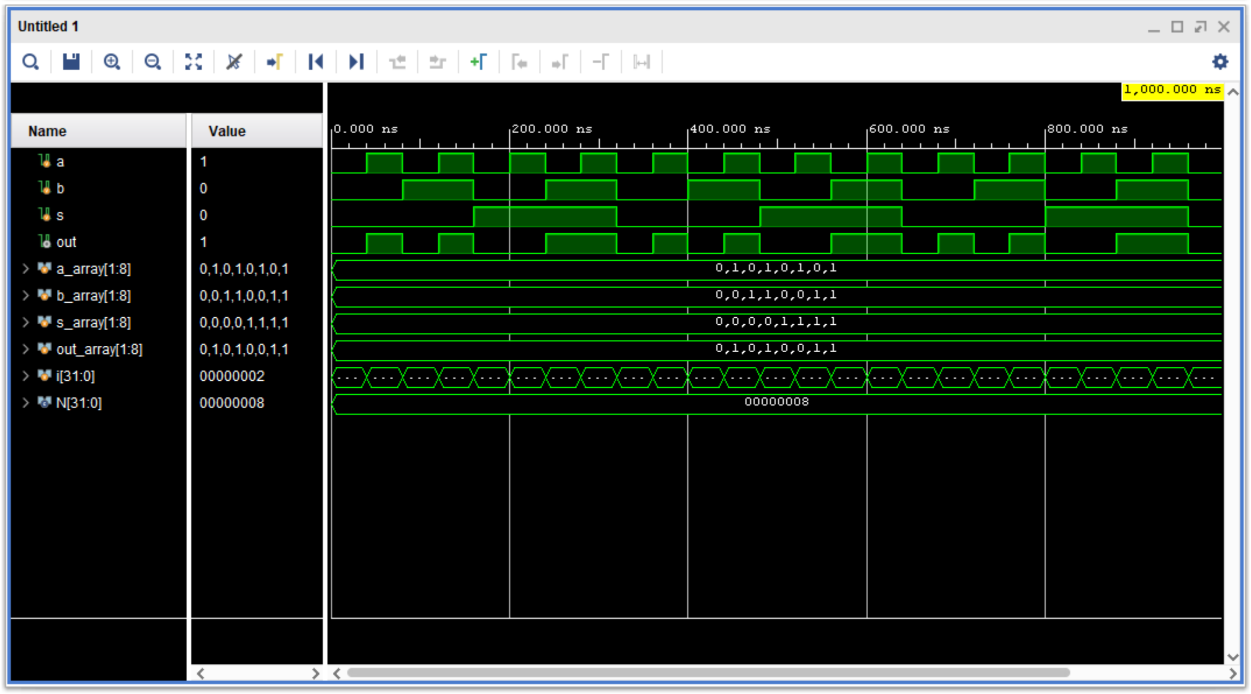
3.2 Design a 2-channel 16-bit multiplexer (25 points)

Requirement: Only 1-bit multiplexers designed in 3.1 are allowed.

Verilog code, RTL schematic, and simulation result:

```
module mux_2c_16b(out, s, a, b);

    output[0:15] out;
    input[0:15] a, b;
    input s;

    mux_2c_1b_dataflow bit0(out[0], s, a[0], b[0]);
    mux_2c_1b_dataflow bit1(out[1], s, a[1], b[1]);
    mux_2c_1b_dataflow bit2(out[2], s, a[2], b[2]);
    mux_2c_1b_dataflow bit3(out[3], s, a[3], b[3]);
    mux_2c_1b_dataflow bit4(out[4], s, a[4], b[4]);
    mux_2c_1b_dataflow bit5(out[5], s, a[5], b[5]);
    mux_2c_1b_dataflow bit6(out[6], s, a[6], b[6]);
    mux_2c_1b_dataflow bit7(out[7], s, a[7], b[7]);
    mux_2c_1b_dataflow bit8(out[8], s, a[8], b[8]);
    mux_2c_1b_dataflow bit9(out[9], s, a[9], b[9]);
    mux_2c_1b_dataflow bit10(out[10], s, a[10], b[10]);
    mux_2c_1b_dataflow bit11(out[11], s, a[11], b[11]);
    mux_2c_1b_dataflow bit12(out[12], s, a[12], b[12]);
    mux_2c_1b_dataflow bit13(out[13], s, a[13], b[13]);
    mux_2c_1b_dataflow bit14(out[14], s, a[14], b[14]);
    mux_2c_1b_dataflow bit15(out[15], s, a[15], b[15]);

endmodule
```

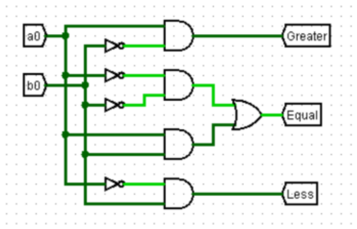




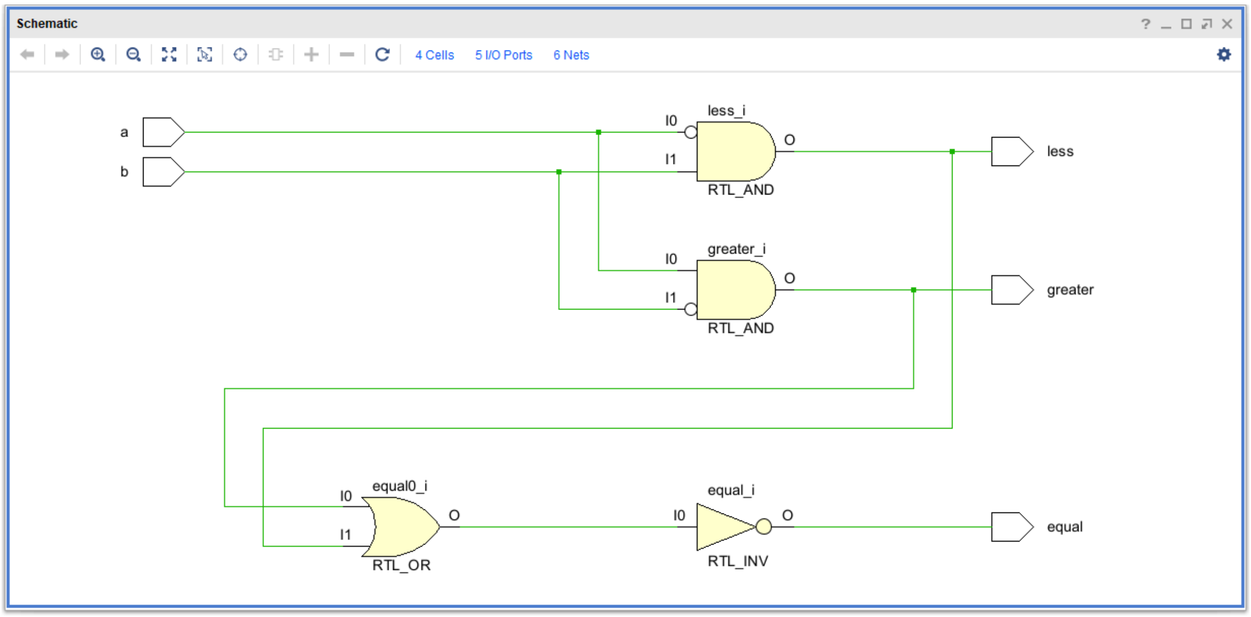
Name	Value
a	1
b	0
s	0
out	1
a_array[1:8]	0,1,0,1,0,1,0,1
b_array[1:8]	0,0,1,1,0,0,1,1
s_array[1:8]	0,0,0,0,1,1,1,1
out_array[1:8]	0,1,0,1,0,0,1,1
i[31:0]	00000002
N[31:0]	00000008

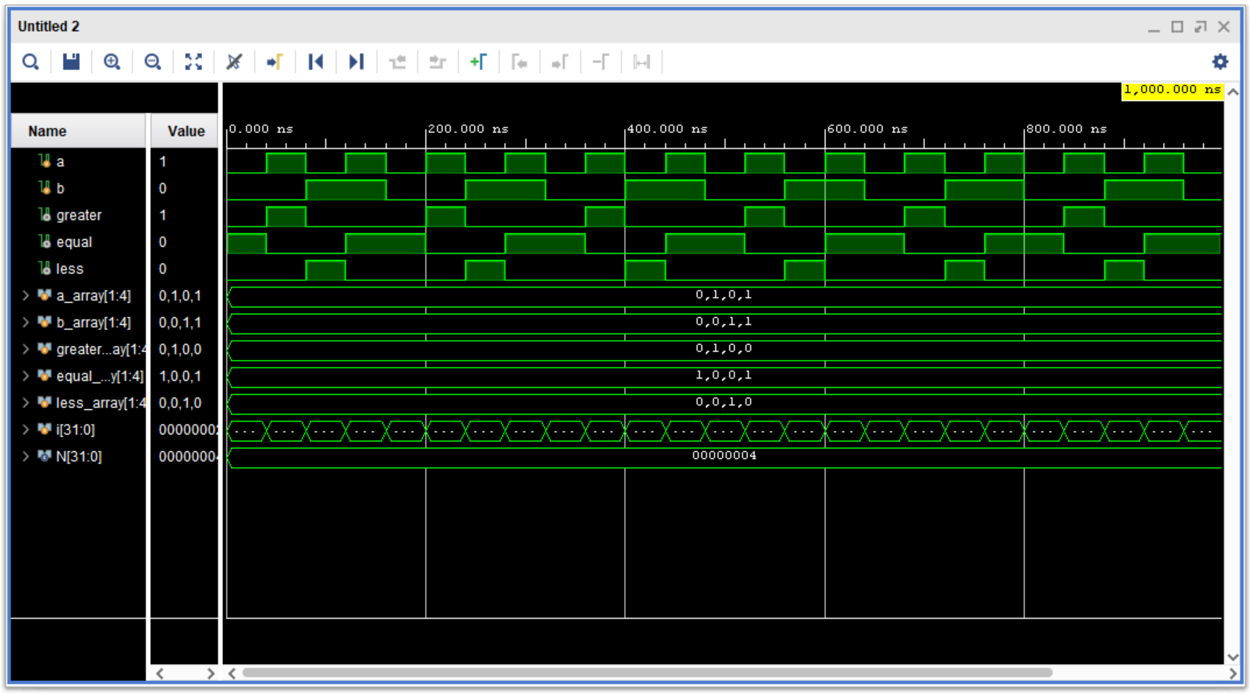
3.3 Design a 1-bit unsigned comparator using dataflow modeling methods (15 points)

Verilog code, RTL schematic, and simulation result:



```
module comparator_u1b_dataflow(greater, equal, less, a, b);

    output greater, equal, less;
    input a, b;

    assign greater = a & ~b;
    assign equal = ~(greater | less);
    assign less = ~a & b;

endmodule
```





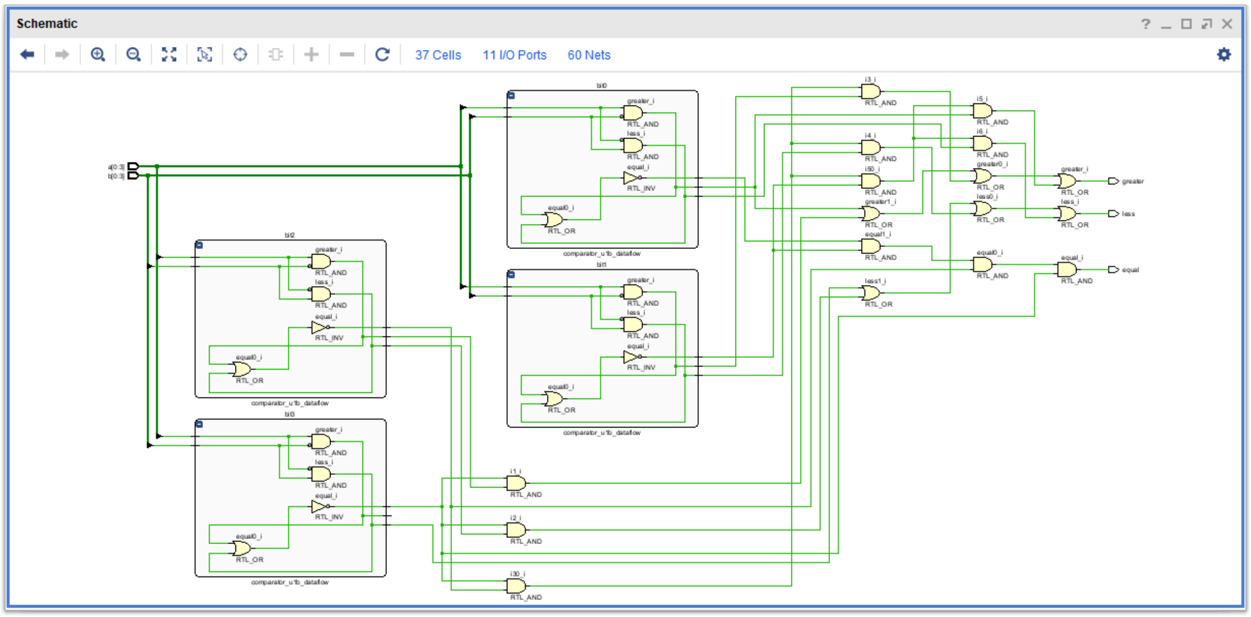
Name	Value
a	1
b	0
greater	1
equal	0
less	0
a_array[1:4]	0,1,0,1
b_array[1:4]	0,0,1,1
greater...ay[1:4	0,1,0,0
equal_...y[1:4]	1,0,0,1
less_array[1:4	0,0,1,0
i[31:0]	0000000
N[31:0]	0000000

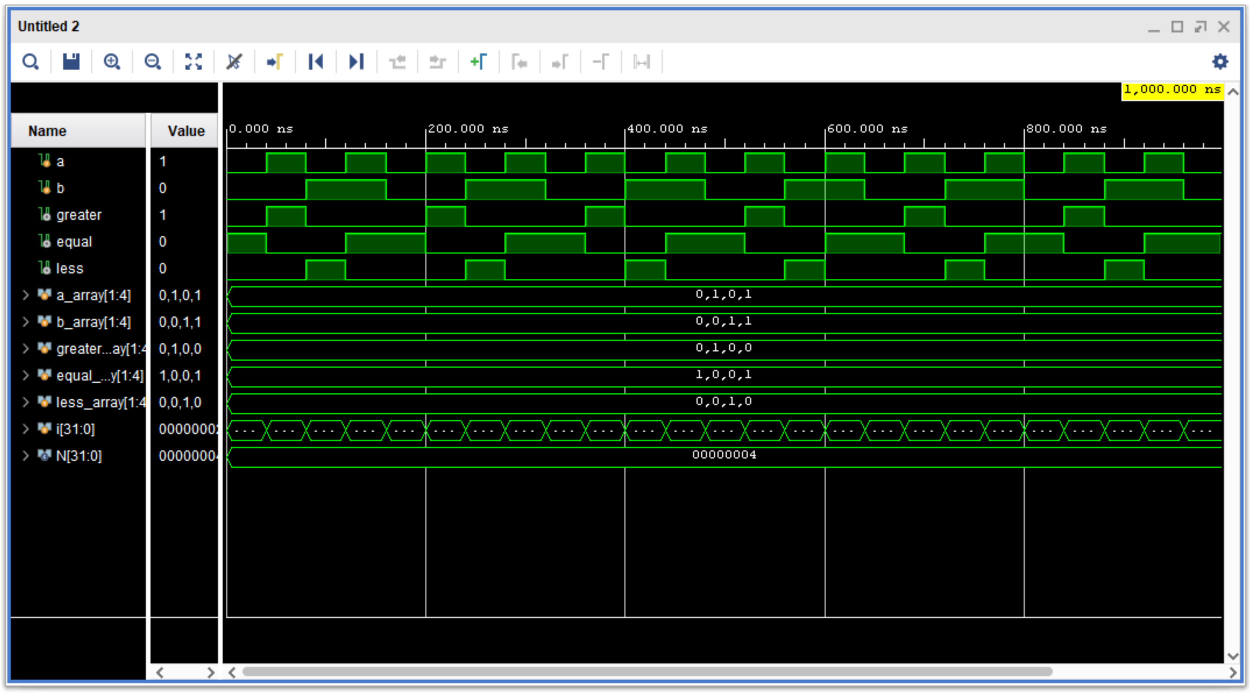
3.4 Design a 4-bit unsigned comparator (25 points)

<u>Requirement:</u> <u>only 1-bit unsigned comparators designed in 3.3 are allowed.</u>

Verilog code, RTL schematic, and simulation result:

```
module comparator_u4b(greater, equal, less, a, b);

    output greater, equal, less;
    input[0:3] a, b;

    wire g0, g1, g2, g3, e0, e1, e2, e3, l0, l1, l2, l3;

    comparator_u1b_dataflow bit0(g0, e0, l0, a[0], b[0]);
    comparator_u1b_dataflow bit1(g1, e1, l1, a[1], b[1]);
    comparator_u1b_dataflow bit2(g2, e2, l2, a[2], b[2]);
    comparator_u1b_dataflow bit3(g3, e3, l3, a[3], b[3]);

    wire i1, i2, i3, i4, i5, i6;

    assign i1 = e3 & g2;
    assign i2 = e3 & l2;
    assign i3 = e3 & e2 & g1;
    assign i4 = e3 & e2 & l1;
    assign i5 = e3 & e2 & e1 & g0;
    assign i6 = e3 & e2 & e1 & l0;

    assign greater = g0 | i1 | i3 | i5;
    assign equal = e0 & e1 & e2 & e3;
    assign less = l3 | i2 | i4 | i6;

endmodule
```





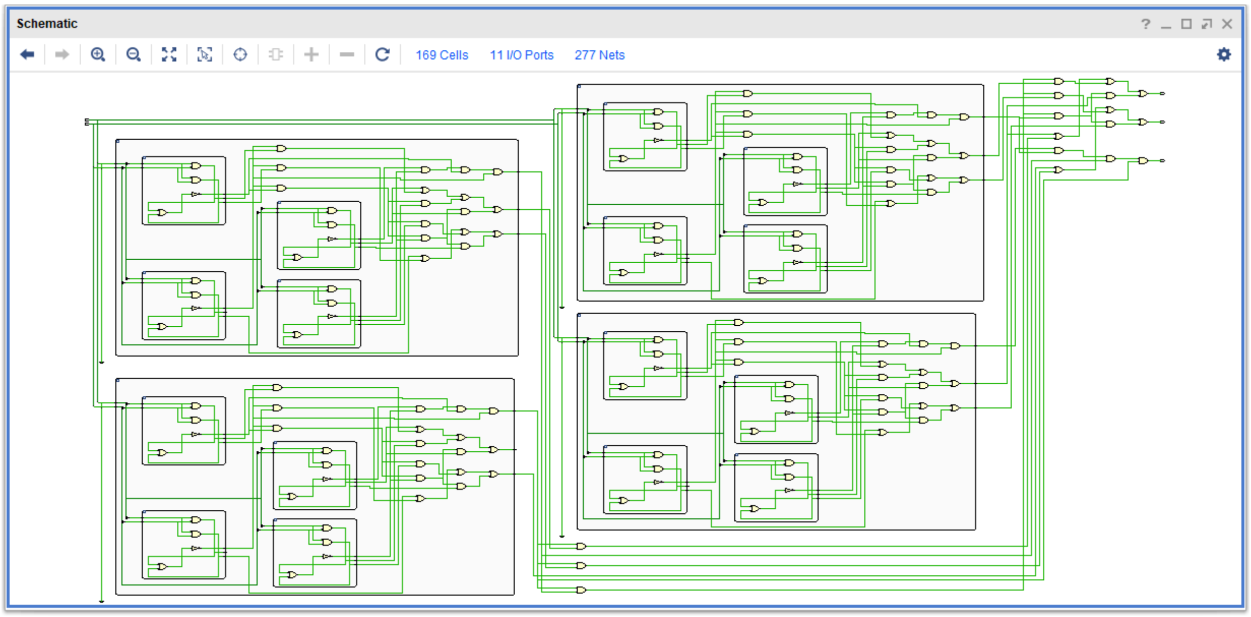
3.5 Design a 16-bit unsigned comparator (20 points)

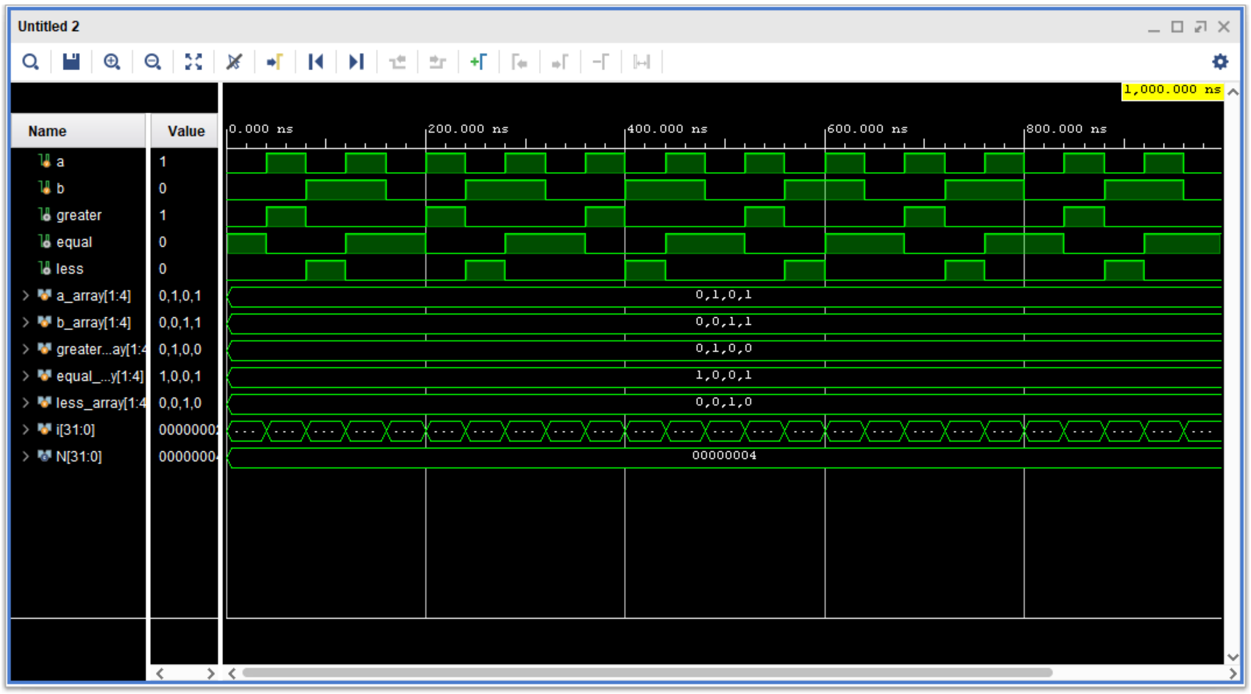
Requirement: only 4-bit unsigned comparators designed in 3.4 are allowed.

Verilog code, RTL schematic, and simulation result:

```
module comparator_u16b(greater, equal, less, a, b);

    output greater, equal, less;
    input[0:3] a, b;

    wire g0, g1, g2, g3, e0, e1, e2, e3, l0, l1, l2, l3;

    comparator_u4b bits0_3(g0, e0, l0, a[0], b[0]);
    comparator_u4b bits4_7(g1, e1, l1, a[1], b[1]);
    comparator_u4b bits6_11(g2, e2, l2, a[2], b[2]);
    comparator_u4b bits12_15(g3, e3, l3, a[3], b[3]);

    wire i1, i2, i3, i4, i5, i6;

    assign i1 = e3 & g2;
    assign i2 = e3 & l2;
    assign i3 = e3 & e2 & g1;
    assign i4 = e3 & e2 & l1;
    assign i5 = e3 & e2 & e1 & g0;
    assign i6 = e3 & e2 & e1 & l0;

    assign greater = g0 | i1 | i3 | i5;
    assign equal = e0 & e1 & e2 & e3;
    assign less = l3 | i2 | i4 | i6;
```

```
endmodule
```





Name	Value
a	1
b	0
greater	1
equal	0
less	0
a_array[1:4]	0,1,0,1
b_array[1:4]	0,0,1,1
greater...ay[1:4	0,1,0,0
equal_...y[1:4]	1,0,0,1
less_array[1:4	0,0,1,0
i[31:0]	0000000:
N[31:0]	0000000

```
0,1,0,1
0,0,1,1
0,1,0,0
1,0,0,1
0,0,1,0
00000004
```