DIGITAL DESIGN

LAB7 COMBINATORIAL CIRCUIT

2021 FALL TERM @ CSE . SUSTECH

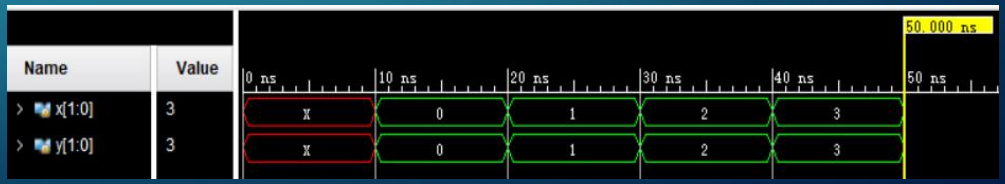
LAB7

- Verilog
  - Sequential block vs Parallel block
- Combinatorial circuit
  - 1bit full adder, 2bits full adder
  - Lighting 7 segment digital tube
- Practice

SEQUENTIAL BLOCK VS PARALLEL BLOCK

```
module block2();
    reg [1:0]x,y;
    initial
    begin
        #10 x=2'd0;
        #10 x=2'd1;
        #10 x=2'd2;
        #10 x=2'd3;
    end

    initial
    fork
        #10 y=2'd0;
        #20 y=2'd1;
        #30 y=2'd2;
        #40 y=2'd3;
    join

    initial
        #50 $finish(1);
endmodule
```

- In one module **all the block executes at the same time(time 0)**
- **Sequential block(begin … end):**
  - **synthesizable(could be used in the circuit design)**
  - all the statements in one sequential block executes with the order of writing.
- **Parallel block(fork … join):**
  - **Not synthesizable(CAN NOT be used in the circuit design)**
  - all the statements in one parallel block executes at same time

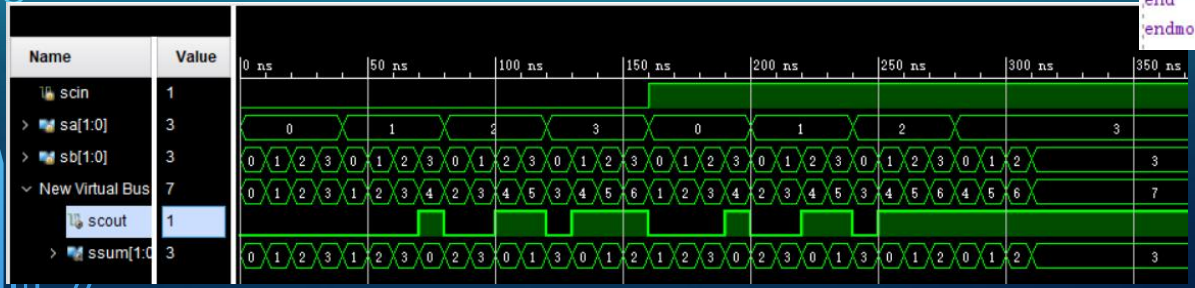


1BIT FULL ADDER VS MULTI-BITS FULL ADDER

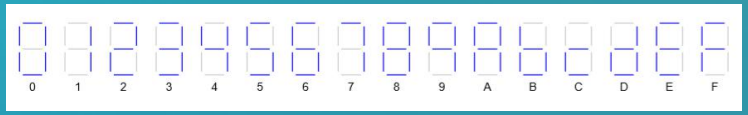
```
module full_add_1b(a, b, cin, sum, cout);
input a, b, cin;
output sum, cout;
assign {cout, sum}=a+b+cin;
endmodule
```

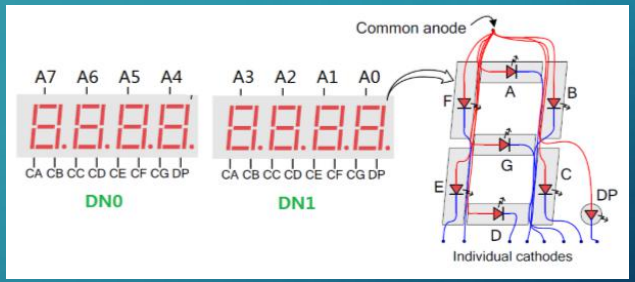
```
module full_add_2b(a, b, cin, sum, cout);
input [1:0]a, b;
input cin;
output[1:0] sum;
output cout;

wire cout1;
full_add_1b u0(a[0], b[0], cin, sum[0], cout1);
full_add_1b u1(a[1], b[1], cout1, sum[1], cout);
endmodule
```

```
module full_add_sm(    );
reg scin;
reg [1:0]sa, sb;
wire [1:0]ssum;
wire scout;
full_add_2b u2(sa, sb, scin, ssum, scout);
initial
{scin, sa, sb} = 5'd0;

initial
begin
    repeat(31)
    #10 {scin, sa, sb} ={scin, sa, sb} +1;
end
endmodule
```



LIGHTING A 7-SEG-TUBE

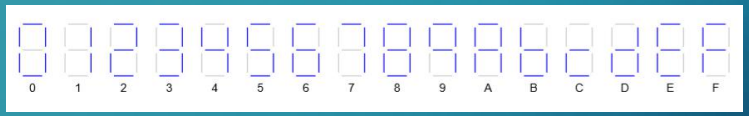




- There is an 4-bits width binary number , show its hexadecimal number.
- Using minisystem develop board
- 4*dial switch are the inputs while 8*7seg-tubes are the outputs

LIGHTING A 7-SEG-TUBE(DESIGN)

```
module light_7seg(input [3:0] sw,output reg [7:0] seg_out,output [7:0] seg_en );
    assign seg_en = ~8'hff;
    always @ *   begin
    case(sw)
            4'h0: seg_out = 8'b01000000; // 0
            4'h1: seg_out = 8'b01111001; // 1
            4'h2: seg_out = 8'b00100100; // 2
            4'h3: seg_out = 8'b00110000; // 3
            4'h4: seg_out = 8'b00011001; // 4
            4'h5: seg_out = 8'b00010010; // 5
            4'h6: seg_out = 8'b00000010; // 6
            4'h7: seg_out = 8'b01111000; // 7
            4'h8: seg_out = 8'b00000000; // 8
            4'h9: seg_out = 8'b00010000; // 9
            4'ha: seg_out = 8'b00001000; // A
            4'hb: seg_out = 8'b00000011; // b
            4'hc: seg_out = 8'b01000110; // c
            4'hd: seg_out = 8'b00100001; // d
            4'he: seg_out = 8'b00000110; // E
            4'hf: seg_out = 8'b00001110; // F
            default: seg_out = 8'b0000000;
        endcase
    end
```



Eight 7-seg-tubes, each one has a enable pin, while it's 0 it enable the tube while it's 1 it disable the tube

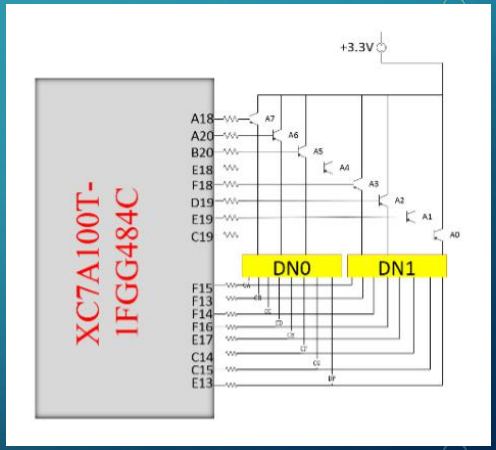
While 'seg_en' is 8'h00, it means enable all the tubes

From DP to CA ~CG, all are negative enable

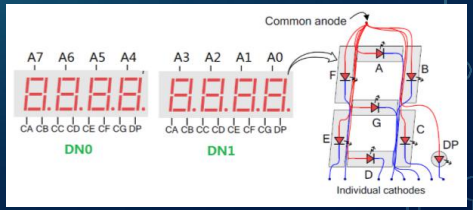
LIGHTING A 7-SEG-TUBE(CONSTRAINTS FILE)

```
set_property PACKAGE_PIN C19 [get_ports {seg_en[0]}]
set_property PACKAGE_PIN E19 [get_ports {seg_en[1]}]
set_property PACKAGE_PIN D19 [get_ports {seg_en[2]}]
set_property PACKAGE_PIN F18 [get_ports {seg_en[3]}]
set_property PACKAGE_PIN E18 [get_ports {seg_en[4]}]
set_property PACKAGE_PIN B20 [get_ports {seg_en[5]}]
set_property PACKAGE_PIN A20 [get_ports {seg_en[6]}]
set_property PACKAGE_PIN A18 [get_ports {seg_en[7]}]
```

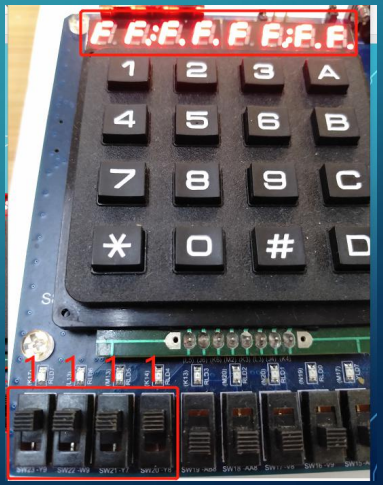
```
set_property PACKAGE_PIN F15 [get_ports {seg_out[0]}]
set_property PACKAGE_PIN F13 [get_ports {seg_out[1]}]
set_property PACKAGE_PIN F14 [get_ports {seg_out[2]}]
set_property PACKAGE_PIN F16 [get_ports {seg_out[3]}]
set_property PACKAGE_PIN E17 [get_ports {seg_out[4]}]
set_property PACKAGE_PIN C14 [get_ports {seg_out[5]}]
set_property PACKAGE_PIN C15 [get_ports {seg_out[6]}]
set_property PACKAGE_PIN E13 [get_ports {seg_out[7]}]
```



C19,E19~A18 are select pins on eight 7-seg-tubes
F15,F13~C15 are pins of CA~CG on 7-seg-tubes while E13 is the pin of DP on 7-seg-tube
Constrains on other ports can be set by user



TESTING RESULT



PRACTICES(1)

- There are sixteen wards, which are numbered from 0 to F respectively, among which #0 ward has the highest priority, #F has the lowest priority(Priority decreases as the number increases).
- Each room has an call bell, it could be turn on or turn off. In the main control room there is a display screen which shows the ID of the room whose bell is on with highest priority of all the bell on room.
- Please write a circuit to implement this function and test.
  - Do the design and verify the function of your design.
  - Create the constrain file, do the synthetic and implementation, generate the bitstream file and program the device, then test on the minisys develop board.

PRACTICES(2)

Design a 3bit width full adder with 3 inputs (a, b, cin) and 2 outputs(sum, cout), the width of a,b and sum are both 3bit width while cin and cout are both 1bit width

1. Do the design by using structure design.
2. Make a testbench to verify its function. Try to build the testbench by using Parallel block in verilog
3. Create the constrain file, do the synthetic and implementation, generate the bitstream file and program the device, then test on the minisys develop board.
    1. Switches are suggested to be used as input device
    2. Select one of the following three options as the output device:
        1. one led as the 'cout' and 7-segment tubes as the 'sum'
        2. one led as the 'cout' and other leds as the 'sum'
        3. 7-segment tubes as the 'cout' and 'sum'