lamin Y. Jammeh
EE417 Summer 2024
HW02

UDP1 Verilog code

```
primitive Adder_sum (sum_lsb, c_in, a, b);
output sum_lsb;
input c_in, a, b;
//define the truth table [note only enter the LSB as the output]
table
//if ans=0 therfore LSB=0 if ans=1 therfore LSB=1 if ans=2 therfore LSB=0 if ans=3 therfore LSB=1

// c_in, a,  b:  sum_lsb    make sure this follows the defination of he primitive from above
    0   0   0:    0;
    0   0   1:    1;
    0   1   0:    1;
    0   1   1:    0;
    1   0   0:    1;
    1   0   1:    0;
    1   1   0:    0;
    1   1   1:    1;
    endtable
    endprimitive
```

UDP2 Verilog code

```
primitive Adder_carry (carry_msb, c_in, a, b);

//define the inouts and outputs
output carry_msb;
input c_in, a, b;

//define the truth table [note only enter the LSB as the output]
table
//if ans=0 therfore MSB=0 if ans=1 therfore MSB=0 if ans=2 therfore MSB=1 if ans=3 therfore MSB=1

// c_in, a,  b:  carry_msb    make sure this follows the defination of he primitive from above
    0     0   0:    0;
    0     0   1:    0;
    0     1   0:    0;
    0     1   1:    1;
    1     0   0:    0;
    1     0   1:    1;
    1     1   0:    1;
    1     1   1:    1;
    endtable
    endprimitive
```

lamin Y. Jammeh
EE417 Summer 2024
HW02

Full Adder Module

```
module full_adder(
    input a,
    input b,
    input c_in,
    output sum,
    output carry
    );
//wire sum;
//wire carry;

 // Instantiate the Adder_sum
Adder_sum UDP1(
    .a(a),
    .b(b),
    .c_in(c_in),
    .sum_lsb(sum)
);

 // Instantiate the Adder_carry
Adder_carry UDP2(
    .a(a),
    .b(b),
    .c_in(c_in),
    .carry_msb(carry)
);

 endmodule
```

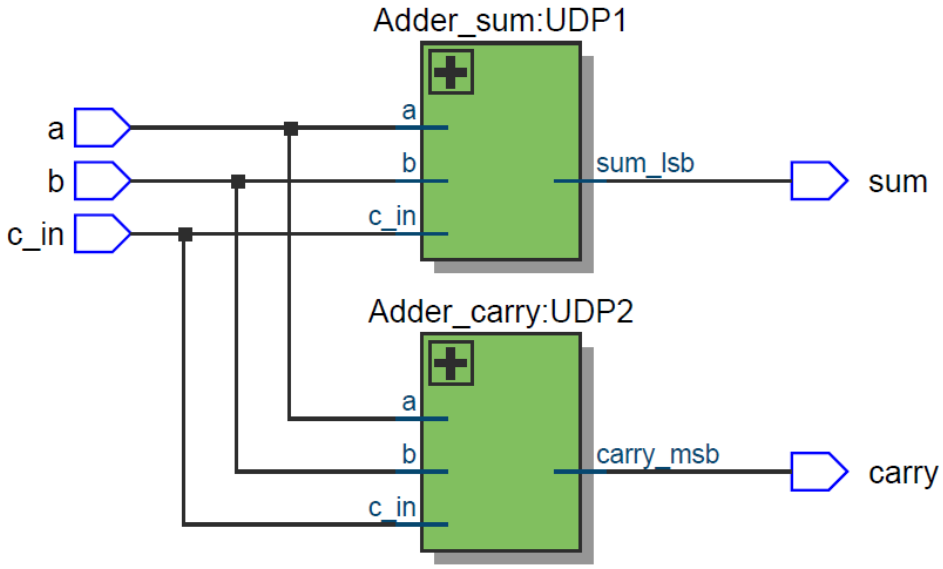
Test-Bench for Full Adder

```
module full_adder_tb();  //defines the module for the test bench

//defines the output as a wire
wire    sum;
wire carry;

reg a, b, c_in;        //defines the inputs as a register

//instantiate the Module under test or UUT=unit under test
full_adder UUT (a, b, c_in, sum, carry);

//Test the different combination to check if the output is the expected answer
initial                     //initial block
    begin                       //begine block
        a = 0; b = 0; c_in = 0;
        #10000;
        a = 0; b = 0; c_in = 1;
        #10000;
        a = 0; b = 1; c_in = 0;
        #10000;
        a = 0; b = 1; c_in = 1;
        #10000;
        a = 1; b = 0; c_in = 0;
        #10000;
        a = 1; b = 0; c_in = 1;
        #10000;
        a = 1; b = 1; c_in = 0;
        #10000;
        a = 1; b = 1; c_in = 1;
        #10000;

    end               // ends the begin block for the different combinations
endmodule          //ends the entire module
```

---

lamin Y. Jammeh
EE417 Summer 2024
HW02

Netlist of Full Adder



Four-bit Input Adder

Verilog Code for one bit full adder

```
''
module oneBit_full_Adder(A, B, C_in, Sum, C_out);

input A, B, C_in;
output Sum, C_out;

assign {C_out, Sum} = A + B + C_in;

endmodule
```

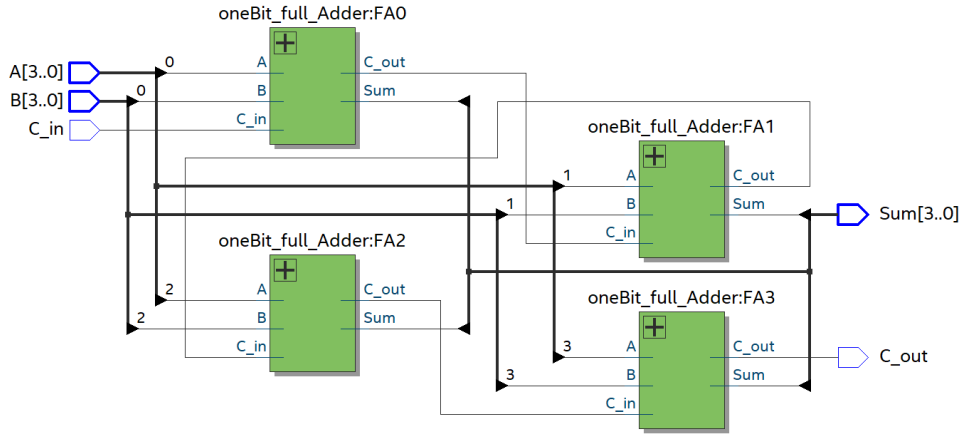
lamin Y. Jammeh
EE417 Summer 2024
HW02

Verilog Code for four bit full adder

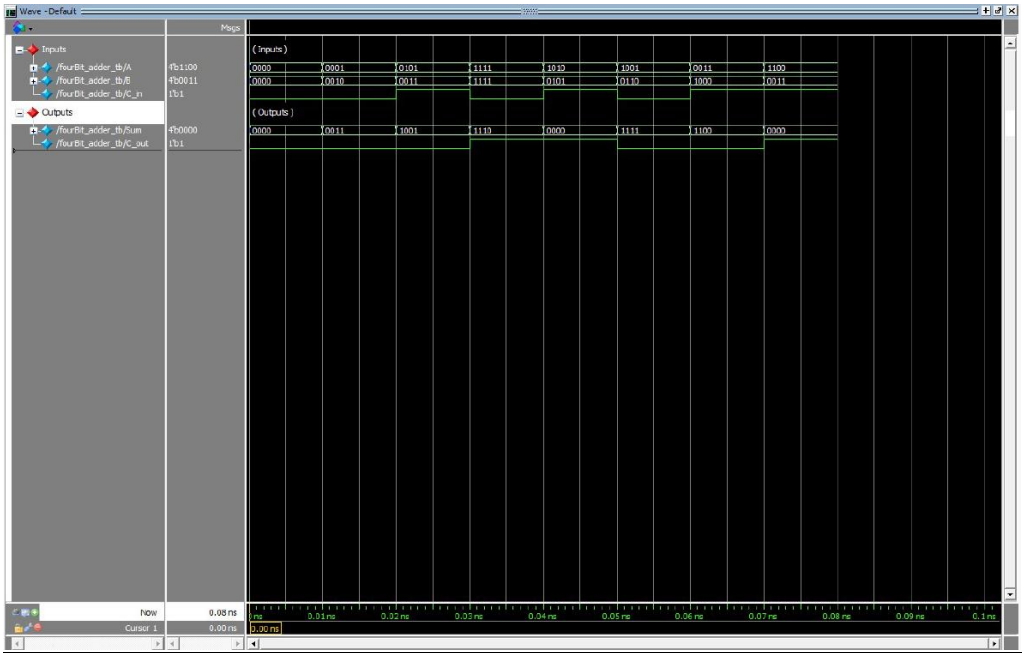
```
module fourBit_adder (
    input [3:0] A,
    input [3:0] B,
    input C_in,
    output [3:0] Sum,
    output C_out
);

     wire C1, C2, C3; //define all the wire between the oneBit full adders

    // Instantiate four oneBit full adders and call then FA[3:0]
    oneBit_full_Adder FA0 (
        .A(A[0]),
        .B(B[0]),
        .C_in(C_in),
        .Sum(Sum[0]),
        .C_out(C1)
    );

    oneBit_full_Adder FA1 (
        .A(A[1]),
        .B(B[1]),
        .C_in(C1),
        .Sum(Sum[1]),
        .C_out(C2)
    );

    oneBit_full_Adder FA2 (
        .A(A[2]),
        .B(B[2]),
        .C_in(C2),
        .Sum(Sum[2]),
        .C_out(C3)
    );

    oneBit_full_Adder FA3 (
        .A(A[3]),
        .B(B[3]),
        .C_in(C3),
        .Sum(Sum[3]),
        .C_out(C_out)
    );

endmodule
```

lamin Y. Jammeh
EE417 Summer 2024
HW02

Netlist of four bit Full Adder



Wave Form four bit full adder



lamin Y. Jammeh
EE417 Summer 2024
HW02

Part 2

Verilog Code for divBy3

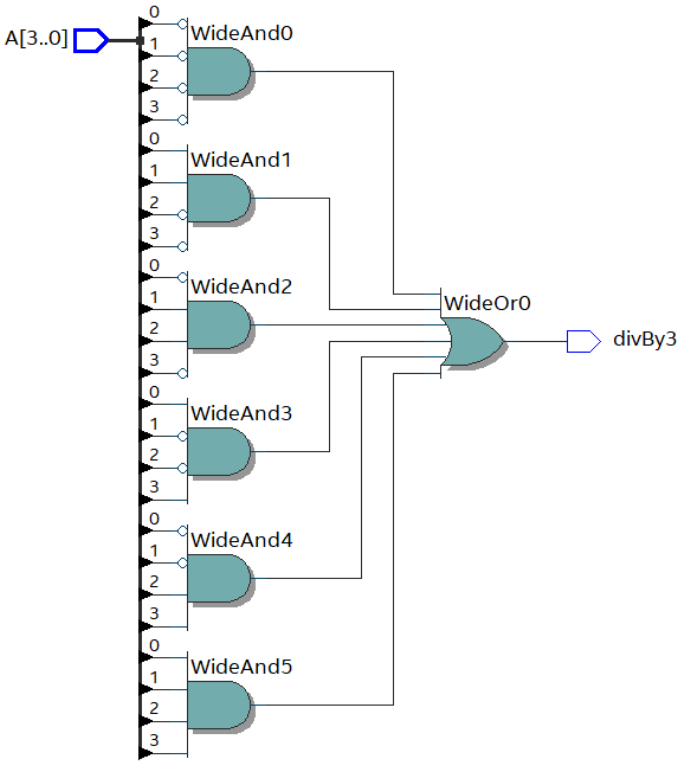
```
//define the module name and 4 bit input with single bit output
module divBy3(
    input [3:0] A,
    output divBy3
);
    //deine all the internal wires
    wire nA, nB, nC, nD;
    wire t1, t2, t3, t4;
    //define inverted logical inputs
    not(nA, A[3]);
    not(nB, A[2]);
    not(nC, A[1]);
    not(nD, A[0]);

    //define the intermediate logics
    and(t1, nA,   nB,   nC,   nD);          //t1 = A' B' C' D'
    and(t2, nA,   nB,   A[1], A[0]);        //t2 = A' B' C  D
    and(t3, nA,   A[2], A[1], nD);          //t3 = A' B  C  D'
    and(t4, A[3], nB,   nC,   A[0]);        //t4 = A  B  C  D
    and(t5, A[3], A[2], nC,   nD);          //t5 = A  B  C' D'
    and(t6, A[3], A[2], A[1], A[0]);        //t6 = A  B  C  D
    or(divBy3, t1, t2, t3, t4, t5, t6);
endmodule
```

Test-Bench for diivBy3

```
module divBy3_tb();
    reg [3:0] A;
    wire divBy3;

    // Instantiate the divBy5 module
    divBy3 uut (
        .A(A),
        .divBy3(divBy3)
    );

    initial
    begin
        A = 4'b0000; #10;  //0
        A = 4'b0001; #10;  // 1
        A = 4'b0010; #10;  // 2
        A = 4'b0011; #10;  // 3
        A = 4'b0100; #10;  // 4
        A = 4'b0101; #10;  // 5
        A = 4'b0110; #10;  // 6
        A = 4'b0111; #10;  // 7
        A = 4'b1000; #10;  // 8
        A = 4'b1001; #10;  // 9
        A = 4'b1010; #10;  // 10
        A = 4'b1011; #10;  // 11
        A = 4'b1100; #10;  // 12
        A = 4'b1101; #10;  // 13
        A = 4'b1110; #10;  // 14
        A = 4'b1111; #10;  // 15
    end
endmodule
```

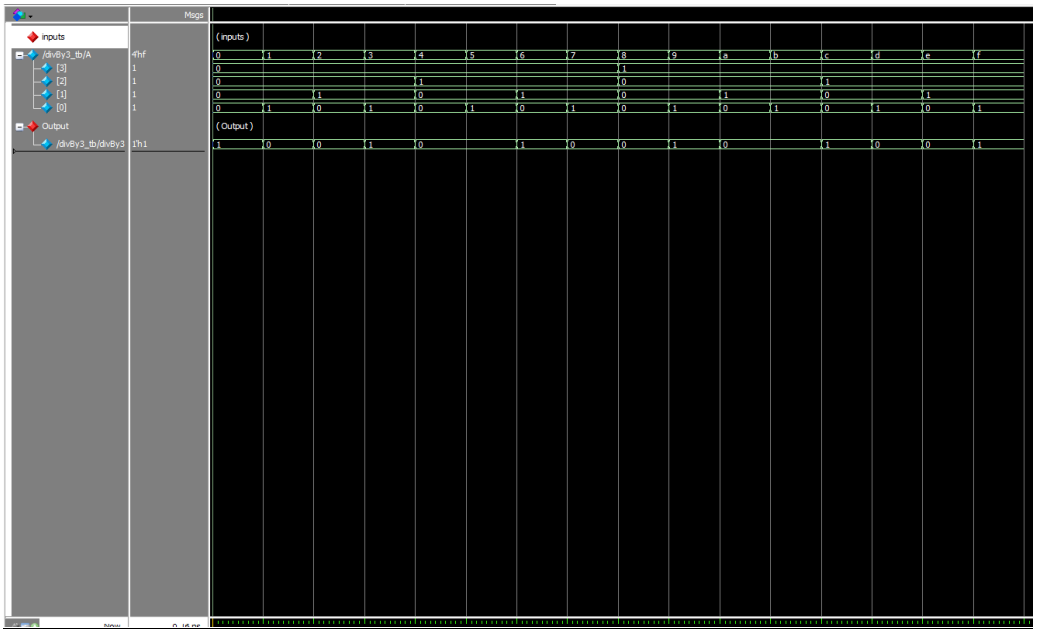
lamin Y. Jammeh
EE417 Summer 2024
HW02

Netlist divBy5



Wave Form divBy5

lamin Y. Jammeh
EE417 Summer 2024
HW02



Verilog Code for divBy5

```
//define the module name and 4 bit input with single bit output
module divBy5(
    input [3:0] A,
    output divBy5
);
    //deine all the internal wires
    wire nA, nB, nC, nD;
    wire t1, t2, t3, t4;
    //define inverted logical inputs
    not(nA, A[3]);
    not(nB, A[2]);
    not(nC, A[1]);
    not(nD, A[0]);

    //define the intermediate logics
    and(t1, nA, nB, nC, nD);            //t1 = A' B' C' D'
    and(t2, nA, A[2], nC, A[0]);        //t2 = A' B  C' D
    and(t3, A[3], nB, A[1], nD);        //t3 = A  B' C  D'
    and(t4, A[3], A[2], A[1], A[0]);    //t4 = A  B  C  D
    or(divBy5, t1, t2, t3, t4);
endmodule
```

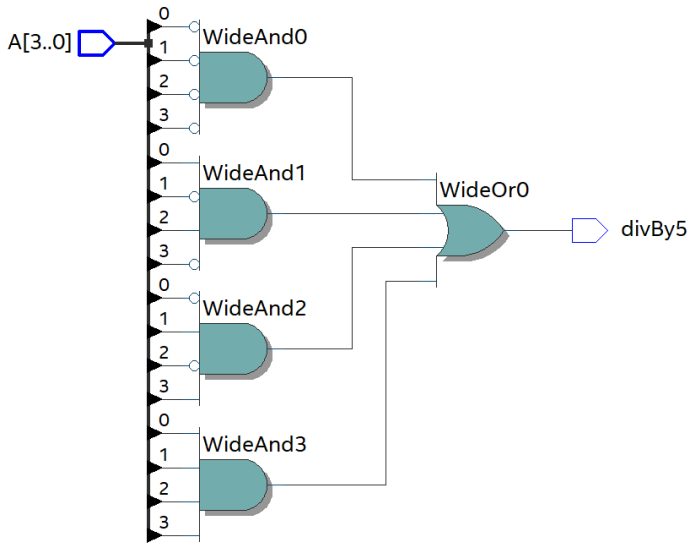
Test-Bench for diivBy5

lamin Y. Jammeh
EE417 Summer 2024
HW02

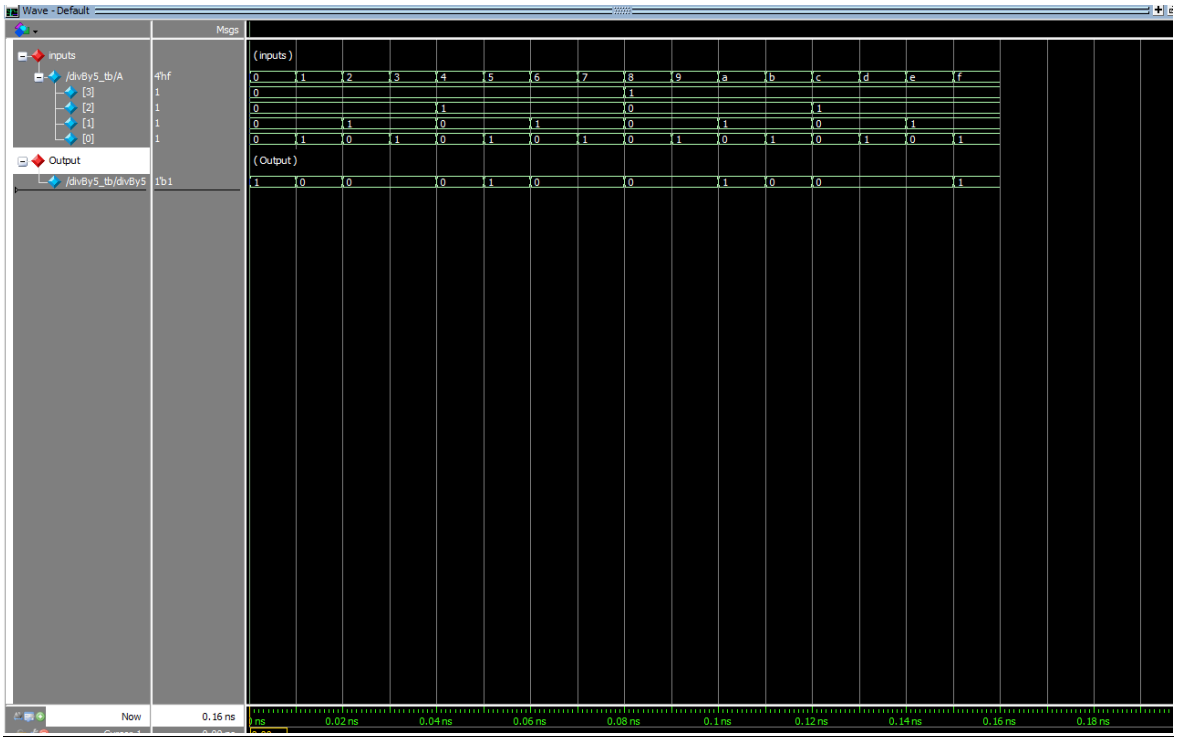
```
module divBy5_tb();
    reg [3:0] A;
    wire divBy5;

    // Instantiate the divBy5 module
    divBy5 uut (
        .A(A),
        .divBy5(divBy5)
    );

    initial begin

        // Test cases
        A = 4'b0000; #10;   // 0
        A = 4'b0001; #10;   // 1
        A = 4'b0010; #10;   // 2
        A = 4'b0011; #10;   // 3
        A = 4'b0100; #10;   // 4
        A = 4'b0101; #10;   // 5
        A = 4'b0110; #10;   // 6
        A = 4'b0111; #10;   // 7
        A = 4'b1000; #10;   // 8
        A = 4'b1001; #10;   // 9
        A = 4'b1010; #10;   // 10
        A = 4'b1011; #10;   // 11
        A = 4'b1100; #10;   // 12
        A = 4'b1101; #10;   // 13
        A = 4'b1110; #10;   // 14
        A = 4'b1111; #10;   // 15
    end
endmodule
```

lamin Y. Jammeh
EE417 Summer 2024
HW02

Netlist divBy5



Wave Form divBy5



lamin Y. Jammeh
EE417 Summer 2024
HW02

Verilog Code for combine divider

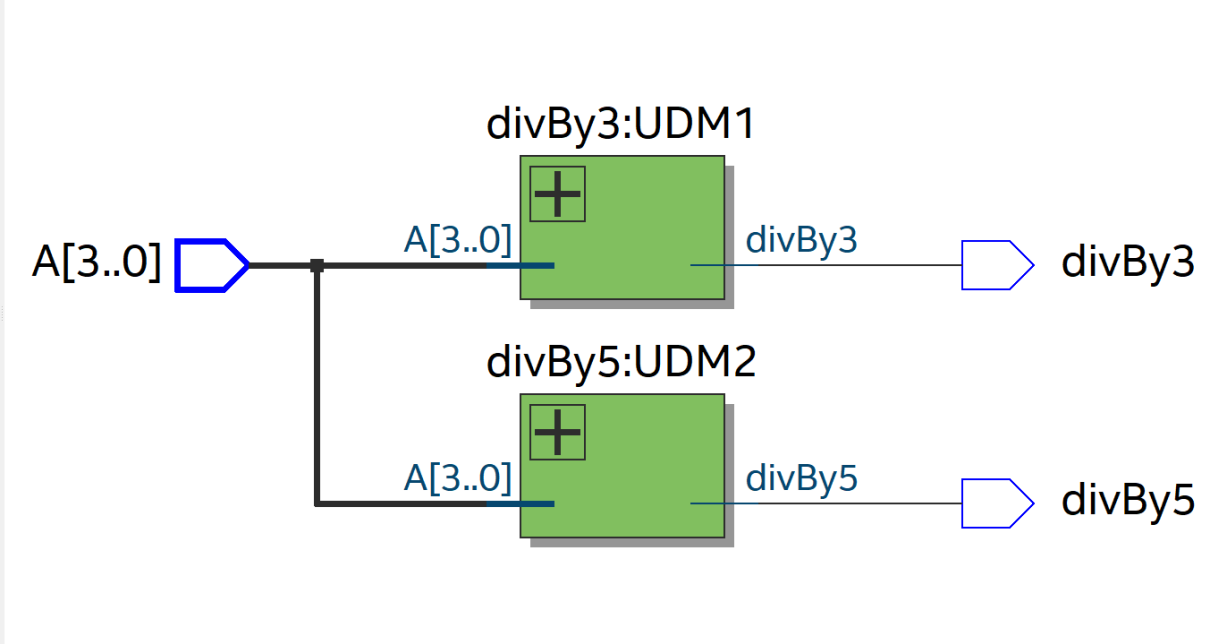
```
module combine_divBy3and5(
          input [3:0] A,
          output divBy3,
          output divBy5
          );
 // call the fiiles for divBy3 and divBy5 name it UDM user define module tie them inputs to the current desing
divBy3 UDM1 (
             .A(A),
             .divBy3(divBy3)
             );
divBy5 UDM2 (
             .A(A),
             .divBy5(divBy5)
             );
 endmodule
```

Test-Bench for combine divider

```
module combine_divBy3and5_tb ();
    reg [3:0] A;
    wire divBy3, divBy5;

    combine_divBy3and5 UUT (
        .A(A),
        .divBy3(divBy3),
        .divBy5(divBy5)
    );

    initial
    begin
        A = 4'b0000; #10;
        A = 4'b0011; #10;
        A = 4'b0101; #10;
        A = 4'b0110; #10;
        A = 4'b1001; #10;
        A = 4'b1010; #10;
        A = 4'b1100; #10;
        A = 4'b1111; #10;
        A = 4'b0001; #10;
        A = 4'b0010; #10;
        A = 4'b0100; #10;
        A = 4'b0111; #10;
        A = 4'b1000; #10;
        A = 4'b1011; #10;
        A = 4'b1101; #10;
        A = 4'b1110; #10;
        end
endmodule
```

lamin Y. Jammeh
EE417 Summer 2024
HW02

Netlist divBy5



Wave Form divBy5