Digital Design Lab 4 Report

Palaskar Adarsh Mahesh
B20EE087

February 5, 2022

1 BCD Adder:

A BCD adder unit takes two binary numbers and returns their output in a format that can be represented on a 7 segment display.

```
timescale 1ns / 1ps

module bcd_adder(a,b,carry_in,sum,carry);


    input [3:0] a,b;
    input carry_in;

    output [3:0] sum;
    output carry;

    reg [4:0] sum_temp;
    reg [3:0] sum;
    reg carry;


    always @(a,b,carry_in)
    begin
        sum_temp = a+b+carry_in;
        if(sum_temp > 9)
            begin
            sum_temp = sum_temp+6;
            carry = 1;
            sum = sum_temp[3:0];
            end
        else
            begin
            carry = 0;
            sum = sum_temp[3:0];
            end
    end

endmodule
```

Figure 1: Verilog code for BCD Adder unit

```
`timescale 1ns / 1ps

module test_bcd_adder;


    reg [3:0] a,b;
    reg carry_in;

    wire [3:0] sum;
    wire carry;

    bcd_adder bcda(a,b,carry_in,sum,carry);
    initial
        begin

        a = 4'b0000;  b = 4'b0000;  carry_in = 1'b0;

        #10 a = 4'b0001;  b = 4'b0010;
        #10 a = 4'b0100;  b = 4'b0011;
        #10 a = 4'b0110;  b = 4'b1000;
        #10 a = 4'b0010;  b = 4'b0010;
        #10 a = 4'b1000;  b = 4'b0100;
        end

    initial #100 $finish;

endmodule
```

Figure 2: Test bench for BCD Adder unit

Figure 3: Simulation wave forms for BCD Adder unit

We can observe that the carry 1 is generated when sum is greater than 9 and six is added so that we observe the correct digits on the display after the addition.

2 BCD Subtractor:

A BCD subtractor unit takes two binary numbers and returns their difference in a format that can be represented on a 7 segment display. We take the 10's compliment of the number to be subtracted and then feed it the BCD Adder and may or may not use another 10's compliment depending on the sign of the output obtained

```
`timescale 1ns / 1ps


module tens_compliment(b,c);

    input [3:0] b;
    output [3:0] c;

    wire [3:0] y,temp;
    wire [4:0] temp2;

    assign y[0] = ~b[0];
    assign y[1] = b[1];
    assign y[2] = (b[2] & (~b[1]))|(b[1]&(~b[2]));
    assign y[3] = (~b[1])&(~b[2])& (~b[3]);

    assign temp = 4'b0001;

    four_bit_adder fba(y,temp,temp2);

    assign c = temp2 [3:0];

endmodule
```

Figure 4: Verilog code for 10's compliment of a number

```
`timescale 1ns / 1ps

module bcd_subtractor(a,b,sign,magnitude);

    input [3:0] a,b;
    output sign;
    output [3:0] magnitude;

    reg [3:0] f_ans;
    wire carry;
    wire [3:0] b1,ans1,ans2;

    tens_compliment tc1(b,b1);
    bcd_adder bcda(a,b1,0,ans1,carry);

    tens_compliment tc2(ans1,ans2);

    always @*
    if (carry == 1)
    begin
    f_ans = ans1;
    end

    else
    f_ans = ans2;

    assign magnitude = f_ans;
    assign sign = ~carry;
endmodule
```

Figure 5: Verilog code for a BCD subtractor

```
`timescale 1ns / 1ps

module test_bcd_subtractor;

    reg [3:0] a,b;

    wire sign;
    wire [3:0] magnitude;

    bcd_subtractor sub1(a,b,sign,magnitude);

    initial
        begin
        a = 4'b0000; b = 4'b0000; // b is the number to be subtracted

        #10 a = 4'b0010; b = 4'b0001;
        #10 a = 4'b0100; b = 4'b0011;
        #10 a = 4'b1001; b = 4'b0001;
        #10 a = 4'b1110; b = 4'b0010;
        #10 a = 4'b0010; b = 4'b0100;
        #10 a = 4'b1000; b = 4'b1000;
        end

    initial #80 $finish;
endmodule
```

Figure 6: Test bench for BCD Subtractor unit

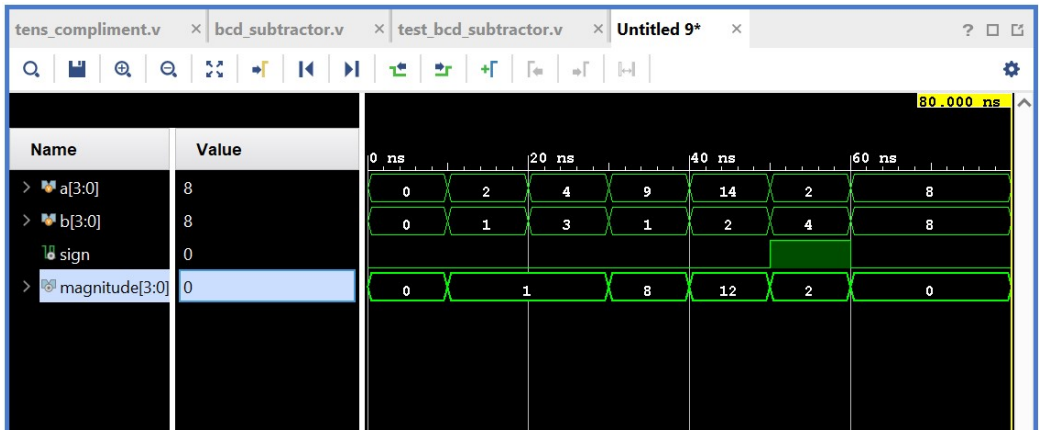

Figure 7: Simulation wave forms for BCD Subtractor unit

When sign variable becomes high, it indicates that the answer is a negative number. We observe that correct answer is obtained for all the cases in the test bench.

3 3 bit by 3 bit multiplier:

A multiplier unit takes two binary numbers and returns their multiplication as output.

The verilog code is as follows:

```
`timescale 1ns / 1ps

module multiplier(a, b, ans,d1,d2);

input   [2:0] a, b;
output [5:0] ans;
output [6:0] d1,d2;

wire [7:0] c,c1;
wire [3:0] temp;
assign c[0] = 1'b0;

assign ans[0] = a[0]&b[0];

full_adder f1(a[1]&b[0],a[0]&b[1],c[0],ans[1],c[1]);

full_adder f2(a[2]&b[0],a[1]&b[1],c[1],c1[0],c1[1]);
full_adder f3(c1[0],a[0]&b[2],c[0],ans[2],temp[1]);

half_adder h1(c1[1],temp[1],c[3],c1[2]);
full_adder f4(a[1]&b[2],a[2]&b[1],c[3],ans[3],temp[2]);

half_adder h2(c1[2],temp[2],c[4],c1[3]);
full_adder f5(a[2]&b[2],c[0],c[4],ans[4],temp[3]);

assign ans[5] = temp[3]|c1[3];

bcd_to_sevenseg bcd71(ans[5:3],d1);
bcd_to_sevenseg bcd72(ans[2:0],d2);

endmodule
```

Figure 8: Verilog code for Multiplier unit

```
`timescale 1ns / 1ps


module test_multiplier();

    wire [6:0] d1,d2;
    wire [5:0] out;

    reg [2:0] x,y;

    multiplier m1(x,y,out,d1,d2);

    initial
        begin

        x = 3'b000; y = 3'b000;

        #10 x = 3'b001; y = 3'b001;
        #10 x = 3'b000; y = 3'b001;
        #10 x = 3'b001; y = 3'b111;
        #10 x = 3'b010; y = 3'b100;
        #10 x = 3'b100; y = 3'b100;
        #10 x = 3'b100; y = 3'b001;
        #10 x = 3'b111; y = 3'b111;

        end
    initial #100 $finish;

endmodule
```

Figure 9: Test bench for Multiplier unit

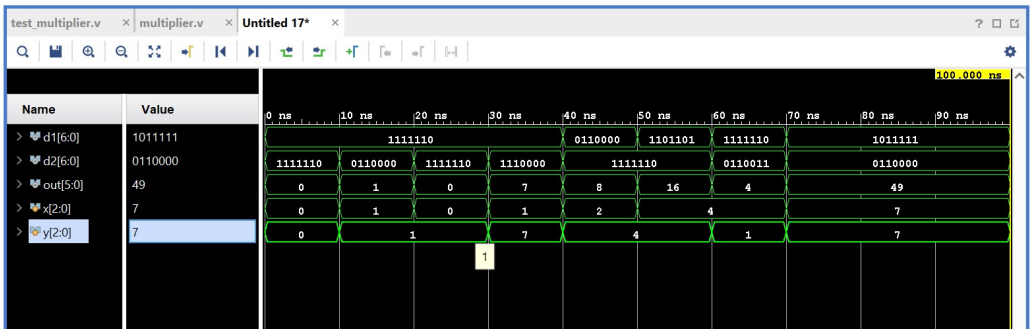

Figure 10: Simulation wave forms for Multiplier unit

We can observe that correct multiplication(converted to unsigned decimal in the wave forms) and corresponding output relation for the seven segment display is obtained.