Lab. 08

Logic Design Lab.

Spring 2022

Prof. ChangGun Lee

(cglee@snu.ac.kr)

TA. Hayeon Park

TA. Haejoo Jeon

TA. Seohwan Yoo

(rubis.ld.ta@gmail.com)

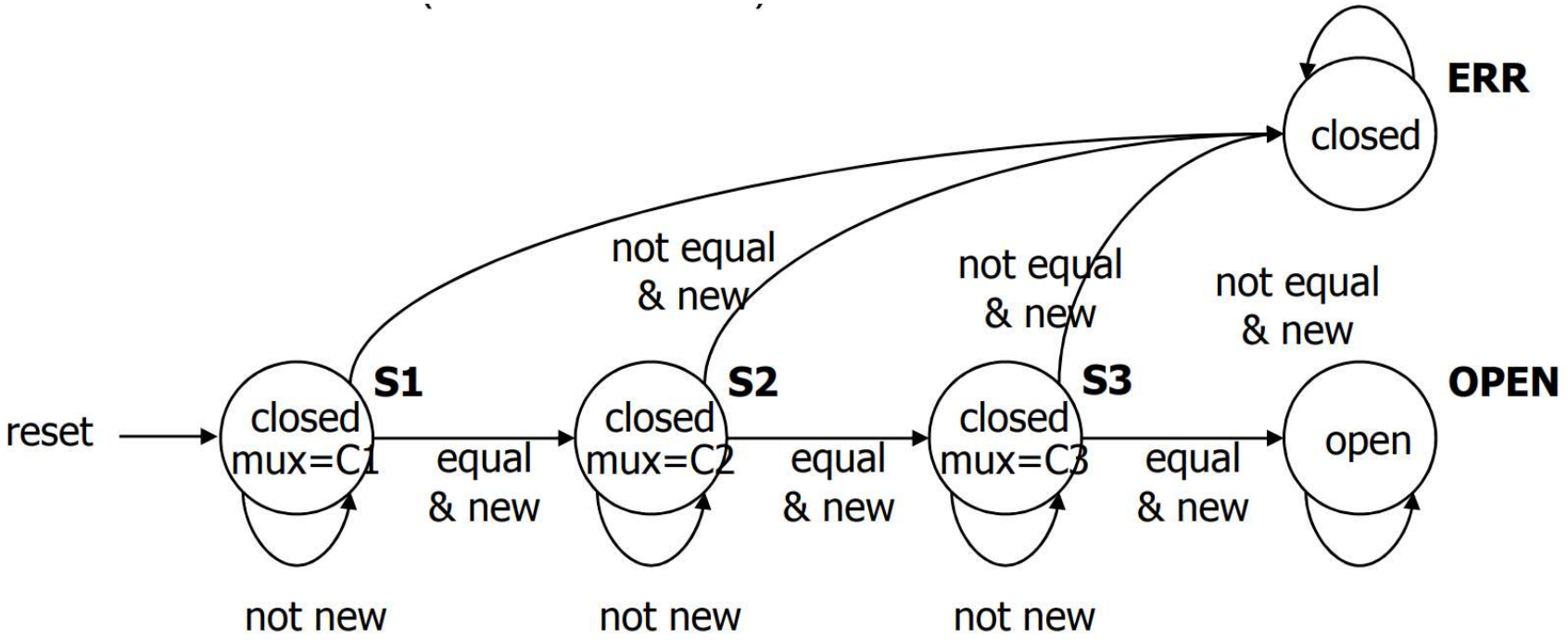
Contents

- Finite state machine
- Moore/Mealy machine
- Implementation on Logic Design Board
- Vending machine

Announcement

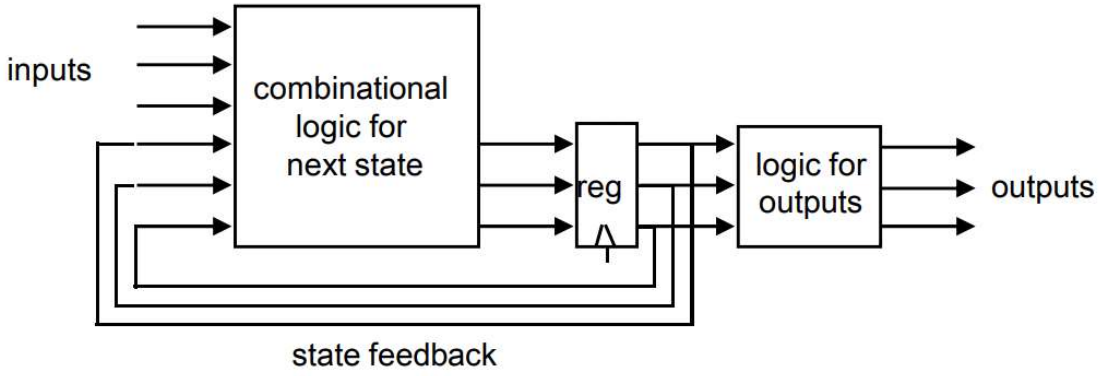
- There will be an announcement of the final project in next lab session.
- Please make sure your board is working correctly

Finite state machine

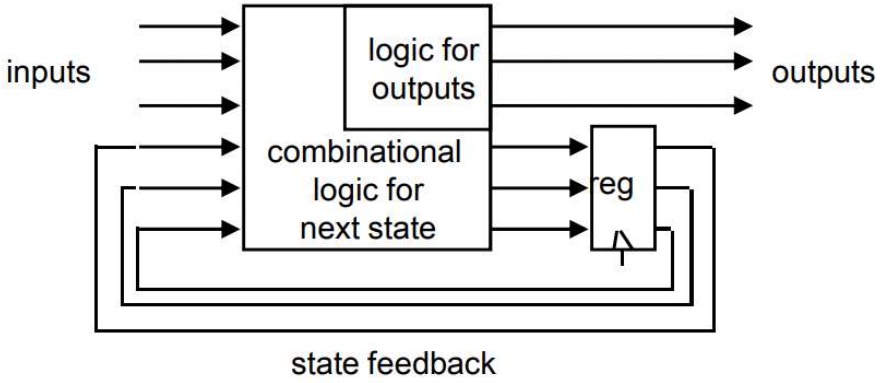


Moore/Mealy machine

- MOORE

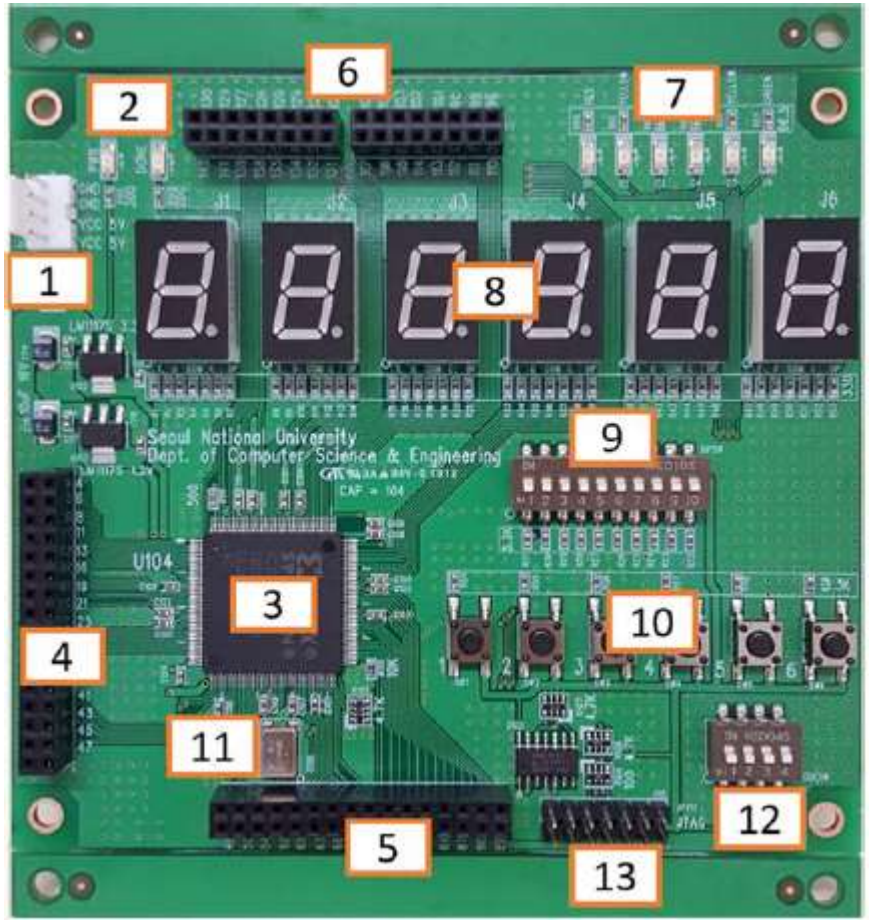


- MEALY



Implementation on Logic Design Board (Review)

SNU Logic Design Board



① Power Input : Molex Connector (DC 5V Only)

② Power, DONE LED

③ FPGA : XC3S50AN-4TQG144C

④ User I/O Ports (P1) : 16X2 2.54mm Pitch

⑤ User I/O Ports (P2) : 16X2 2.54mm Pitch

⑥ User I/O Ports (P3, P4) : 2 8X2 2.54mm Pitch

⑦ User Output LEDs : 6 output LEDs

⑧ User Output LEDs : 6 7-segment LEDs

⑨ User Input Switches : 10pin Dip Switch

⑩ User Input Switches : 6 Tactile Switches

⑪ Oscillator : 50MHz

⑫ Mode Select Switch

⑬ JTAG Header

SNU Logic Design Board

- **Pin Number Mapping**

Pin Num		Component	
P1	3	7-Segment Display [J1]	A
	4		B
	5		C
	6		D
	7		E
	8		F
	10		G
	11	7-Segment Display [J2]	A
	12		B
	13		C
	15		D
	16		E
	18		F
	19		G
	20	7-Segment Display [J3]	A
	21		B
	24		C
	25		D
	27		E
	28		F
	29		G
	30	DIP Switch [DipSW1]	1
	31		2
	32		3
	33		4
	41		5
	42		6
	43		7
	44		8
	45		9
	46		10
	47	Tactile Switch [SW1]	

Pin Num		Component	
P2	48	Tactile Switch [SW2]	
	49	Tactile Switch [SW3]	
	50	Tactile Switch [SW4]	
	51	Tactile Switch [SW5]	
	54	Tactile Switch [SW6]	
	55	7-Segment Display [J4]	A
	58		B
	59		C
	60		D
	62		E
	63		F
	64		G
	68	7-Segment Display [J5]	A
	69		B
	70		C
	71		D
	72		E
	75		F
	76		G
	77	7-Segment Display [J6]	A
	78		B
	79		C
	82		D
	83		E
	84		F
	85		G
	87	LED [D1]	Red
	88	LED [D2]	Yellow
	90	LED [D3]	Green
	91	LED [D4]	Red
	92	LED [D5]	Yellow
	93	LED [D6]	Green

Pin Num	
P3	96
	98
	99
	101
	102
	103
	104
	105
	110
	111
	112
	113
	114
	115
	116
	117

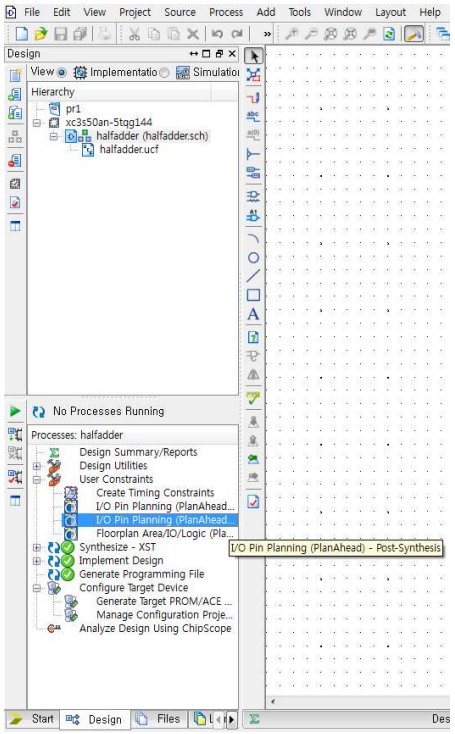
Pin Num	
P4	120
	121
	124
	125
	126
	127
	129
	130
	131
	132
	134
	135
	138
	139
	141
	142

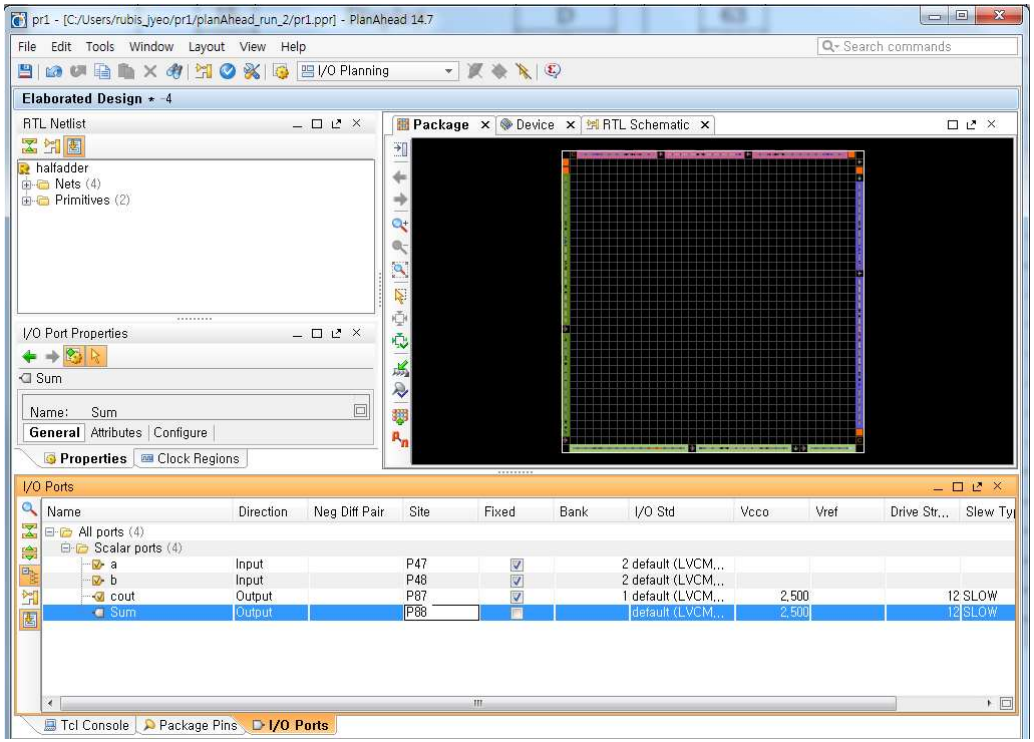
Pin		Component	
JP101	1	JTAG	TMS
	2		TDI
	107		TDO
	109		TCK
Mode SW	37	Mode SW	M1
	38		M0
	39		M2
	144		PROG
FPGA	67	Configuration	INIT
	73		DONE
	74		SUSPEND
FPGA	35	Not Connected (Input Only/VREF)	
	53		
	80		
	97		
	123		
	140		

Pin		Component
FPGA	57	Clock 50MHz
	14	VCCO
	23	
	40	
	61	
	86	
	95	
	119	
	136	
	36	VCCAUX
	66	
	108	
	133	
	22	VCCINT
	52	
	94	
	122	
	9	GND
	17	
	26	
	34	
	56	
	65	
	81	
	89	
	100	
	106	
	118	
	128	
	137	

SNU Logic Design Board

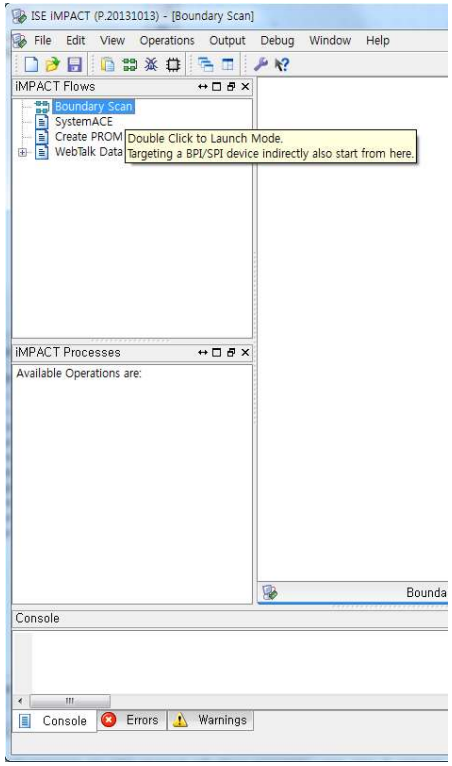
- Assign I/O pins for the half adder

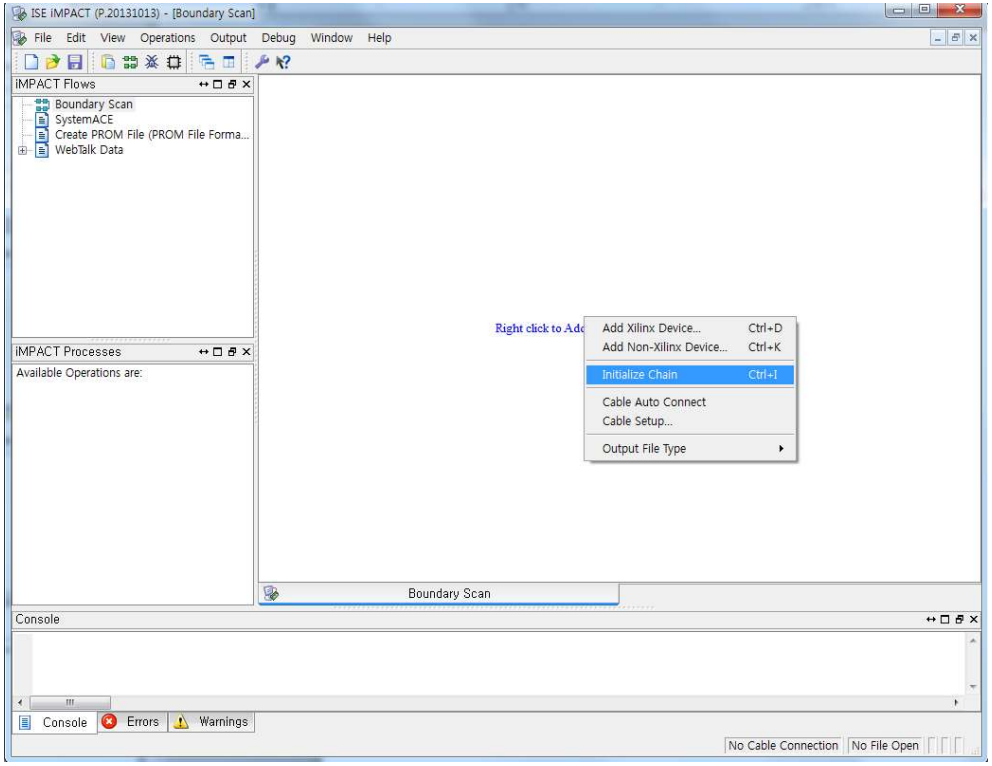




SNU Logic Design Board

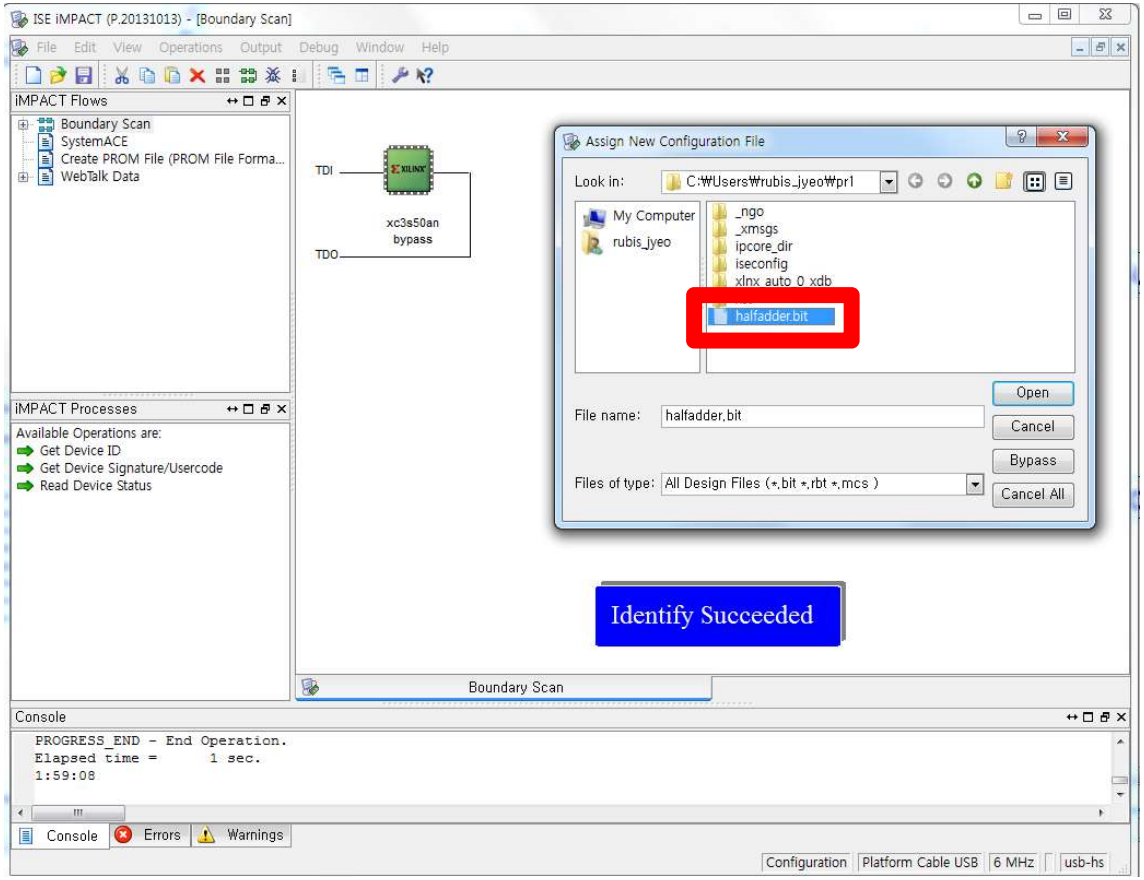
- Boundary Scan > Initialize Chain





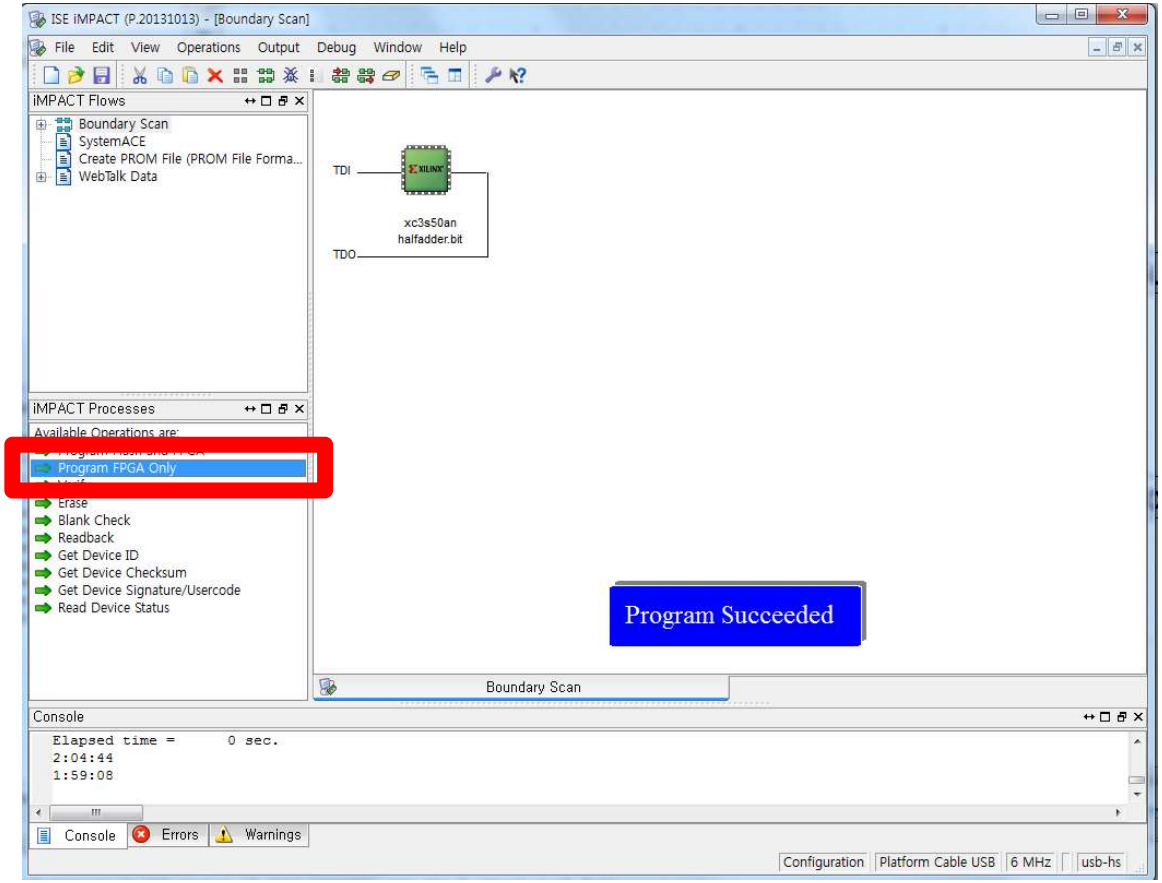
SNU Logic Design Board

- Choose the schematic source you've written.



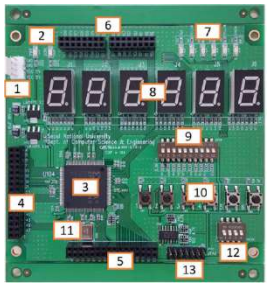
SNU Logic Design Board

- Program FPGA Only > Program Succeeded



LAB

Vending Machine Structure



50MHz Oscillator

Tactile Switch X 3

Vending machine

7 Segment Display X 2

LED

Frequency Divider

(Optional)

Core

Core logic

BCD to 7 Segment

Decoder

LAB 1

1. Implement **core logic of vending machine** in Verilog. It accept nickels(5 cent) and dimes(10 cent). It has only single coin slot for nickels and dimes. The cost of a drink is 15 cents. It release item without change after 15 cents are deposited. Additionally, your implementation should handle external asynchronous reset signal, it should set the deposit back to 0 when the signal is HIGH. If the deposit is greater than 15 cents, consider it as 15 cents.

LAB 2

2. Implement the **Vending machine** in Verilog and upload your implementation to the Logic Design Board. It accept nickels(5 cent) and dimes(10 cent). It has only single coin slot for nickels and dimes. The cost of a drink is 15 cents. It release item without change after 15 cents are deposited. Additionally, your implementation should handle external asynchronous reset signal, it should set the deposit back to 0 when the signal is HIGH. If the deposit is greater than 15 cents, consider it as 15 cents. Use 50MHz oscillator as clock input, and a tactile switch as reset button and coin inputs. For outputs, use two 7-segment displays as your deposit status, and led as release status.

Report

- **Core logic of vending machine**
  1. Result of simulation
     - All possible case of coin inputs
     - Deposit and release status should be 0 when reset = 1

- **Vending machine**
  1. All Verilog source code
     - should include code for module, submodule
       (i.e., Vending machine, core logic, BCD-7 seg decoder)
     - only module code, **no test bench code**
  2. Result of implementation
     - Pictures of all possible states that you design

- **Discussion**

(Optional) Study and compare Moore and Mealy machine

Sample Code (Behavioral Description)

```
`timescale 1ns / 1ps

module freq_divider(
    input clr,
    input clk,
    output reg clkout
    );
         reg[31:0] cnt;
         always@ (posedge clk) begin
                if(clr) begin
                        cnt <=32'd0;
                        clkout <= 1'b0;
                end
                else if (cnt == 32'd25000000) begin
                        cnt <= 32'd0;
                        clkout <= ~clkout;
                end
                else begin
                        cnt <= cnt + 1;
                end
        end

endmodule
```

Frequency Divider

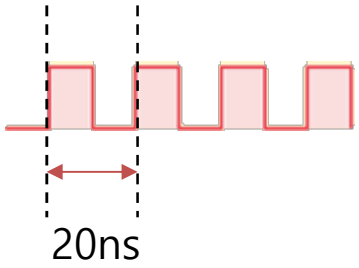
```
1  `timescale 1ns / 1ps
2
3  module bcd_to_7(
4      input [3:0] bcd,
5      output reg [6:0] seg
6      );
7
8      always@(bcd) begin
9        case(bcd)
10           4'd0: seg <= 7'b0111111;
11           4'd1: seg <= 7'b0000110;
12           4'd2: seg <= 7'b1011011;
13           4'd3: seg <= 7'b1001111;
14           4'd4: seg <= 7'b1100110;
15           4'd5: seg <= 7'b1101101;
16           4'd6: seg <= 7'b1111101;
17           4'd7: seg <= 7'b0000111;
18           4'd8: seg <= 7'b1111111;
19           4'd9: seg <= 7'b1101111;
20       endcase
21     end
22
23 endmodule
```

BCD to 7-segment Decoder

Homework

- **Simulating 50MHz clock in test bench**
  - 50MHz : period is 20ns

    (So we should flip clock every 10ns)




```
`timescale 1ns / 1ps

module counter_sim;

    // Inputs
    reg clk;
    reg reset;

    // Outputs
    wire [6:0] cnt;

    // Instantiate the Unit Under Test (UUT)
    counter uut (
        .clk(clk),
        .reset(reset),
        .cnt(cnt)
    );

    always #10 clk=~clk;

    initial begin
        // Initialize Inputs
        clk = 0;
        reset = 1;
    end

endmodule
```

Report

- Write a report
  - Either in Korean or in English
  - Your report should include
    - Discussion
    - Homework
  - # of pages doesn't matter
  - Documents should be submitted as PDF file(**less than 25Mb**)
  - **Attach source code and waveform screenshot**
  - Due :
    Class 001 - May, 23th (Before class begin at 7:00pm)
    Class 002 - May, 24th (Before class begin at 7:00pm)
    Class 003 - May, 25th (Before class begin at 7:00pm)