ELC 2137 Lab ##: Lab Title

Put your name(s) here

October 15, 2020

Summary

In this lab, we built a conditional adder, 6, and 11-bit binary to BCD converter using the dibble dabble algorithm.

Code

Listing 1: Add3

```
module add3(
    input [3:0] num,
    output [3:0] modnum
    );

    reg [3:0] num, modnum;

    always @*
        begin
            if (num > 4'b0100)
                modnum = num + 4'b0011;
            else
                modnum = num;
        end


endmodule
```

Listing 2: Add3 Test

```
module add3_test();

reg [3:0] num_t;
reg [3:0] modnum_t;


add3 dut(
    .num(num_t),
    .modnum(modnum_t)

   );

 integer i;
```

```
initial begin
        for (i=0; i<=4'b1111; i=i+1) begin
        num_t = i;
        #10;
        end
        $finish;
end

endmodule
```

Listing 3: 6 bit BCD

```
module BCD6b(
    input [5:0] B,
    output [3:0] tens,
    output [3:0] ones
    );

    wire [2:0] C1_out, C2_out;




add3 C1(
    .num({1'b0,B[5:3]}),
    .modnum({tens[2],C1_out})
    );


add3 C2(
    .num({C1_out,B[2]}),
    .modnum({tens[1],C2_out})
    );

add3 C3(
    .num({C2_out,B[1]}),
    .modnum({tens[0],ones[3:1]})
    );

assign tens[3] = 1'b0;
assign ones[0] = B[0];




endmodule
```

Listing 4: Sseg 6 bit BCD Test

```
module BCD6b_test();

reg [5:0] B_t;
reg [3:0] tens_t, ones_t;
```

```
BCD6b dut(

    .B(B_t),
    .tens(tens_t),
    .ones(ones_t)
    );

integer i;

initial begin
    for (i=0; i<=6'b111111; i=i+1) begin
    B_t=i;
    #10;
    end
    $finish;
end




endmodule
```

Listing 5: 11 bit BCD

```
module BCD11b(
    input [10:0] B,
    output [3:0] thousands,
    output [3:0] hundreds,
    output [3:0] tens,
    output [3:0] ones
    );

    wire [3:0] C1_out, C2_out, C3_out, C4_out, C5_out, C6_out, C7_out,
       C9_out, C10_out, C11_out, C12_out, C14_out;



add3 C1(
    .num({1'b0,B[10:8]}),
    .modnum(C1_out)
    );

add3 C2(
    .num({C1_out[2:0],B[7]}),
    .modnum(C2_out)
    );

add3 C3(
    .num({C2_out[2:0],B[6]}),
    .modnum(C3_out)
    );
```

```
add3 C4(
    .num({C3_out[2:0],B[5]}),
    .modnum(C4_out)
    );

add3 C5(
    .num({C4_out[2:0],B[4]}),
    .modnum(C5_out)
    );

add3 C6(
    .num({C5_out[2:0],B[3]}),
    .modnum(C6_out)
    );

add3 C7(
    .num({C6_out[2:0],B[2]}),
    .modnum(C7_out)
    );

add3 C8(
    .num({C7_out[2:0],B[1]}),
    .modnum({tens[0],ones[3:1]})
    );


add3 C9(
    .num({1'b0,C1_out[3],C2_out[3],C3_out[3]}),
    .modnum(C9_out)
    );

add3 C10(
    .num({C9_out[2:0],C4_out[3]}),
    .modnum(C10_out)
    );

add3 C11(
    .num({C10_out[2:0],C5_out[3]}),
    .modnum(C11_out)
    );

add3 C12(
    .num({C11_out[2:0],C6_out[3]}),
    .modnum(C12_out)
    );

add3 C13(
    .num({C12_out[2:0],C7_out[3]}),
    .modnum({hundreds[0],tens[3:1]})
    );


add3 C14(
    .num({1'b0,C9_out[3],C10_out[3],C11_out[3]}),
```

```
    .modnum(C14_out)
    );

add3 C15(
    .num({C14_out[2:0],C12_out[3]}),
    .modnum({thousands[0],hundreds[3:1]})
    );


assign thousands[1] = C14_out[3];
assign thousands[2] = 1'b0;
assign thousands[3] = 1'b0;
assign ones[0] = B[0];

endmodule
```

Listing 6: 11 bit BCD Test

```
module BCD11b_test();

reg [10:0] B_t;
reg [3:0] thousands_t, hundreds_t, tens_t, ones_t;


BCD11b dut(
    .B(B_t),
    .thousands(thousands_t),
    .hundreds(hundreds_t),
    .tens(tens_t),
    .ones(ones_t)
    );

 initial begin
    B_t=11'b00000000000; #10
    B_t=11'b00000000001; #10
    B_t=11'b00001010101; #10
    B_t=11'b01001010101; #10
    B_t=11'b11111010101; #10
    $finish;
    end

endmodule
```

Listing 7: Sseg1 for BCD

```
module sseg1_BCD(

  input [15:0] sw,
  input clk,
  output [3:0] an,
  output dp,
  output [6:0] seg
    );
```

```
wire  [3:0] thousands, hundreds, tens_w, ones_w;

BCD11b my_BCD11b(
    .B(sw[10:0]),
    .thousands(thousands),
    .hundreds(hundreds),
    .tens(tens_w),
    .ones(ones_w)
    );

sseg1 my_sseg1(
    .A(tens_w),
    .B(ones_w),
    .sel(sw[15]),
    .seg_un(an[3:2]),
    .dp(dp),
    .seg(seg),
    .seg_L(an[1]),
    .seg_R(an[0])
    );



endmodule
```

Results

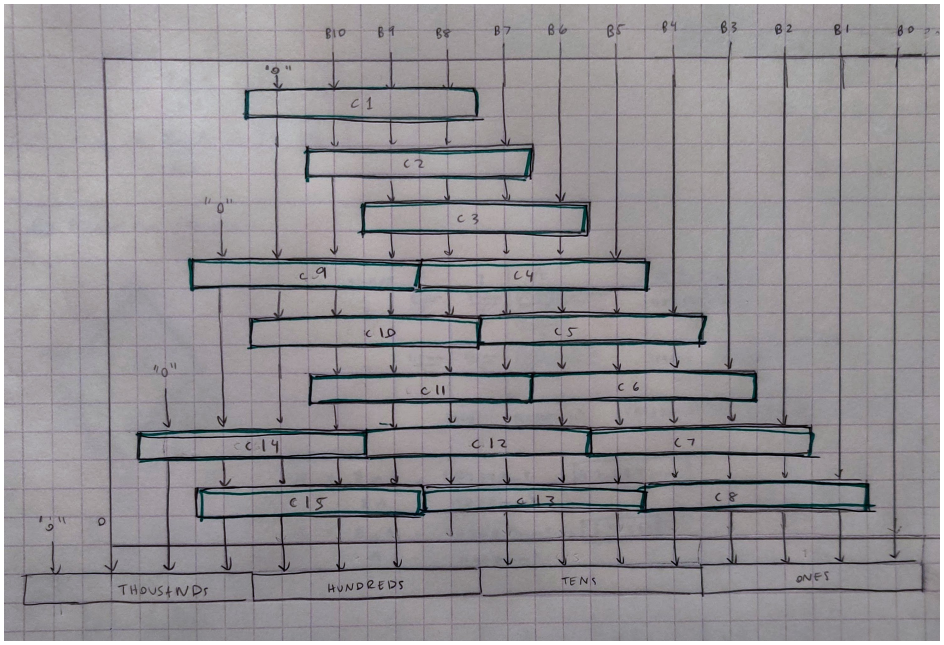

Figure 1: Circuit Diagram for 11 bit converter

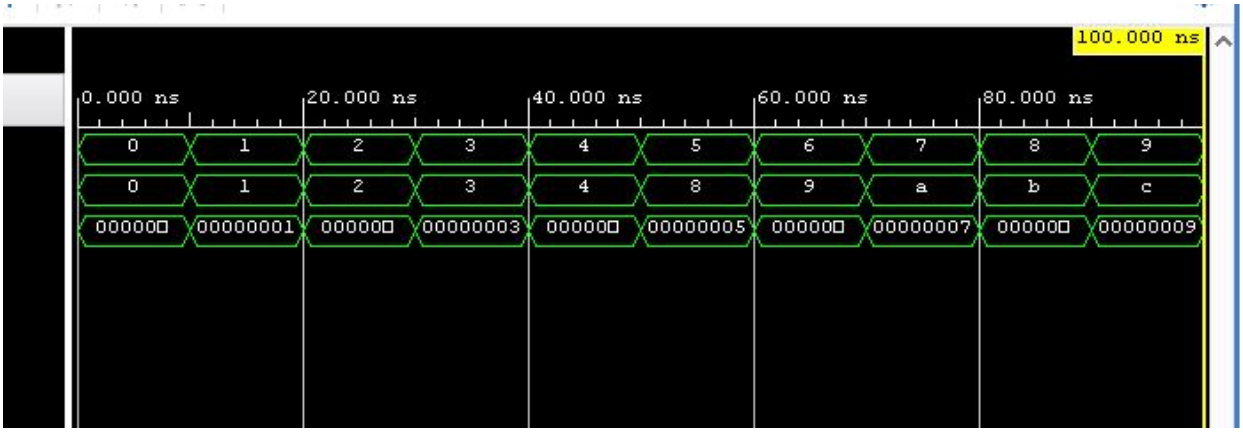

Figure 2: Add 3

B	tens	ones
000000	0000	0000
000001	0000	0001
000010	0000	0010
000011	0000	0011
000100	0000	0100

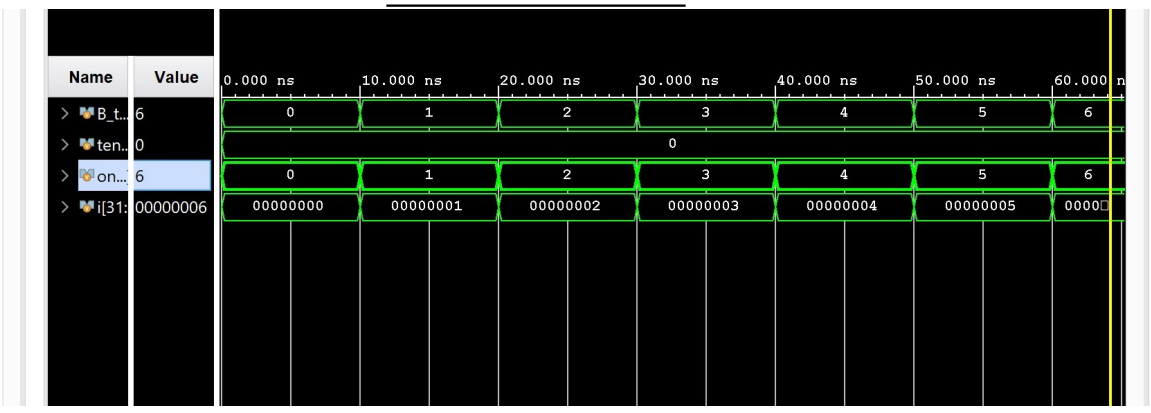

Figure 3: 6 bit BCD beginning

B	tens	ones
111011	0101	1001
111100	0110	0000
111101	0110	0001
111110	0110	0010
111111	0110	0011

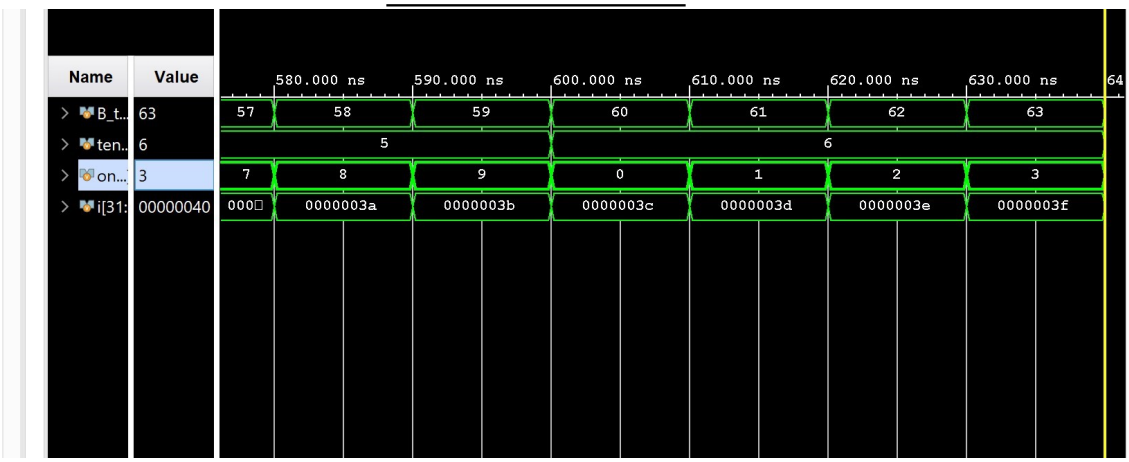

Figure 4: 6 bit BCD end

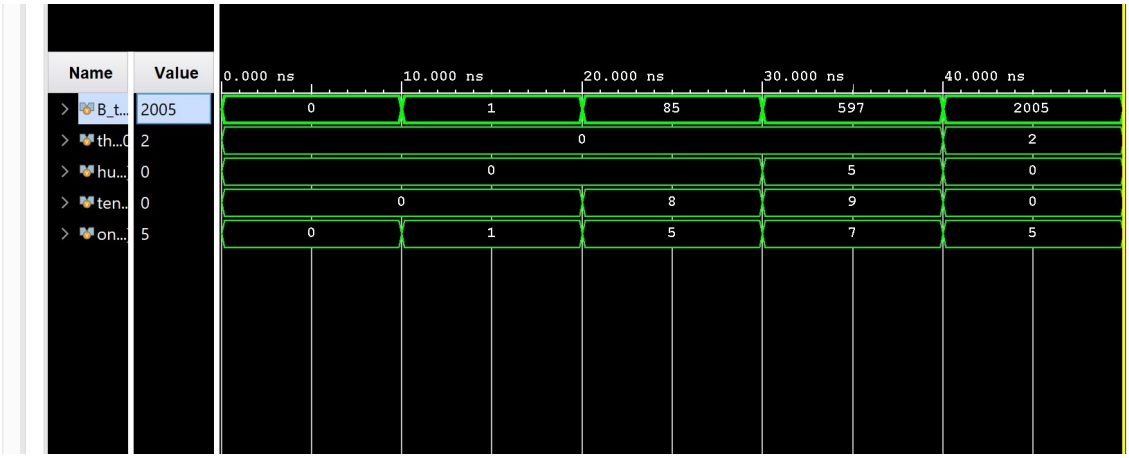

Figure 5: 11 bit BCD test cases