CS 223 Digital Design

Section 1

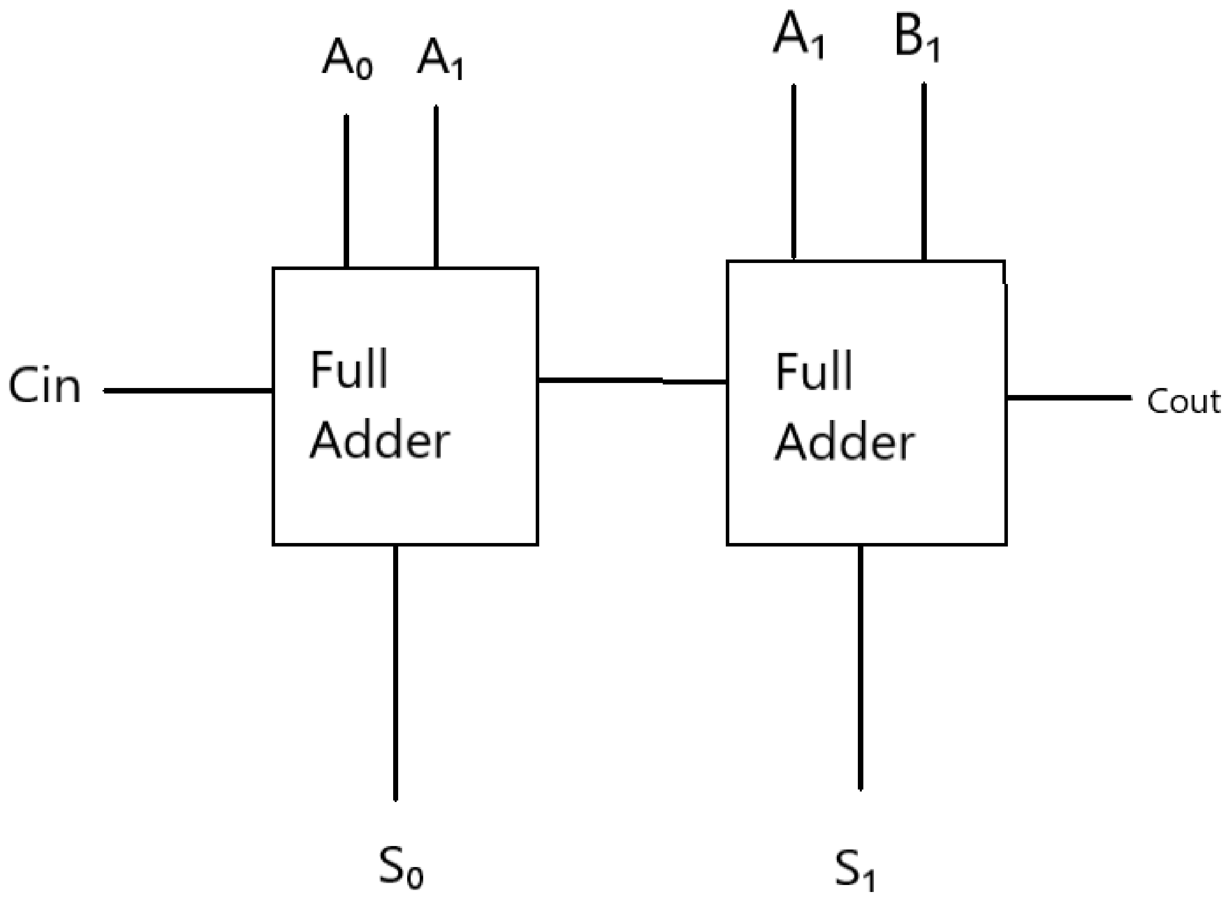
Lab 2

Ahmet Faruk Ulutaş

21803717

17.10.2020

B) Circuit schematic of 2-bit adder made from two full adders



C) Behavioral SystemVerilog module for the full adder

```
`timescale 1ns / 1ps
module behavioral_sv_of_full_adder( input logic a, b, cin, output logic sum, cout);

    logic xor_a_b, and_a_b, and_cin_xor0;

    assign xor_a_b = a ^ b;
    assign and_a_b = a & b;
    assign and_cin_xor0 = cin & xor_a_b;
    assign sum = xor_a_b ^ cin;
    assign cout = and_a_b | and_cin_xor0;
endmodule
```

D) Structural SystemVerilog module for the full adder and a testbench for it

Structural SV Module

```
`timescale 1ns / 1ps
module structural_sv_of_full_adder( input logic a, b, cin, output logic sum, carry);

    logic xor_a_b, and_cin_xor0, and_a_b;

    xor0 xor0( a, b, xor_a_b);
    and0 and0( cin, xor_a_b, and_cin_xor0);
    and0 and1( a, b, and_a_b);
    xor0 xor1( xor_a_b, cin, sum);
    or0 or0( and_cin_xor0, and_a_b, carry);
endmodule

module xor0( input logic a, b, output logic out);

    assign out = a ^ b;
endmodule

module and0( input logic a, b, output logic out);

    assign out = a & b;
endmodule

module or0( input logic a, b, output logic out);

    assign out = a | b;
```

```
endmodule
```

Testbench

```
`timescale 1ns / 1ps
module testbench_of_structural_full_adder();

    logic a, b, cin, sum, carry;

    structural_sv_of_full_adder dut( a, b, cin, sum, carry);

    initial begin
        a = 0; b = 0; cin = 0; #10;
        cin = 1; #10;
        b = 1; cin = 0; #10;
        cin = 1; #10;
        a = 1; b = 0; cin = 0; #10;
        cin = 1; #10;
        b = 1; cin = 0; #10;
        cin = 1; #10;
    end
endmodule
```

E) Structural SystemVerilog module for the full subtractor and a testbench for it

Structural SV Module

```
`timescale 1ns / 1ps
 module structural_sv_of_full_subtractor( input logic a, b, bin, output logic d, bo);


    logic xor_a_b, inv_a, and_inv0_b, inv_xor0, and_bin_inv1;


    xor0 xor0( a, b, xor_a_b);
    inv inv0( a, inv_a);
    and0 and0( inv_a, b, and_inv0_b);
    inv inv1( xor_a_b, inv_xor0);
    and0 and1( bin, inv_xor0, and_bin_inv1);
    xor0 xor1( bin, xor_a_b, d);
    or0 or0( and_bin_inv1, and_inv0_b, bo);
endmodule


module xor0( input logic a, b, output logic out);


    assign out = a ^ b;
endmodule


module inv( input logic a, output logic out);


    assign out = ~a;
endmodule


module and0( input logic a, b, output logic out);
```

```
    assign out = a & b;
endmodule


module or0( input logic a, b, output logic out);


    assign out = a | b;
endmodule
```

Testbench

```
`timescale 1ns / 1ps
module testbench_of_structural_full_subtractor();


    logic a, b, cin, d, bo;


    structural_sv_of_full_subtractor dut( a, b, cin, d, bo);


    initial begin
        a = 0; b = 0; cin = 0; #10;
        cin = 1; #10;
        b = 1; cin = 0; #10;
        cin = 1; #10;
        a = 1; b = 0; cin = 0; #10;
        cin = 1; #10;
        b = 1; cin = 0; #10;
        cin = 1; #10;
    end
endmodule
```

F) Structural SystemVerilog module for the 2-bit adder and a testbench for it

Structural SV Module

```
`timescale 1ns / 1ps
module structural_sv_of_two_bit_adder( input logic a0, b0, cin, a1, b1, output logic s0, s1,
cout1);


    logic cout0;


    full_adder full_adder0( a0, b0, cin, s0, cout0);
    full_adder full_adder1( a1, b1, cout0, s1, cout1);
endmodule


module full_adder( input logic a, b, cin, output logic s, cout);


    logic xor_a_b, and_cin_xor0, and_a_b;


    assign xor_a_b = a ^ b;
    assign and_cin_xor0 = cin & xor_a_b;
    assign and_a_b = a & b;
    assign s = xor_a_b ^ cin;
    assign cout = and_cin_xor0 | and_a_b;
endmodule
```

Testbench

```
`timescale 1ns / 1ps
module testbench_of_structural_two_bit_adder();


    logic a0, b0, cin, a1, b1, cout0, s0, s1, cout1;
```

```
structural_sv_of_two_bit_adder dut( a0, b0, cin, a1, b1, s0, s1, cout1);


initial begin
    a0 = 0; b0 = 0; cin = 0; a1 = 0; b1 = 0; #10;
    b1 = 1; #10;
    a1 = 1; b1 = 0; #10;
    b1 = 1; #10;
    cin = 1; a1 = 0; b1 = 0; #10;
    b1 = 1; #10;
    a1 = 1; b1 = 0; #10;
    b1 = 1; #10;
    b0 = 1; cin = 0; a1 = 0; b1 = 0; #10;
    b1 = 1; #10;
    a1 = 1; b1 = 0; #10;
    b1 = 1; #10;
    cin = 1; a1 = 0; b1 = 0; #10;
    b1 = 1; #10;
    a1 = 1; b1 = 0; #10;
    b1 = 1; #10;
    a0 = 1; b0 = 0; cin = 0; a1 = 0; b1 = 0; #10;
    b1 = 1; #10;
    a1 = 1; b1 = 0; #10;
    b1 = 1; #10;
    cin = 1; a1 = 0; b1 = 0; #10;
    b1 = 1; #10;
    a1 = 1; b1 = 0; #10;
    b1 = 1; #10;
    b0 = 1; cin = 0; a1 = 0; b1 = 0; #10;
    b1 = 1; #10;
    a1 = 1; b1 = 0; #10;
```

```
        b1 = 1; #10;
        cin = 1; a1 = 0; b1 = 0; #10;
        b1 = 1; #10;
        a1 = 1; b1 = 0; #10;
        b1 = 1; #10;
    end
endmodule
```