**DIGITAL DESIGN**

**CS223**

**LAB-02 PRELIMINARY REPORT**

**NAME: MUHAMMAD ARHAM KHAN**

**BILKENT ID: 21701848**

**SECTION: 2**

**DATE: 23 Oct, 2018**

**TRAINING PACK: 47**

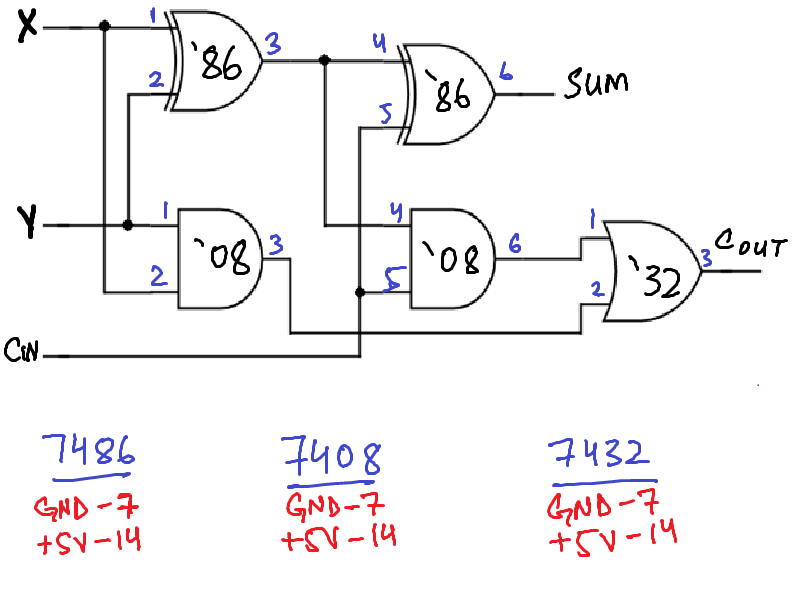
**1-BIT FULL ADDER**

**BOOLEAN EQUATION:**

SUM = (X XOR Y) XOR CIN

COUT = ((X XOR Y) . CIN) + (X . Y)

**CIRCUIT SCHEMATIC:**

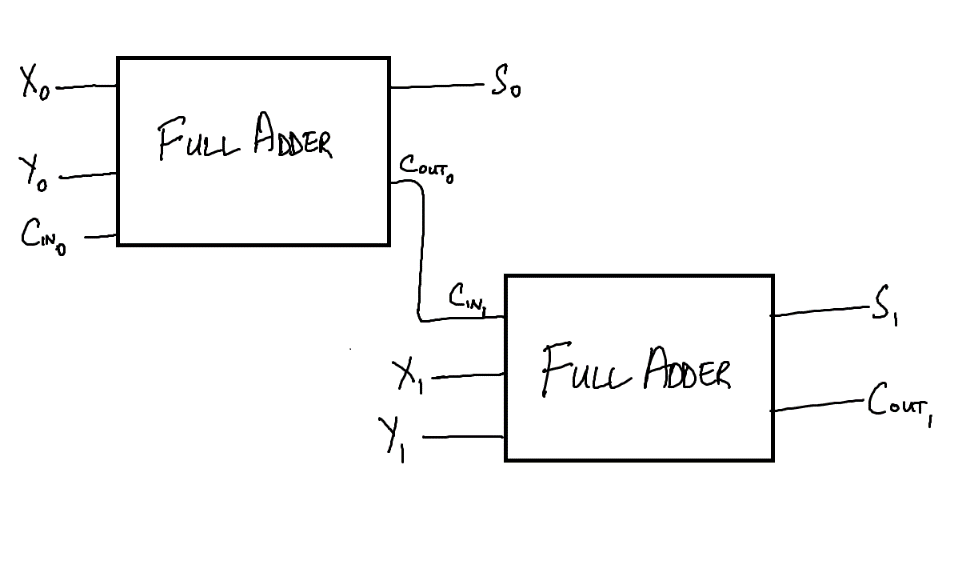


**IC LIST:**

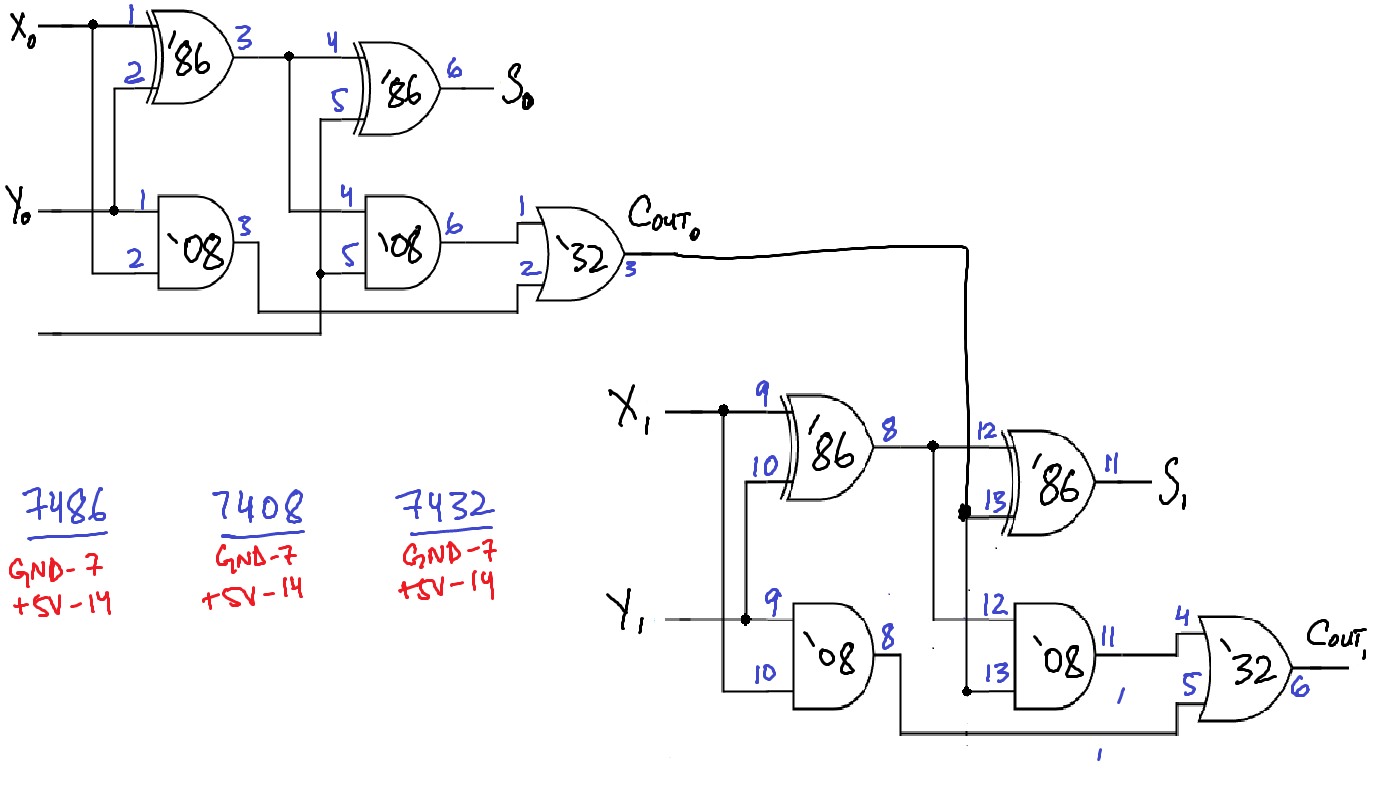
* ONE 7408 QUAD 2-INPUT AND GATE
* ONE 7486 QUAD 2-INPUT XOR GATE
* ONE 7432 QUAD 2-INPUT OR GATE

**2-BIT ADDER**

**LOGIC DIAGRAM (BLACK-BOXED):**



**CIRCUIT SCHEMATIC:**



**IC LIST:**

* ONE 7408 QUAD 2-INPUT AND GATE
* ONE 7486 QUAD 2-INPUT XOR GATE
* ONE 7432 QUAD 2-INPUT OR GATE

**1-BIT FULL ADDER DATAFLOW(BEHAVORIAL) SYSTEMVERILOG MODULES**

module oneBitFullAdder( input logic x, y, cin

Output logic s, cout);

logic p, g;

assign p=x^y;

assign g=x&y;

assign s=p^cin;

assign cout=g | (p & cin);

endmodule

**TESTBENCH:**

module oneBitFullAdder\_testb();

logic x, y, cin, s, cout;

//instantiate the device in testbench

oneBitFullAdder dut( x, y, cin, s, cout);

//start applying inputs( one by one)

initial begin

x=0; y=0; cin=0; #10;

cin=1; #10;

y=1; cin=0; #10;

cin=1; #10;

x=1; y=0; cin=0; #10;

cin=1; #10;

y=1; cin=0; #10;

cin=1; #10;

end

endmodule

**1-BIT FULL ADDER STRUCTURAL SYSTEMVERILOG MODULES**

module oneBitFullAdder( input logic x, y, cin

Output logic s, cout);

logic a, b, c;

xor( a, x, y);

and( b, x, y);

xor( s, a, cin);

and( c, a, cin);

or( cout, b, c);

endmodule

**TESTBENCH:**

module oneBitFullAdder\_testb();

logic x, y, cin, s, cout;

//instantiate the device in testbench

oneBitFullAdder dut( x, y, cin, s, cout);

//start applying inputs( one by one)

initial begin

x=0; y=0; cin=0; #10;

cin=1; #10;

y=1; cin=0; #10;

cin=1; #10;

x=1; y=0; cin=0; #10;

cin=1; #10;

y=1; cin=0; #10;

cin=1; #10;

end

endmodule

**2-BIT ADDER STRUCTURAL SYSTEMVERILOG MODULES**

module twoBitAdder( input logic[1:0] x, y ,

input logic cin0,

output logic[1:0] s,

output logic cout1);

logic cout0;

oneBitFullAdder( x[0], y[0], cin0, s[0], cout0);

oneBitFullAdder( x[1], y[1], cout0, s[1], cout1);

endmodule

**TESTBENCH:**

module twoBitAdder\_testb();

logic[1:0] x, y, s;

logic cin0, cout1;

//instantiate the device in testbench

twoBitAdder dut( x, y, cin0, s, cout1);

//start applying inputs( one by one)

initial begin

x[0]=0; x[1]=0; y[0]=0; y[1]=0; cin0=0; #10;

cin0=1; #10;

y[1]=1; cin0=0; #10;

cin0=1; #10;

y[0]=1; y[1]=0; cin0=0; #10;

cin0=1; #10;

y[1]=1; cin0=0; #10;

cin0=1; #10;

x[1]=1; y[0]=0; y[1]=0; cin0=0; #10;

cin0=1; #10;

y[1]=1; cin0=0; #10;

cin0=1; #10;

y[0]=1; y[1]=0; cin0=0; #10;

cin0=1; #10;

y[1]=1; cin0=0; #10;

cin0=1; #10;

x[0]=1; x[1]=0; y[0]=0; y[1]=0; cin0=0; #10;

cin0=1; #10;

y[1]=1; cin0=0; #10;

cin0=1; #10;

y[0]=1; y[1]=0; cin0=0; #10;

cin0=1; #10;

y[1]=1; cin0=0; #10;

cin0=1; #10;

x[1]=1; y[0]=0; y[1]=0; cin0=0; #10;

cin0=1; #10;

y[1]=1; cin0=0; #10;

cin0=1; #10;

y[0]=1; y[1]=0; cin0=0; #10;

cin0=1; #10;

y[1]=1; cin0=0; #10;

cin0=1; #10;

end

endmodule