CS223 - DIGITAL DESIGN

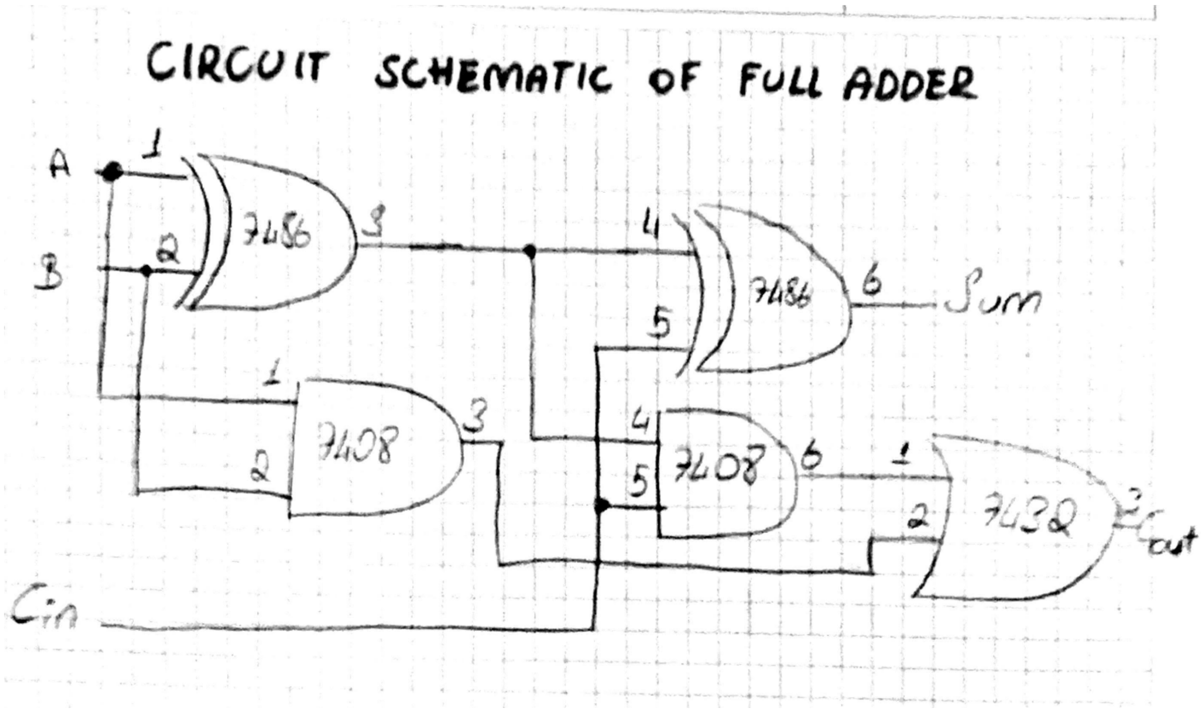
SECTION 3 - LAB 2

ARDA TAVUSBAY

21902722

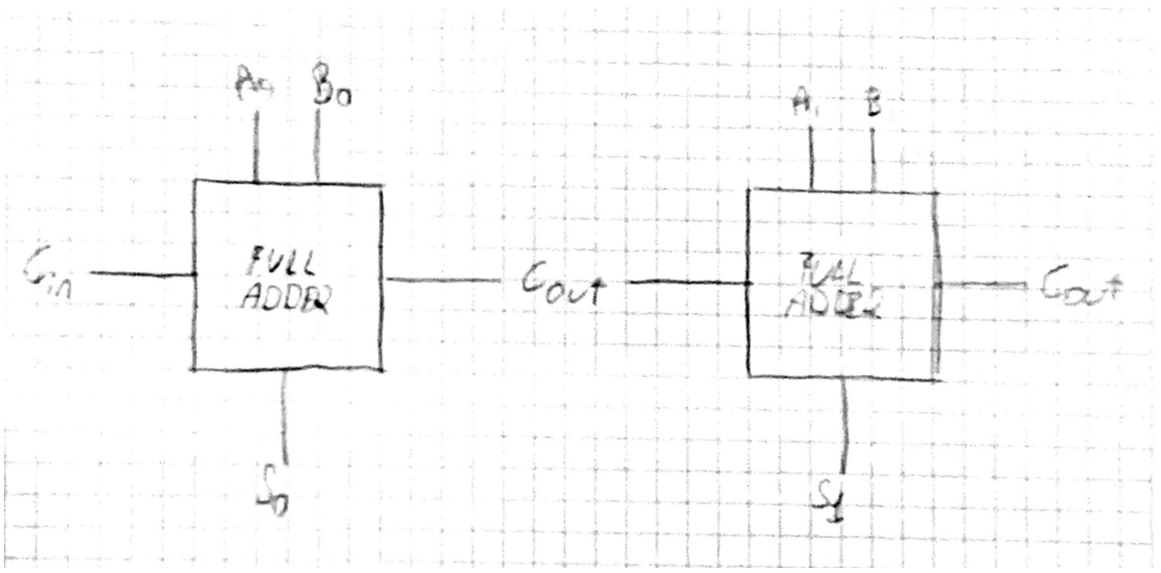
17/10/2021

(B) Circuit schematic for full adder using 2-input XOR, AND, and OR gates

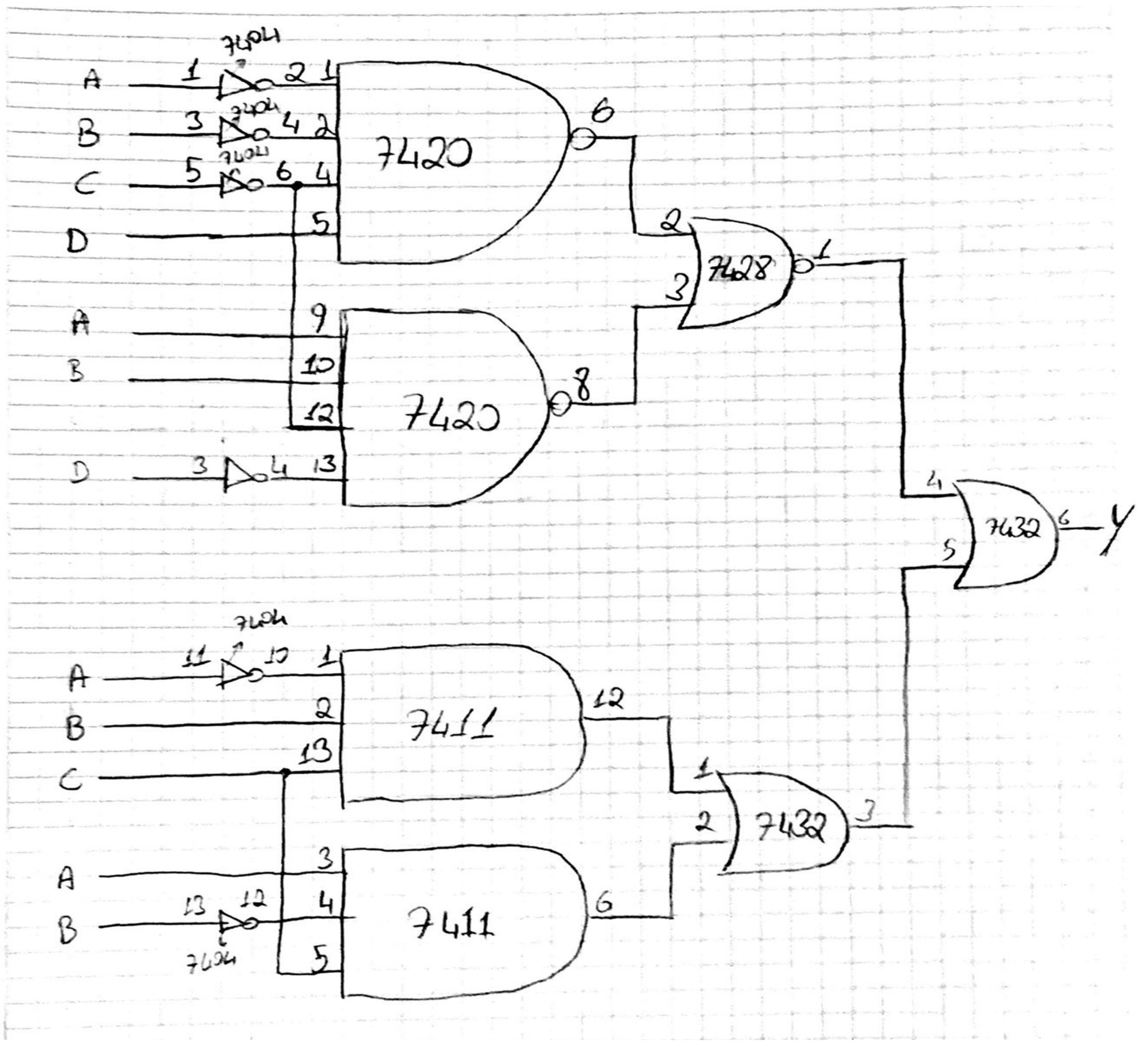


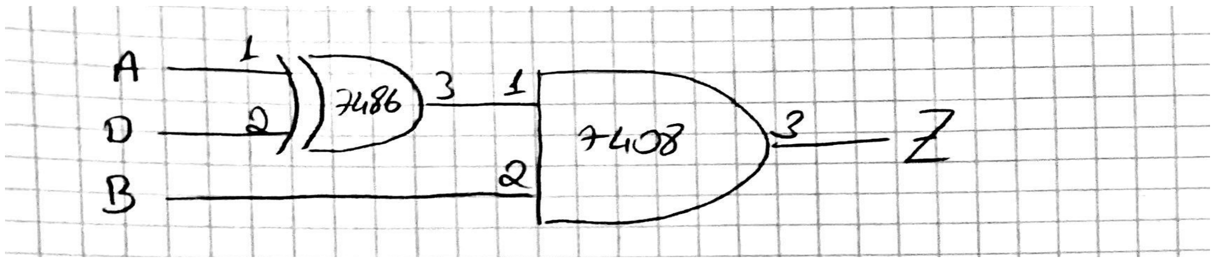
IC LIST:
1-) One Quad 2-Input XOR Gate (7486)
2-) One Quad 2-Input AND Gate (7408)
3-) One Quad 2-Input OR Gate (7432)

(C) Circuit schematic for a 2-bit adder made from two full adders



(D) Circuit Schematic for Lab Calculator (Made with K-maps)





IC LIST:

For Y:

1-) One Dual 4-Input NAND Gate (7420)

2-) One Hex Inverter NOT Gate (7404)

3-) One Quad 2-Input NOR Gate (7428)

4-) One Triple 3-Input AND Gate (7411)

5-) One Quad 2-Input OR Gate (7432)

For Z:

1-) One Quad 2-Input XOR Gate (7486)

2-) One Quad 2-Input AND Gate (7408)

(E) Behavioral SystemVerilog Module for the Full Adder and a Testbench

Behavioral Full Adder:

```
`timescale 1ns / 1ps

module BehavioralFullAdder( input logic a, b, cin, output logic sum, cout );

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

endmodule
```

Testbench:

```
`timescale 1ns / 1ps

module TestBenchBehavioralFA();

   logic a, b, cin, sum, cout;
   BehavioralFullAdder dut( a, b, cin, sum, cout );

   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 Full Adder and a Testbench

Structural Full Adder:

```
`timescale 1ns / 1ps

module StructuralFullAdder( input logic a, b , cin, output logic sum, cout );

   logic x, y, z;
   XorGate xorgate( a, b, x );
   XorGate xorgate2( x, cin, sum );
   AndGate andgate( x, cin, y  );
   AndGate andgate2( a, b, z  );
   OrGate orgate( y, z, cout );

endmodule

module AndGate( input logic a, b, output logic c);

   assign c = a & b;

endmodule

module XorGate( input logic a, b, output logic c);

   assign c = a ^ b;

endmodule

module OrGate( input logic a, b, output logic c);

   assign c = a | b;

endmodule
```

Testbench:

```
`timescale 1ns / 1ps

module TestBenchStructuralFA();

   logic a, b, cin, sum, cout;
   StructuralFullAdder dut( a, b, cin, sum, cout );

   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
```

(G) Structural SystemVerilog Module for the 2-bit Adder and a Testbench for it. Use the Full Adder Module in part (f).

Structural 2-Bit Adder:

```
`timescale 1ns / 1ps

module StructuralTwoBitAdder( input logic a0, a1, b0, b1, cin, output logic sum0, sum1, cout);

    logic firstCout;
    StructuralFullAdder fa1(a0, b0, cin, sum0, firstCout);
    StructuralFullAdder fa2(a1, b1, firstCout, sum1, cout);

endmodule
```

TestBench:

```
`timescale 1ns / 1ps

module TestBenchTwoBitAdder();

    logic a0, a1, b0, b1, cin, sum0, sum1, cout;
    StructuralTwoBitAdder dut( a0, a1, b0, b1, cin, sum0, sum1, cout );

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

endmodule
```

(H) Structural SystemVerilog Module for the Lab Calculator and a Testbench for it.

Structural Lab Calculator:

```
`timescale 1ns / 1ps

module LabCalculator( input logic c, d, a, b, output logic y, z);

   logic temp1, temp2, temp3, temp4;
   HalfSubtractor hs(a, b, temp1, temp2);
   HalfAdder ha(a, b, temp3, temp4);
   assign y = c ? (d ? temp1 : temp3) : (d ? ~(a | b) : (a & b));
   assign z = c ? (d ? temp2 : temp4) : (d ? 0 : 0);

endmodule

module HalfAdder( input logic a, b, output logic sum, carry);

   assign sum = a ^ b;
   assign carry = a & b;

endmodule

module HalfSubtractor( input logic a, b, output logic diff, borrow );

   assign diff = a ^ b;
   assign borrow = ~a & b;

endmodule
```

Testbench:

```
`timescale 1ns / 1ps

module TestBenchLabCalculator();

   logic c, d, a, b, y, z;
   LabCalculator dut( c, d, a, b, y, z );

   initial begin
      c = 0; d = 0; a = 0; b = 0; #10;
      c = 0; d = 0; a = 0; b = 1; #10;
      c = 0; d = 0; a = 1; b = 0; #10;
      c = 0; d = 0; a = 1; b = 1; #10;
      c = 0; d = 1; a = 0; b = 0; #10;
      c = 0; d = 1; a = 0; b = 1; #10;
      c = 0; d = 1; a = 1; b = 0; #10;
      c = 0; d = 1; a = 1; b = 1; #10;
      c = 1; d = 0; a = 0; b = 0; #10;
      c = 1; d = 0; a = 0; b = 1; #10;
      c = 1; d = 0; a = 1; b = 0; #10;
      c = 1; d = 0; a = 1; b = 1; #10;
      c = 1; d = 1; a = 0; b = 0; #10;
      c = 1; d = 1; a = 0; b = 1; #10;
      c = 1; d = 1; a = 1; b = 0; #10;
      c = 1; d = 1; a = 1; b = 1; #10;
   end

endmodule
```