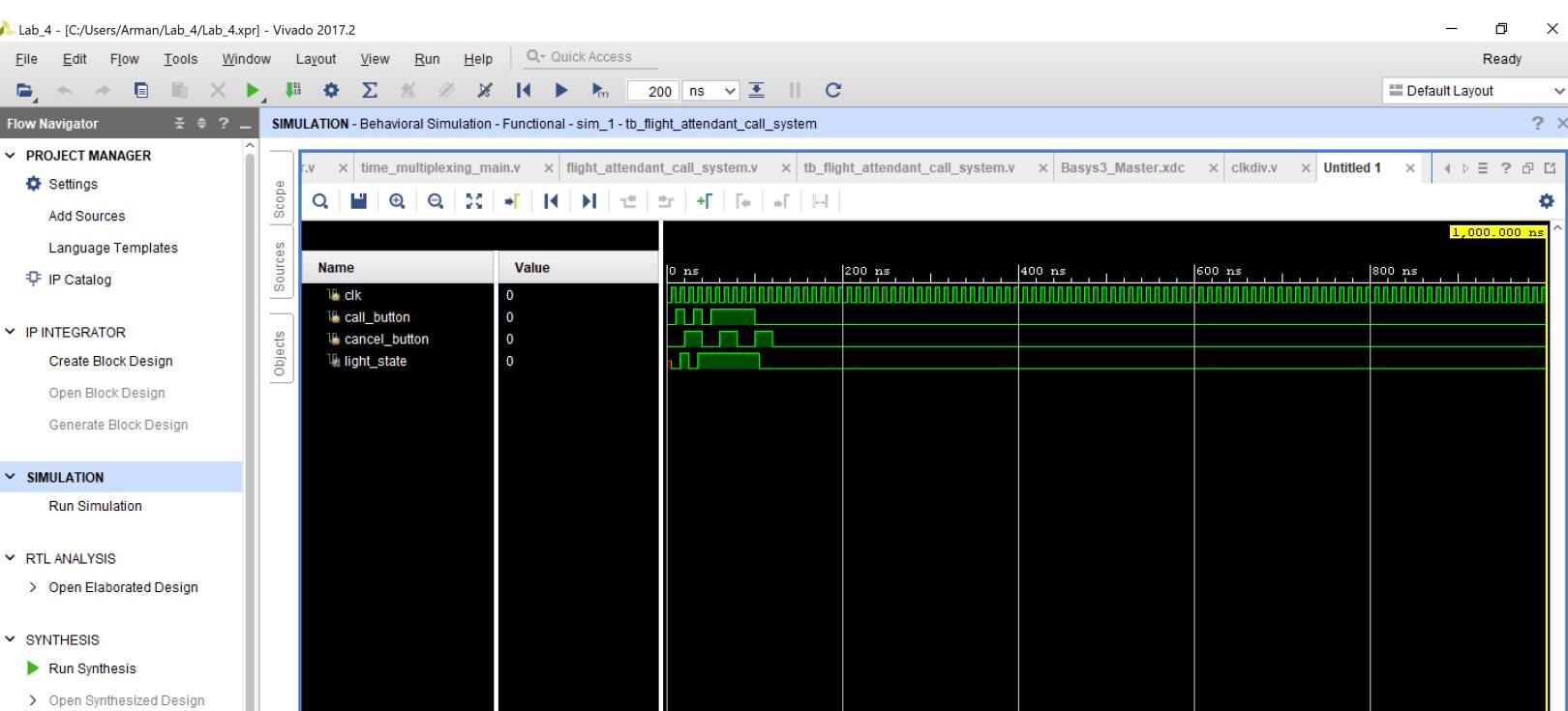


PART 1

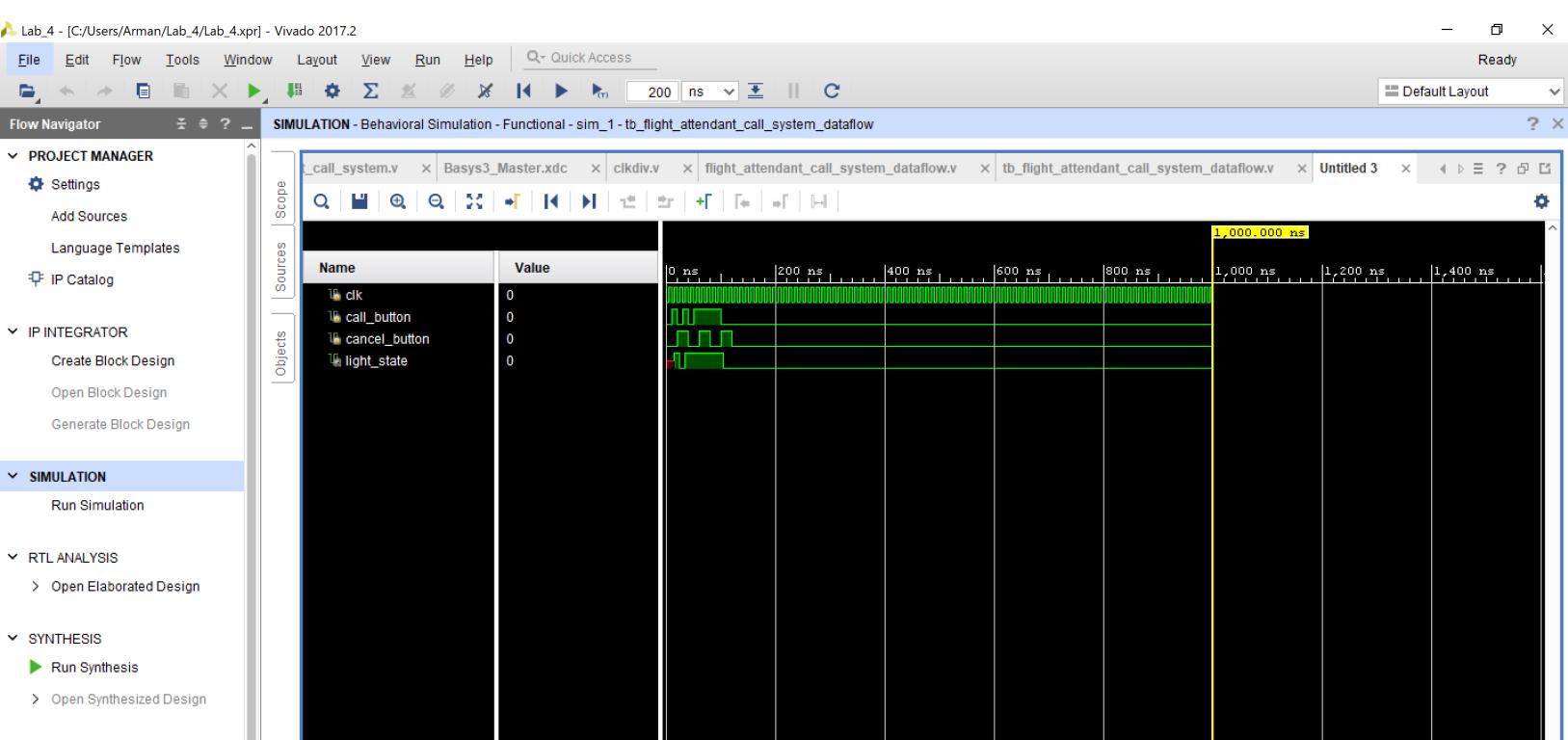
Arman Khondker
EID: aak2464

Simulation waveform Flight attendant Call system



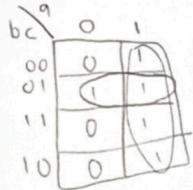
Arman Khondker
EID: aak2464

Simulation Waveform Flight Attendant Call System Data flow



Part 1

K-map



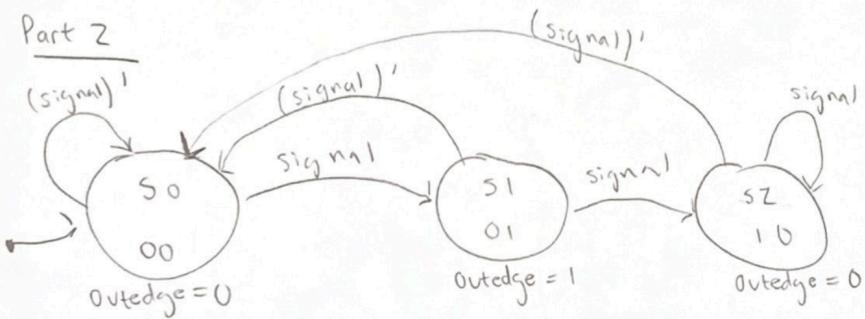
EE 316 Lab 4

Arman Khondker
aak2464

$$\begin{aligned} a &= \text{call} \\ b &= \text{cancel} \\ c &= \text{light-state} \end{aligned}$$

$$y = a + b'c$$

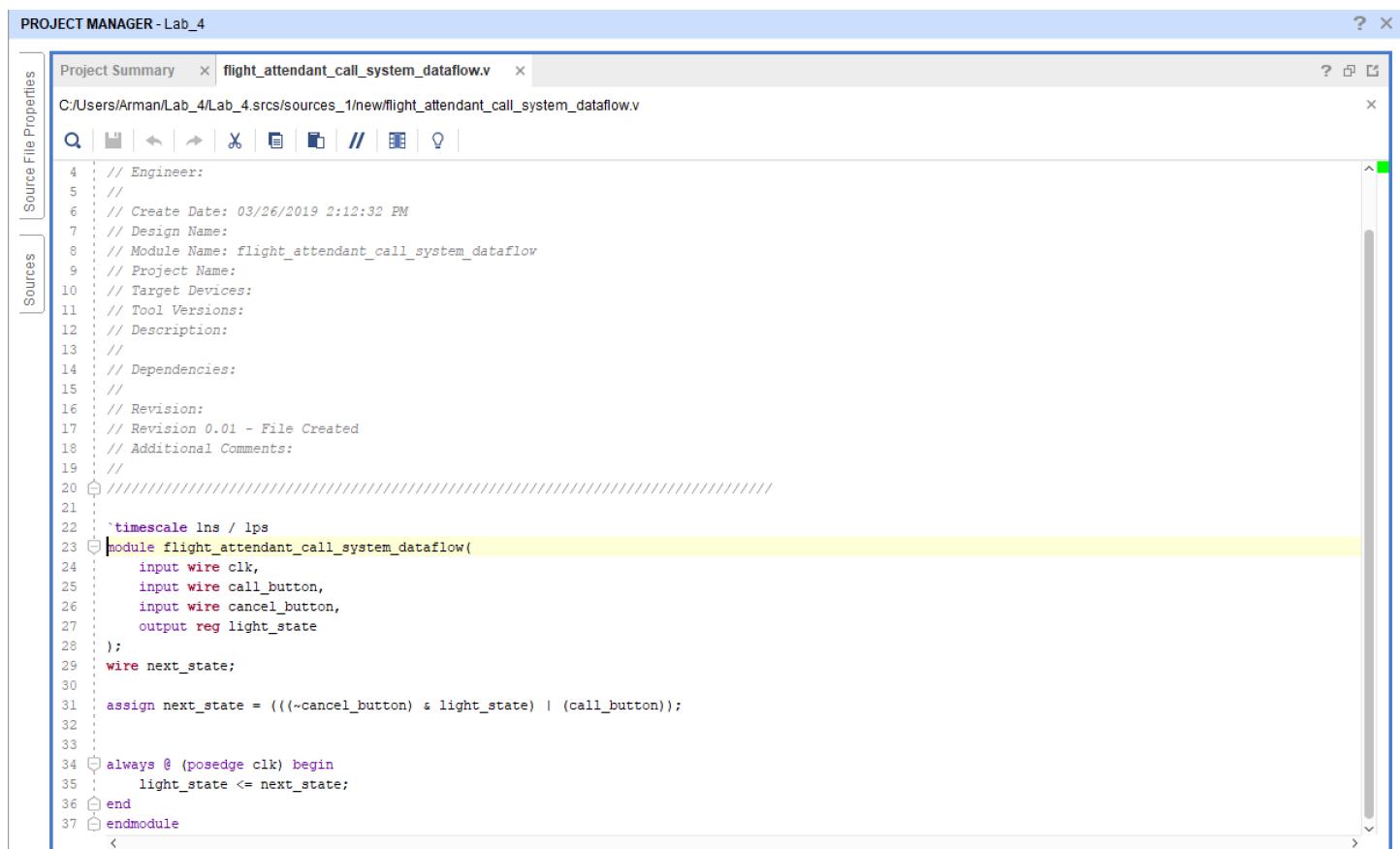
$$\text{Next state} = \text{call} \parallel (b'c)$$

Part 2

Arman Khondker

EID: aak2464

Flight attendant Call System Dataflow



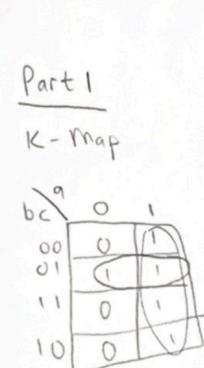
The screenshot shows a software interface titled "PROJECT MANAGER - Lab_4". The main window displays a code editor for a Verilog module named "flight_attendant_call_system_dataflow.v". The code is as follows:

```
4 // Engineer:  
5 //  
6 // Create Date: 03/26/2019 2:12:32 PM  
7 // Design Name:  
8 // Module Name: flight_attendant_call_system_dataflow  
9 // Project Name:  
10 // Target Devices:  
11 // Tool Versions:  
12 // Description:  
13 //  
14 // Dependencies:  
15 //  
16 // Revision:  
17 // Revision 0.01 - File Created  
18 // Additional Comments:  
19 //  
20 ///////////////////////////////////////////////////////////////////  
21  
22 `timescale 1ns / 1ps  
23 module flight_attendant_call_system_dataflow(  
24     input wire clk,  
25     input wire call_button,  
26     input wire cancel_button,  
27     output reg light_state  
28 );  
29     wire next_state;  
30  
31     assign next_state = ((~cancel_button) & light_state) | (call_button);  
32  
33  
34     always @ (posedge clk) begin  
35         light_state <= next_state;  
36     end  
37 endmodule
```

PART 2

Arman Khondker
EID: aak2464

State Diagram of Rising Edge Detector



EE 316 Lab 4

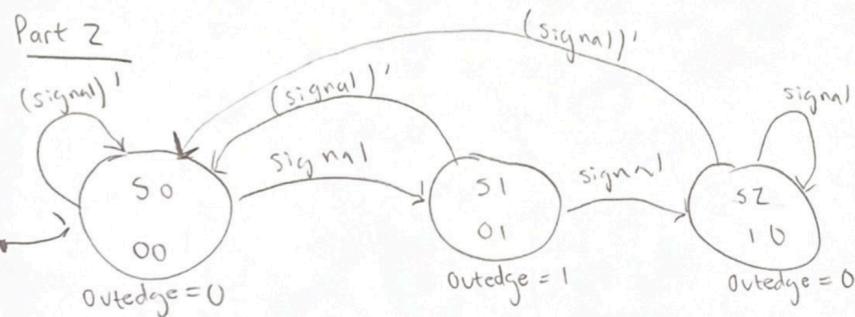
Arman Khondker
aak2464

$$a = \text{call} \quad b = \text{cancel}$$

C = light-state

$$y = a + b'c$$

$$\text{Next state} = \text{call} \parallel (b'c)$$



Arman Khondker

EID: aak2464

The screenshot shows a software interface with a tab bar at the top containing five tabs: "attendant_call_system.v", "rising_edge_detector.v", "tb_rising_edge_detector.v", "clkdiv.v", and several others partially visible. Below the tabs is a toolbar with icons for search, file operations, and help. The main area is a code editor displaying Verilog code for a clock divider module. The code is as follows:

```
15 //  
16 // Revision:  
17 // Revision 0.01 - File Created  
18 // Additional Comments:  
19 //  
20 ///////////////////////////////////////////////////////////////////  
21  
22          Clckdiv  
23 module clkdiv(  
24     input clk,  
25     input reset,  
26     output clk_out  
27 );  
28  
29     reg [1:0] COUNT;  
30  
31     initial begin  
32         COUNT = 0;  
33     end  
34  
35     assign clk_out = COUNT[1];  
36  
37     always @(posedge clk)  
38     begin  
39         if (reset)  
40             COUNT = 0;  
41         else  
42             COUNT = COUNT + 1;  
43     end  
44  
45 endmodule  
46  
47
```

Arman Khondker
EID: aak2464

Rising Edge Detector

```
22
23 module rising_edge_detector(
24     input clk,
25     input signal,
26     input reset,
27     output reg outedge
28 );
29
30 wire slow_clk;
31
32 reg [1:0] state;
33 reg [1:0] next_state;
34
35 clkdiv cl(clk, reset, slow_clk);
36
37
38 always @(*)begin
39
40     case (state)
41
42         2'b00 : begin
43
44             outedge = 1'b0;
45             if (~signal)
46                 next_state = 2'b00;
47             else
48                 next_state = 2'b01;
49             end
50
51         2'b01 : begin
52             outedge = 1'b1;
53             if (~signal)
54                 next_state = 2'b00;
55             else
```

Arman Khondker

EID: aak2464

Rising Edge Detector Continued

The screenshot shows a Verilog code editor window with the following details:

- Project Summary:** flight_attendant_call_system.v
- File:** rising_edge_detector.v
- Path:** C:/Users/Arman/Lab_4/Lab_4.srcts/sources_1/new/rising_edge_detector.v

The code itself is as follows:

```
54         next_state = 2'b00;
55     else
56         next_state = 2'b10;
57 end
58
59 2'b10 : begin
60     outedge = 1'b0;
61     if(~signal)
62         next_state = 2'b00;
63     else
64         next_state = 2'b10;
65 end
66
67 default : begin
68     next_state = 2'b00;
69     outedge = 1'b0;
70 end
71
72 endcase
73
74
75
76 end
77
78 always @(posedge slow_clk) begin
79     if (reset)
80         state <= 2'b00;
81     else
82         state <= next_state;
83
84 end
85
86 endmodule
87
```

The line `state <= next_state;` is highlighted with a yellow background.

Arman Khondker

EID: aak2464

Tb Rising Edge Detector

The screenshot shows a text editor window with a toolbar at the top containing icons for search, file operations, and help. The code is a Verilog testbench for a rising edge detector. It includes declarations for reg clk, signal, and reset, and a wire for outedge. A rising_edge_detector module is instantiated with its ports (.clk, .signal, .reset, .outedge). The initial block sets clk, signal, and reset to 0, then waits #100, sets signal to 1, and then waits another #100.

```
19 //////////////////////////////////////////////////////////////////
20 module tb_rising_edge_detector;
21
22
23     reg clk;
24     reg signal;
25     reg reset;
26     wire outedge;
27
28     rising_edge_detector u1 (
29         .clk(clk),
30         .signal(signal),
31         .reset(reset),
32         .outedge(outedge)
33     );
34
35
36
37
38 initial
39 begin
40
41     clk = 0;
42     signal = 0;
43     reset = 0;
44
45
46     #100;
47
48     signal = 1;
49     reset = 0;
50
51     #100;
52 
```

Arman Khondker

EID: aak2464

Tb Rising Edge Detector Continued

C:/Users/Arman/Lab_4/Lab_4.srcts/sim_1/new/tb_rising_edge_detector.v

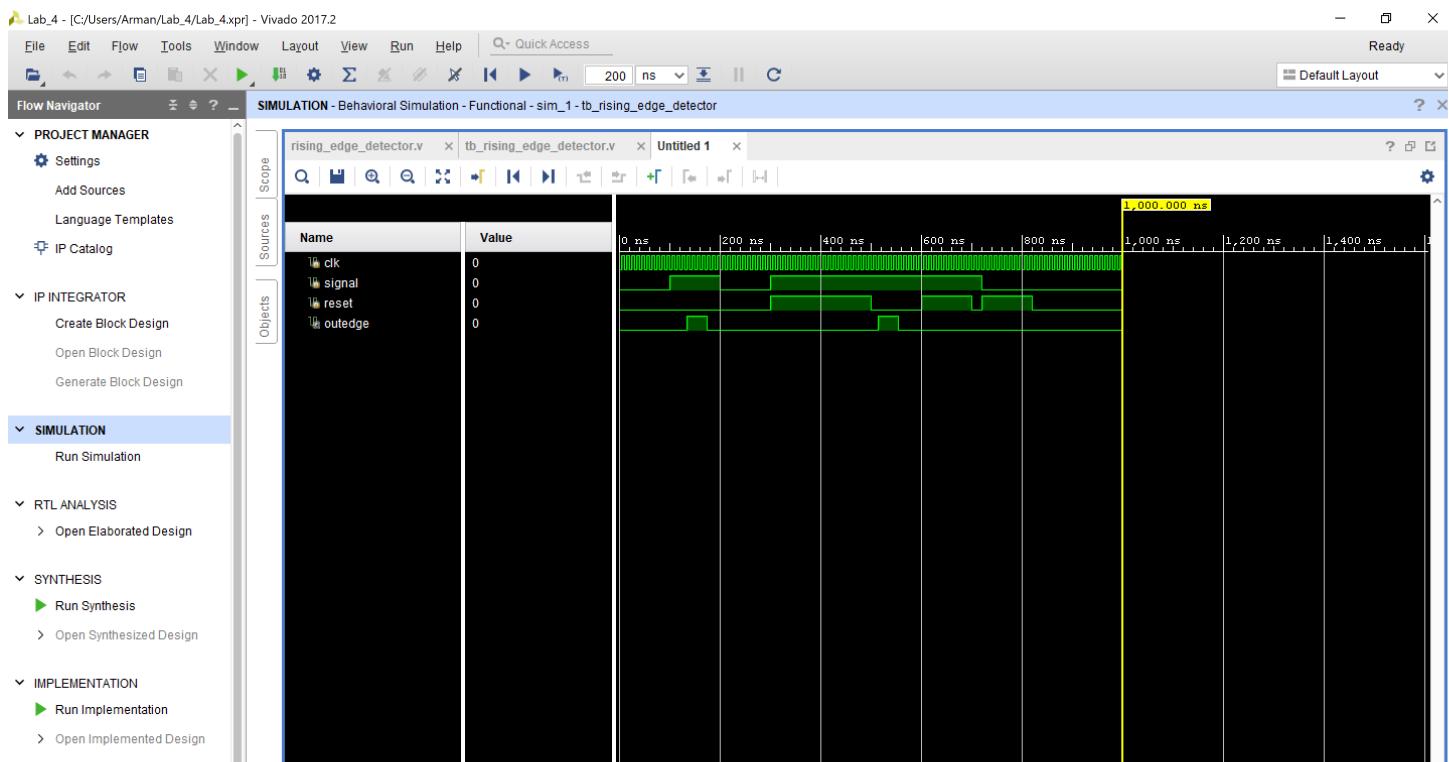
```
64      signal = 1;
65      reset = 1;
66
67      #100;
68
69      signal = 1;
70      reset = 0;
71
72      #100;
73
74      reset = 1;
75
76      #100;
77
78      reset = 0;
79
80      #20;
81
82      signal = 0;
83      reset = 1;
84
85      #100
86
87
88      signal = 0;
89      reset = 0;
90
91 end
92
93 always
94 #5 clk = ~clk;
95
96 endmodule
97
```

uns

Arman Khondker

EID: aak2464

Simulation wave form



Arman Khondker

EID: aak2464

Constraint File

The screenshot shows a software interface with a toolbar at the top and a code editor below. The code editor displays an XDC constraint file for a Basys3 board. The file contains configuration statements for various pins, including setting package pins, IOSTANDARD, and creating clocks. A yellow highlight covers the bottom portion of the code editor.

```
Project Summary  x rising_edge_detector.v  x flight_attendant_call_system.v  x tb_rising_edge_ < > ? < >
C:/Users/Arman/Lab_4/Lab_4.srscs/constrs_1/imports/Downloads/Basys3_Master_Part2.xdc  x

Q | H | ← | → | X | D | C | // | E | ? |

1 ## This file is a general .xdc for the Basys3 rev B board
2 ## To use it in a project:
3 ## - uncomment the lines corresponding to used pins
4 ## - rename the used ports (in each line, after get_ports) according to the top level signal name
5
6 ## Clock signal
7 set_property PACKAGE_PIN W5 [get_ports clk]
8     set_property IOSTANDARD LVCMS33 [get_ports clk]
9     create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
10
11 ## Switches
12 set_property PACKAGE_PIN V17 [get_ports {signal}]
13     set_property IOSTANDARD LVCMS33 [get_ports {signal}]
14
15
16 ## LEDs
17 set_property PACKAGE_PIN U16 [get_ports {outedge}]
18     set_property IOSTANDARD LVCMS33 [get_ports {outedge}]
19
20
21 ##Buttons
22 set_property PACKAGE_PIN U18 [get_ports reset]
23     set_property IOSTANDARD LVCMS33 [get_ports reset]
24 #set_property PACKAGE_PIN T18 [get_ports btnU]
25     #set_property IOSTANDARD LVCMS33 [get_ports btnU]
26 #set_property PACKAGE_PIN W19 [get_ports btnL]
27     #set_property IOSTANDARD LVCMS33 [get_ports btnL]
```

Arman Khondker

EID: aak2464

PART 3

clkdiv

The screenshot shows a code editor window with a Verilog file named "clkdiv.v". The file contains a module definition for a clock divider. The code includes comments, input and output ports, an initial block, and an always block that increments a counter on each rising edge of the input clock. The line "assign clk_out = COUNT[25];" is highlighted in yellow.

```
Project Summary × time_multiplexing_main.v × clkdiv.v × ? ⓘ ⓘ
C:/Users/Arman/Lab_4/Lab_4.srcts/sources_1/new/clkdiv.v ×
Q | H | ← | → | X | ⌂ | ⌂ | // | ⌂ | ? |
15 // 
16 // Revision: 
17 // Revision 0.01 - File Created 
18 // Additional Comments: 
19 // 
20 /////////////////////////////////////////////////////////////////// 
21 
22 
23 module clkdiv( 
24     input clk, 
25     input reset, 
26     output clk_out 
27 ); 
28 
29 ); 
30 
31 reg [25:0] COUNT; 
32 
33 initial begin 
34     COUNT = 0; 
35 end 
36 
37 assign clk_out = COUNT[25]; 
38 
39 always @(posedge clk) 
40 begin 
41     if (reset) 
42         COUNT = 0; 
43     else 
44         COUNT = COUNT + 1; 
45 end 
46 
47 endmodule 
48
```

Arman Khondker

EID: aak2464

Hexto7segment

The screenshot shows a code editor window with the following details:

- Project Summary**: A tab in the top bar.
- time_multiplexing_main.v**: An open file tab.
- clkdiv.v**: Another open file tab.
- hexto7segment.v**: The active file tab.
- C:/Users/Arman/Lab_4/Lab_4.srcts/sources_1/new/hexto7segment.v**: The full path displayed below the tabs.
- Toolbar icons**: Standard file operations like Open, Save, Find, and Print.
- Code Area**: The main content of the hexto7segment.v module:

```
19 //  
20 ///////////////////////////////////////////////////////////////////  
21  
22  
23 module hexto7segment(  
24     input [3:0] x,  
25     output reg [6:0] r  
26 );  
27  
28     always @(*)  
29     case (x)  
30         //abcdef  
31         4'b0000 : r = 7'b0000001; //0  
32         4'b0001 : r = 7'b1001111; //1  
33         4'b0010 : r = 7'b0010010; //2  
34         4'b0011 : r = 7'b0000110;//3  
35         4'b0100 : r = 7'b1001100;//4  
36         4'b0101 : r = 7'b0100100; //5  
37         4'b0110 : r = 7'b0100000;//6  
38         4'b0111 : r = 7'b0001111;//7  
39         4'b1000 : r = 7'b0000000;//8  
40         4'b1001 : r = 7'b00001100;//9  
41         4'b1010 : r = 7'b00001000;//a  
42         4'b1011 : r = 7'b1100000;//b  
43         4'b1100 : r = 7'b0110001;//c  
44         4'b1101 : r = 7'b1000010;//d  
45         4'b1110 : r = 7'b0110000;//e  
46         4'b1111 : r = 7'b0111000;//f  
47  
48  
49     endcase  
50  
51 endmodule  
52
```

Arman Khondker

EID: aak2464

Time mux state machine

Synthesis and Implementation Out-of-date [details](#) 

Default Layout 

?

X

x time_multiplexing_main.v x clkdiv.v x hexto7segment.v x time_mux_state_machine.v x

C:/Users/Arman/Lab_4/Lab_4.srcts/sources_1/new/time_mux_state_machine.v

Q | G | ← | → | X | D | F | // | E | ? |

```
15 //  
16 // Revision:  
17 // Revision 0.01 - File Created  
18 // Additional Comments:  
19 //  
20 ///////////////////////////////////////////////////////////////////  
21  
22  
23 module time_mux_state_machine(  
24     input [6:0] in0,  
25     input [6:0] in1,  
26     input [6:0] in2,  
27     input [6:0] in3,  
28     input clk,  
29     input reset,  
30     output reg [3:0] an,  
31     output reg [6:0] sseg  
32 );  
33  
34     reg [1:0] state;  
35     reg [1:0] next_state;  
36  
37 always @ (*)begin  
38     case(state)  
39         2'b00 : next_state = 2'b01;  
40         2'b01 : next_state = 2'b10;  
41         2'b10 : next_state = 2'b11;  
42         2'b11 : next_state = 2'b00;  
43  
44         default next_state = 2'b00;  
45  
46     endcase  
47 end
```

Arman Khondker

EID: aak2464

time mux state machine continued

The screenshot shows a text editor window with multiple tabs at the top: "time_multiplexing_main.v", "clkdiv.v", "hexto7segment.v", and "time_mux_state_machine.v". The "time_mux_state_machine.v" tab is active, displaying the following Verilog code:

```
48 end
49
50 always @(*) begin
51     case (state)
52
53         2'b00 : sseg = in0;
54         2'b01 : sseg = in1;
55         2'b10 : sseg = in2;
56         2'b11 : sseg = in3;
57
58         default : sseg = in0;
59     endcase
60
61     case (state)
62         2'b00 : an= 4'b1110;
63         2'b01 : an= 4'b1101;
64         2'b10 : an= 4'b1011;
65         2'b11 : an= 4'b0111;
66
67         default : an = 4'b1110;
68
69     endcase
70 end
71
72 always @(posedge clk or posedge reset) begin
73     if (reset)
74         state <= 2'b00;
75     else
76         state <= next_state;
77
78 end
79
80 endmodule
81
```

Arman Khondker

EID: aak2464

Time mux main

```
C:/Users/Arman/Lab_4/Lab_4.srcts/sources_1/new/time_multiplexing_main.v

22
23 module time_multiplexing_main(
24     input clk,
25     input reset,
26     input [15:0] sw,
27     output [3:0] an,
28     output [6:0] sseg
29 );
30
31     wire [6:0] in0, in1, in2, in3;
32     wire slow_clk;
33
34     // module instantiation of hexto7segment decoder
35     hexto7segment c1 (.x(sw[3:0]), .r(in0));
36     hexto7segment c2 (.x(sw[7:4]), .r(in1));
37     hexto7segment c3 (.x(sw[11:8]), .r(in2));
38     hexto7segment c4 (.x(sw[15:12]), .r(in3));
39
40     //module instantiation of the clock divider
41     clkdiv c5 (.clk(clk), .reset(reset), .clk_out(slow_clk));
42
43     //module instantiation of the multiplexer
44     time_mux_state_machine c6 (
45         .clk (slow_clk),
46         .reset (reset),
47         .in0 (in0),
48         .in1 (in1),
49         .in2 (in2),
50         .in3 (in3),
51         .an (an),
52         .sseg (sseg));
53 endmodule
```

Arman Khondker

tb mux main

EID: aak2464

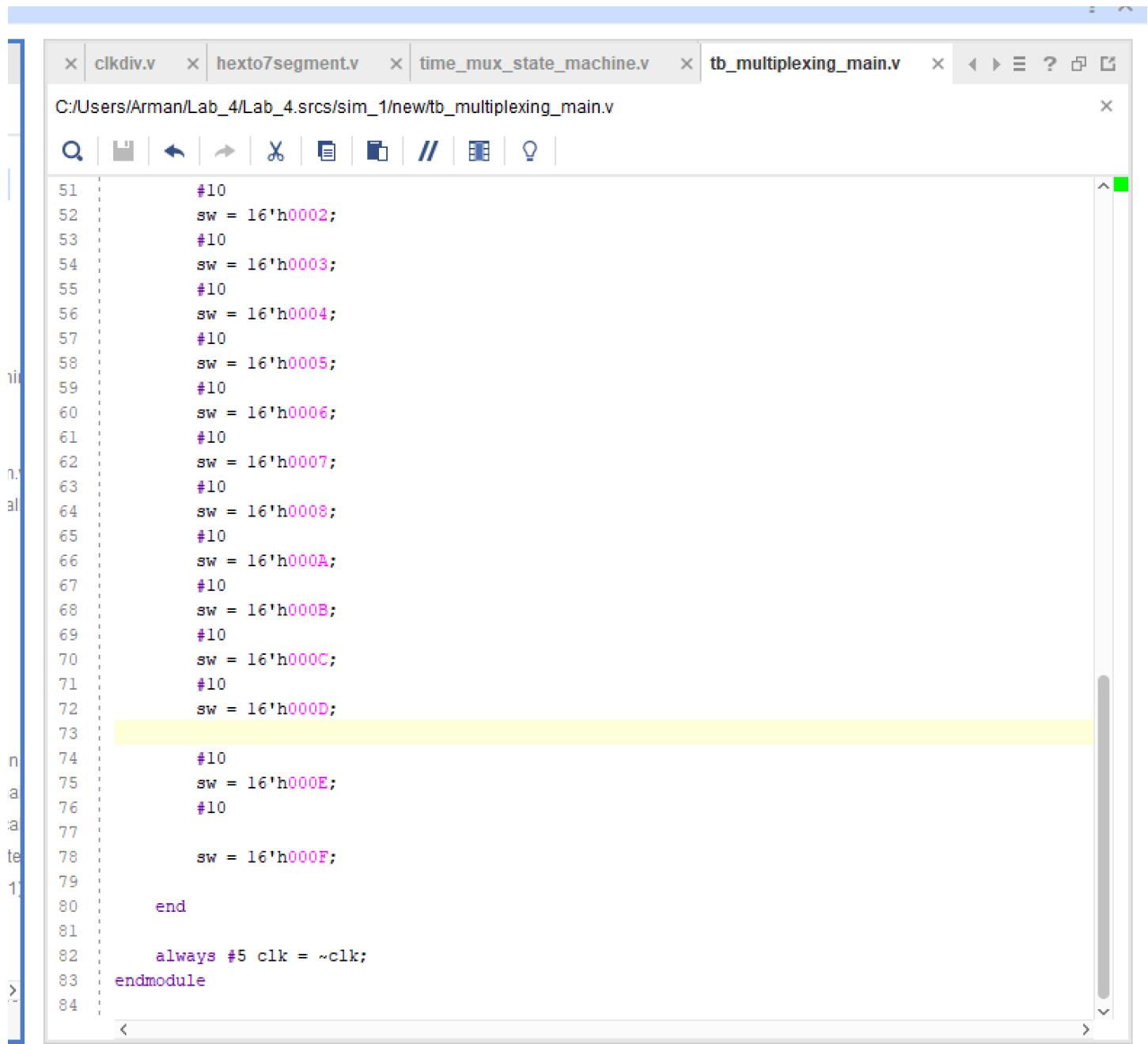
The screenshot shows a text editor window with multiple tabs at the top. The active tab is 'tb_multiplexing_main.v'. Below the tabs, the file path 'C:/Users/Arman/Lab_4/Lab_4.srcts/sim_1/new/tb_multiplexing_main.v' is displayed. The main area contains Verilog testbench code. The code includes comments for revision history and additional comments. It defines a module 'tb_time_mux_state_machine' with various ports and parameters. Inside the module, it instantiates a 'time_multiplexing_main' unit and provides initial values for its inputs: clk = 0, reset = 0, and sw = 16'h0000.

```
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////////////////////
21
22
23 module tb_time_mux_state_machine;
24     reg clk;
25     reg reset;
26     reg [15:0] sw;
27     wire [3:0] an;
28     wire [6:0] sseg;
29
30
31     time_multiplexing_main u1 (
32         .clk(clk),
33         .reset(reset),
34         .sw(sw),
35         .an(an),
36         .sseg(sseg)
37     );
38
39
40     initial
41     begin
42         clk = 0;
43         reset = 0;
44         sw = 0;
45
46         #10
47         reset = 1;
48         sw = 16'h0000;
49         #10
50
51 endmodule
```

Arman Khondker

EID: aak2464

tb mux main continued



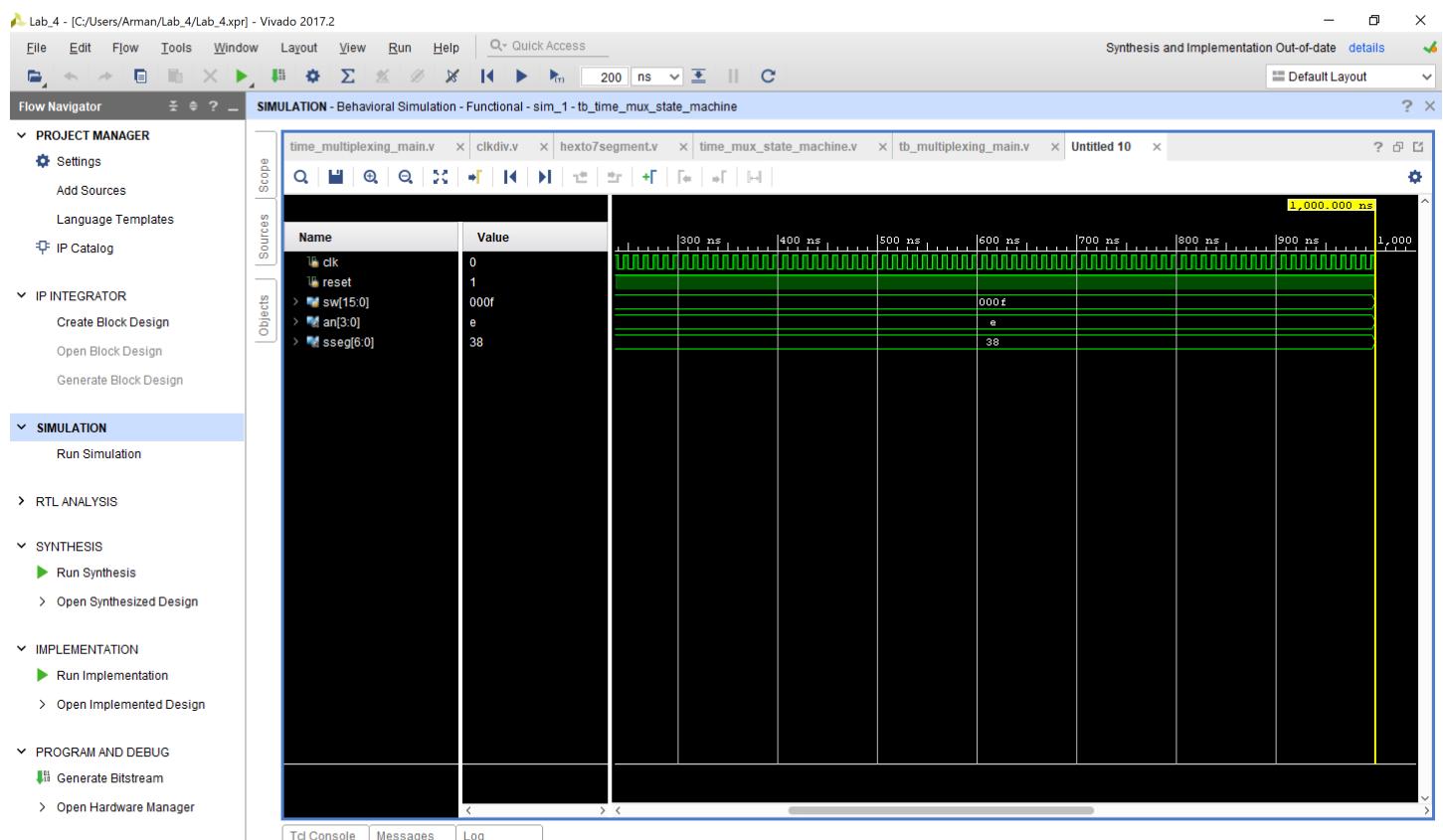
```
clkdiv.v  hexto7segment.v  time_mux_state_machine.v  tb_multiplexing_main.v
C:/Users/Arman/Lab_4/Lab_4.srscs/sim_1/new/tb_multiplexing_main.v

51      #10
52      sw = 16'h0002;
53      #10
54      sw = 16'h0003;
55      #10
56      sw = 16'h0004;
57      #10
58      sw = 16'h0005;
59      #10
60      sw = 16'h0006;
61      #10
62      sw = 16'h0007;
63      #10
64      sw = 16'h0008;
65      #10
66      sw = 16'h000A;
67      #10
68      sw = 16'h000B;
69      #10
70      sw = 16'h000C;
71      #10
72      sw = 16'h000D;
73
74      #10
75      sw = 16'h000E;
76      #10
77
78      sw = 16'h000F;
79
80      end
81
82      always #5 clk = ~clk;
83 endmodule
84 }
```

Arman Khondker

EID: aak2464

Simulation Waveform



Constraint

Arman Khondker
EID: aak2464

The screenshot shows a software interface with multiple tabs at the top: "time_mux_state_machine.v", "tb_multiplexing_main.v", "Untitled 10", "Basys3_Master_Part3.xdc", and several others. The main window displays an XDC (Hardware Constraint Description) file named "Basys3_Master_Part3.xdc". The code in the file is as follows:

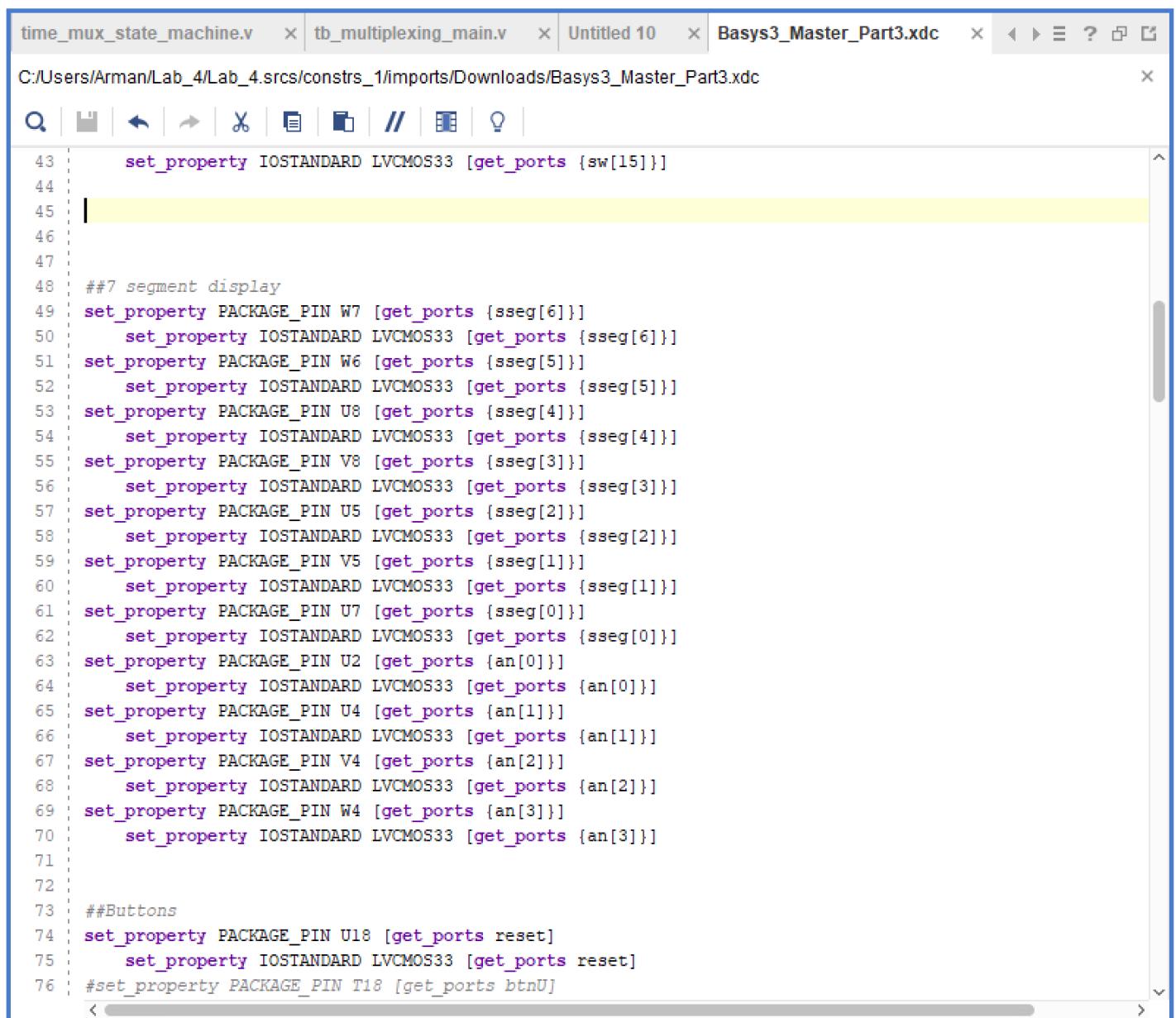
```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the Verilog files
##
## Clock signal
set_property PACKAGE_PIN W5 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

## Switches
set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]
set_property PACKAGE_PIN W16 [get_ports {sw[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[2]}]
set_property PACKAGE_PIN W17 [get_ports {sw[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[3]}]
set_property PACKAGE_PIN W15 [get_ports {sw[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[4]}]
set_property PACKAGE_PIN V15 [get_ports {sw[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[5]}]
set_property PACKAGE_PIN W14 [get_ports {sw[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[6]}]
set_property PACKAGE_PIN W13 [get_ports {sw[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[7]}]
set_property PACKAGE_PIN V2 [get_ports {sw[8]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[8]}]
set_property PACKAGE_PIN T3 [get_ports {sw[9]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[9]}]
set_property PACKAGE_PIN T2 [get_ports {sw[10]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[10]}]
set_property PACKAGE_PIN R3 [get_ports {sw[11]}]
```

Arman Khondker

EID: aak2464

Constraint continued



The screenshot shows a software interface with a code editor window. The title bar of the window displays four tabs: "time_mux_state_machine.v", "tb_multiplexing_main.v", "Untitled 10", and "Basys3_Master_Part3.xdc". Below the title bar, the path "C:/Users/Arman/Lab_4/Lab_4.srcts/constrs_1/imports/Downloads/Basys3_Master_Part3.xdc" is shown. The code editor contains XDC constraint code for a Basys3 Master Part 3 device. The code defines properties for various pins, including segments (sseg) and buttons (reset, btnU). The code is as follows:

```
43 set_property IOSTANDARD LVCMOS33 [get_ports {sw[15]}]
44
45
46
47
48 ##7 segment display
49 set_property PACKAGE_PIN W7 [get_ports {sseg[6]}]
50     set_property IOSTANDARD LVCMOS33 [get_ports {sseg[6]}]
51 set_property PACKAGE_PIN W6 [get_ports {sseg[5]}]
52     set_property IOSTANDARD LVCMOS33 [get_ports {sseg[5]}]
53 set_property PACKAGE_PIN U8 [get_ports {sseg[4]}]
54     set_property IOSTANDARD LVCMOS33 [get_ports {sseg[4]}]
55 set_property PACKAGE_PIN V8 [get_ports {sseg[3]}]
56     set_property IOSTANDARD LVCMOS33 [get_ports {sseg[3]}]
57 set_property PACKAGE_PIN U5 [get_ports {sseg[2]}]
58     set_property IOSTANDARD LVCMOS33 [get_ports {sseg[2]}]
59 set_property PACKAGE_PIN V5 [get_ports {sseg[1]}]
60     set_property IOSTANDARD LVCMOS33 [get_ports {sseg[1]}]
61 set_property PACKAGE_PIN U7 [get_ports {sseg[0]}]
62     set_property IOSTANDARD LVCMOS33 [get_ports {sseg[0]}]
63 set_property PACKAGE_PIN U2 [get_ports {an[0]}]
64     set_property IOSTANDARD LVCMOS33 [get_ports {an[0]}]
65 set_property PACKAGE_PIN U4 [get_ports {an[1]}]
66     set_property IOSTANDARD LVCMOS33 [get_ports {an[1]}]
67 set_property PACKAGE_PIN V4 [get_ports {an[2]}]
68     set_property IOSTANDARD LVCMOS33 [get_ports {an[2]}]
69 set_property PACKAGE_PIN W4 [get_ports {an[3]}]
70     set_property IOSTANDARD LVCMOS33 [get_ports {an[3]}]
71
72
73 ##Buttons
74 set_property PACKAGE_PIN U18 [get_ports reset]
75     set_property IOSTANDARD LVCMOS33 [get_ports reset]
76 #set_property PACKAGE_PIN T18 [get_ports btnU]
```