UNIVERSITY OF BIRMINGHAM

Department of Electronic, Electrical and Computer Engineering

VHDL

Assignment

Name: Konstantinos Tsimpoukas

I.D.: 1130191

Course: MSc Embedded Systems

CONTENTS

1 INTRODUCTION & OVERVIEW

The new user of VHDL can easily understand the reasons why this hardware description language is so spread. Designing in VHDL, we can have a design of 10 or 20 thousand gates quickly and easily. Also, in comparison with the schematics (OrCad) and the Boolean equations, with VHDL we can reach the register-transfer level very quickly.

As Kevin Skahill (VHDL for Programmable Logic,1996, p.4) describes, VHDL provides the following capabilities:

1. Power and Flexibility
2. Device-Independent Design
3. Portability
4. Benchmarking Capabilities
5. ASIC Migration
6. Quick time-to-market and Low Cost

Furthermore, Skahill (1996, p9) explains the six stages of the design process:

1. Define the design requirements
2. Describe the design in VHDL (formulate and code the design)
3. Simulate the source code
4. Synthesize, optimise, and fit (place and route) the design
5. Simulate the post-layout (fit) design model
6. Program the device

2 DESIGN

2.1 STAGE 1

In the first stage, we had to design a synchronous finite state machine. Figure 1 shows the state diagram that I had to implement (no.18).

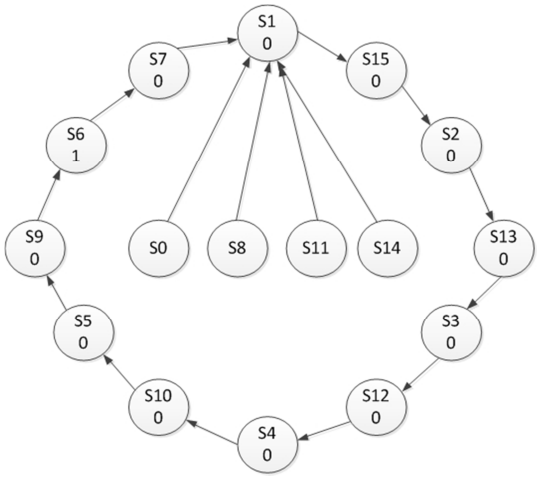

Figure 1 State Diagram (no.18)

Firstly, I declared 16 constants (4 bits each) which hold the binary code for each state (S0, S1…S15) and two signals, state and next_state, to store the current and the next stage. Also, we wanted to display the current state in the LEDs (first 4 LEDs), so we had to slow down the rate that the states are changing. For this reason, we introduced the signal signal_clk, which is the slow clock, and the signal count (32 bits), with which we count up until we reach the constant MAX_VALUE (X"01FFFFFF"). Finally, the component display used to display the states in the first digit of the 7 segment display.

The program is consisted of three processes. The first process is serving three purposes. First of all, it implements the manual reset (forcing to unallocated states). Secondly, it rolls the states and thirdly checks if we are in the state (S6) in which output is 1. The algorithmic logic for the first process is presented below.

```
pr1: process (slow clock and switches)
                begin
                        if (switch 7 is on ) then
                                signal state gets value from the 4 first switches
                        elsif ( in every rising edge of the slow clock ) then
                                give state_next to state
                                        if(we are in state S6) then
                                                leds(7) on
                                        else
                                                leds(7) off
                                        end if;
                        end if;
                end process;
```

The second process checks the current state and moves to the next. The algorithmic logic is shown below.

```
pr2:    process
                begin
                in every rising edge of the slow clock
                if (state=S1) then
                        state_next<=S15;
                elsif (state=S15) then
                        state_next<=S2;
                elsif (state=S2) then
                        state_next<=S13;
```

```
elsif (state=S13) then
       state_next<=S3;
elsif (state=S3) then
       state_next<=S12;
elsif (state=S12) then
       state_next<=S4;
elsif (state=S4) then
       state_next<=S10;
elsif (state=S10) then
       state_next<=S5;
elsif (state=S5) then
       state_next<=S9;
elsif (state=S9) then
       state_next<=S6;
elsif (state=S6) then
       state_next<=S7;
elsif (state=S7) then
       state_next<=S1;
else
       state_next<=S1;  --we use S1 as the Idle State
end if;
end process;
```

The third and final process, produces the slow clock (signal_clk). The process is driven from the fast clock. In every rising edge the signal count is increasing. When count reaches the MAX_VALUE (corresponds to the delay) the slow clock (signal_clk) becomes 1 and the count is zeroed. In any other case, the slow clock is zero.

```
pr3: process(fast clock)
       begin
       if (in every rising edge of the fast clock) then
       increase signal count
              if (signal count equals MAX_VALUE) then
                     slow clock becomes 1
```

```
                        and count is zeroed

                else

                        slow clock equals 0

                end if;

        end if;

        end process;
```

2.2 STAGE 2

In this stage, we had to implement an adjustable delay. From the switches, we get an 8 bit number and we multiply this with a quarter of a second delay. However, the new clock rate (or delay) has to be adjusted to the circuit only when a button is pressed.

For this implementation, we had to introduce signal n to store the value that we are going to input from the switches, constant DELAY to hold the value in hex which corresponds to the quarter of a second delay and finally, signal TOTAL_DELAY to store the desired multiplication (TOTAL_DELAY(40 bits) =n(8 bits)*DELAY (32 bits)).

We have again three processes. The second process is exactly the same. So, we will explain here the modifications that have been made to process 1 and 3.

In process 1, functionality added for button 2. The rest of the code is the same. When button 2 is pushed, signal n stores the value that has been set to the switches (8 bit value).

Although, the logic in process 3 is exactly the same with the one in stage one, we had to add another if statement to check if TOTAL_DELAY is bigger than the count signal. Without this if statement, if TOTAL_DELAY was bigger than count, the count was counting up until it reaches TOTAL_DELAY again.

Finally, we have to notice that when we multiply with zero the FSM is stopped. When we multiply with one, we have a quarter of a second delay and so on.

2.3 STAGE 3

In this stage, we had to write a program which will implement the state sequence of any person's number (35 numbers). We based again in stage 1. Also, we display the current state not only to the LEDs, but also to the first digit of the 7 segments display. Again, we use LED(7) for the output.

We use the switches for two reasons. The first is to force the FSM into the unallocated states and the other is to take the person's ID number. Therefore, we use button 0 and 1.

The logic for the first process is presented below.

```
pr1: process (signal_clk,buttons,state_next)

                begin
```

```
        if (rising edge of the clock) then


                if(button 0 is pushed) then

                        force state from the switches

                elsif (button 1 is pushed) then

                        take person's ID number from the switches

                else

                        give state_next to state

                end if;

        end if;

end process;
```

The second process which generates the slow clock is the same. We produce a delay just to notice the changes from state to state.

The code which sets the output is now moved into the last process which returns the corresponding state sequence for a specific ID number (choice signal in our code).

```
choice_process: process(signal_clk,choice)

        begin

        if (rising edge of the slow clock) then


                if(check ID number)then

                        set on LED(7) in the correct state, for the appropriate ID

                else

                        reset LED(7)

                end if;


                if(check ID number)then

                        give the corresponding sequence

                elsif(check ID nuber)then

                        ..

                ...

                else  --number = 35

                        give the corresponding sequence
```

```
                end if;

        end if;

end process;
```

2.4 STAGE 4

Stage 4 is based in stage 3. The only difference is that we turned signal choice to generic. By initializing the generic choice in the beginning, inside the entity, we accomplish to synthesise only the sequence for this specified person.

3 SIMULATION

3.1 STAGE 1

Figure 2 shows the way that states are changing. Signal state and next_state are changing every two clock periods. However, signal state holds the current state. That's the reason that the four first bits of the leds are following this signal.

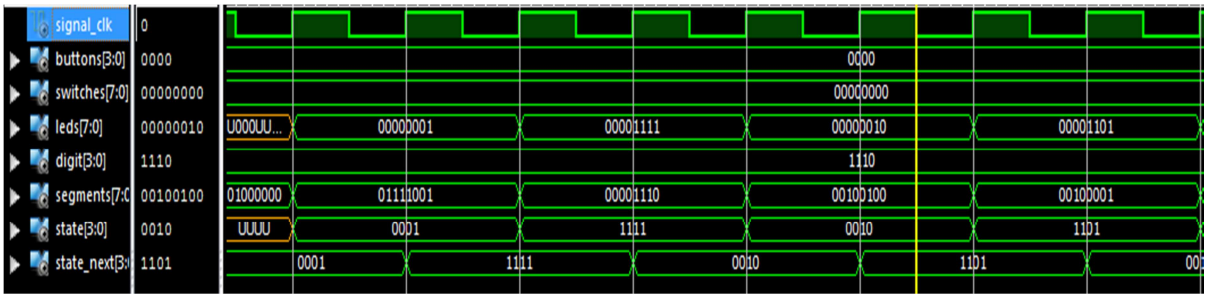

Figure 2 Changing states

Figure 3 shows that when the current state is S6, the output (LED(7)) is on.

Figure 3 Output in S6

Figure 4 illustrates the way that we can force states in our FSM. When switch 7 is on, we can force a state using the first four switches. Here, we force state S0 (0000). S0 is an unallocated state for my FSM. So, when switch 7 is off, current state moves to S1 and then continues through the correct sequence.

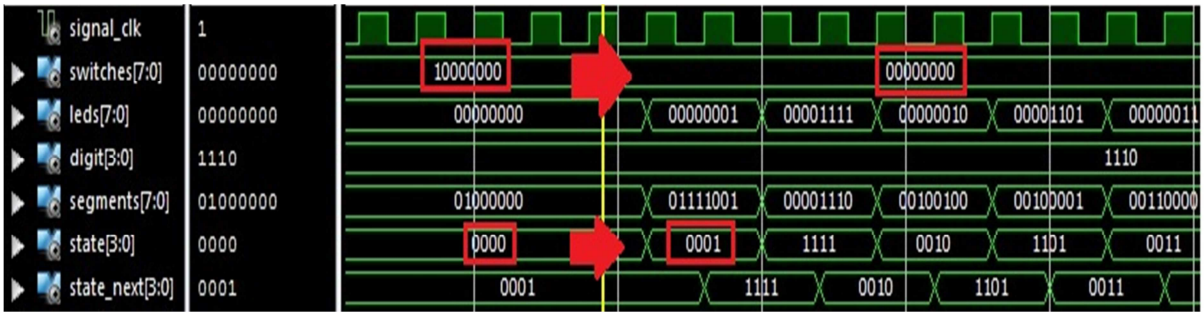

Figure 4 Forcing unallocated state (S0)

Figure 5 shows how we can force our FSM in a normal state (like the S10). When switch 7 is off, state goes to the next state S5, as it was expected.

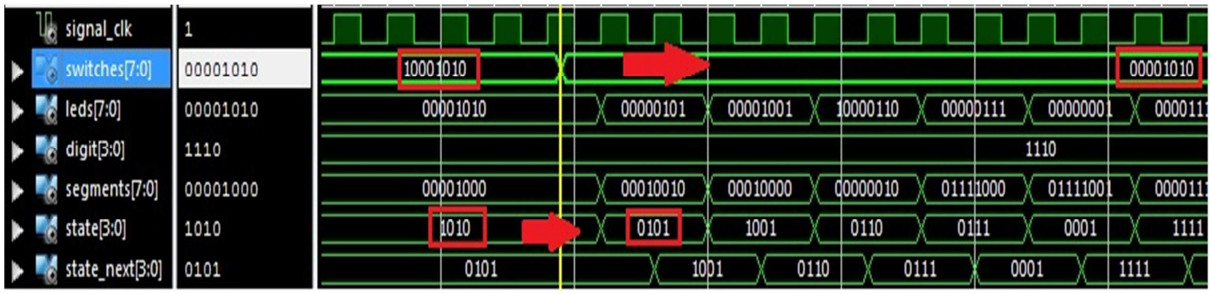

Figure 5 Forcing Normal State (S10)

3.2 STAGE 2

Figure 6 shows the basic functionality of stage 2. The slow clock is generated according to the TOTAL_DELAY. The signal n (8 bit binary number) is initialised 1, so the states are displayed with a delay of a quarter of the second (this delay corresponds to BEBC20 in HEX).

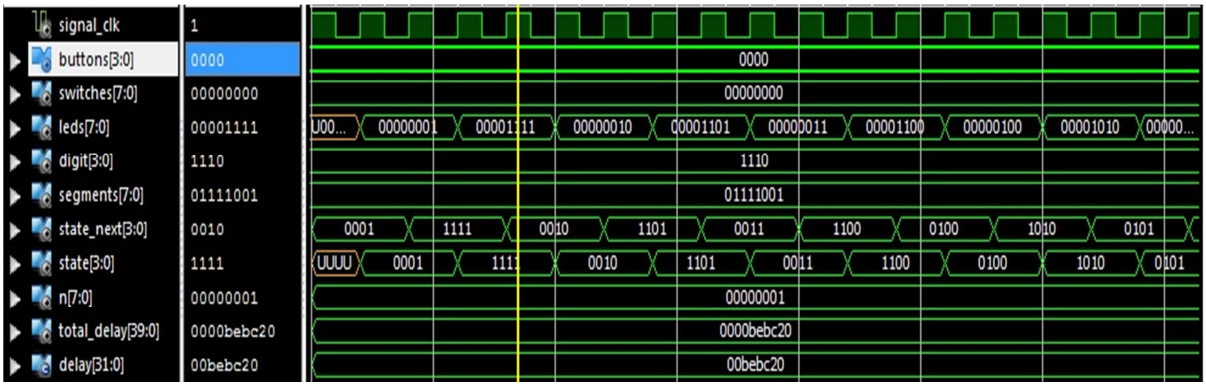

Figure 6 Quarter of a second delay

To take a new n from the switches, we have to push button 1. As we can see in Figure 7, first we change the positions of the switches to the desired n (n=3), we push button 1 and then n changes to 3. Finally, we notice that TOTAL_DELAY takes its new value (00000011 (3 Dec) x BEBC20 (hex)=23C3460 (hex)).

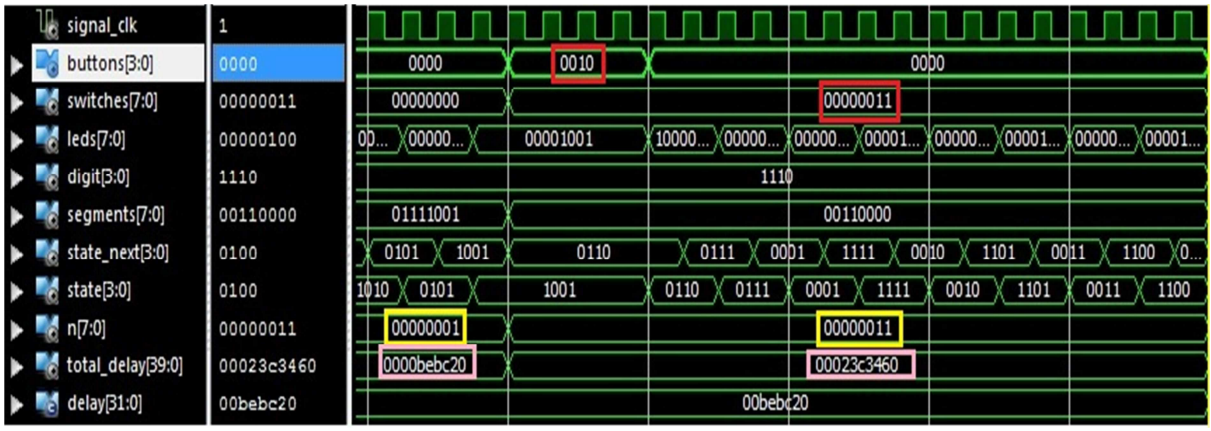

Figure 7 Input new n

Also, in this stage, we have the capability to force states using button 0. Here we force S0.

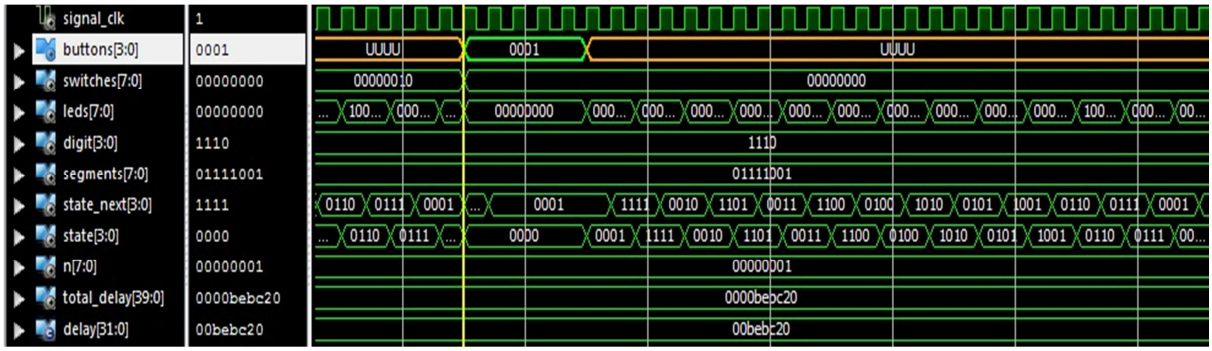

Figure 8 Forcing unallocated state (S0) from button 0

When we multiply with zero, the state is stuck. Here is stuck in S9 (1001). Again, we push button1 to take the input (n) from the switches. As a result, TOTAL_DELAY is zeroed.

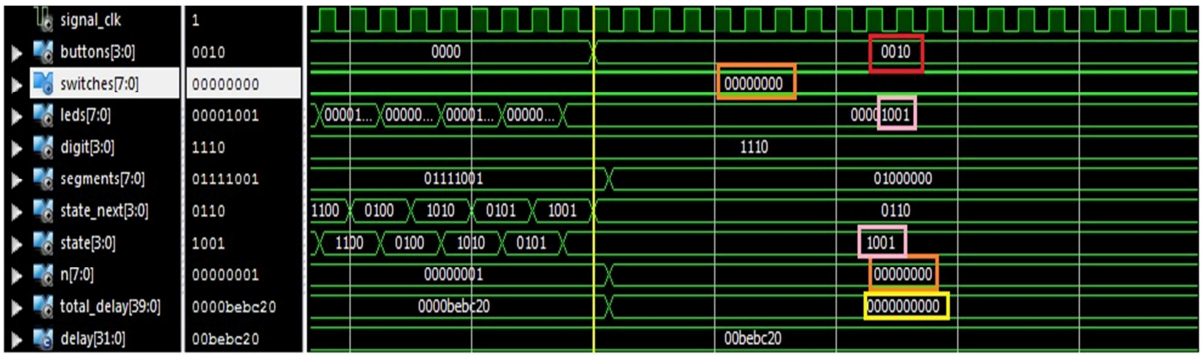

Figure 9 Multiply with zero

3.3 STAGE 3

Figure 10 illustates how we can choose another person's ID number (1 to 35) and display his sequence. By default, the program runs the sequence for no.18. So, if we want to display the sequence for no.1, we have to set number 1 in the switches and push button 1. Now the choice changed. After that, we set the switches off and push button 0 (force to unallocated state S0 and start from the first state of the specific sequence (S2)).

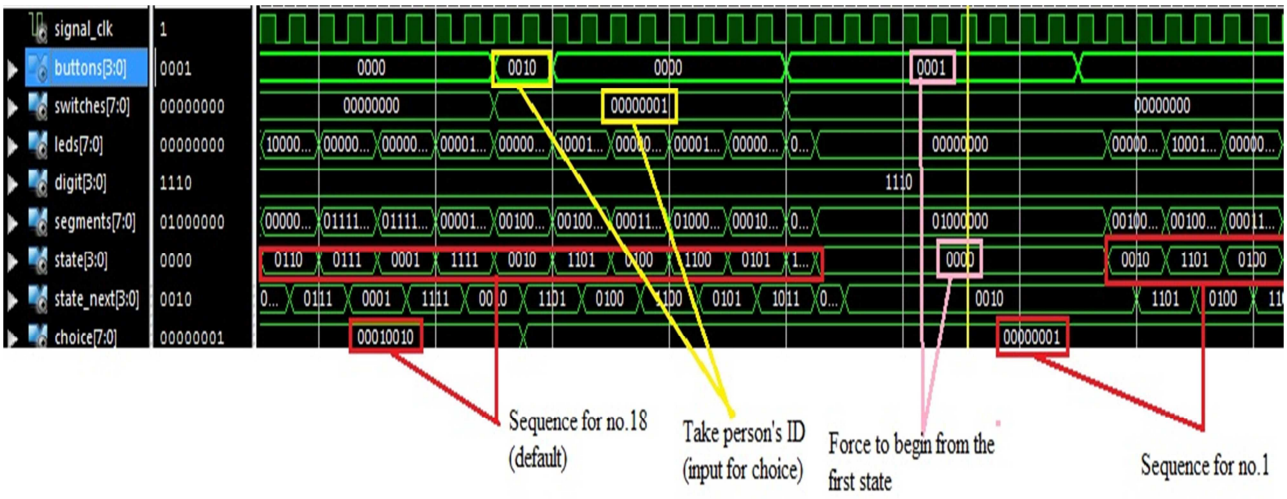

Figure 10 Changing Person's ID (choice)

3.4 STAGE 4

In stage 4 we synthesise the solution for only one person. Therefore, the simulations are the same as in stage 1. The generic parameter will determine which sequence to display.

4 SYNTHESIS

4.1 STAGE 1

In this stage, we notice the lower percentages in the Utilization column. We use only a small part of our FPGA, so an implementation in a smaller device would be fitting.

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	42	3,840	1%	
Number of 4 input LUTs	29	3,840	1%	
Logic Distribution				
Number of occupied Slices	36	1,920	1%	
Number of Slices containing only related logic	36	36	100%	
Number of Slices containing unrelated logic	0	36	0%	
Total Number of 4 input LUTs	60	3,840	1%	
Number used as logic	29			
Number used as a route-thru	31			
Number of bonded IOBs				
Number of bonded	33	173	19%	
Number of BUFGMUXs	1	8	12%	

Figure 11 Device Utilization Summary (stage 1)

4.2 STAGE 2

The percentages in this stage are significantly increased. The multiplier (8 x 32 bits= 40 bits) consumes the most of the hardware.

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Total Number Slice Registers	50	3,840	1%	
Number used as Flip Flops	42			
Number used as Latches	8			
Number of 4 input LUTs	133	3,840	3%	
Logic Distribution				
Number of occupied Slices	96	1,920	5%	
Number of Slices containing only related logic	96	96	100%	
Number of Slices containing unrelated logic	0	96	0%	
Total Number of 4 input LUTs	172	3,840	4%	
Number used as logic	133			
Number used as a route-thru	39			
Number of bonded IOBs				
Number of bonded	33	173	19%	
Number of MULT18X18s	2	12	16%	
Number of BUFGMUXs	1	8	12%	

Figure 12 Device Utilization Summary (stage 2)

4.3 STAGE 3

This stage is the most demanding of the four (from the perspective of the hardware). But even this stage could be implemented in a smaller device.

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	70	3,840	1%	
Number of 4 input LUTs	384	3,840	10%	
Logic Distribution				
Number of occupied Slices	235	1,920	12%	
Number of Slices containing only related logic	235	235	100%	
Number of Slices containing unrelated logic	0	235	0%	
Total Number of 4 input LUTs	415	3,840	10%	
Number used as logic	384			
Number used as a route-thru	31			
Number of bonded IOBs				
Number of bonded	33	173	19%	
Number of BUFGMUXs	2	8	25%	

Figure 13 Device Utilization Summary (stage 3)

4.4 STAGE 4

As it was expected, the percentages in this stage are exactly the same with these in stage 1.

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	42	3,840	1%	
Number of 4 input LUTs	26	3,840	1%	
Logic Distribution				
Number of occupied Slices	31	1,920	1%	
Number of Slices containing only related logic	31	31	100%	
Number of Slices containing unrelated logic	0	31	0%	
Total Number of 4 input LUTs	57	3,840	1%	
Number used as logic	26			
Number used as a route-thru	31			
Number of bonded IOBs				
Number of bonded	33	173	19%	
Number of BUFGMUXs	1	8	12%	

Figure 14 Device Utilization Summary (stage 4)

4.5 COMPARING RTL

Comparing Figure 15 and 16, we can notice that the RTL schematic for stage 3 is by far larger than the RTL schematic in stage 4. Also, stage 1 and stage 4 have the same RTL schematic.

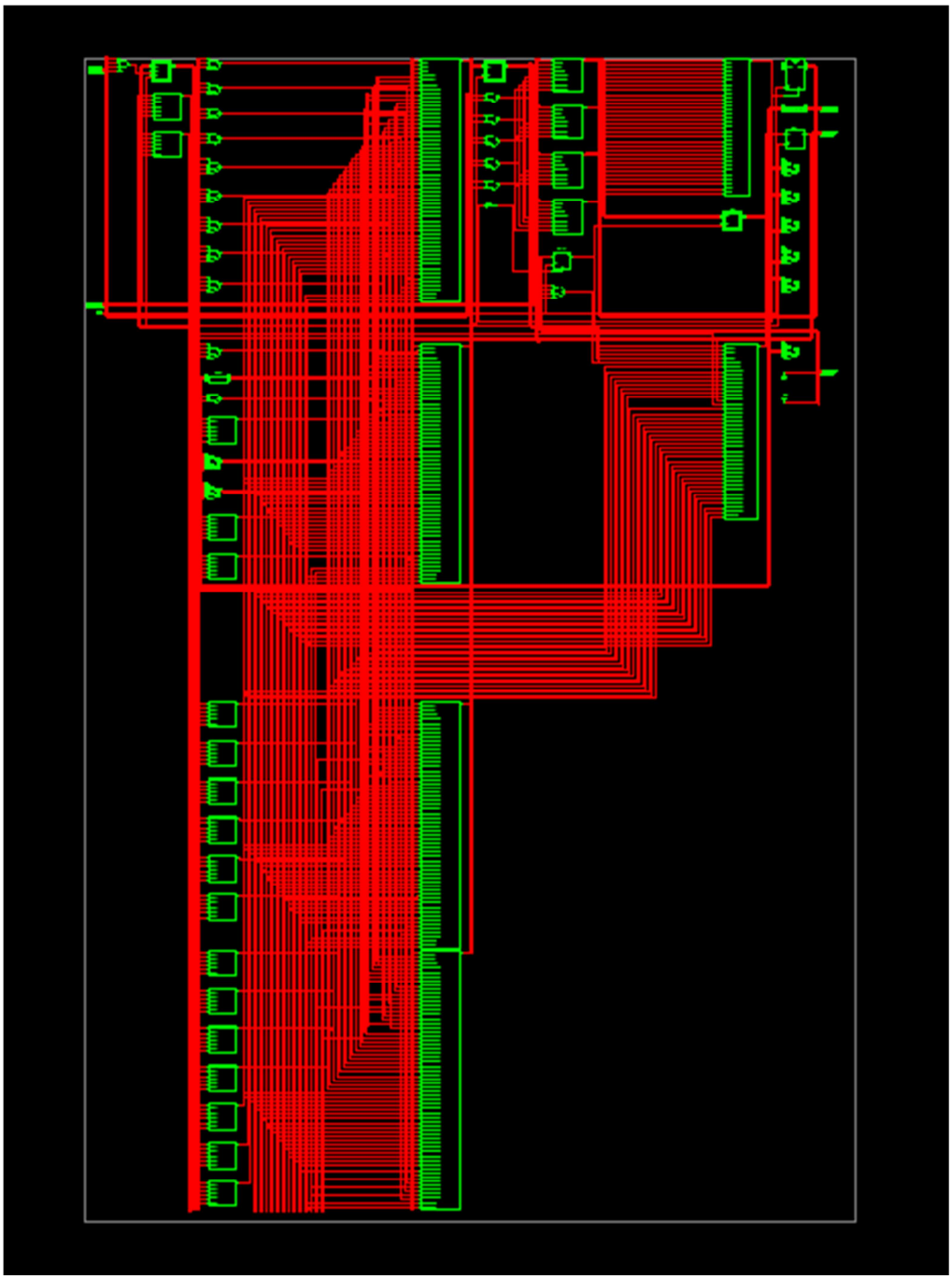

Figure 15 RTL for stage 3

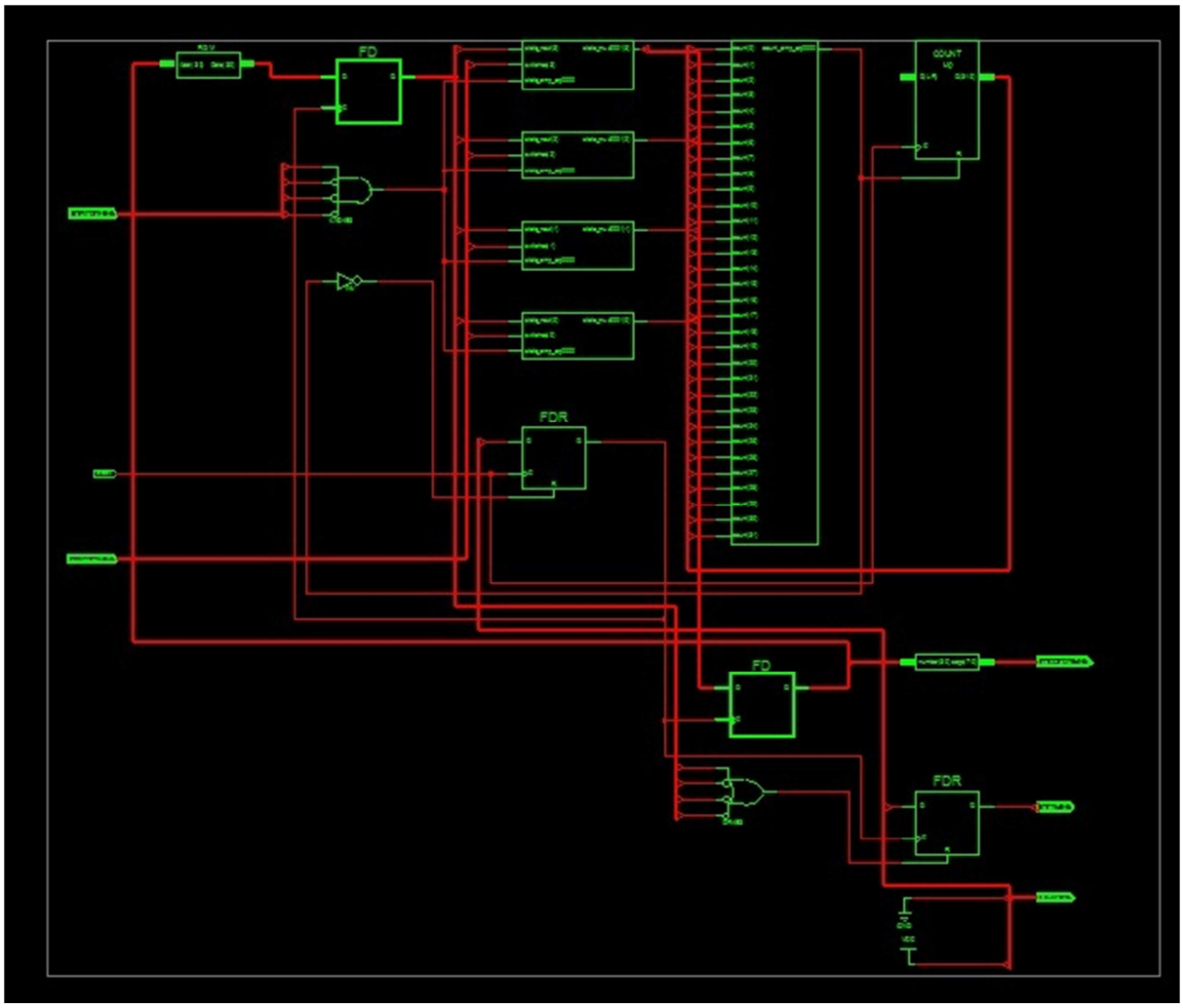

Figure 16 RTL for stage 4 (very similar with stage 1)

5 CONCLUSIONS

The design and the implementation were the two important stages in this assignment. During the design procedure, we fitted the given specifications to our code. But, synthesising and implementing the code in the development kit was the real experience. There, we could see the real behaviour of our code. This is very important because when you program in VHDL you have to be sure about the correctness of the code's logic.

6 APPENDIX I

6.1 STAGE 1

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer: Konstantinos Tsimpoukas
--
-- Create Date:    18:23:11 11/21/2010
-- Design Name:              assignment stage 1
-- Module Name:    assign - Behavioral
-- Project Name:
-- Target Devices: Spartan 3
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;


entity assign is
    Port ( clock : in  STD_LOGIC;
           buttons : in  STD_LOGIC_VECTOR (3 downto 0);
           switches : in  STD_LOGIC_VECTOR (7 downto 0);    --switches(7)=reset
          leds : out  STD_LOGIC_VECTOR (7 downto 0);
           digit : out  STD_LOGIC_VECTOR (3 downto 0);
           segments : out  STD_LOGIC_VECTOR (7 downto 0));
```

```
                        -- Assign inputs and outputs to appropriate pins on FPGA
          attribute LOC : string ;
          attribute LOC of clock : signal is "T9";
          attribute LOC of switches : signal is "K13,K14,J13,J14,H13,H14,G12,F12";
          attribute LOC of buttons : signal is "L14,L13,M14,M13";
          attribute LOC of leds : signal is "P11,P12,N12,P13,N14,L12,P14,K12";
          attribute LOC of digit : signal is "E13,F14,G14,D14";
          attribute LOC of segments : signal is "P16,N16,F13,R16,P15,N15,G13,E14";

end assign;

architecture Behavioral of assign is

        component display
    Port ( segs : out  STD_LOGIC_VECTOR (7 downto 0);
           number : in  STD_LOGIC_VECTOR (3 downto 0));
        end component;

        signal state,state_next : STD_LOGIC_VECTOR(3 downto 0);
        signal signal_clk:std_logic;

        signal count:STD_LOGIC_VECTOR(31 downto 0);
        constant MAX_VALUE:STD_LOGIC_VECTOR(31 downto 0):=X"01FFFFFF";

        constant S0 : STD_LOGIC_VECTOR(3 downto 0) := "0000";
        constant S1 : STD_LOGIC_VECTOR(3 downto 0) := "0001";
        constant S2 : STD_LOGIC_VECTOR(3 downto 0) := "0010";
        constant S3 : STD_LOGIC_VECTOR(3 downto 0) := "0011";
        constant S4 : STD_LOGIC_VECTOR(3 downto 0) := "0100";
        constant S5 : STD_LOGIC_VECTOR(3 downto 0) := "0101";
        constant S6 : STD_LOGIC_VECTOR(3 downto 0) := "0110";
        constant S7 : STD_LOGIC_VECTOR(3 downto 0) := "0111";
        constant S8 : STD_LOGIC_VECTOR(3 downto 0) := "1000";
```

```
        constant S9 : STD_LOGIC_VECTOR(3 downto 0) := "1001";
        constant S10 : STD_LOGIC_VECTOR(3 downto 0) := "1010";
        constant S11 : STD_LOGIC_VECTOR(3 downto 0) := "1011";
        constant S12 : STD_LOGIC_VECTOR(3 downto 0) := "1100";
        constant S13 : STD_LOGIC_VECTOR(3 downto 0) := "1101";
        constant S14 : STD_LOGIC_VECTOR(3 downto 0) := "1110";
        constant S15 : STD_LOGIC_VECTOR(3 downto 0) := "1111";

begin


leds(3 downto 0)<=state;

--Initializing Unused Outputs
leds(6 downto 4)<="000";
digit(3 downto 0)<="1110";
--segments(7 downto 0)<=X"FF";
dd : display port map(segs => segments, number => state);

pr1: process (signal_clk,switches)
               begin

                      if(switches(7)='1') then            --switches(7)=reset
                             state<=switches(3 downto 0);     --S1 as the idle state
                      elsif (rising_edge(signal_clk)) then
                             state<=state_next;
                                    if(state_next=S6) then
                                           leds(7)<='1';
                                    else
                                           leds(7)<='0';
                                    end if;
                      end if;
               end process;
```

```
pr2:   process
               begin
               wait until signal_clk'event and signal_clk='1';

               if (state=S1) then
                      state_next<=S15;
               elsif (state=S15) then
                      state_next<=S2;
               elsif (state=S2) then
                      state_next<=S13;
               elsif (state=S13) then
                      state_next<=S3;
               elsif (state=S3) then
                      state_next<=S12;
               elsif (state=S12) then
                      state_next<=S4;
               elsif (state=S4) then
                      state_next<=S10;
               elsif (state=S10) then
                      state_next<=S5;
               elsif (state=S5) then
                      state_next<=S9;
               elsif (state=S9) then
                      state_next<=S6;
               elsif (state=S6) then
                      state_next<=S7;
               elsif (state=S7) then
                      state_next<=S1;
               else
                      state_next<=S1;  --S1 idle State

               end if;
               end process;
```

```
pr3: process(clock)
                begin
                if rising_edge(clock) then
                count<=count+1;
                        if count=MAX_VALUE then
                                signal_clk<='1';
                                count<=(others=>'0');
                        else
                                signal_clk<='0';
                        end if;
                end if;
                end process;

end Behavioral;
```

6.2 STAGE 2

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer: Konstantinos Tsimpoukas
--
-- Create Date:    18:23:11 11/21/2010
-- Design Name:              assignment stage 2
-- Module Name:    assign - Behavioral
-- Project Name:
-- Target Devices: Spartan 3
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
```

```
-- Additional Comments:

--

----------------------------------------------------------------------------------

library IEEE;

use IEEE.STD_LOGIC_1164.ALL;

use ieee.std_logic_unsigned.all;


entity assign is

    Port ( clock : in  STD_LOGIC;

           buttons : in  STD_LOGIC_VECTOR (3 downto 0);     --button0=reset button1=restart
button2=set the n*1/4 sec to action

           switches : in   STD_LOGIC_VECTOR (7 downto 0);      --switches for 2 purposes
(force state and input n binary number)

           leds : out  STD_LOGIC_VECTOR (7 downto 0);

           digit : out  STD_LOGIC_VECTOR (3 downto 0);

           segments : out  STD_LOGIC_VECTOR (7 downto 0));


                        -- Assign inputs and outputs to appropriate pins on FPGA

         attribute LOC : string ;

         attribute LOC of clock : signal is "T9";

         attribute LOC of switches : signal is "K13,K14,J13,J14,H13,H14,G12,F12";

         attribute LOC of buttons : signal is "L14,L13,M14,M13";

         attribute LOC of leds : signal is "P11,P12,N12,P13,N14,L12,P14,K12";

         attribute LOC of digit : signal is "E13,F14,G14,D14";

         attribute LOC of segments : signal is "P16,N16,F13,R16,P15,N15,G13,E14";


end assign;


architecture Behavioral of assign is

        component display

         Port ( segs : out  STD_LOGIC_VECTOR (7 downto 0);

                           number : in  STD_LOGIC_VECTOR (7 downto 0));
```

```
	end component;

	signal state,state_next : STD_LOGIC_VECTOR(3 downto 0);

	signal signal_clk:std_logic;                              --this is the delayed clock
that i program

	signal n:STD_LOGIC_VECTOR(7 downto 0):=X"01";                  --here i will store
the n binary number

	signal TOTAL_DELAY:STD_LOGIC_VECTOR(39 downto 0):=X"0000BEBC20";     --here i will
store the n*1/4 seconds  (40 bits=8 bits x 32 bits)

	signal count:STD_LOGIC_VECTOR(31 downto 0);                  --this is the counter
variable to delay the clock

	constant  DELAY:STD_LOGIC_VECTOR(31  downto  0):=X"00BEBC20";    --delay  1/4  sec
(50000000/4->to HEX=BEBC20)



	--definition of the states

	constant S0 : STD_LOGIC_VECTOR(3 downto 0) := "0000";

	constant S1 : STD_LOGIC_VECTOR(3 downto 0) := "0001";

	constant S2 : STD_LOGIC_VECTOR(3 downto 0) := "0010";

	constant S3 : STD_LOGIC_VECTOR(3 downto 0) := "0011";

	constant S4 : STD_LOGIC_VECTOR(3 downto 0) := "0100";

	constant S5 : STD_LOGIC_VECTOR(3 downto 0) := "0101";

	constant S6 : STD_LOGIC_VECTOR(3 downto 0) := "0110";

	constant S7 : STD_LOGIC_VECTOR(3 downto 0) := "0111";

	constant S8 : STD_LOGIC_VECTOR(3 downto 0) := "1000";

	constant S9 : STD_LOGIC_VECTOR(3 downto 0) := "1001";

	constant S10 : STD_LOGIC_VECTOR(3 downto 0) := "1010";

	constant S11 : STD_LOGIC_VECTOR(3 downto 0) := "1011";

	constant S12 : STD_LOGIC_VECTOR(3 downto 0) := "1100";

	constant S13 : STD_LOGIC_VECTOR(3 downto 0) := "1101";

	constant S14 : STD_LOGIC_VECTOR(3 downto 0) := "1110";

	constant S15 : STD_LOGIC_VECTOR(3 downto 0) := "1111";


begin
```

```
TOTAL_DELAY<=n*DELAY;

--states always look to the first 4 LEDS

leds(3 downto 0)<=state;


--Initializing Unused Outputs

leds(6 downto 4)<="000";

digit(3 downto 0)<="1110";

--segments(7 downto 0)<=X"FF";


dd : display port map(segs => segments, number => n);


pr1: process (signal_clk,switches)

               begin


                     if(buttons="0001") then          --if (button(0)=ON) force state from
the position that the switches (3 downto 0) have

                            state<=switches(3 downto 0);    --S1 as the idle state in our
FSM Design

                     elsif (buttons="0010") then          --if (button(1)=ON) take the n
binary number from the positions of the switches that they already have

                        n<=switches(7 downto 0);

                     --elsif  (buttons="0100")  then            --if  (button(2)=ON)  Give
TOTAL_DELAY the final price

                            --TOTAL_DELAY<=n+DELAY;

                     elsif (rising_edge(signal_clk)) then       --here    is    the    main
function of the program

                            state<=state_next;
       --state_next looks always in the state

                                  if(state_next=S6) then
               --if (state=S6)--> Output(leds(7))='1'

                                        leds(7)<='1';
                     --else Output(leds(7))='0'

                                  else

                                        leds(7)<='0';

                                  end if;
```

```
                end if;
        end process;

pr2:    process
        begin
        wait until signal_clk'event and signal_clk='1';

        if (state=S1) then
                state_next<=S15;
        elsif (state=S15) then
                state_next<=S2;
        elsif (state=S2) then
                state_next<=S13;
        elsif (state=S13) then
                state_next<=S3;
        elsif (state=S3) then
                state_next<=S12;
        elsif (state=S12) then
                state_next<=S4;
        elsif (state=S4) then
                state_next<=S10;
        elsif (state=S10) then
                state_next<=S5;
        elsif (state=S5) then
                state_next<=S9;
        elsif (state=S9) then
                state_next<=S6;
        elsif (state=S6) then
                state_next<=S7;
        elsif (state=S7) then
                state_next<=S1;
        else
```

```
                        state_next<=S1;         --use S1 as the Idle State


                end if;

                end process;


pr3: process(clock)                 --this is the process in which i produce the delayed
clock(signal_clk)

                begin

                if rising_edge(clock) then

                count<=count+1;


                        if (count=TOTAL_DELAY) then         --TOTAL_DELAY = n*DELAY = n*1/4
seconds

                                signal_clk<='1';

                                count<=(others=>'0');

                        elsif(count>TOTAL_DELAY) then

                                count<=(others=>'0');

                        elsif(count<TOTAL_DELAY) then

                                signal_clk<='0';

                        end if;

                end if;

                end process;




end Behavioral;
```

6.3 STAGE 3

```
----------------------------------------------------------------------------------

-- Company:

-- Engineer: Konstantinos Tsimpoukas

--
```

```
-- Create Date:    18:23:11 11/21/2010
-- Design Name:              assignment stage 3
-- Module Name:    assign - Behavioral
-- Project Name:
-- Target Devices: Spartan 3
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity assign is
    Port ( clock : in  STD_LOGIC;
           buttons : in  STD_LOGIC_VECTOR (3 downto 0);
           switches : in  STD_LOGIC_VECTOR (7 downto 0);
           leds : out  STD_LOGIC_VECTOR (7 downto 0);
           digit : out  STD_LOGIC_VECTOR (3 downto 0);
           segments : out  STD_LOGIC_VECTOR (7 downto 0));

                        -- Assign inputs and outputs to appropriate pins on FPGA
         attribute LOC : string ;
         attribute LOC of clock : signal is "T9";
         attribute LOC of switches : signal is "K13,K14,J13,J14,H13,H14,G12,F12";
         attribute LOC of buttons : signal is "L14,L13,M14,M13";
```

```
         attribute LOC of leds : signal is "P11,P12,N12,P13,N14,L12,P14,K12";
         attribute LOC of digit : signal is "E13,F14,G14,D14";
         attribute LOC of segments : signal is "P16,N16,F13,R16,P15,N15,G13,E14";

end assign;

architecture Behavioral of assign is
   component display
       Port ( segs : out  STD_LOGIC_VECTOR (7 downto 0);
           number : in  STD_LOGIC_VECTOR (3 downto 0));
                      end component;
       signal state,state_next : STD_LOGIC_VECTOR(3 downto 0);
       signal signal_clk:std_logic;                              --this is the new delayed
clock
       signal choice:std_logic_vector(7 downto 0):=X"12";                    --choose
the student's ID with this variable
       signal count:STD_LOGIC_VECTOR(31 downto 0):=(others=>'0');

       constant MAX_VALUE:STD_LOGIC_VECTOR(31 downto 0):=X"01FFFFFF";  --delay in order to
see the changes in the LEDS

       constant S0 : STD_LOGIC_VECTOR(3 downto 0) := "0000";
       constant S1 : STD_LOGIC_VECTOR(3 downto 0) := "0001";
       constant S2 : STD_LOGIC_VECTOR(3 downto 0) := "0010";
       constant S3 : STD_LOGIC_VECTOR(3 downto 0) := "0011";
       constant S4 : STD_LOGIC_VECTOR(3 downto 0) := "0100";
       constant S5 : STD_LOGIC_VECTOR(3 downto 0) := "0101";
       constant S6 : STD_LOGIC_VECTOR(3 downto 0) := "0110";
       constant S7 : STD_LOGIC_VECTOR(3 downto 0) := "0111";
       constant S8 : STD_LOGIC_VECTOR(3 downto 0) := "1000";
       constant S9 : STD_LOGIC_VECTOR(3 downto 0) := "1001";
       constant S10 : STD_LOGIC_VECTOR(3 downto 0) := "1010";
       constant S11 : STD_LOGIC_VECTOR(3 downto 0) := "1011";
```

```
        constant S12 : STD_LOGIC_VECTOR(3 downto 0) := "1100";
        constant S13 : STD_LOGIC_VECTOR(3 downto 0) := "1101";
        constant S14 : STD_LOGIC_VECTOR(3 downto 0) := "1110";
        constant S15 : STD_LOGIC_VECTOR(3 downto 0) := "1111";

begin

dd:display port map (segs=>segments, number=>state);
leds(3 downto 0)<=state;

--Initializing Unused Outputs
leds(6 downto 4)<="000";
digit(3 downto 0)<="1110";
--segments(7 downto 0)<=X"FF";

pr1: process (signal_clk,switches,buttons,state_next)
               begin

                     if (rising_edge(signal_clk)) then

                              if(buttons="0001") then     --this is button 0         --force
unused (switches must be already set in the right positions)
                                      state<=switches(3 downto 0);                  --i have
to push reset for a big amount of time to go to the next that i already picked quickly
                              elsif (buttons="0010") then    --this is button 1     --select
choice (switches must be already set in the right positions)
                                      choice<=switches(7 downto 0);
                              else
                                      state<=state_next;
                              end if;
                     end if;
               end process;
```

```
pr2: process(clock)
		begin
		if rising_edge(clock) then
		count<=count+1;

			if count=MAX_VALUE then
				signal_clk<='1';
				count<=(others=>'0');
			else
				signal_clk<='0';
			end if;
		end if;
		end process;

choice_process:
	process(signal_clk,choice)
	begin
	if (rising_edge(signal_clk)) then

		if ( choice=X"01" or choice=X"02" or choice=X"03" or choice=X"13"
				or choice=X"16" or choice=X"17" or choice=X"19" or
choice=X"1A"
				or choice=X"1C" or choice=X"1F" or choice=X"22") then
					if(state_next=S13) then
						leds(7)<='1';
					else
						leds(7)<='0';
					end if;
				elsif(choice=X"04" or choice=X"05" or choice=X"06" or
choice=X"14" or choice=X"1B"
				or choice=X"1D" or choice=X"1E" or choice=X"20" ) then
					if(state_next=S12) then
```

```
                                        leds(7)<='1';
                                    else
                                        leds(7)<='0';
                                    end if;
                                elsif(choice=X"07"   or   choice=X"08"   or   choice=X"15"   or
choice=X"18" or choice=X"21") then
                                    if(state_next=S11) then
                                        leds(7)<='1';
                                    else
                                        leds(7)<='0';
                                    end if;
                                elsif(choice=X"09"   or   choice=X"0A"   or   choice=X"0B"   or
choice=X"0C") then
                                    if(state_next=S10) then
                                        leds(7)<='1';
                                    else
                                        leds(7)<='0';
                                    end if;
                                elsif(choice=X"0D" ) then
                                    if(state_next=S7)then
                                        leds(7)<='1';
                                    else
                                        leds(7)<='0';
                                    end if;
                                elsif(choice=X"0E" or choice=X"0F" ) then
                                    if(state_next=S8) then
                                        leds(7)<='1';
                                    else
                                        leds(7)<='0';
                                    end if;
                                elsif (choice=X"10" or choice=X"11" or choice=X"12") then
                                    if(state_next=S6) then
```

```
                                leds(7)<='1';
                        else
                                leds(7)<='0';
                        end if;
                else
                        if(state_next=S12) then
                                leds(7)<='1';
                        else
                                leds(7)<='0';
                        end if;
                end if;

if (choice="00000001") then     --n=1
        if (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S12;
        elsif (state=S12) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S8;
        elsif (state=S8) then state_next<=S2;
        else state_next<=S2;  --S2 idle State
        end if;
elsif (choice="00000010") then  --n=2
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S14;
        elsif (state=S14) then state_next<=S5;
        elsif (state=S5) then state_next<=S13;
        elsif (state=S13) then state_next<=S7;
        elsif (state=S7) then state_next<=S12;
```

```
        elsif (state=S12) then state_next<=S8;
        elsif (state=S8) then state_next<=S11;
        elsif (state=S11) then state_next<=S10;
        elsif (state=S10) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00000011") then  --n=3
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S4;
        elsif (state=S4) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S7;
        elsif (state=S7) then state_next<=S11;
        elsif (state=S11) then state_next<=S8;
        elsif (state=S8) then state_next<=S10;
        elsif (state=S10) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00000100") then      --n=4
        if (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S12;
        elsif (state=S12) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S9;
        elsif (state=S9) then state_next<=S7;
        elsif (state=S7) then state_next<=S8;
        elsif (state=S8) then state_next<=S3;
        else state_next<=S3;  --S3 idle State
```

```
        end if;
elsif (choice="00000101") then      --n=5
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S4;
        elsif (state=S4) then state_next<=S13;
        elsif (state=S13) then state_next<=S6;
        elsif (state=S6) then state_next<=S12;
        elsif (state=S12) then state_next<=S8;
        elsif (state=S8) then state_next<=S10;
        elsif (state=S10) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00000110") then  --n=6
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S6;
        elsif (state=S6) then state_next<=S11;
        elsif (state=S11) then state_next<=S9;
        elsif (state=S9) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00000111") then  --n=7
        if (state=S3) then state_next<=S13;
```

```
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S6;
        elsif (state=S6) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S8;
        elsif (state=S8) then state_next<=S3;
        else state_next<=S3;  --S3 idle State
        end if;
elsif (choice="00001000") then      --n=8
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S10;
        elsif (state=S10) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00001001") then      --n=9
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S12;
```

```
        elsif (state=S12) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00001010") then      --n=10
        if (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S6;
        elsif (state=S6) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S3;
        else state_next<=S3;  --S3 idle State
        end if;
elsif (choice="00001011") then      --n=11
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
```

```
        end if;
elsif (choice="00001100") then      --n=12
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S5;
        elsif (state=S5) then state_next<=S10;
        elsif (state=S10) then state_next<=S7;
        elsif (state=S7) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00001101") then      --n=13
        if (state=S3) then state_next<=S14;
        elsif (state=S14) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S6;
        elsif (state=S6) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S3;
        else state_next<=S3;  --S3 idle State
        end if;
elsif (choice="00001110") then      --n=14
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
```

```
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S10;
        elsif (state=S10) then state_next<=S5;
        elsif (state=S5) then state_next<=S8;
        elsif (state=S8) then state_next<=S6;
        elsif (state=S6) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00001111") then      --n=15
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S5;
        elsif (state=S5) then state_next<=S9;
        elsif (state=S9) then state_next<=S6;
        elsif (state=S6) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010000") then      --n=16
        if (state=S3) then state_next<=S14;
        elsif (state=S14) then state_next<=S5;
        elsif (state=S5) then state_next<=S13;
        elsif (state=S13) then state_next<=S6;
        elsif (state=S6) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
```

```
        elsif (state=S7) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S3;
        else state_next<=S3;  --S3 idle State
        end if;
elsif (choice="00010001") then      --n=17
        if (state=S1) then state_next<=S14;
        elsif (state=S14) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S11;
        elsif (state=S11) then state_next<=S4;
        elsif (state=S4) then state_next<=S9;
        elsif (state=S9) then state_next<=S5;
        elsif (state=S5) then state_next<=S7;
        elsif (state=S7) then state_next<=S6;
        elsif (state=S6) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010010") then      --n=18
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S10;
        elsif (state=S10) then state_next<=S5;
        elsif (state=S5) then state_next<=S9;
        elsif (state=S9) then state_next<=S6;
        elsif (state=S6) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
```

```
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010011") then      --n=19
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S14;
        elsif (state=S14) then state_next<=S5;
        elsif (state=S5) then state_next<=S13;
        elsif (state=S13) then state_next<=S7;
        elsif (state=S7) then state_next<=S11;
        elsif (state=S11) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010100") then      --n=20
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S14;
        elsif (state=S14) then state_next<=S5;
        elsif (state=S5) then state_next<=S13;
        elsif (state=S13) then state_next<=S7;
        elsif (state=S7) then state_next<=S12;
        elsif (state=S12) then state_next<=S8;
        elsif (state=S8) then state_next<=S11;
        elsif (state=S11) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010101") then      --n=21
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
```

```
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S11;
        elsif (state=S11) then state_next<=S4;
        elsif (state=S4) then state_next<=S10;
        elsif (state=S10) then state_next<=S5;
        elsif (state=S5) then state_next<=S8;
        elsif (state=S8) then state_next<=S6;
        elsif (state=S6) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010110") then      --n=22
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S7;
        elsif (state=S7) then state_next<=S11;
        elsif (state=S11) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010111") then      --n=23
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S4;
        elsif (state=S4) then state_next<=S13;
        elsif (state=S13) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S7;
```

```
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011000") then      --n=24
        if (state=S1) then state_next<=S14;
        elsif (state=S14) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S11;
        elsif (state=S11) then state_next<=S4;
        elsif (state=S4) then state_next<=S9;
        elsif (state=S9) then state_next<=S5;
        elsif (state=S5) then state_next<=S8;
        elsif (state=S8) then state_next<=S6;
        elsif (state=S6) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011001") then      --n=25
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S11;
        elsif (state=S11) then state_next<=S5;
        elsif (state=S5) then state_next<=S9;
        elsif (state=S9) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
```

```
elsif (choice="00011010") then      --n=26
       if (state=S1) then state_next<=S15;
       elsif (state=S15) then state_next<=S2;
       elsif (state=S2) then state_next<=S14;
       elsif (state=S14) then state_next<=S3;
       elsif (state=S3) then state_next<=S13;
       elsif (state=S13) then state_next<=S5;
       elsif (state=S5) then state_next<=S11;
       elsif (state=S11) then state_next<=S7;
       elsif (state=S7) then state_next<=S10;
       elsif (state=S10) then state_next<=S9;
       elsif (state=S9) then state_next<=S1;
       else state_next<=S1;  --S1 idle State
       end if;
elsif (choice="00011011") then      --n=27
       if (state=S1) then state_next<=S15;
       elsif (state=S15) then state_next<=S2;
       elsif (state=S2) then state_next<=S14;
       elsif (state=S14) then state_next<=S4;
       elsif (state=S4) then state_next<=S13;
       elsif (state=S13) then state_next<=S5;
       elsif (state=S5) then state_next<=S12;
       elsif (state=S12) then state_next<=S7;
       elsif (state=S7) then state_next<=S11;
       elsif (state=S11) then state_next<=S8;
       elsif (state=S8) then state_next<=S9;
       elsif (state=S9) then state_next<=S1;
       else state_next<=S1;  --S1 idle State
       end if;
elsif (choice="00011100") then      --n=28
       if (state=S1) then state_next<=S15;
       elsif (state=S15) then state_next<=S3;
```

```
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S5;
        elsif (state=S5) then state_next<=S9;
        elsif (state=S9) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011101") then      --n=29
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011110") then      --n=30
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
```

```
        elsif (state=S12) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011111") then      --n=31
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S6;
        elsif (state=S6) then state_next<=S9;
        elsif (state=S9) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00100000") then      --n=32
        if (state=S1) then state_next<=S14;
        elsif (state=S14) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S10;
        elsif (state=S10) then state_next<=S5;
        elsif (state=S5) then state_next<=S8;
        elsif (state=S8) then state_next<=S6;
```

```
        elsif (state=S6) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00100001") then      --n=33
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00100010") then      --n=34
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
```

```
                else    --n=35
                        if (state=S1) then state_next<=S15;
                        elsif (state=S15) then state_next<=S3;
                        elsif (state=S3) then state_next<=S14;
                        elsif (state=S14) then state_next<=S5;
                        elsif (state=S5) then state_next<=S13;
                        elsif (state=S13) then state_next<=S7;
                        elsif (state=S7) then state_next<=S12;
                        elsif (state=S12) then state_next<=S8;
                        elsif (state=S8) then state_next<=S10;
                        elsif (state=S10) then state_next<=S9;
                        elsif (state=S9) then state_next<=S1;

                        else state_next<=S1;  --S1 idle State
                        end if;

                end if;
        end if;



        end process;

end Behavioral;
```

6.4 STAGE 4

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer: Konstantinos Tsimpoukas
--
-- Create Date:    18:23:11 11/21/2010
-- Design Name:              assignment stage 4
```

```
-- Module Name:    assign - Behavioral
-- Project Name:
-- Target Devices: Spartan 3
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity assign is
               generic (choice:std_logic_vector(7 downto 0):=X"12"); --18 mine
    Port ( clock : in  STD_LOGIC;
           buttons : in  STD_LOGIC_VECTOR (3 downto 0);
           switches : in  STD_LOGIC_VECTOR (7 downto 0);
           leds : out  STD_LOGIC_VECTOR (7 downto 0);
           digit : out  STD_LOGIC_VECTOR (3 downto 0);
           segments : out  STD_LOGIC_VECTOR (7 downto 0));

                        -- Assign inputs and outputs to appropriate pins on FPGA
         attribute LOC : string ;
         attribute LOC of clock : signal is "T9";
         attribute LOC of switches : signal is "K13,K14,J13,J14,H13,H14,G12,F12";
         attribute LOC of buttons : signal is "L14,L13,M14,M13";
         attribute LOC of leds : signal is "P11,P12,N12,P13,N14,L12,P14,K12";
```

```
         attribute LOC of digit : signal is "E13,F14,G14,D14";
         attribute LOC of segments : signal is "P16,N16,F13,R16,P15,N15,G13,E14";

end assign;

architecture Behavioral of assign is
   component display
       Port ( segs : out  STD_LOGIC_VECTOR (7 downto 0);
           number : in  STD_LOGIC_VECTOR (3 downto 0));
                      end component;
       signal state,state_next : STD_LOGIC_VECTOR(3 downto 0);
       signal signal_clk:std_logic;                              --this is the new delayed
clock
       --signal choice:std_logic_vector(7 downto 0):=X"01";                    --choose
the student's ID with this variable
       signal count:STD_LOGIC_VECTOR(31 downto 0):=(others=>'0');

       constant MAX_VALUE:STD_LOGIC_VECTOR(31 downto 0):=X"01FFFFFF";  --delay in order to
see the changes in the LEDS

       constant S0 : STD_LOGIC_VECTOR(3 downto 0) := "0000";
       constant S1 : STD_LOGIC_VECTOR(3 downto 0) := "0001";
       constant S2 : STD_LOGIC_VECTOR(3 downto 0) := "0010";
       constant S3 : STD_LOGIC_VECTOR(3 downto 0) := "0011";
       constant S4 : STD_LOGIC_VECTOR(3 downto 0) := "0100";
       constant S5 : STD_LOGIC_VECTOR(3 downto 0) := "0101";
       constant S6 : STD_LOGIC_VECTOR(3 downto 0) := "0110";
       constant S7 : STD_LOGIC_VECTOR(3 downto 0) := "0111";
       constant S8 : STD_LOGIC_VECTOR(3 downto 0) := "1000";
       constant S9 : STD_LOGIC_VECTOR(3 downto 0) := "1001";
       constant S10 : STD_LOGIC_VECTOR(3 downto 0) := "1010";
       constant S11 : STD_LOGIC_VECTOR(3 downto 0) := "1011";
       constant S12 : STD_LOGIC_VECTOR(3 downto 0) := "1100";
```

```
        constant S13 : STD_LOGIC_VECTOR(3 downto 0) := "1101";
        constant S14 : STD_LOGIC_VECTOR(3 downto 0) := "1110";
        constant S15 : STD_LOGIC_VECTOR(3 downto 0) := "1111";

begin

dd:display port map (segs=>segments, number=>state);
leds(3 downto 0)<=state;

--Initializing Unused Outputs
leds(6 downto 4)<="000";
digit(3 downto 0)<="1110";
--segments(7 downto 0)<=X"FF";

pr1: process (signal_clk,buttons)
               begin

                     if (rising_edge(signal_clk)) then

                              if(buttons="0001") then     --this is button 0         --force
unused (switches must be already set in the right positions)
                                    state<=switches(3 downto 0);                  --i have
to push reset for a big amount of time to go to the next that i already picked quickly
                              --elsif (buttons="0010") then  --this is button 1     --select
choice (switches must be already set in the right positions)
                                --choice<=switches(7 downto 0);
                              else
                                    state<=state_next;
                              end if;
                     end if;
               end process;

pr2: process(clock,count)
```

```
		begin
		if rising_edge(clock) then
		count<=count+1;

			if count=MAX_VALUE then
				signal_clk<='1';
				count<=(others=>'0');
			else
				signal_clk<='0';
			end if;
		end if;
		end process;

choice_process:
	process(signal_clk)
	begin
	if (rising_edge(signal_clk)) then

		if ( choice=X"01" or choice=X"02" or choice=X"03" or choice=X"13"
					or choice=X"16" or choice=X"17" or choice=X"19" or
choice=X"1A"
					or choice=X"1C" or choice=X"1F" or choice=X"22") then
				if(state_next=S13) then
					leds(7)<='1';
				else
					leds(7)<='0';
				end if;
			elsif(choice=X"04" or choice=X"05" or choice=X"06" or
choice=X"14" or choice=X"1B"
				or choice=X"1D" or choice=X"1E" or choice=X"20" ) then
				if(state_next=S12) then
					leds(7)<='1';
```

```
                                        else
                                                leds(7)<='0';
                                        end if;
                                elsif(choice=X"07"   or   choice=X"08"   or   choice=X"15"   or
choice=X"18" or choice=X"21") then
                                        if(state_next=S11) then
                                                leds(7)<='1';
                                        else
                                                leds(7)<='0';
                                        end if;
                                elsif(choice=X"09"   or   choice=X"0A"   or   choice=X"0B"   or
choice=X"0C") then
                                        if(state_next=S10) then
                                                leds(7)<='1';
                                        else
                                                leds(7)<='0';
                                        end if;
                                elsif(choice=X"0D" ) then
                                        if(state_next=S7)then
                                                leds(7)<='1';
                                        else
                                                leds(7)<='0';
                                        end if;
                                elsif(choice=X"0E" or choice=X"0F" ) then
                                        if(state_next=S8) then
                                                leds(7)<='1';
                                        else
                                                leds(7)<='0';
                                        end if;
                                elsif (choice=X"10" or choice=X"11" or choice=X"12") then
                                        if(state_next=S6) then
                                                leds(7)<='1';
```

```
                                else
                                        leds(7)<='0';
                                end if;
                        else
                                if(state_next=S12) then
                                        leds(7)<='1';
                                else
                                        leds(7)<='0';
                                end if;
                        end if;

        if (choice="00000001") then     --n=1
                if (state=S2) then state_next<=S13;
                elsif (state=S13) then state_next<=S4;
                elsif (state=S4) then state_next<=S12;
                elsif (state=S12) then state_next<=S5;
                elsif (state=S5) then state_next<=S11;
                elsif (state=S11) then state_next<=S7;
                elsif (state=S7) then state_next<=S9;
                elsif (state=S9) then state_next<=S8;
                elsif (state=S8) then state_next<=S2;
                else state_next<=S2;  --S2 idle State
                end if;
        elsif (choice="00000010") then  --n=2
                if (state=S1) then state_next<=S15;
                elsif (state=S15) then state_next<=S3;
                elsif (state=S3) then state_next<=S14;
                elsif (state=S14) then state_next<=S5;
                elsif (state=S5) then state_next<=S13;
                elsif (state=S13) then state_next<=S7;
                elsif (state=S7) then state_next<=S12;
                elsif (state=S12) then state_next<=S8;
```

```
        elsif (state=S8) then state_next<=S11;
        elsif (state=S11) then state_next<=S10;
        elsif (state=S10) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00000011") then  --n=3
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S4;
        elsif (state=S4) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S7;
        elsif (state=S7) then state_next<=S11;
        elsif (state=S11) then state_next<=S8;
        elsif (state=S8) then state_next<=S10;
        elsif (state=S10) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00000100") then      --n=4
        if (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S12;
        elsif (state=S12) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S9;
        elsif (state=S9) then state_next<=S7;
        elsif (state=S7) then state_next<=S8;
        elsif (state=S8) then state_next<=S3;
        else state_next<=S3;  --S3 idle State
        end if;
```

```
elsif (choice="00000101") then      --n=5
       if (state=S1) then state_next<=S15;
       elsif (state=S15) then state_next<=S2;
       elsif (state=S2) then state_next<=S14;
       elsif (state=S14) then state_next<=S4;
       elsif (state=S4) then state_next<=S13;
       elsif (state=S13) then state_next<=S6;
       elsif (state=S6) then state_next<=S12;
       elsif (state=S12) then state_next<=S8;
       elsif (state=S8) then state_next<=S10;
       elsif (state=S10) then state_next<=S9;
       elsif (state=S9) then state_next<=S1;
       else state_next<=S1;  --S1 idle State
       end if;
elsif (choice="00000110") then  --n=6
       if (state=S1) then state_next<=S15;
       elsif (state=S15) then state_next<=S2;
       elsif (state=S2) then state_next<=S14;
       elsif (state=S14) then state_next<=S3;
       elsif (state=S3) then state_next<=S13;
       elsif (state=S13) then state_next<=S5;
       elsif (state=S5) then state_next<=S12;
       elsif (state=S12) then state_next<=S6;
       elsif (state=S6) then state_next<=S11;
       elsif (state=S11) then state_next<=S9;
       elsif (state=S9) then state_next<=S8;
       elsif (state=S8) then state_next<=S1;
       else state_next<=S1;  --S1 idle State
       end if;
elsif (choice="00000111") then  --n=7
       if (state=S3) then state_next<=S13;
       elsif (state=S13) then state_next<=S5;
```

```
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S6;
        elsif (state=S6) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S8;
        elsif (state=S8) then state_next<=S3;
        else state_next<=S3;  --S3 idle State
        end if;
elsif (choice="00001000") then      --n=8
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S10;
        elsif (state=S10) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00001001") then      --n=9
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S12;
        elsif (state=S12) then state_next<=S6;
```

```
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00001010") then      --n=10
        if (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S6;
        elsif (state=S6) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S3;
        else state_next<=S3;  --S3 idle State
        end if;
elsif (choice="00001011") then      --n=11
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
```

```
elsif (choice="00001100") then      --n=12
       if (state=S1) then state_next<=S15;
       elsif (state=S15) then state_next<=S2;
       elsif (state=S2) then state_next<=S14;
       elsif (state=S14) then state_next<=S3;
       elsif (state=S3) then state_next<=S13;
       elsif (state=S13) then state_next<=S4;
       elsif (state=S4) then state_next<=S11;
       elsif (state=S11) then state_next<=S5;
       elsif (state=S5) then state_next<=S10;
       elsif (state=S10) then state_next<=S7;
       elsif (state=S7) then state_next<=S8;
       elsif (state=S8) then state_next<=S1;
       else state_next<=S1;  --S1 idle State
       end if;
elsif (choice="00001101") then      --n=13
       if (state=S3) then state_next<=S14;
       elsif (state=S14) then state_next<=S5;
       elsif (state=S5) then state_next<=S12;
       elsif (state=S12) then state_next<=S6;
       elsif (state=S6) then state_next<=S11;
       elsif (state=S11) then state_next<=S7;
       elsif (state=S7) then state_next<=S10;
       elsif (state=S10) then state_next<=S8;
       elsif (state=S8) then state_next<=S3;
       else state_next<=S3;  --S3 idle State
       end if;
elsif (choice="00001110") then      --n=14
       if (state=S1) then state_next<=S15;
       elsif (state=S15) then state_next<=S2;
       elsif (state=S2) then state_next<=S13;
       elsif (state=S13) then state_next<=S3;
```

```
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S10;
        elsif (state=S10) then state_next<=S5;
        elsif (state=S5) then state_next<=S8;
        elsif (state=S8) then state_next<=S6;
        elsif (state=S6) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00001111") then      --n=15
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S5;
        elsif (state=S5) then state_next<=S9;
        elsif (state=S9) then state_next<=S6;
        elsif (state=S6) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010000") then      --n=16
        if (state=S3) then state_next<=S14;
        elsif (state=S14) then state_next<=S5;
        elsif (state=S5) then state_next<=S13;
        elsif (state=S13) then state_next<=S6;
        elsif (state=S6) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S10;
```

```
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S3;
        else state_next<=S3;  --S3 idle State
        end if;
elsif (choice="00010001") then      --n=17
        if (state=S1) then state_next<=S14;
        elsif (state=S14) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S11;
        elsif (state=S11) then state_next<=S4;
        elsif (state=S4) then state_next<=S9;
        elsif (state=S9) then state_next<=S5;
        elsif (state=S5) then state_next<=S7;
        elsif (state=S7) then state_next<=S6;
        elsif (state=S6) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010010") then      --n=18
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S10;
        elsif (state=S10) then state_next<=S5;
        elsif (state=S5) then state_next<=S9;
        elsif (state=S9) then state_next<=S6;
        elsif (state=S6) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
```

```
        end if;
elsif (choice="00010011") then      --n=19
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S14;
        elsif (state=S14) then state_next<=S5;
        elsif (state=S5) then state_next<=S13;
        elsif (state=S13) then state_next<=S7;
        elsif (state=S7) then state_next<=S11;
        elsif (state=S11) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010100") then      --n=20
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S14;
        elsif (state=S14) then state_next<=S5;
        elsif (state=S5) then state_next<=S13;
        elsif (state=S13) then state_next<=S7;
        elsif (state=S7) then state_next<=S12;
        elsif (state=S12) then state_next<=S8;
        elsif (state=S8) then state_next<=S11;
        elsif (state=S11) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010101") then      --n=21
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
```

```
        elsif (state=S3) then state_next<=S11;
        elsif (state=S11) then state_next<=S4;
        elsif (state=S4) then state_next<=S10;
        elsif (state=S10) then state_next<=S5;
        elsif (state=S5) then state_next<=S8;
        elsif (state=S8) then state_next<=S6;
        elsif (state=S6) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010110") then      --n=22
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S7;
        elsif (state=S7) then state_next<=S11;
        elsif (state=S11) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00010111") then      --n=23
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S4;
        elsif (state=S4) then state_next<=S13;
        elsif (state=S13) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
```

```
        elsif (state=S9) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011000") then      --n=24
        if (state=S1) then state_next<=S14;
        elsif (state=S14) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S11;
        elsif (state=S11) then state_next<=S4;
        elsif (state=S4) then state_next<=S9;
        elsif (state=S9) then state_next<=S5;
        elsif (state=S5) then state_next<=S8;
        elsif (state=S8) then state_next<=S6;
        elsif (state=S6) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011001") then      --n=25
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S11;
        elsif (state=S11) then state_next<=S5;
        elsif (state=S5) then state_next<=S9;
        elsif (state=S9) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011010") then      --n=26
```

```
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S10;
        elsif (state=S10) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011011") then      --n=27
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S4;
        elsif (state=S4) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S7;
        elsif (state=S7) then state_next<=S11;
        elsif (state=S11) then state_next<=S8;
        elsif (state=S8) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011100") then      --n=28
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
```

```
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S5;
        elsif (state=S5) then state_next<=S9;
        elsif (state=S9) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011101") then      --n=29
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011110") then      --n=30
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S12;
        elsif (state=S12) then state_next<=S6;
```

```
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S8;
        elsif (state=S8) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00011111") then      --n=31
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S6;
        elsif (state=S6) then state_next<=S9;
        elsif (state=S9) then state_next<=S7;
        elsif (state=S7) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00100000") then      --n=32
        if (state=S1) then state_next<=S14;
        elsif (state=S14) then state_next<=S2;
        elsif (state=S2) then state_next<=S13;
        elsif (state=S13) then state_next<=S3;
        elsif (state=S3) then state_next<=S12;
        elsif (state=S12) then state_next<=S4;
        elsif (state=S4) then state_next<=S10;
        elsif (state=S10) then state_next<=S5;
        elsif (state=S5) then state_next<=S8;
        elsif (state=S8) then state_next<=S6;
        elsif (state=S6) then state_next<=S1;
```

```
        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00100001") then      --n=33
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S2;
        elsif (state=S2) then state_next<=S14;
        elsif (state=S14) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S4;
        elsif (state=S4) then state_next<=S11;
        elsif (state=S11) then state_next<=S6;
        elsif (state=S6) then state_next<=S10;
        elsif (state=S10) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
elsif (choice="00100010") then      --n=34
        if (state=S1) then state_next<=S15;
        elsif (state=S15) then state_next<=S3;
        elsif (state=S3) then state_next<=S13;
        elsif (state=S13) then state_next<=S5;
        elsif (state=S5) then state_next<=S11;
        elsif (state=S11) then state_next<=S7;
        elsif (state=S7) then state_next<=S9;
        elsif (state=S9) then state_next<=S8;
        elsif (state=S8) then state_next<=S1;

        else state_next<=S1;  --S1 idle State
        end if;
else   --n=35
```

```
                        if (state=S1) then state_next<=S15;
                        elsif (state=S15) then state_next<=S3;
                        elsif (state=S3) then state_next<=S14;
                        elsif (state=S14) then state_next<=S5;
                        elsif (state=S5) then state_next<=S13;
                        elsif (state=S13) then state_next<=S7;
                        elsif (state=S7) then state_next<=S12;
                        elsif (state=S12) then state_next<=S8;
                        elsif (state=S8) then state_next<=S10;
                        elsif (state=S10) then state_next<=S9;
                        elsif (state=S9) then state_next<=S1;

                        else state_next<=S1;  --S1 idle State
                        end if;

                end if;
        end if;



        end process;

end Behavioral;
```