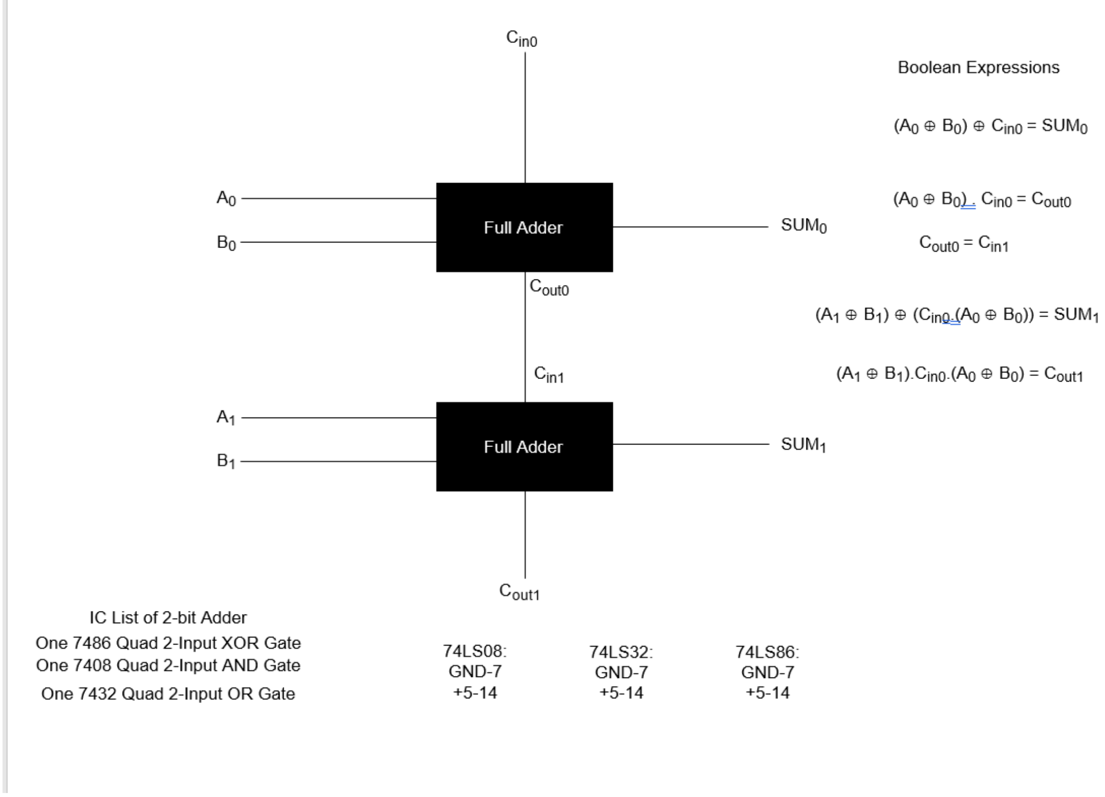
CS223-05 LABORATORY WORK

LAB-02(12.10.2020)

Remzi Tepe
21802713

Circuit Schematic for 2-bit Adder



Boolean Expressions

$(A_0 \oplus B_0) \oplus C_{in0} = SUM_0$

$(A_0 \oplus B_0) . C_{in0} = C_{out0}$

$C_{out0} = C_{in1}$

$(A_1 \oplus B_1) \oplus (C_{in0}.(A_0 \oplus B_0)) = SUM_1$

$(A_1 \oplus B_1).C_{in0}.(A_0 \oplus B_0) = C_{out1}$

IC List of 2-bit Adder

One 7486 Quad 2-Input XOR Gate
One 7408 Quad 2-Input AND Gate
One 7432 Quad 2-Input OR Gate

74LS08:	74LS32:	74LS86:
GND-7	GND-7	GND-7
+5-14	+5-14	+5-14

Behavioral SystemVerilog module for Full Adder:

```
module fullAdder (input logic a, b, cin,
                  output logic sum, cout);
        assign sum = (a ^ b) ^ cin;
        assign cout = (a^b) & cin | (a & b);
endmodule
```

Structural SystemVerilog module for Full Adder:

```
module xor2(input logic a, b,
          output logic c);
        assign c = a ^ b;
endmodule


module and2(input logic a, b,
          output logic c);
        assign c = a & b;
endmodule
module or2(input logic a, b,
          output logic c);
        assign c = a | b;
endmodule
module fullAdder(input logic a, b, cin
                 output logic sum, cout);
        logic n1, n2, n3;
        xor2 xorgate(a, b, n1);
        xor2 xorgate2(n1, cin, sum);

        and2 andgate(n1, cin, n2);

        and2 andgate2(a, b, n3);

        or2 orgate(n2, n3, cout);

endmodule
```

Testbench for Full Adder:

```
module testbenchAdder();

        logic a, b, cin;

        logic sum, cout;

        fullAdder dut(a, b, cin, sum, cout);

        initial begin

        a = 0; b = 0; cin = 0; #10;

        c = 1;                 #10;

        b = 1; c = 0;          #10;

        c = 1;                 #10;

        a = 1; b = 0; c = 0;   #10;

        c = 1;                #10;

        b = 1; c = 0;        #10;

        c = 1;              #10;

    end

endmodule
```

Structural SystemVerilog module for Full Subtractor:

```
module xor2(input logic a, b,

          output logic c);
```

```
assign c = a ^ b;
endmodule
module and2(input logic a, b,
          output logic c);
        assign c = a & b;
endmodule
module or2(input logic a, b,
          output logic c);
        assign c = a | b;
endmodule
module inv(input logic a,
          output logic b);
        assign b = ~a;
module fullSubtractor(input logic a, b, bin
                 output logic d, bo);
        logic n1, n2, n3, n4, n5;
        xor2 xorgate(a, b, n1);
        inv inv1(a,n2);
        and2 andgate(n2, b, n3);
        xor2 xorgate2(bin, n1, d);
        inv inv2(n1, n4);
        and2 andgate2(n4, bin, n5);
        or2 orgate(n3, n5, bo);
endmodule
```

Testbench for full Subtractor:

```
module testbenchSubtractor();

            logic a, b, bin;

            logic d, bo;

            fullSubtractor dut(a, b, bin, d, bo);

            initial begin

            a = 0; b = 0; bin = 0; #10;

            bin = 1;                   #10;

            b = 1; bin = 0;          #10;

            bin = 1;             #10;

            a = 1; b = 0; bin = 0; #10;

            bin = 1;             #10;

            b = 1; bin = 0;       #10;

            bin = 1;             #10;

      end

endmodule
```

Structural SystemVerilog module for 2-bit Adder:

```
module fullAdder (input logic a, b, cin,

                          output logic sum, cout);

            assign sum = (a ^ b) ^ cin;

            assign cout = (a^b) & cin | (a & b);

endmodule

module 2bitAdder(input logic a0, b0, cin0, a1, b1, cin1

                         output logic sum0, cout0, sum1, cout1);

            logic n1;
```

```
        fullAdder fullAdder1(a0, b0, cin0, sum0, n1);

        fullAdder fullAdder2(a1, b1, n1, sum1, cout1);

endmodule
```

Testbench for 2-bit Adder:

```
module testbench2bitAdder();

        logic a, b, bin;

        logic d, bo;

        2bitAdder dut(a0, b0, cin0, a1, b1, cin1, sum0, cout0, sum1, 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
```