

San Jose State University
Department of Computer Engineering

CMPE 140 Lab Report

Lab 1 Report

Title System-Level Design Review

Semester Fall/18

Date 9/11/18

by

Name Anahit Sarao
(typed)

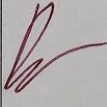

SID 008435583
(typed)

Name Colin Schardt
(typed)

SID 012080185
(typed)

Lab Checkup Record

100%

Week	Performed By (signature)	Checked By (signature)	Tasks Successfully Completed*	Tasks Partially Completed*	Tasks Failed or Not Performed*
1	A.S. CS.		100%		
2	A.S. CS.		100%		

* Detailed descriptions must be given in the report.

Factorial Calculator:

Introduction:

The purpose of this lab was to create a factorial calculator using Xilinx Vivado Design Suite through Verilog programming. This module takes a 4-bit binary value and outputs the factorial on a 7-segment display. The module is only able to process factorials as high as 12!, and will give an error message if any the input is a larger number than 12.

The list of tasks were fully completed in lab:

- Waveform validation of factorial
- Design system datapath using basic building blocks
- Design block diagrams
- Construct output table
- Hardware validation using Nexys4 DDR board
- Stimuli and Hardware Verification of factorial

The list of tasks were not fully completed in lab but were finished afterword:

- Simulation and hardware verification of the error flag

Design Methodology:

The lab assignment called for a hierarchical design consisting of several modules:

`factorial_top`:

This module takes a 4-bit input, a go signal, and a reset signal, and outputs a done signal, a 32-bit product, and an error flag. The input is processed by the datapath while the go signal is processed by the control unit. The product is generated by the datapath while the done and error flags are generated by the control unit.

control_unit:

This module takes in go and reset signals from the top module, as well as a greater than signal, and a greater than 12 signal from the datapath to generate next state logic. The control unit has 5 states. State 0 is an idle state waiting for the command to begin. State 1 prepares the datapath to begin calculating the factorial. State 2 is a wait state for the internal multiplier to finish its calculations. State 3 is the done state where an output enable is sent to the datapath and the done and error flags will be read. State 4 is a transition state where counter and register values of the datapath are updated.

datapath:

This module takes a 4 bit input from the top module, as well as load counter, enable counter, load register, and output enable signals from the control unit, and outputs a 32-bit product to the top module as well as a greater than signal and a greater than 12 signal to the control unit.

Procedure:

1. Create *factorial* project in Vivado and select the proper hardware: xc7a100csg324-1.
2. Create *factorial_top.v*, *control_unit.v*, *factorial_dp.v*, and self-checking testbenches *factorial_top_tb.v*, *control_unit_tb.v*, and *factorial_dp_tb.v*.
3. Run simulations and verify functionality performs as desired.
4. Create and design *factorial_FPGA.v*, *factorial_FPGA.xdc*, and all modules necessary for board functionality.
5. Run Synthesis, Implementation, and Bitstream Generation, and program the Nexys 4 FPGA board.
6. Verify board Functionality is performing as desired.

Simulation Results:

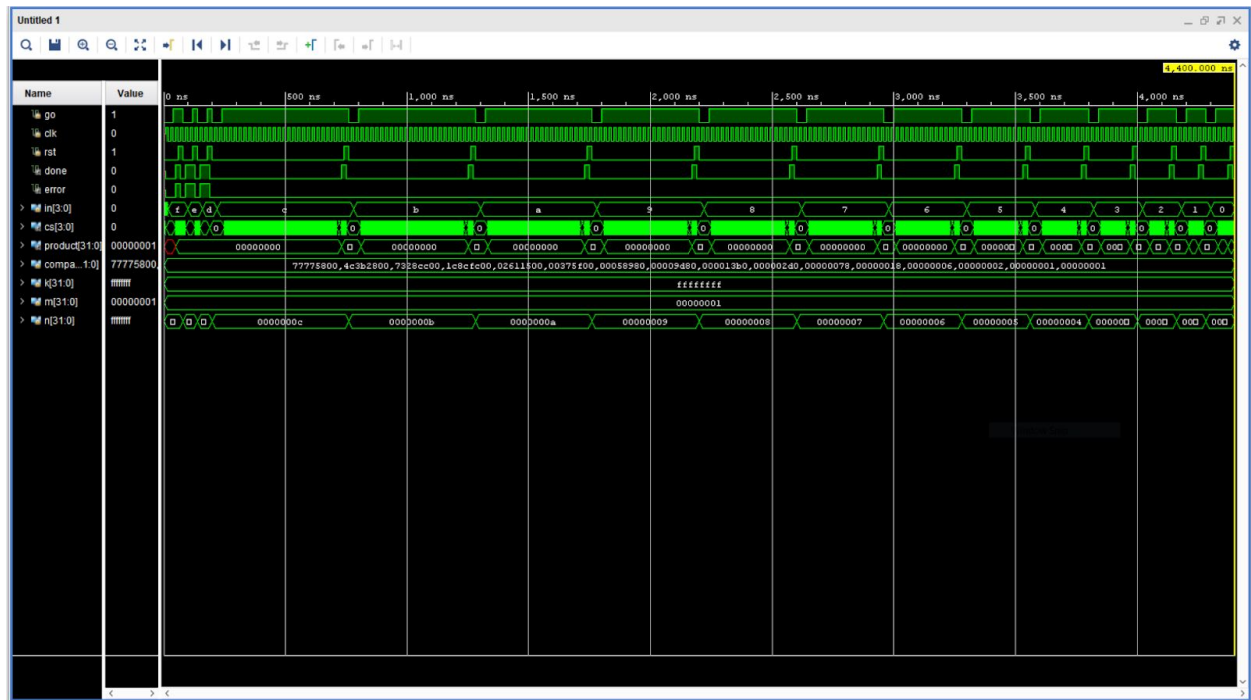


Figure 1a: factorial_top_tb.v simulation result.

```

xsim: Time (s): cpu = 00:00:10 ; elapsed = 00:00:08 . Memory (MB): peak = 761.133 ; gain = 5.023
INFO: [USF-XSim-96] XSim completed. Design snapshot 'factorial_top_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:21 ; elapsed = 00:00:45 . Memory (MB): peak = 761.133 ; gain = 11.879
restart
INFO: [Simtcl 6-17] Simulation restarted
run all
ERROR: input value greater than 12

ERROR: input value greater than 12

ERROR: input value greater than 12

SUCCESS:      12! =  479001600

SUCCESS:      11! =  39916800

SUCCESS:      10! =  3628800

SUCCESS:       9! =   362880

SUCCESS:       8! =   40320

SUCCESS:       7! =    5040

SUCCESS:       6! =     720

SUCCESS:       5! =    120

SUCCESS:       4! =     24

SUCCESS:       3! =      6

SUCCESS:       2! =      2

SUCCESS:       1! =      1

SUCCESS:       0! =      1

```

Figure 1b: factorial_top_tb.v Tcl console output.

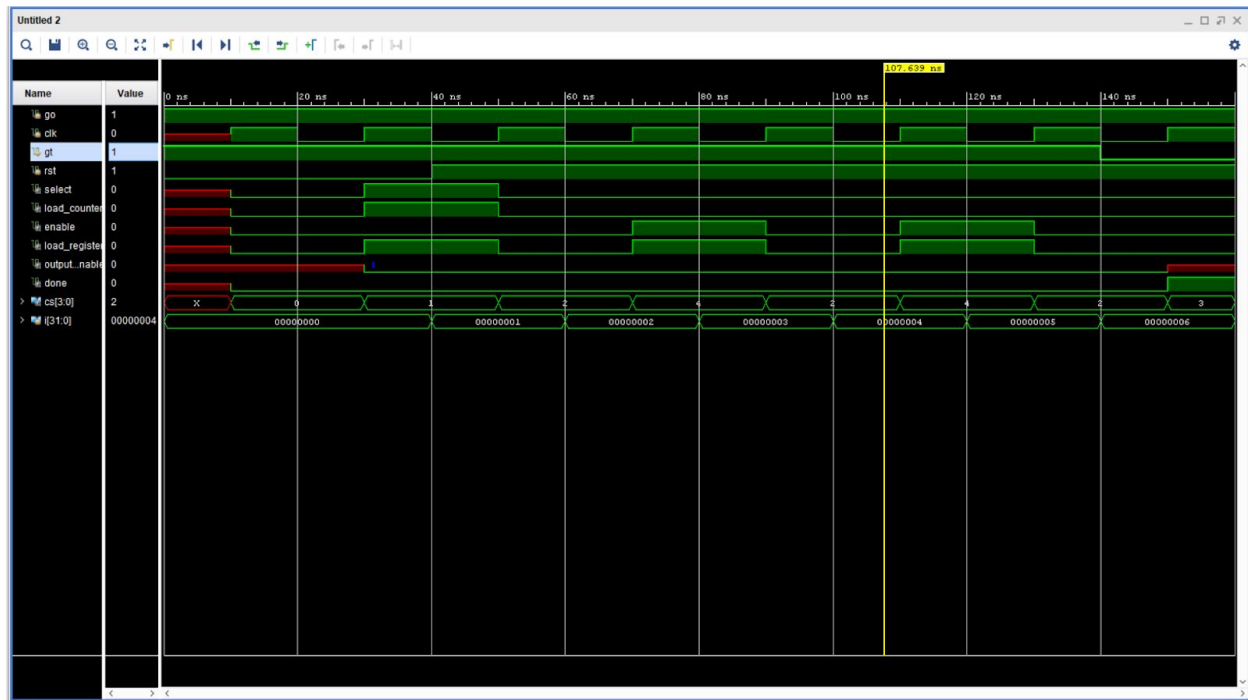


Figure 2a: control_unit_tb.v simulation results.

```

$finish called at time : 160 ns : File "C:/Users/Colin/140/factorial_calculator/factorial_calculator.srcs/sim_1/new/control_unit_tb.v" Line 80
xsim: Time (s): cpu = 00:00:07 ; elapsed = 00:00:05 . Memory (MB): peak = 788.480 ; gain = 0.000
INFO: [USF-XSim-96] XSim completed. Design snapshot 'control_unit_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:10 ; elapsed = 00:00:24 . Memory (MB): peak = 788.480 ; gain = 0.000
restart
INFO: [Simtool 6-17] Simulation restarted
run all
begin testbench.

Test complete.

$finish called at time : 160 ns : File "C:/Users/Colin/140/factorial_calculator/factorial_calculator.srcs/sim_1/new/control_unit_tb.v" Line 80

```

Figure 2b: control_unit_tb.v Tcl console output.

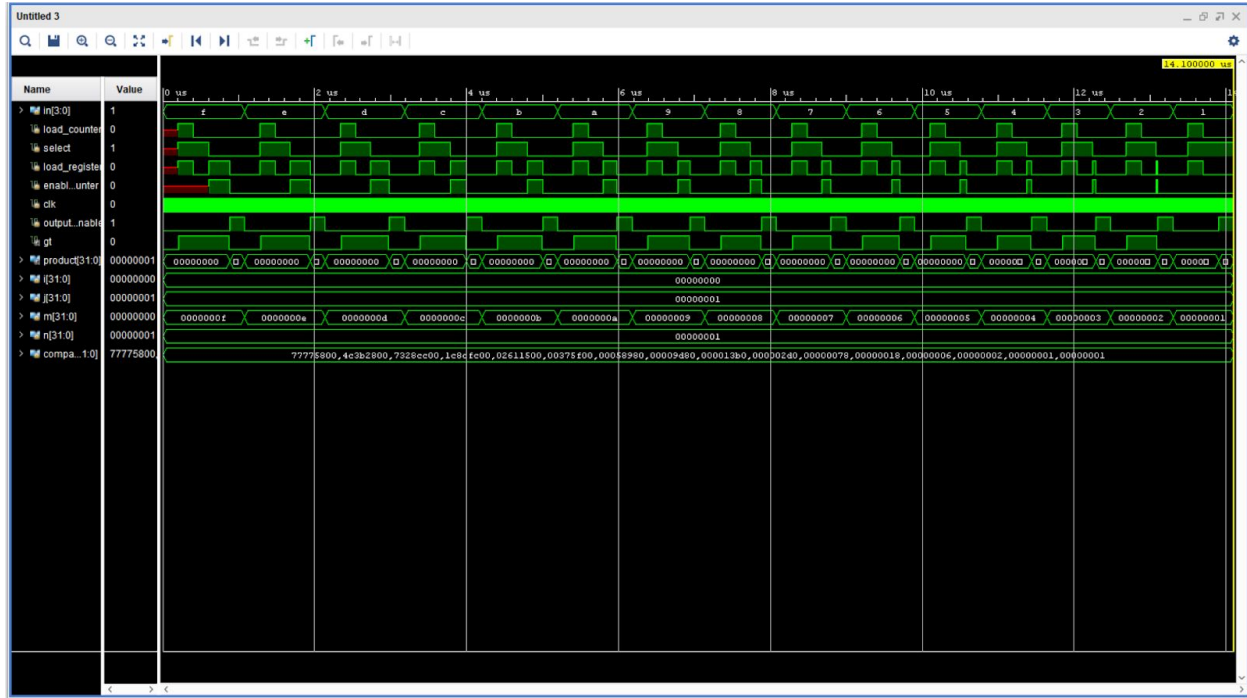


Figure 3a: datapath_tb.v simulation results.

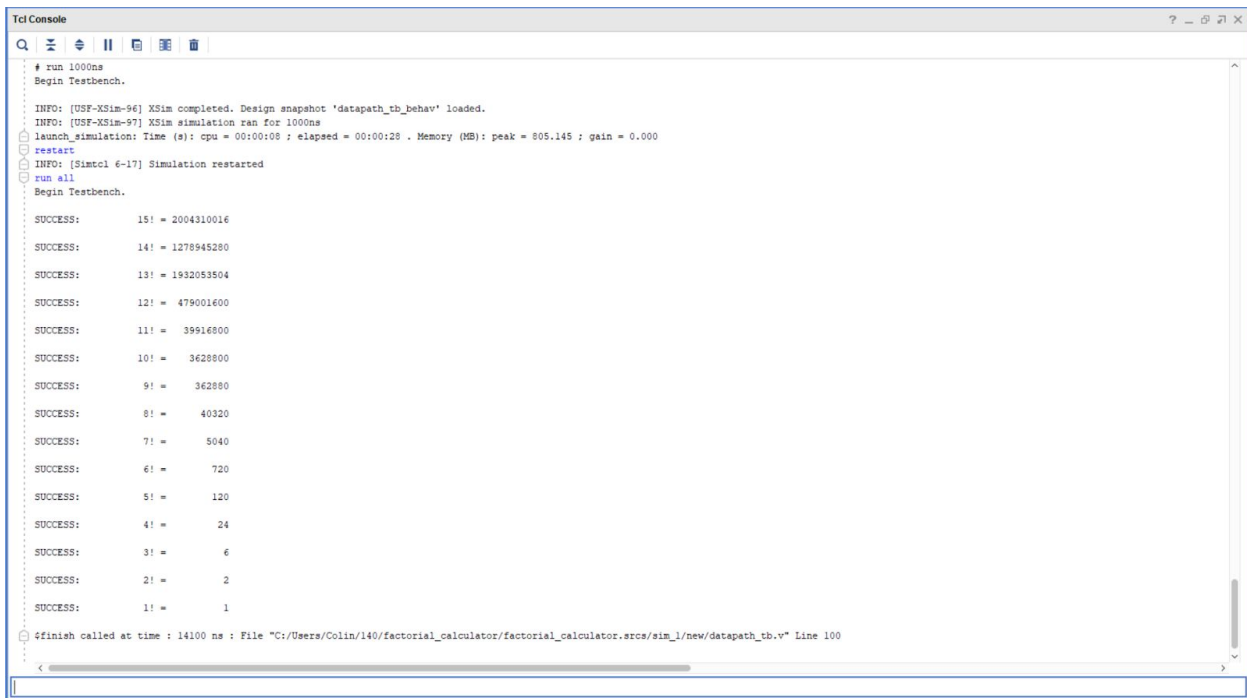


Figure 3b: datapath_tb.v Tcl console output.

FPGA Validation:

For hardware validation, shown in the pictures below of input 7 and 0 which started in state 0 with Go flag set. The results are shown in decimal with the output of 5040, which is correct for 7!, and output 1 which is correct for 0!.

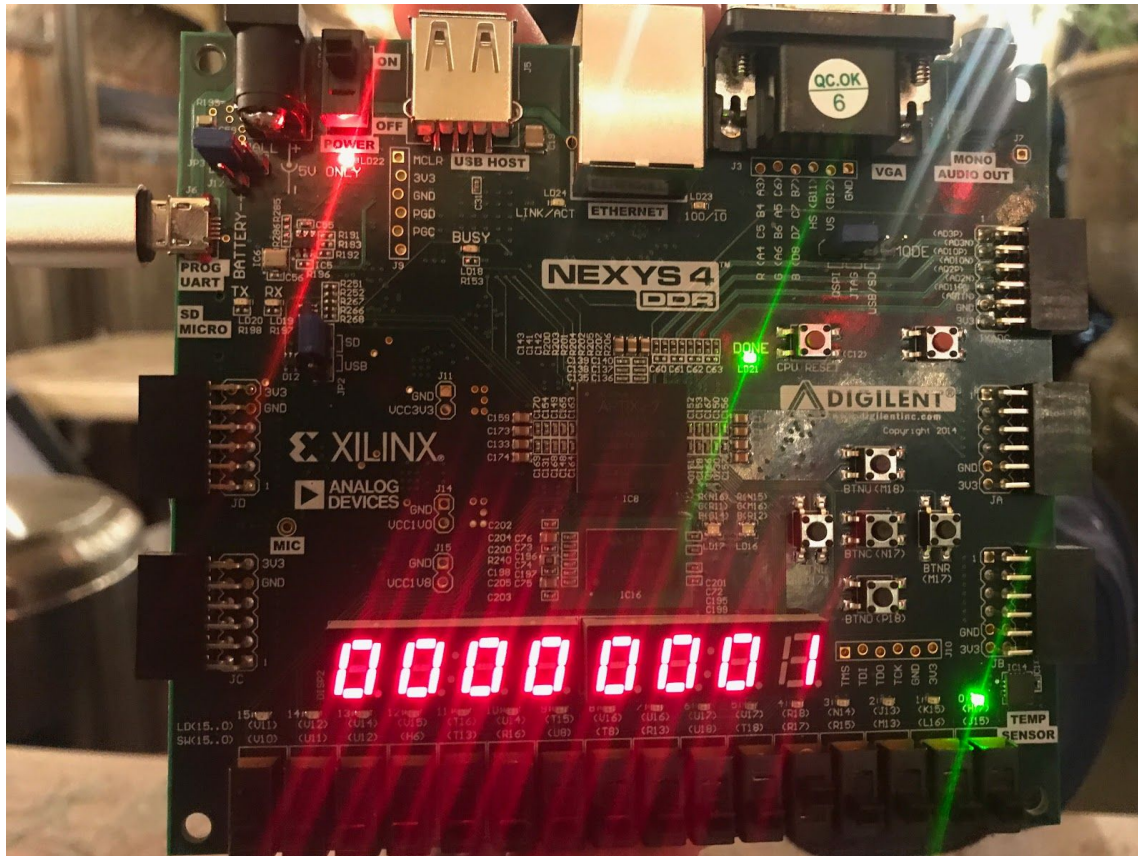


Figure 4a: FPGA Validation Result of Factorial 0.

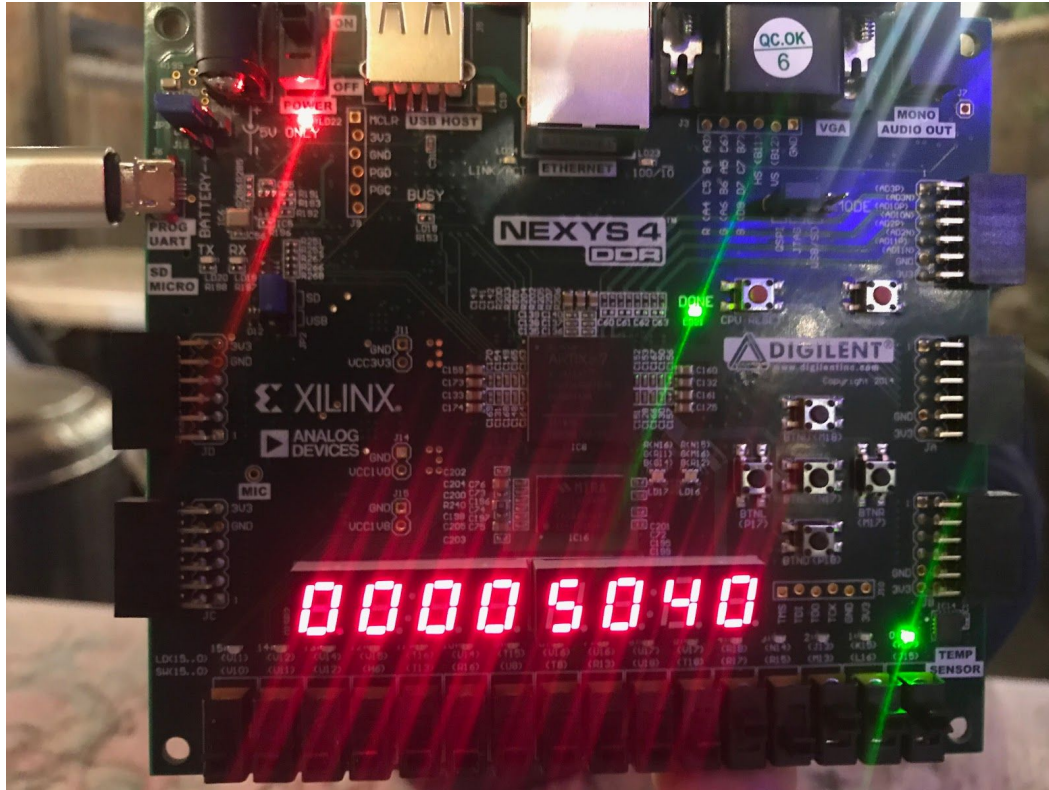


Figure 4b: FPGA Validation Result of Factorial 7.

Conclusion:

All tasks outlined in the *Procedure* section were successful. Conflict originally arose when implementing the top-level design on the FPGA board, but this was rectified and board functionality was completed as intended. Proper functionality of source code was also verified by simulations. This lab was a good test of retained knowledge from prerequisite courses.

Appendix:

Diagrams:

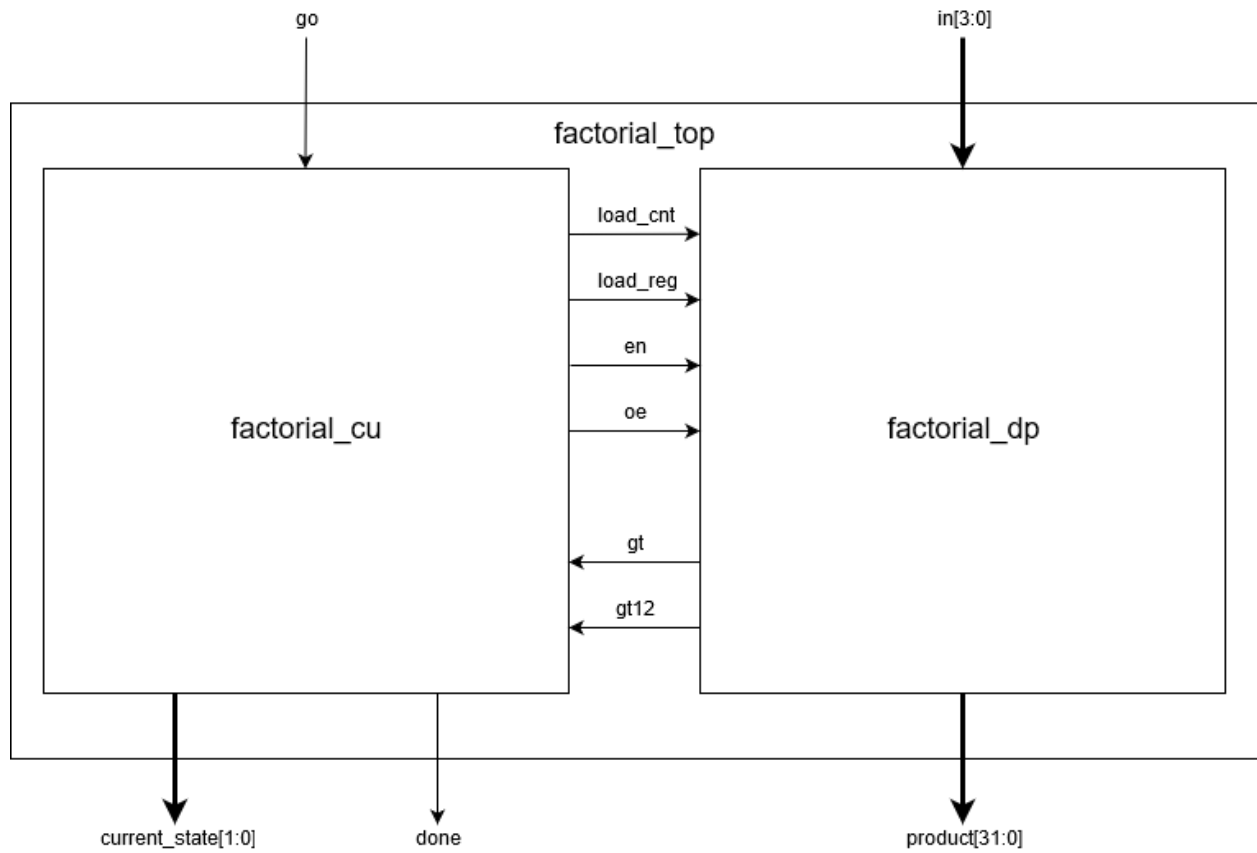


Figure 5: Top Module Block Diagram.

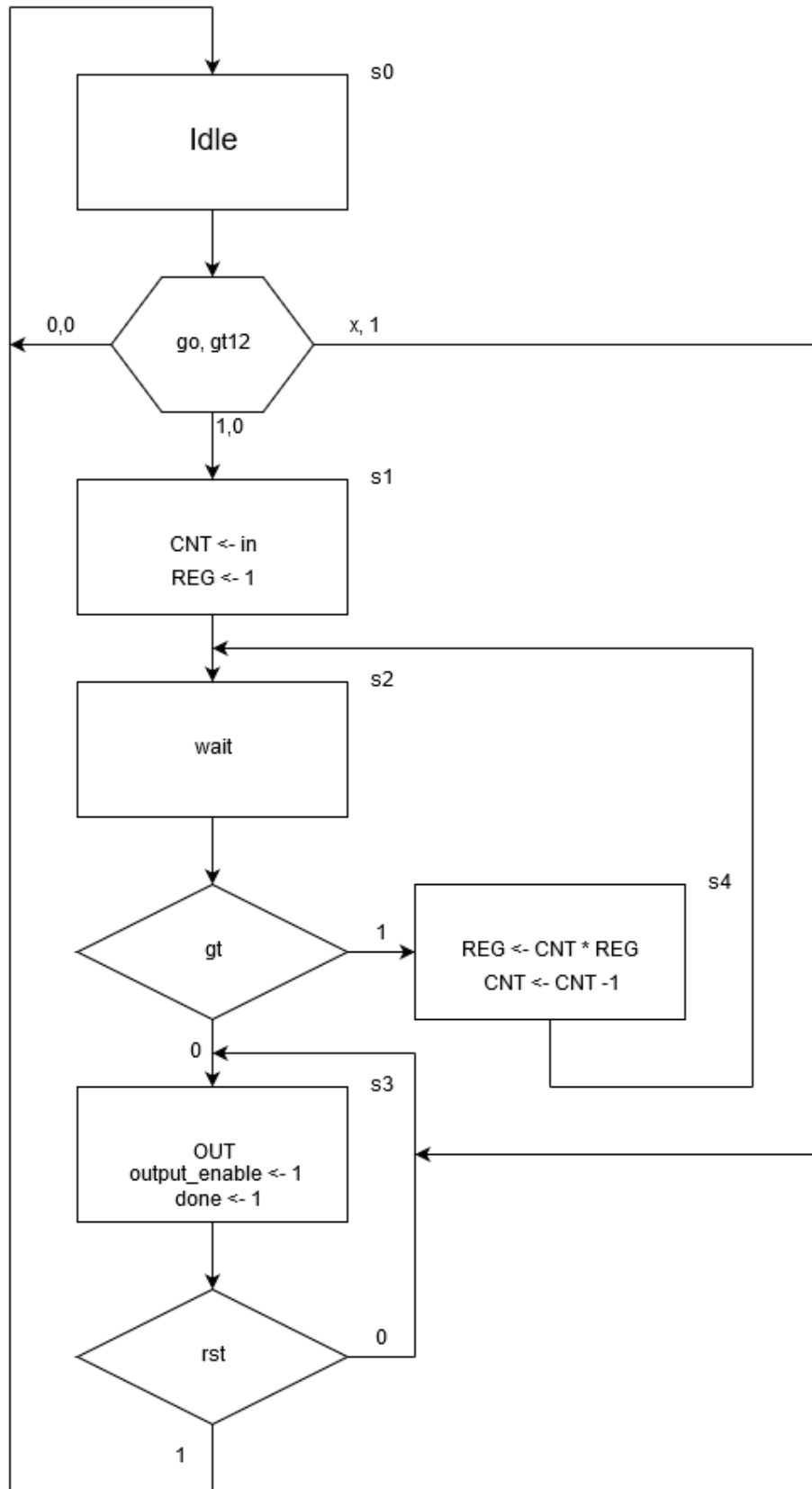


Figure 6: Control Unit Finite State Machine.

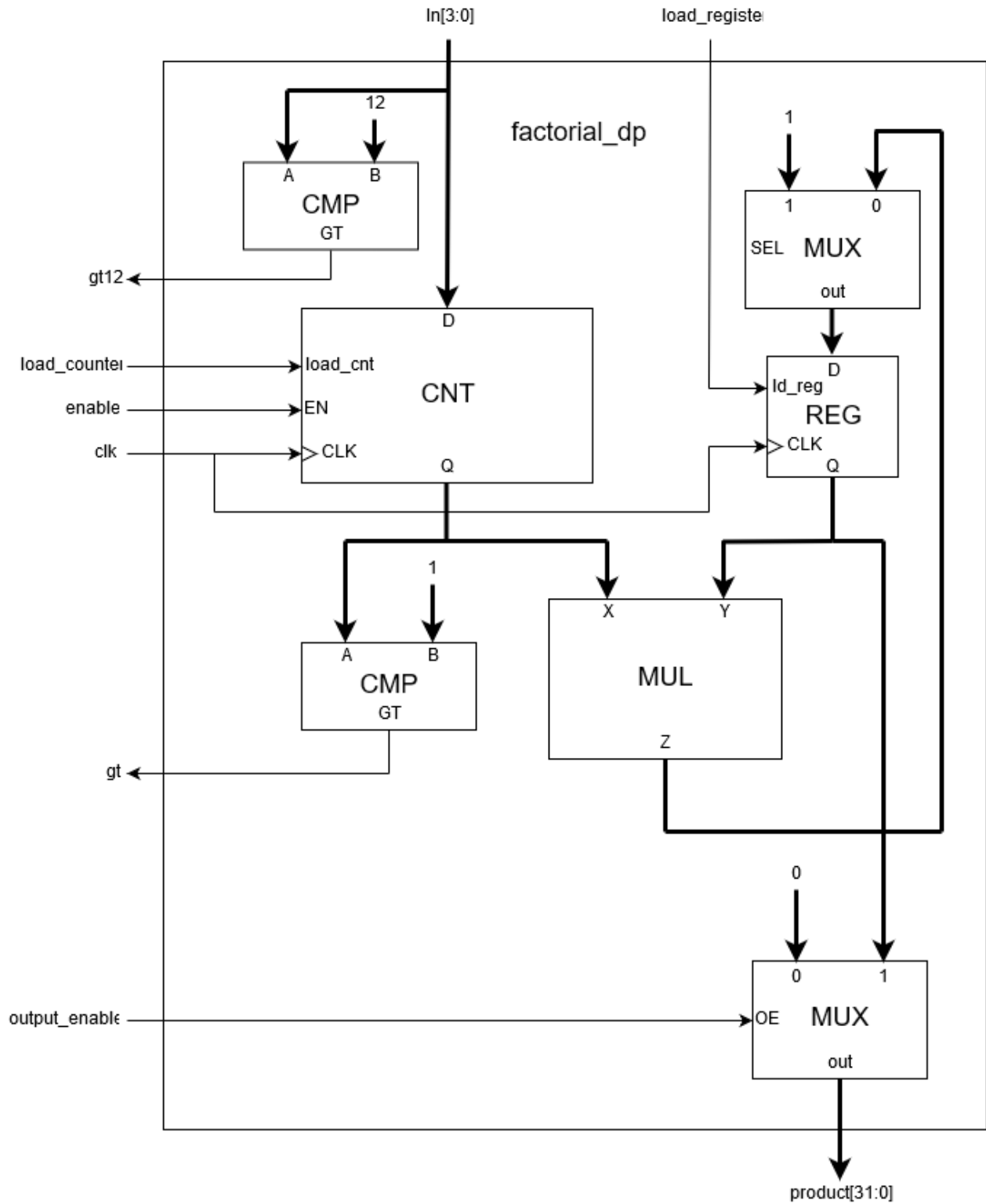


Figure 7: Data Path Module.

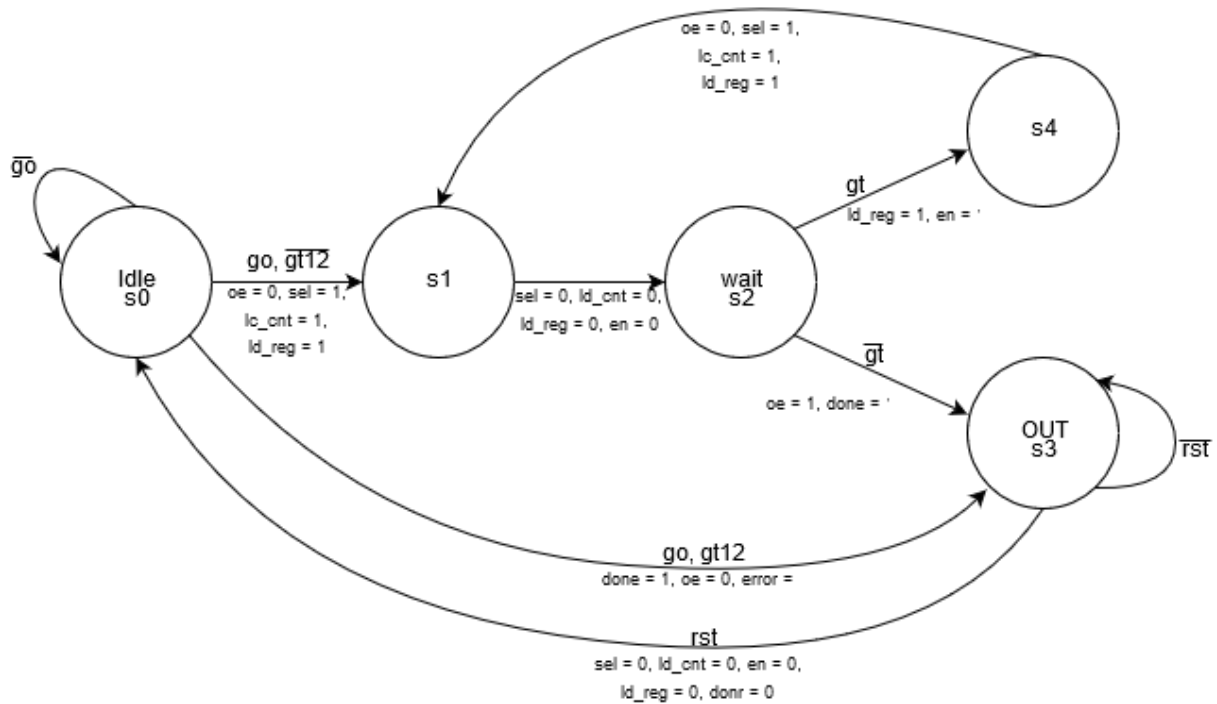


Figure 8: State Transition Diagram.

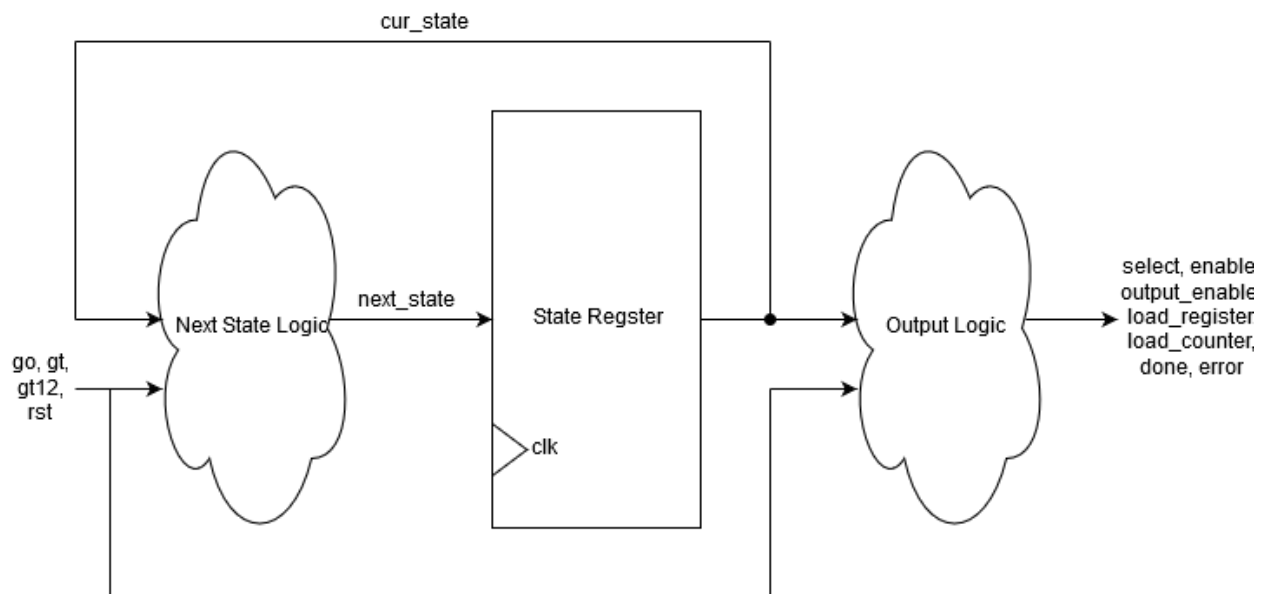


Figure 9: Next State Logic Diagram.

CS	go	load_cnt	load_reg	cnt_en	sel1	sel2	ud	GT	NS	DONE
0	0	0	0	0	1	1	x	x	0	0
0	1	0	0	0	1	1	x	x	1	0
1	x	1	1	1	1	1	x	x	2	0
2	x	0	0	1	0	1	x	0	4	0
2	x	0	0	1	0	1	x	1	3	0
3	x	0	1	1	0	0	0	x	2	0
4	x	0	0	0	0	0	x	x	0	1

Table 1: State Output Table

Source Code:

factorial_top.v
<pre> `timescale 1ns / 1ps /// // Company: // Engineer: // // Create Date: // Design Name: // Module Name: Factorial // Project Name: // Target Devices: // Tool Versions: // Description: // // Dependencies: // // Revision: // Revision 0.01 - File Created // Additional Comments: // /// module Factorial(input [3:0] n, input Go, clk, rst, output Done, output [31:0] out); wire [5:0] ctrl; wire Is_Gt; </pre>


```

Factorial_CU U0 (.Go(Go), .clk(clk), .rst(rst), .Is_Gt(Is_Gt), .Done(Done), .ctrl(ctrl));
Factorial_DP U1 ( .n(n), .ctrl(ctrl), .clk(clk), .rst(rst), .out(out), .Is_GT(Is_GT));
endmodule

```

control_unit.v

```

`timescale 1ns / 1ps
/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:
// Design Name:
// Module Name: Factorial_CU
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
/////////////////////////////////////////////////////////////////

```

```

module Factorial_CU(input Go, clk, rst, Is_Gt,
                    output reg Done,
                    output reg [5:0] ctrl);

```

```

    reg [3:0] ns, cs;

```

```

    parameter

```

```

        IDLE = 6'b0_0_0_0_0_0,
        S1 = 6'b0_1_0_1_1_0,
        S2 = 6'b1_0_0_0_0_0,
        S3 = 6'b1_0_0_0_0_1,
        S4 = 6'b1_1_0_0_1_0;

```

```

always @(Go, Is_Gt, Done, ns, cs)

```

```

begin

```

```

    case(cs)

```

```

        4'd0:

```

```

            begin

```

```

        Done = 0;
        if(Go) begin ns = 4'd1; end
        else begin ns = 4'd0; end
    end
4'd1: begin Done = 0; ns = 4'd2; end
4'd2:
    begin
        if(Is_Gt) begin ns = 4'd4; Done = 0; end
        else begin ns = 4'd3; Done = 0; end
    end
4'd3:
    begin
        Done = 1;
        ns = 4'd0;
    end
4'd4:
    begin
        Done = 0;
        ns = 4'd2;
    end
    default: ns = 4'd0;
endcase
end

always @(posedge clk, posedge rst)
begin
    if(rst)
        cs <= 0;
    else
        cs <= ns;
end

always @(cs, ctrl) //{Sel1, ce, ud, ld1, ld2, sel2} = ctrl;
begin
    case(cs)
        4'd0: ctrl = IDLE;
        4'd1: ctrl = S1;
        4'd2: ctrl = S2;
        4'd3: ctrl = S3;
        4'd4: ctrl = S4;
    endcase
end
endmodule

```

datapath.v

```
`timescale 1ns / 1ps
////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:
// Design Name:
// Module Name: Factorial_DP
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////

module Factorial_DP(input [3:0] n,
                    input [5:0] ctrl,
                    input  clk, rst,
                    output [31:0] out,
                    output Is_GT);

wire [3:0] Cnt_Reg;
wire [31:0] q_out, aluout, q2_out, mux_out1, mux_out2;
wire IS_equal;
Ud_Cnt_4 u1(.D(n), .LD(ctrl[2]), .UD(ctrl[3]), .CE(ctrl[4]), .CLK(clk), .RST(rst), .Q(Cnt_Reg));
comparater u2(.A({28'd0,Cnt_Reg}), .B(32'd1), .greater(Is_GT));
Dreg u3(.D(mux_out1), .en(ctrl[1]), .clk(clk), .rst(rst), .Q(q_out));
Alu u4(.in1(q_out), .in2({28'd0,Cnt_Reg}), .c(2'b00), .aluout(aluout));
Mux u6 (.in1(32'd1), .in2(aluout), .sel(ctrl[5]), .out(mux_out1));
Mux u7 (.in1(32'd0), .in2(aluout), .sel(ctrl[0]), .out(out));
endmodule
```

factorial_FPGA.v

```

`timescale 1ns / 1ps
/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:
// Design Name:
// Module Name: Factorial_fpga
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
/////////////////////////////////////////////////////////////////

module Factorial_fpga(input [3:0] n,
                     input go, clk100MHz, control, rst,
                     output Done,
                     output [7:0] LEDOUT, LEDSEL );
supply1 [7:0] vcc;
wire DONT_USE, clk_5KHz;
wire [31:0] out;
wire [3:0] dig0, dig1, dig2, dig3, dig4, dig5, dig6, dig7;
wire [6:0] out0, out1, out2, out3, out4, out5, out6, out7;
Factorial U0 (.n(n), .Go(go), .clk(debouncedButton), .rst(rst), .Done(Done), .out(out));
bin2bcd32 U1 (.value(out), .dig0(dig0), .dig1(dig1), .dig2(dig2), .dig3(dig3), .dig4(dig4), .dig5(dig5),
.dig6(dig6), .dig7(dig7));
showNumber U2 (.dig0(dig0), .dig1(dig1), .dig2(dig2), .dig3(dig3), .dig4(dig4), .dig5(dig5),
.dig6(dig6), .dig7(dig7),
.out0(out0), .out1(out1), .out2(out2), .out3(out3), .out4(out4), .out5(out5), .out6(out6),
.out7(out7));
led_mux U3(clk_5KHz, rst, {1'b1, out7}, {1'b1, out6}, {1'b1, out5}, {1'b1, out4}, {1'b1, out3}, {1'b1,
out2}, {1'b1, out1}, {1'b1, out0}, LEDOUT, LEDSEL);
clk_gen U4(.clk100MHz(clk100MHz), .rst(rst), .clk_4sec(DONT_USE), .clk_5KHz(clk_5KHz));
button_debouncer U5(.clk(clk_5KHz), .button(control), .debounced_button(debouncedButton));
endmodule

```

factorial_FPGA.xdc

```

set_property -dict { PACKAGE_PIN E3  IOSTANDARD LVCMOS33 } [get_ports { clk100MHz }];
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {clk100MHz}];

set_property -dict { PACKAGE_PIN J15  IOSTANDARD LVCMOS33 } [get_ports { n[0] }];
set_property -dict { PACKAGE_PIN L16  IOSTANDARD LVCMOS33 } [get_ports { n[1] }];
set_property -dict { PACKAGE_PIN M13  IOSTANDARD LVCMOS33 } [get_ports { n[2] }];
set_property -dict { PACKAGE_PIN R15  IOSTANDARD LVCMOS33 } [get_ports { n[3] }];

set_property -dict { PACKAGE_PIN H17  IOSTANDARD LVCMOS33 } [get_ports { Done }];

set_property -dict { PACKAGE_PIN M17  IOSTANDARD LVCMOS33 } [get_ports { rst }];
set_property -dict { PACKAGE_PIN N17  IOSTANDARD LVCMOS33 } [get_ports { control }];

set_property -dict { PACKAGE_PIN R17  IOSTANDARD LVCMOS33 } [get_ports { go }];

set_property -dict { PACKAGE_PIN K13  IOSTANDARD LVCMOS33 } [get_ports { LEDOUT[0]
}];
set_property -dict { PACKAGE_PIN K16  IOSTANDARD LVCMOS33 } [get_ports { LEDOUT[1]
}];
set_property -dict { PACKAGE_PIN P15  IOSTANDARD LVCMOS33 } [get_ports { LEDOUT[2]
}];
set_property -dict { PACKAGE_PIN L18  IOSTANDARD LVCMOS33 } [get_ports { LEDOUT[3]
}];
set_property -dict { PACKAGE_PIN R10  IOSTANDARD LVCMOS33 } [get_ports { LEDOUT[4]
}];
set_property -dict { PACKAGE_PIN T11  IOSTANDARD LVCMOS33 } [get_ports { LEDOUT[5]
}];
set_property -dict { PACKAGE_PIN T10  IOSTANDARD LVCMOS33 } [get_ports { LEDOUT[6]
}];
set_property -dict { PACKAGE_PIN H15  IOSTANDARD LVCMOS33 } [get_ports { LEDOUT[7]
}];

set_property -dict { PACKAGE_PIN J17  IOSTANDARD LVCMOS33 } [get_ports { LEDSEL[0] }];
set_property -dict { PACKAGE_PIN J18  IOSTANDARD LVCMOS33 } [get_ports { LEDSEL[1] }];
set_property -dict { PACKAGE_PIN T9   IOSTANDARD LVCMOS33 } [get_ports { LEDSEL[2] }];
set_property -dict { PACKAGE_PIN J14  IOSTANDARD LVCMOS33 } [get_ports { LEDSEL[3] }];
set_property -dict { PACKAGE_PIN P14  IOSTANDARD LVCMOS33 } [get_ports { LEDSEL[4] }];
set_property -dict { PACKAGE_PIN T14  IOSTANDARD LVCMOS33 } [get_ports { LEDSEL[5]
}];
set_property -dict { PACKAGE_PIN K2   IOSTANDARD LVCMOS33 } [get_ports { LEDSEL[6] }];
set_property -dict { PACKAGE_PIN U13  IOSTANDARD LVCMOS33 } [get_ports { LEDSEL[7]
}];

```

datapath_tb.v

```
`timescale 1ns / 1ps
/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:
// Design Name:
// Module Name: dp_tb
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
/////////////////////////////////////////////////////////////////

module dp_tb();
reg [3:0] n_tb;
reg [5:0] ctrl_tb;
reg clk_tb, rst_tb;
wire [31:0] out_tb;
wire Is_GT_tb;
integer i;

Factorial_DP DUT( .n(n_tb), .ctrl(ctrl_tb), .clk(clk_tb), .rst(rst_tb), .out(out_tb), .Is_GT(Is_GT_tb));

parameter
    IDLE = 6'b0_0_0_0_0_0,
    S1 = 6'b0_1_0_1_1_0,
    S2 = 6'b1_0_0_0_0_0,
    S3 = 6'b1_0_0_0_0_1,
    S4 = 6'b1_1_0_0_1_0;

task tick; begin #5 clk_tb = 1; #5 clk_tb = 0; end endtask
initial
begin
    clk_tb = 0; rst_tb = 0;
```



```

for(i = 0; i < 13; i = i + 1)
begin
    n_tb = i;
    ctrl_tb = IDLE; tick;
    ctrl_tb = S1; tick;
    while(Is_GT_tb)
    begin
        ctrl_tb = S2; tick;
        ctrl_tb = S4; tick;
    end
    ctrl_tb = S3; tick;
    case(i)
        4'd0: begin if(out_tb != 0) $display("Error with 0"); end
        4'd1: begin if(out_tb != 1) $display("Error with 1"); end
        4'd2: begin if(out_tb != 2) $display("Error with 2"); end
        4'd3: begin if(out_tb != 6) $display("Error with 3"); end
        4'd4: begin if(out_tb != 24) $display("Error with 4"); end
        4'd5: begin if(out_tb != 120) $display("Error with 5"); end
        4'd6: begin if(out_tb != 720) $display("Error with 6"); end
        4'd7: begin if(out_tb != 5040) $display("Error with 7"); end
        4'd8: begin if(out_tb != 40320) $display("Error with 8"); end
        4'd9: begin if(out_tb != 362880) $display("Error with 9"); end
        4'd10: begin if(out_tb != 3628800) $display("Error with 10"); end
    endcase

    end
    $display("Sucess");
    $finish;
end

endmodule

```

cu_tb.v

```

`timescale 1ns / 1ps
/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:
// Design Name:
// Module Name: cu_tb
// Project Name:

```

```

// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////

module cu_tb();
reg Go_tb, clk_tb, rst_tb, Is_Gt_tb;
wire Done_tb;
wire [5:0] ctrl_tb;

Factorial_CU DUT(.Go(Go_tb), .clk(clk_tb), .rst(rst_tb), .Is_Gt(Is_Gt_tb), .Done(Done_tb),
.ctrl(ctrl_tb));
parameter
    IDLE = 6'b0_0_0_0_0_0,
    S1 = 6'b0_1_0_1_1_0,
    S2 = 6'b1_0_0_0_0_0,
    S3 = 6'b1_0_0_0_0_1,
    S4 = 6'b1_1_0_0_1_0;

task tick; begin #5 clk_tb = 1; #5 clk_tb = 0; end endtask
initial
begin
    clk_tb = 0; rst_tb = 0; Go_tb = 1; Is_Gt_tb = 1;
    tick;
    if(ctrl_tb != IDLE) $display("Error with IDLE");
    tick;
    if(ctrl_tb != S1) $display("Error with S1");
    tick;
    if(ctrl_tb != S2) $display("Error with S2");
    tick;
    if(ctrl_tb != S4) $display("Error with S4");
    tick;
    Is_Gt_tb = 0;
    tick;
    if(ctrl_tb != S3)
    begin
        $display("Error with S3");
        if(Done_tb != 1) $display("Error");
    end
end

```

```
    $display("Success");
    $finish;
end
endmodule
```

factorial_top_tb.v

```
`timescale 1ns / 1ps
/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:
// Design Name:
// Module Name: Factorial_tb
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
/////////////////////////////////////////////////////////////////

module Factorial_tb();
reg [3:0] n_tb;
reg Go_tb, clk_tb, rst_tb;
wire Done_tb;
wire [31:0] out_tb;
integer i;
Factorial DUT (.n(n_tb), .Go(Go_tb), .clk(clk_tb), .rst(rst_tb),.Done(Done_tb), .out(out_tb));
task tick;
    begin
        #5 clk_tb = 1;
        #5 clk_tb = 0;
    end
endtask
initial
begin
    n_tb = 0; clk_tb = 0; rst_tb = 0;
```

```

Go_tb = 0; tick; tick; tick; tick; tick; tick; tick; tick; tick;
Go_tb = 1;
for(i = 0; i < 11; i = i + 1)
begin
    n_tb = i;
    tick;
    while(!Done_tb)
    begin
        tick;
    end
    case(i)
        4'd0: begin if(out_tb != 1) $display("Error with 0"); end
        4'd1: begin if(out_tb != 1) $display("Error with 1"); end
        4'd2: begin if(out_tb != 2) $display("Error with 2"); end
        4'd3: begin if(out_tb != 6) $display("Error with 3"); end
        4'd4: begin if(out_tb != 24) $display("Error with 4"); end
        4'd5: begin if(out_tb != 120) $display("Error with 5"); end
        4'd6: begin if(out_tb != 720) $display("Error with 6"); end
        4'd7: begin if(out_tb != 5040) $display("Error with 7"); end
        4'd8: begin if(out_tb != 40320) $display("Error with 8"); end
        4'd9: begin if(out_tb != 362880) $display("Error with 9"); end
        4'd10: begin if(out_tb != 3628800) $display("Error with 10"); end
    endcase
end
$display("Sucess");
$finish;
end
endmodule

```