Lab 6

3300L.E01

Dual BCD Up/Down Counters, ALU, and Control Display on 7-Segment

Group U

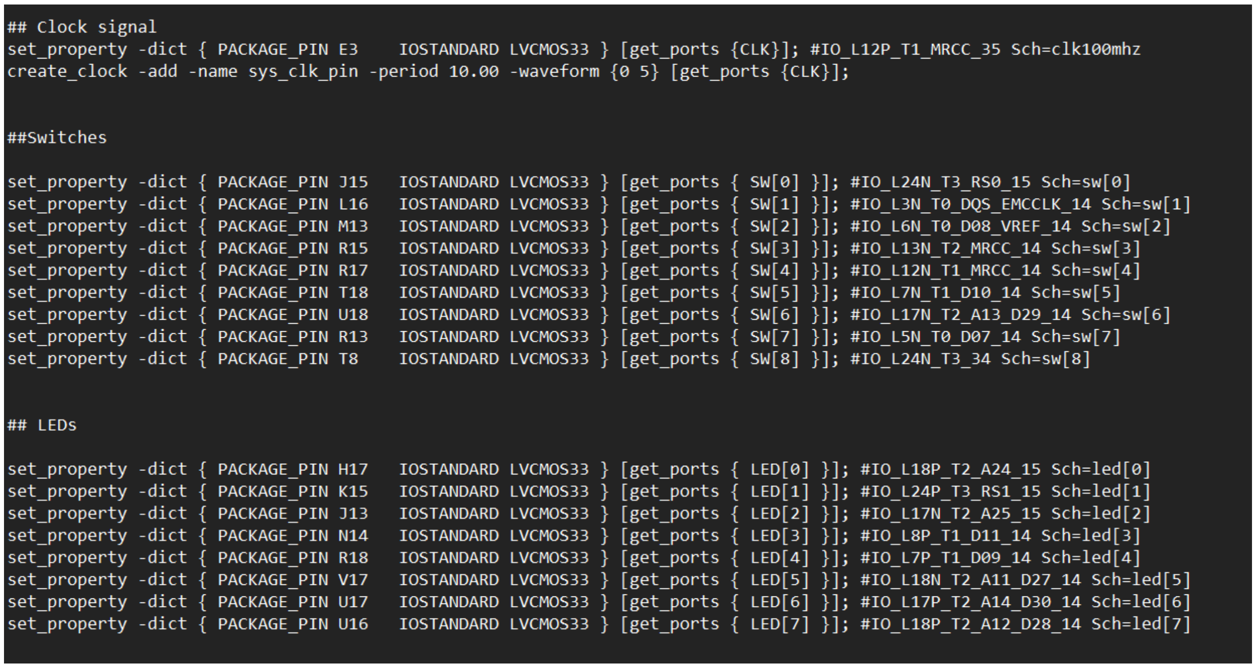
Justin Herrera - Pandarb004

Nathan Rahbar - 9athan

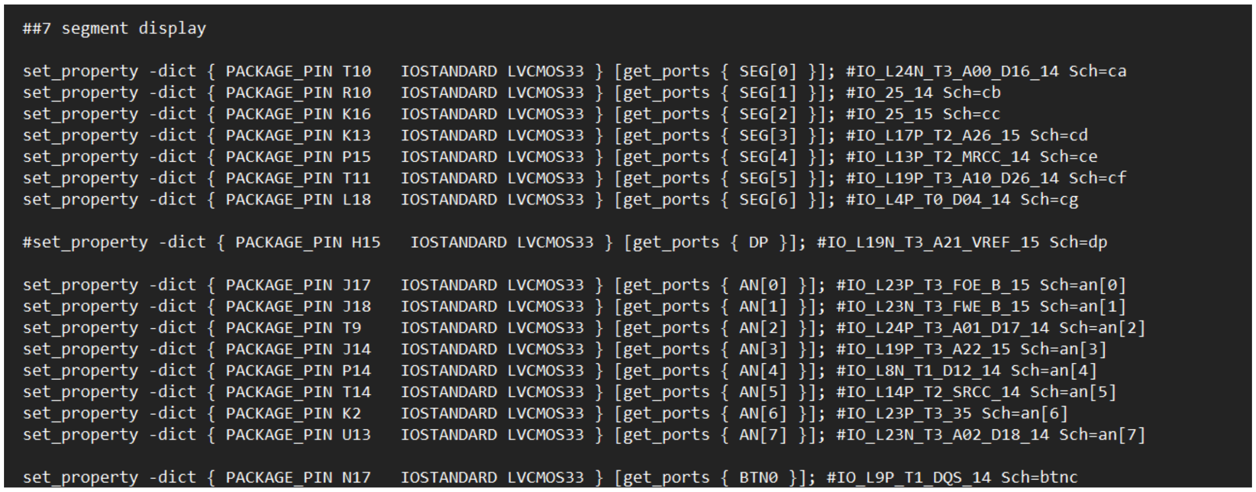
Design

Constraints

For Lab 6, our team implemented the 7 segment displays on the Nexys A7. We had a constraints file in which we initialized all of our needed inputs and outputs.

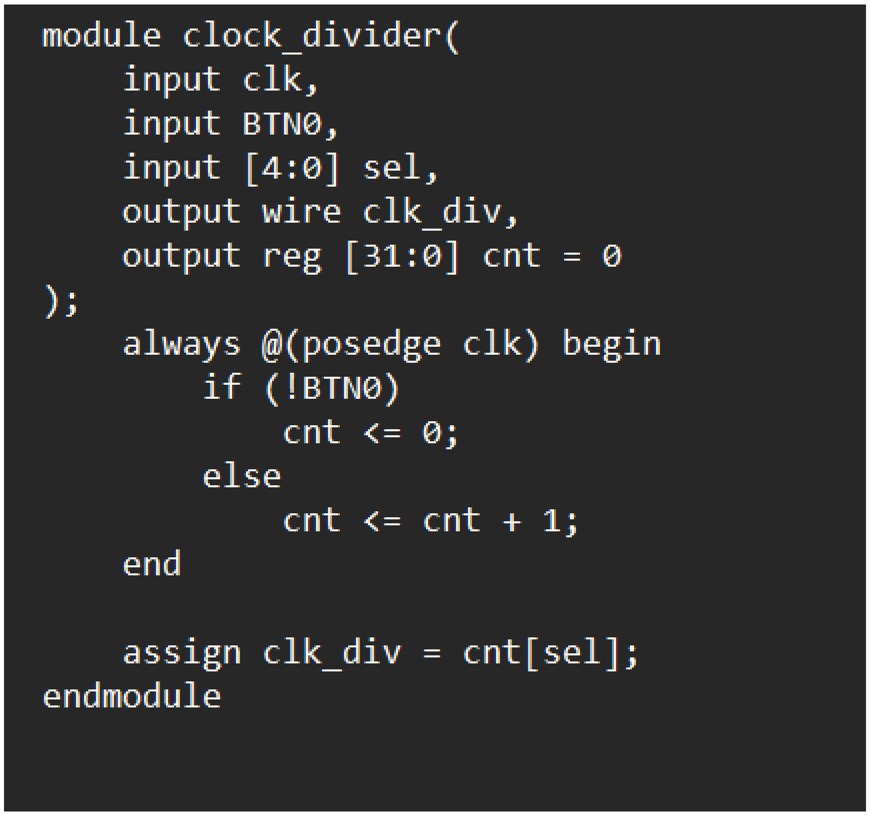


```
## Clock signal
set_property -dict { PACKAGE_PIN E3    IOSTANDARD LVCMOS33 } [get_ports {CLK}]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK}];


##Switches

set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { SW[0] }]; #IO_L24N_T3_RS0_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { SW[1] }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { SW[2] }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
set_property -dict { PACKAGE_PIN R15   IOSTANDARD LVCMOS33 } [get_ports { SW[3] }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]
set_property -dict { PACKAGE_PIN R17   IOSTANDARD LVCMOS33 } [get_ports { SW[4] }]; #IO_L12N_T1_MRCC_14 Sch=sw[4]
set_property -dict { PACKAGE_PIN T18   IOSTANDARD LVCMOS33 } [get_ports { SW[5] }]; #IO_L7N_T1_D10_14 Sch=sw[5]
set_property -dict { PACKAGE_PIN U18   IOSTANDARD LVCMOS33 } [get_ports { SW[6] }]; #IO_L17N_T2_A13_D29_14 Sch=sw[6]
set_property -dict { PACKAGE_PIN R13   IOSTANDARD LVCMOS33 } [get_ports { SW[7] }]; #IO_L5N_T0_D07_14 Sch=sw[7]
set_property -dict { PACKAGE_PIN T8    IOSTANDARD LVCMOS33 } [get_ports { SW[8] }]; #IO_L24N_T3_34 Sch=sw[8]


## LEDs

set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { LED[0] }]; #IO_L18P_T2_A24_15 Sch=led[0]
set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMOS33 } [get_ports { LED[1] }]; #IO_L24P_T3_RS1_15 Sch=led[1]
set_property -dict { PACKAGE_PIN J13   IOSTANDARD LVCMOS33 } [get_ports { LED[2] }]; #IO_L17N_T2_A25_15 Sch=led[2]
set_property -dict { PACKAGE_PIN N14   IOSTANDARD LVCMOS33 } [get_ports { LED[3] }]; #IO_L8P_T1_D11_14 Sch=led[3]
set_property -dict { PACKAGE_PIN R18   IOSTANDARD LVCMOS33 } [get_ports { LED[4] }]; #IO_L7P_T1_D09_14 Sch=led[4]
set_property -dict { PACKAGE_PIN V17   IOSTANDARD LVCMOS33 } [get_ports { LED[5] }]; #IO_L18N_T2_A11_D27_14 Sch=led[5]
set_property -dict { PACKAGE_PIN U17   IOSTANDARD LVCMOS33 } [get_ports { LED[6] }]; #IO_L17P_T2_A14_D30_14 Sch=led[6]
set_property -dict { PACKAGE_PIN U16   IOSTANDARD LVCMOS33 } [get_ports { LED[7] }]; #IO_L18P_T2_A12_D28_14 Sch=led[7]
```



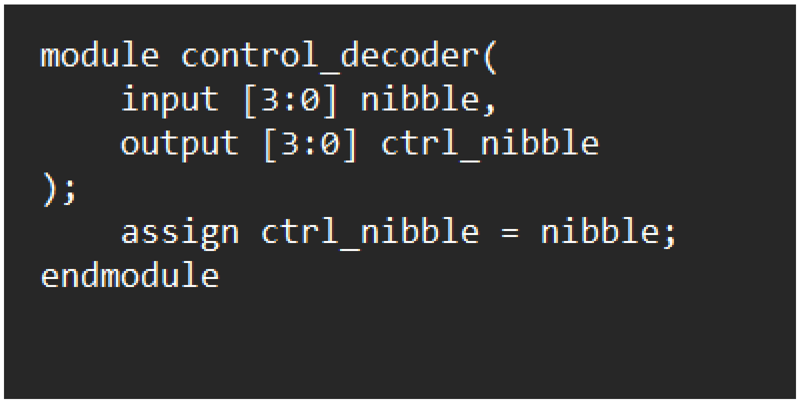
```
##7 segment display

set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { SEG[0] }]; #IO_L24N_T3_A00_D16_14 Sch=ca
set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { SEG[1] }]; #IO_25_14 Sch=cb
set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { SEG[2] }]; #IO_25_15 Sch=cc
set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { SEG[3] }]; #IO_L17P_T2_A26_15 Sch=cd
set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { SEG[4] }]; #IO_L13P_T2_MRCC_14 Sch=ce
set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { SEG[5] }]; #IO_L19P_T3_A10_D26_14 Sch=cf
set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { SEG[6] }]; #IO_L4P_T0_D04_14 Sch=cg

#set_property -dict { PACKAGE_PIN H15   IOSTANDARD LVCMOS33 } [get_ports { DP }]; #IO_L19N_T3_A21_VREF_15 Sch=dp

set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { AN[0] }]; #IO_L23P_T3_FOE_B_15 Sch=an[0]
set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { AN[1] }]; #IO_L23N_T3_FWE_B_15 Sch=an[1]
set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { AN[2] }]; #IO_L24P_T3_A01_D17_14 Sch=an[2]
set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { AN[3] }]; #IO_L19P_T3_A22_15 Sch=an[3]
set_property -dict { PACKAGE_PIN P14   IOSTANDARD LVCMOS33 } [get_ports { AN[4] }]; #IO_L8N_T1_D12_14 Sch=an[4]
set_property -dict { PACKAGE_PIN T14   IOSTANDARD LVCMOS33 } [get_ports { AN[5] }]; #IO_L14P_T2_SRCC_14 Sch=an[5]
set_property -dict { PACKAGE_PIN K2    IOSTANDARD LVCMOS33 } [get_ports { AN[6] }]; #IO_L23P_T3_35 Sch=an[6]
set_property -dict { PACKAGE_PIN U13   IOSTANDARD LVCMOS33 } [get_ports { AN[7] }]; #IO_L23N_T3_A02_D18_14 Sch=an[7]

set_property -dict { PACKAGE_PIN N17   IOSTANDARD LVCMOS33 } [get_ports { BTN0 }]; #IO_L9P_T1_DQS_14 Sch=btnc
```

Clock Divider



```
module clock_divider(
    input clk,
    input BTN0,
    input [4:0] sel,
    output wire clk_div,
    output reg [31:0] cnt = 0
);
    always @(posedge clk) begin
        if (!BTN0)
            cnt <= 0;
        else
            cnt <= cnt + 1;
    end

    assign clk_div = cnt[sel];
endmodule
```

Control Decoder



```
module control_decoder(
    input [3:0] nibble,
    output [3:0] ctrl_nibble
);
    assign ctrl_nibble = nibble;
endmodule
```

Seven Segment Scanner

```
module seg7_scan(
    input clk,
    input [7:0] result,
    input [3:0] ctrl_nibble,
    output reg [6:0] SEG,
    output reg [7:0] AN
);

    reg [16:0] digit_scan_counter;
    wire [1:0] digit_mux;
    reg [3:0] value;

    always @(posedge clk)
        digit_scan_counter <= digit_scan_counter + 1;

    assign digit_mux = digit_scan_counter[16:15];

    always @(*) begin
        case (digit_mux)
            2'b00: begin
                value = result[3:0];
                AN = 8'b11111110;
            end
            2'b01: begin
                value = result[7:4];
                AN = 8'b11111101;
            end
            2'b10: begin
                value = ctrl_nibble;
                AN = 8'b11111011;
            end
            default: begin
                value = 4'b0000;
                AN = 8'b11111111;
            end
        endcase
    end
```

```verilog
    always @(*) begin
        case (value)
            4'h0: SEG = 7'b1000000;
            4'h1: SEG = 7'b1111001;
            4'h2: SEG = 7'b0100100;
            4'h3: SEG = 7'b0110000;
            4'h4: SEG = 7'b0011001;
            4'h5: SEG = 7'b0010010;
            4'h6: SEG = 7'b0000010;
            4'h7: SEG = 7'b1111000;
            4'h8: SEG = 7'b0000000;
            4'h9: SEG = 7'b0010000;
            4'hA: SEG = 7'b0001000;
            4'hB: SEG = 7'b0000011;
            4'hC: SEG = 7'b1000110;
            4'hD: SEG = 7'b0100001;
            4'hE: SEG = 7'b0000110;
            4'hF: SEG = 7'b0001110;
            default: SEG = 7'b1111111;
        endcase
    end
endmodule
```

Top Module

```
module top_lab6(
    input CLK,
    input BTN0,
    input [8:0] SW,
    output [7:0] LED,
    output [7:0] AN,
    output [6:0] SEG
);
    wire clk_div;
    wire [31:0] cnt;
    wire [3:0] unit_bcd, tens_bcd, ctrl_nibble;
    wire [7:0] logic_result;

    // Instantiate clock divider
    clock_divider u_div(
        .clk(CLK),
        .BTN0(!BTN0),
        .sel(SW[4:0]),
        .cnt(cnt),
        .clk_div(clk_div)
    );

    // BCD counters
    bcd_counter u_unit(
        .clk(clk_div),
        .rst_n(BTN0),
        .dir(SW[7]),
        .bcd(unit_bcd)
    );

    bcd_counter u_tens(
        .clk(clk_div),
        .rst_n(BTN0),
        .dir(SW[8]),
        .bcd(tens_bcd)
    );
    // Control Decoder
    control_decoder u_ctrl(
        .nibble({SW[8], SW[7], SW[6], SW[5]}),
        .ctrl_nibble(ctrl_nibble)
    );

    // 7-segment scanner
    seg7_scan u_scan(
        .clk(CLK),
        .result(logic_result),
        .ctrl_nibble(ctrl_nibble),
        .SEG(SEG),
        .AN(AN)
    );

    assign LED[3:0] = unit_bcd;
    assign LED[7:4] = tens_bcd;

endmodule
```

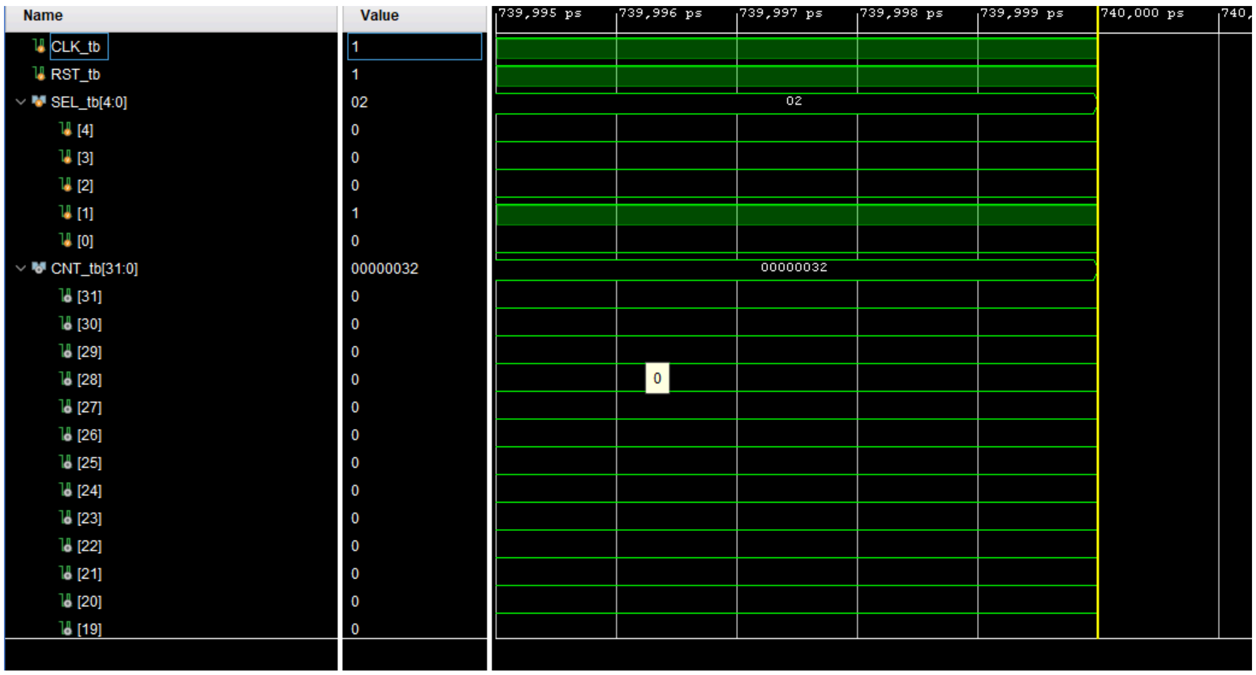
ALU

```
module alu(
    input [3:0] A,
    input [3:0] B,
    input [1:0] ctrl,
    output reg [7:0] result
);
    always @(*) begin
        case (ctrl)
            2'b00: result = A + B;
            2'b01: result = A - B;
            default: result = 8'b00000000;
        endcase
    end
endmodule
```

BCD Counter

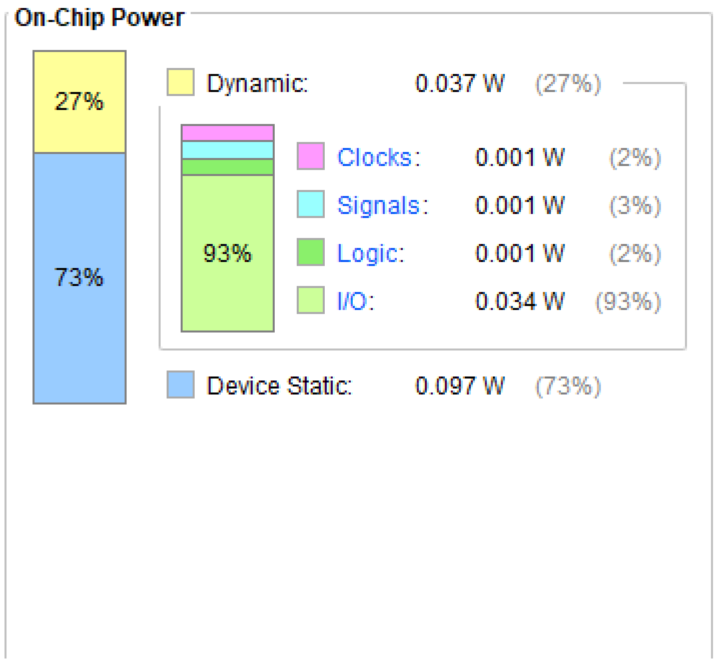
```verilog
module bcd_counter(
    input clk,
    input rst_n,
    input dir,
    output reg [3:0] bcd = 0
);
    always @(posedge clk or negedge rst_n) begin
        if (!rst_n)
            bcd <= 0;
        else if (dir) begin
            if (bcd == 9)
                bcd <= 0;
            else
                bcd <= bcd + 1;
        end else begin
            if (bcd == 0)
                bcd <= 9;
            else
                bcd <= bcd - 1;
        end
    end
endmodule
```

Testbench



Name	Value
CLK_tb	1
RST_tb	1
SEL_tb[4:0]	02
[4]	0
[3]	0
[2]	0
[1]	1
[0]	0
CNT_tb[31:0]	00000032
[31]	0
[30]	0
[29]	0
[28]	0
[27]	0
[26]	0
[25]	0
[24]	0
[23]	0
[22]	0
[21]	0
[20]	0
[19]	0

Implementation



On-Chip Power

🟨	Dynamic:	0.037 W	(27%)
🟪	Clocks:	0.001 W	(2%)
🟦	Signals:	0.001 W	(3%)
🟩	Logic:	0.001 W	(2%)
🟩	I/O:	0.034 W	(93%)
🟦	Device Static:	0.097 W	(73%)

Contributions

Justin - Testbench, Report. - 50%
Nathan - Driver Module, Constraints. - 50%

Demo

https://youtu.be/FXDZODTNaR8