Midterm 2 – EE4301

Problem 1. (20 pts.) The following code is written using structural Verilog – rewrite this code for a behavioral version (note that we have delays in this code).

```
module expl_str(x,y,a,b,c);
        input a, b, c;
        output x, y;
        wire a, b, c, x, y;
        wire na, nb, nc, t3, t5, t6;
        not n1(na,a);
        not n2(nb,b);
        not n3(nc,c);
        and #1 a1(t3,na,b,c);
        and a2(t5,a,nb,c);
        and a3(t6,a,b,nc);
        or o1(x,t3,t6);
        or #3 o2(y,a,t5);
endmodule
```

Problem 2 (20 pts.) : Given the following Verilog codes below Draw a RTL – hardware realization of these codes.

Code A (10 pts)

```
module tproblem_1(x, y, z, a, b, op);
        input a, b, op;
        output x, y, z;
        wire [7:0] a, b;
        wire [1:0] op;
        reg [7:0] x, y, z;
        always @( op or a or b ) begin
        if ( a == 0 ) y = b;
        if ( a < b ) z = a; else z = b;
        case ( op )
                0: x = a + b;
                1: x = a;
                2: x = b;
        endcase
end
endmodule
```

CodeB (10 pts.)

```
module tproblem-2(output [6:0] sum, input [15:0] nibbles, a, b, c);
        logic [15:0] n2;
        logic last_c;
        always @( posedge a )
        if ( !b ) begin
        sum = 0;
        end else begin
        if ( c != last_c ) begin
        n2 = nibbles;
        for ( int i=0; i < 4; i++ ) begin
        sum = sum + n2[3:0];
        n2 = n2 >> 4;
        end
end
        last_c = c;
end
endmodule
```

Problem 3 (15 pts.) A 5-variable function f(a, b, c, d, e) is expanded about a using Shannon's Expansion theorem to get

$$f(a, b, c, d, e) = a' f1(b, c, d, e) + a .f2(b, c, d, e).$$

Show how this would be implemented using two LUT4s and a 2-1 MUX. Label all inputs and outputs for the LUT4s and the MUX.

Problem 4 (20pts.). Consider computing the product of "10 1010 ×10 1001" using the shift-add multiplier, for unsigned binary numbers. Both the multiplicand, "10 1010", and multiplier, "10 1001", are 6-bit unsigned binary integers. Assume a 12-bit product register and an adder with the appropriate number of bits. The multiplier is initially loaded in the least significant 6 bits of the product register and shifts are to the right.

(a) **(5 pts.)Show the initial contents of the product register and multiplicand register**
(b) **(10 pts.)Identify all arithmetic and shift operations for each iteration in the correct order. You donot need to show the contents of the product register after each iteration.**

(c) **(5 pts.) Show the contents of the product register after the first iteration.**

Problem 5 (25 pts.) Given the Flow chart below – write a Verilog code to implement the operation of this chart: