ECEN 429: Introduction to Digital Systems Design Laboratory

North Carolina Agricultural and Technical State University

Department of Electrical and Computer Engineering

Ian Parker (Reporter)

Tayanna Lee (Lab Partner)

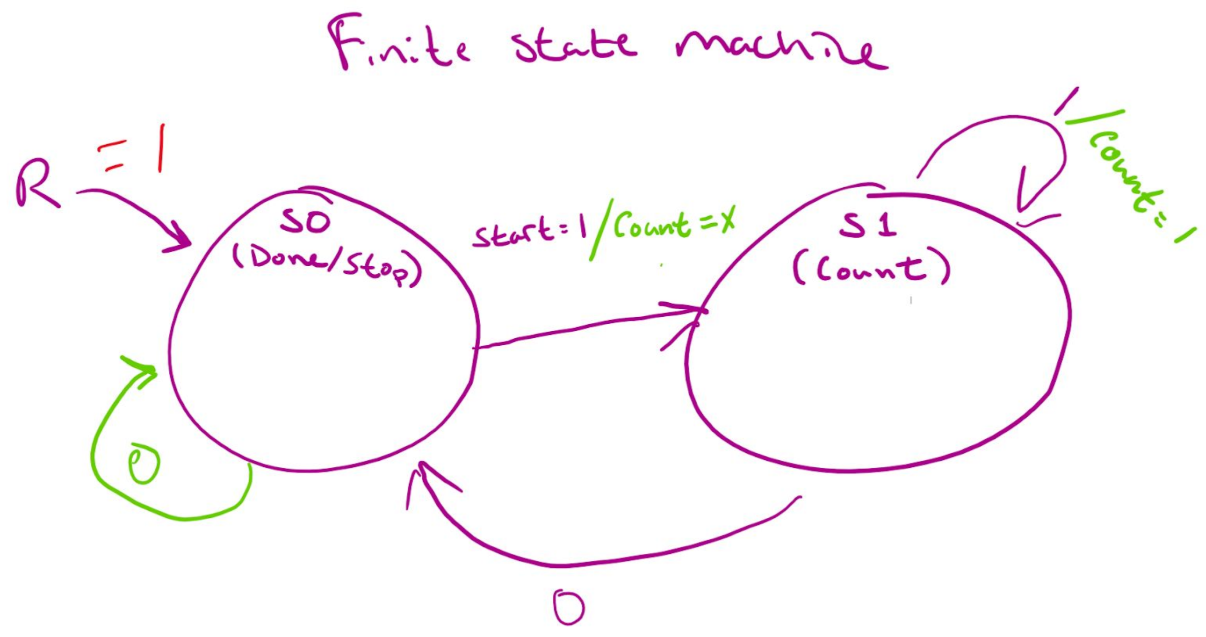
April 4, 2019

Lab #8

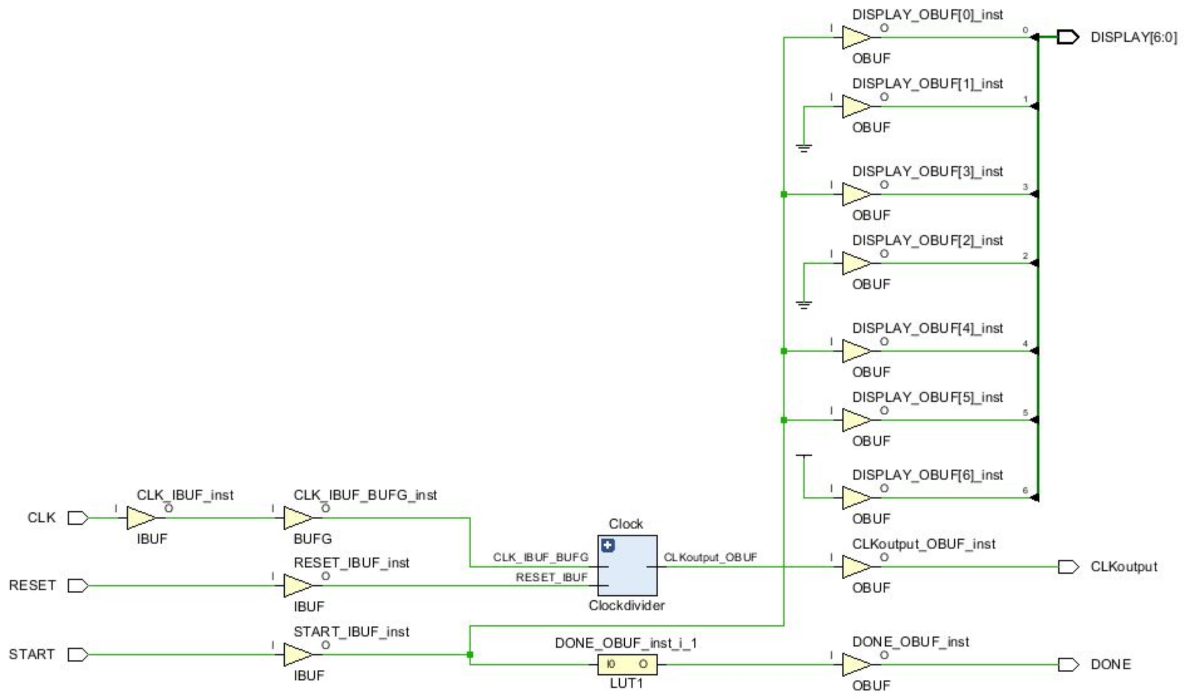
Introduction:

For this lab, we learn how to create a counter with multiple functionalities. We display how the counter changes from one function to the next by implementing a finite state machine. Finally, after combining both into a top-level design, we make sure to display the results on the Seven-Segment Display.

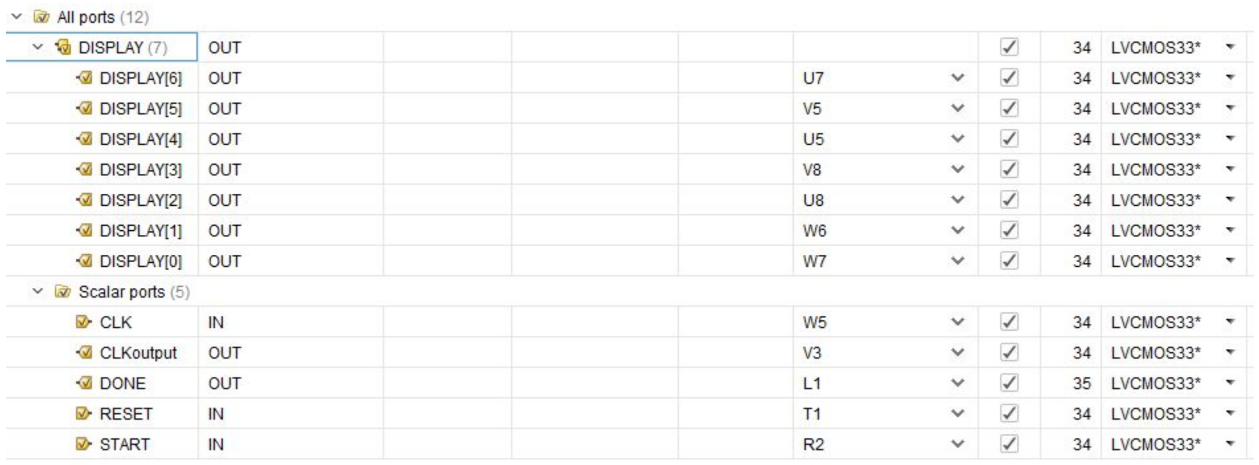
Part 1
Finite State Machine



Schematic



PIN ASSIGNMENTS



All ports (12)

DISPLAY (7)	OUT				✓	34	LVCMOS33*
DISPLAY[6]	OUT			U7	✓	34	LVCMOS33*
DISPLAY[5]	OUT			V5	✓	34	LVCMOS33*
DISPLAY[4]	OUT			U5	✓	34	LVCMOS33*
DISPLAY[3]	OUT			V8	✓	34	LVCMOS33*
DISPLAY[2]	OUT			U8	✓	34	LVCMOS33*
DISPLAY[1]	OUT			W6	✓	34	LVCMOS33*
DISPLAY[0]	OUT			W7	✓	34	LVCMOS33*
Scalar ports (5)							
CLK	IN			W5	✓	34	LVCMOS33*
CLKoutput	OUT			V3	✓	34	LVCMOS33*
DONE	OUT			L1	✓	35	LVCMOS33*
RESET	IN			T1	✓	34	LVCMOS33*
START	IN			R2	✓	34	LVCMOS33*

Results

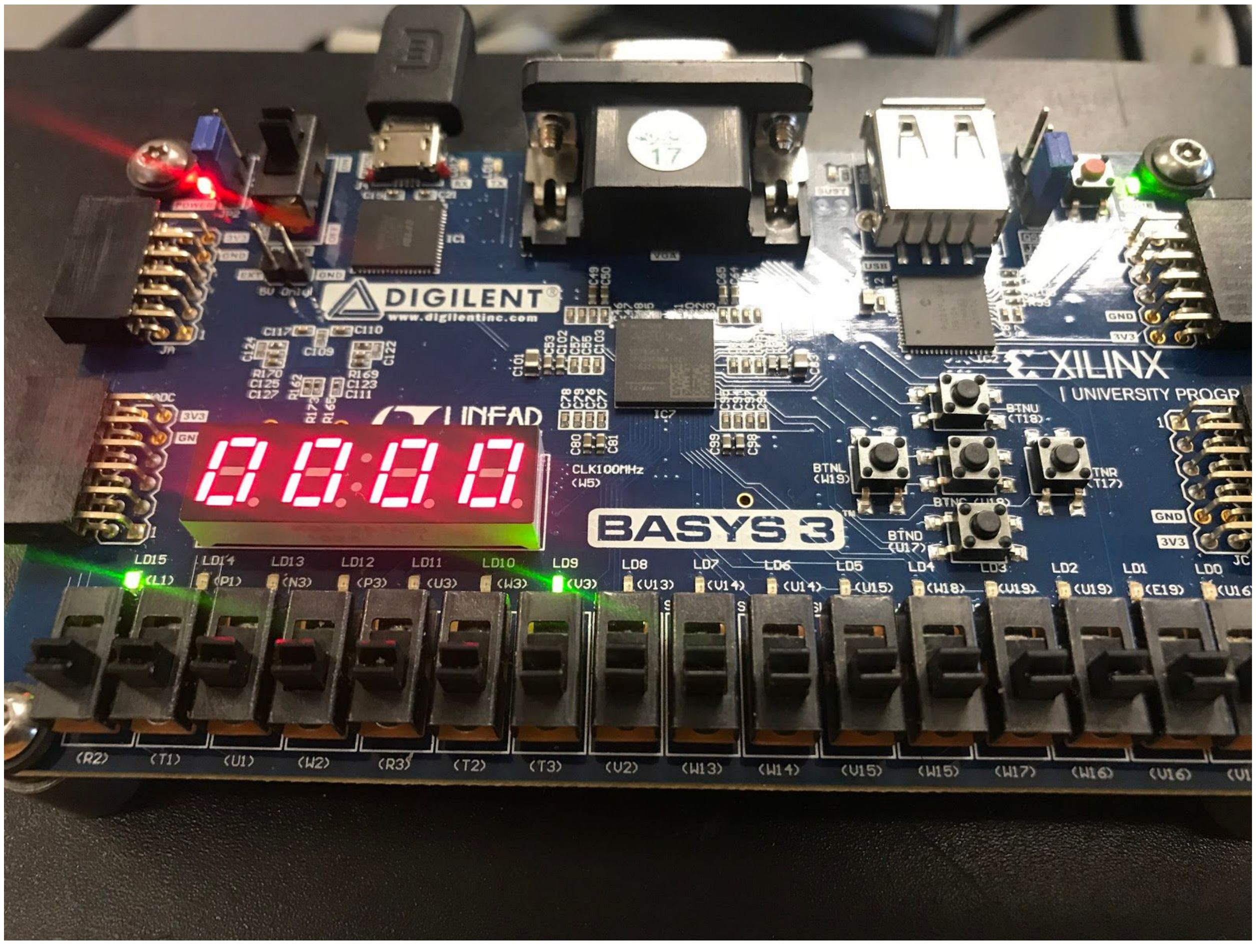


Figure 1.1) Low Clock. State 0

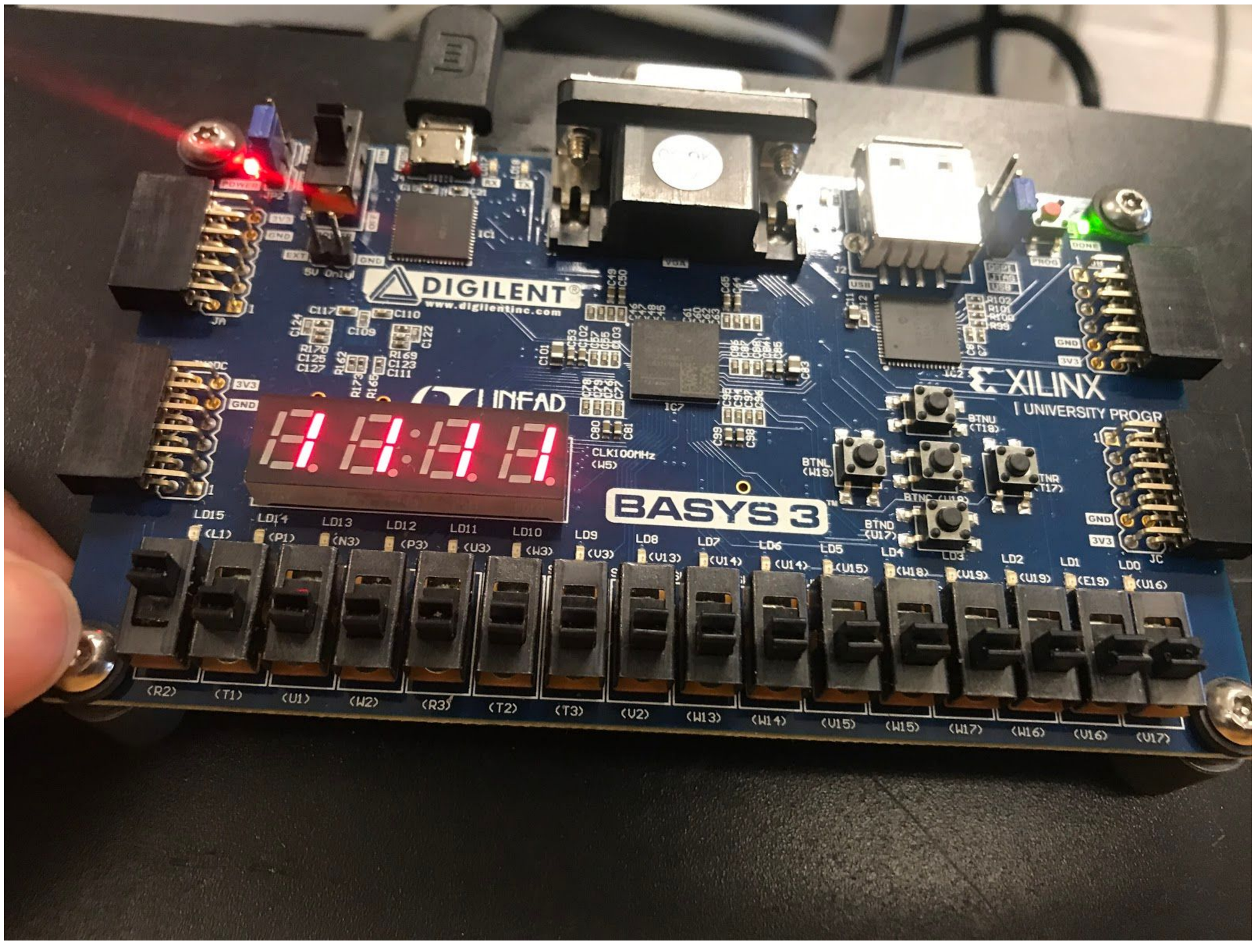
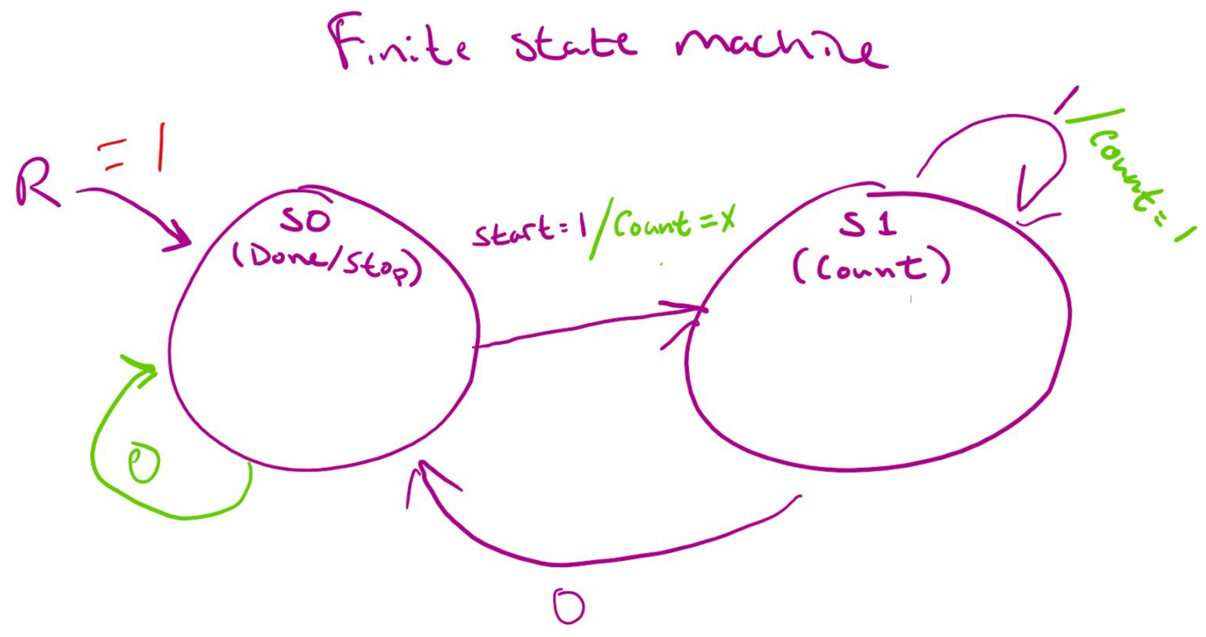
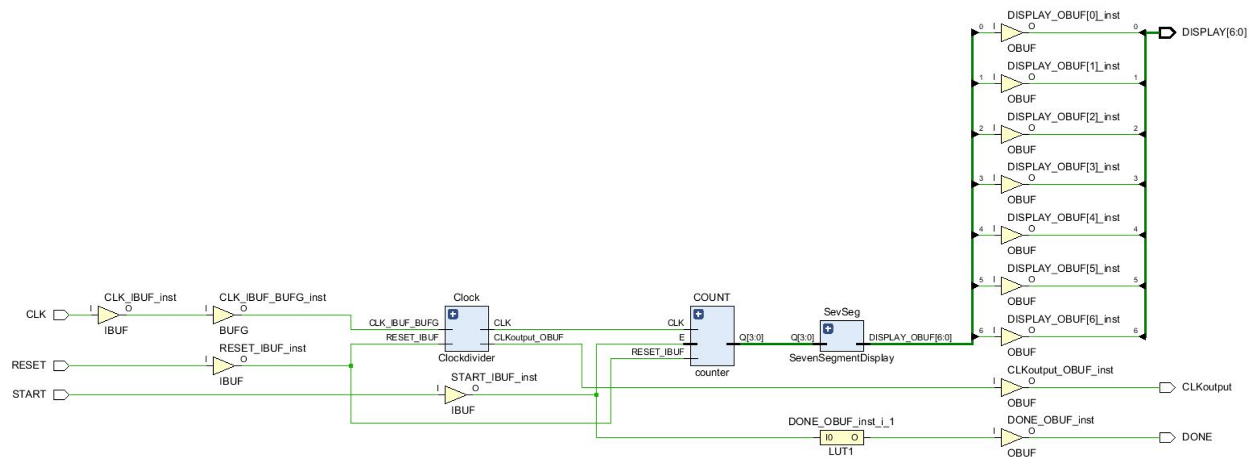


Figure 1.2) High Clock. State 1

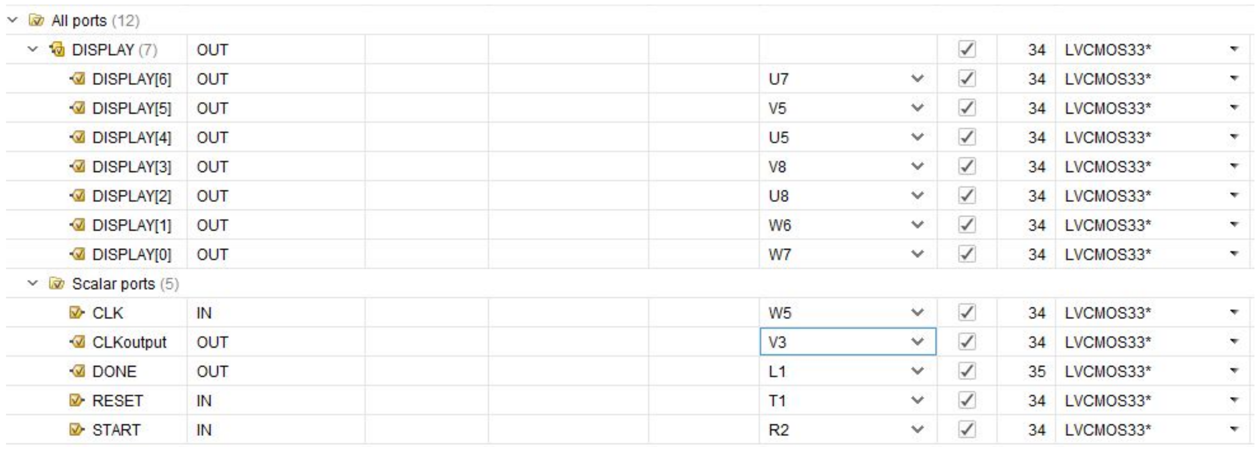
Part 2
Finite State Machine



Schematic



PIN ASSIGNMENTS



All ports (12)								
DISPLAY (7)	OUT					✓	34	LVCMOS33*
DISPLAY[6]	OUT				U7	✓	34	LVCMOS33*
DISPLAY[5]	OUT				V5	✓	34	LVCMOS33*
DISPLAY[4]	OUT				U5	✓	34	LVCMOS33*
DISPLAY[3]	OUT				V8	✓	34	LVCMOS33*
DISPLAY[2]	OUT				U8	✓	34	LVCMOS33*
DISPLAY[1]	OUT				W6	✓	34	LVCMOS33*
DISPLAY[0]	OUT				W7	✓	34	LVCMOS33*
Scalar ports (5)								
CLK	IN				W5	✓	34	LVCMOS33*
CLKoutput	OUT				V3	✓	34	LVCMOS33*
DONE	OUT				L1	✓	35	LVCMOS33*
RESET	IN				T1	✓	34	LVCMOS33*
START	IN				R2	✓	34	LVCMOS33*

Results

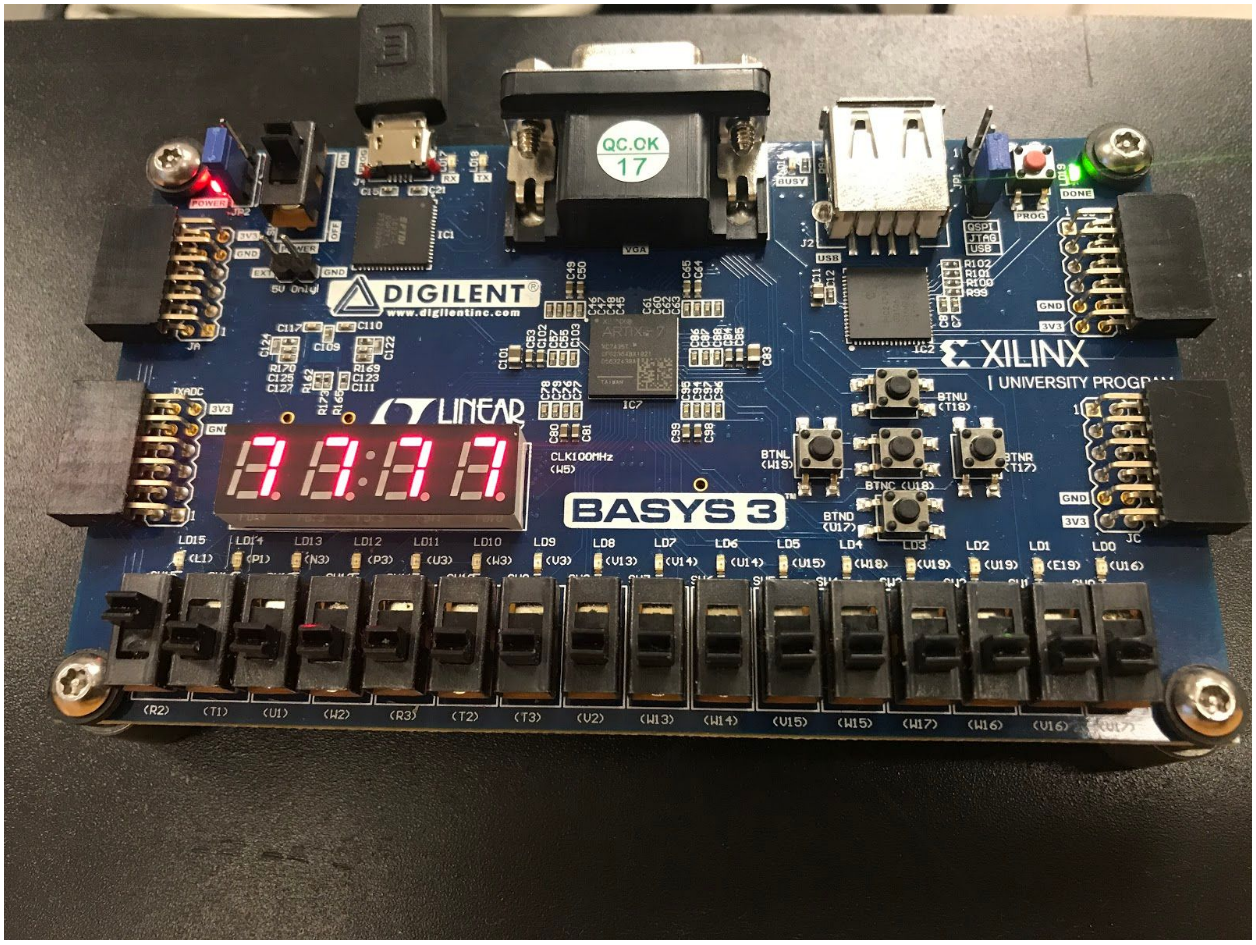


Figure 2.1) High Clock. The count is at 7

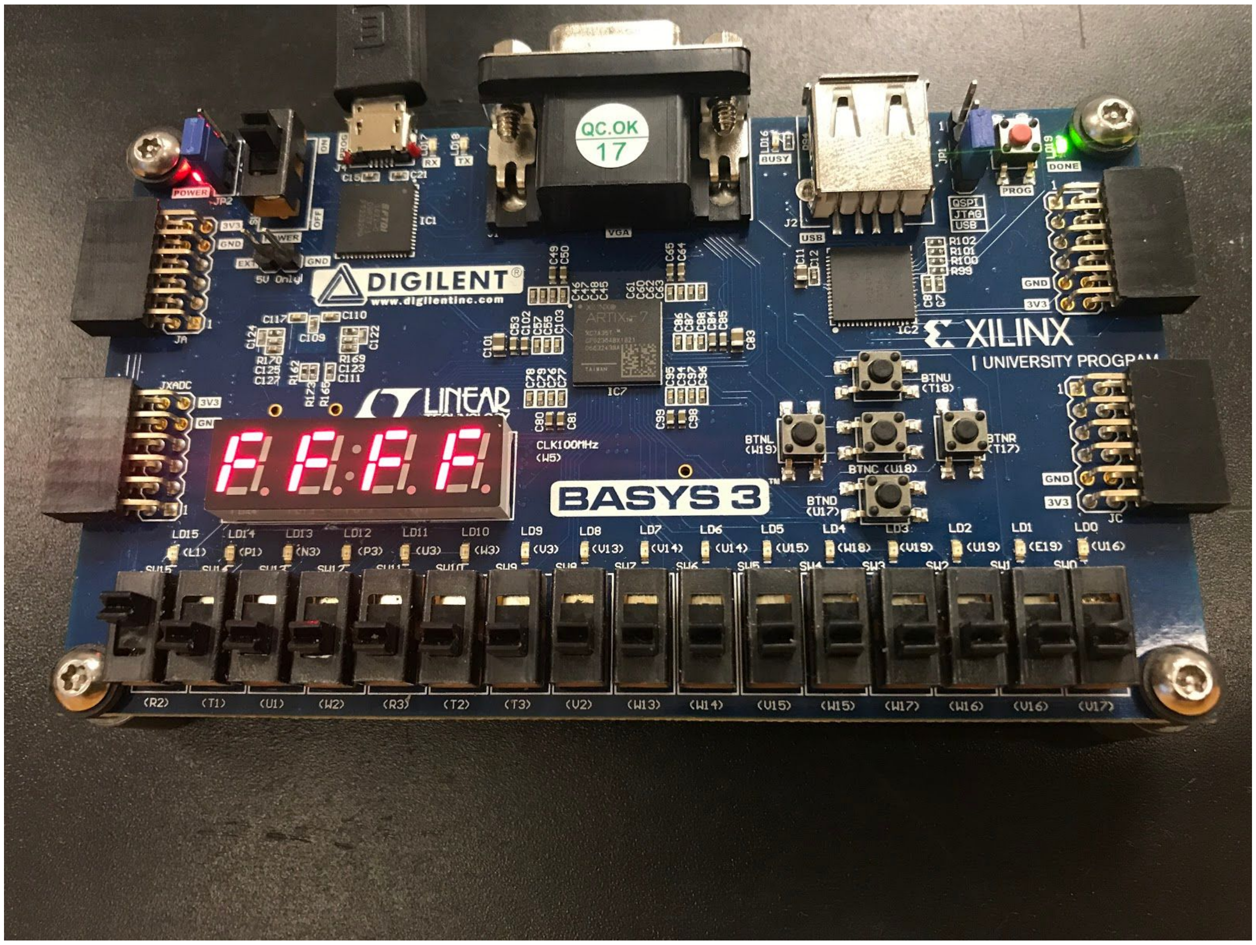


Figure 2.1) High Clock. The count is at F or 15 in HEX

Conclusion:

In conclusion, this continues to strengthen our use of components and understanding of\ FSM, and how they can be used as a control unit (in this case the counter). The previous lab was very difficult and gave us a cushion for this lab and labs to come, when it comes to implementing FSMs.

PART 1 VHDL CODE

CLOCK DIVIDER

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;


entity Clockdivider is
Port ( CLK : in  STD_LOGIC;
       RESET : in  STD_LOGIC;
       SlowClock,ledO : out  STD_LOGIC);
end Clockdivider;

architecture behavioral of Clockdivider is
signal slowSig: std_logic;
begin
   process
    variable cnt :    std_logic_vector(26 downto 0):= "000000000000000000000000000";
      begin                           -- calculations
       wait until ((CLK'EVENT) AND (CLK = '1'));
       if (RESET = '1') then
           cnt := "000000000000000000000000000";
        else
         cnt := cnt + 1; -- count to 26
        end if;

     SlowClock <= cnt(26);
     slowSig<=cnt(26);

    if(slowSig='1')then
     ledO<='1';
    else
     ledO<='0';
    end if;

  end process;
end Behavioral;
```

DISPLAY

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity SevenSegmentDisplay is
 Port (input: in std_logic_vector (3 downto 0);
       output:out std_logic_vector(6 downto 0));-- lsb
end SevenSegmentDisplay;

architecture Behavioral of SevenSegmentDisplay is

begin
process(input)
begin
case input is
when "0000" =>
   output <= "1000000"; --0
when "0001" =>
   output <= "1111001"; --1
when "0010" =>
   output <= "0100100";--2
when "0011" =>
   output <= "0110000"; --3
when "0100" =>
   output <= "0011001"; --4
when "0101" =>
   output <= "0010010";--5
when "0110" =>
   output <= "0000010";--6
when "0111" =>
   output <= "1111000";--7
when "1000" =>
   output <= "0000000";--8
when "1001" =>
   output <= "0010000";--9
when "1010" =>
    output <= "0100000";--10 A
when "1011" =>
    output <= "0000011";--11 B
when "1100" =>
    output <= "1000110";--12 C
```

```
when "1101" =>
      output <= "0100001";--13 D
when "1110" =>
      output <= "0000110";--14 E
when "1111" =>
      output <= "0001110";--15 F
end case;
end process;

end Behavioral;
```

COUNTER

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity counter is
  Port (CLK,R,ENABLE,CLEAR: in std_logic;
        COUNTER :out std_logic_vector(3 downto 0));
end counter;

architecture Behavioral of counter is
signal tmp: std_logic_vector(3 downto 0);--temporary signal for counter

begin
process(R,CLK)
begin
    if(R='1')then
    tmp<="0000";--reset the count
    elsif(CLK'event and (CLK='1'))then
        if(ENABLE='1') then
        tmp <= tmp + "0001";--start the count
        end if;
    end if;
    COUNTER <= tmp;
    end process;
end Behavioral;
```

FSM

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity FSM is
 Port (CLK,R,START: in std_logic;
        STATE : out std_logic_vector(3 downto 0);
     DONE,ENABLE,CLEAR: out std_logic);
end FSM;

architecture Behavioral of fsm is
signal CS,NS: std_logic;
signal STATEDISPLAYtmp : std_logic_vector(3 downto 0);
    begin
    process(CLK, R)
    begin
        if(R='1') then
        CS<='0';
        CLEAR<='1';
        --STATEDISPLAYtmp <= "0000";
        elsif(CLK' event and (CLK='1'))then
            CS<=NS;
            STATEDISPLAYtmp<= STATEDISPLAYtmp;
      end if;
      end process;
    process(CS,START)
    begin
        case CS is
        when '0' => --State 0 handles both the DONE state and the INITIAL state of the FSM
                  if(START ='1') then
```

```
                    NS<='1';
                    ENABLE<='1';
                    DONE<='0';
                    STATEDISPLAYtmp <= "0001";
                elsif(START='0')then
                    ENABLE<='0';
                    NS<= '0';
                    DONE<='1';
                    STATEDISPLAYtmp <= "0000";
                end if;
        when '1' => --count state
               if(START ='1') then
                    NS<='1';
                    ENABLE<='1';
                    DONE<='0';
                    STATEDISPLAYtmp <= "0000";
               elsif(START='0')then
                    ENABLE<='0';
                    NS<= '0';
                    DONE<='1';
                    STATEDISPLAYtmp <= "0001";
               end if;
        end case;
    end process;
        STATE<= STATEDISPLAYtmp;
end Behavioral;
```

TOP LEVEL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity TopLevel is
 Port (CLK,START,RESET: in std_logic;
       CLKoutput,DONE: out std_logic;
       DISPLAY: out std_logic_vector(6 downto 0) );
end TopLevel;

architecture Behavioral of TopLevel is

signal tmpEnable,tmpCLEAR,tmpCLK: std_logic;
signal tmpDisplay: std_logic_vector( 3 downto 0);
```

```
    component FSM Port (CLK,R,START: in std_logic;
                   DONE,ENABLE,CLEAR: out std_logic);
    end component;
    --ClockDivider Component Declaration
    component ClockDivider Port ( clk : in  STD_LOGIC;
                              reset : in  STD_LOGIC;
                              SlowClock,ledO : out  STD_LOGIC);
    end component;
    --Counter Component Declaration
    component Counter Port (CLK,R,ENABLE,CLEAR: in std_logic;
                      COUNTER :out std_logic_vector(3 downto 0));
    end component;
    --SevenSegmentDisplay Component Declaration
    component SevenSegmentDisplay  Port (input: in std_logic_vector (3 downto 0);
                      output:out std_logic_vector(6 downto 0));
    end component;

        begin
        --Component Instantiation
        Clock: ClockDivider port
map(CLK=>CLK,RESET=>RESET,SlowClock=>tmpCLK,ledO=>CLKoutput);
        Machine: FSM port
map(CLK=>tmpCLK,R=>RESET,START=>START,DONE=>DONE,ENABLE=>tmpEnable,CLE
AR=>tmpCLEAR);
        COUNT : COUNTER port
map(CLK=>tmpClk,R=>RESET,ENABLE=>tmpEnable,CLEAR=>tmpCLEAR,COUNTER=>tmp
Display);
        SevSeg: SevenSegmentDisplay port map(input=>tmpDisplay,output=>DISPLAY);

end Behavioral;
```

CONSTRAINTS

```
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports CLK]
set_property IOSTANDARD LVCMOS33 [get_ports CLKoutput]
set_property IOSTANDARD LVCMOS33 [get_ports DONE]
set_property IOSTANDARD LVCMOS33 [get_ports RESET]
set_property IOSTANDARD LVCMOS33 [get_ports START]
set_property PACKAGE_PIN U7 [get_ports {DISPLAY[6]}]
set_property PACKAGE_PIN V5 [get_ports {DISPLAY[5]}]
set_property PACKAGE_PIN U5 [get_ports {DISPLAY[4]}]
set_property PACKAGE_PIN V8 [get_ports {DISPLAY[3]}]
set_property PACKAGE_PIN U8 [get_ports {DISPLAY[2]}]
set_property PACKAGE_PIN W6 [get_ports {DISPLAY[1]}]
set_property PACKAGE_PIN W7 [get_ports {DISPLAY[0]}]
set_property PACKAGE_PIN L1 [get_ports DONE]
set_property PACKAGE_PIN T1 [get_ports RESET]
set_property PACKAGE_PIN W5 [get_ports CLK]
set_property PACKAGE_PIN V3 [get_ports CLKoutput]
set_property PACKAGE_PIN R2 [get_ports START]
```

PART 2 VHDL CODE

CLOCK DIVIDER

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;


entity Clockdivider is
Port ( CLK : in  STD_LOGIC;
          RESET : in  STD_LOGIC;
          SlowClock,ledO : out  STD_LOGIC);
end Clockdivider;

architecture behavioral of Clockdivider is
signal slowSig: std_logic;
begin
     process
      variable cnt :    std_logic_vector(26 downto 0):= "000000000000000000000000000";
         begin                        -- calculations
           wait until ((CLK'EVENT) AND (CLK = '1'));
           if (RESET = '1') then
                cnt := "000000000000000000000000000";
            else
              cnt := cnt + 1; -- count to 26
            end if;

        SlowClock <= cnt(26);
        slowSig<=cnt(26);

      if(slowSig='1')then
        ledO<='1';
      else
        ledO<='0';
      end if;

   end process;
end Behavioral;
```

Display VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity SevenSegmentDisplay is
 Port (input: in std_logic_vector (3 downto 0);
       output:out std_logic_vector(6 downto 0));-- lsb
end SevenSegmentDisplay;

architecture Behavioral of SevenSegmentDisplay is

begin
process(input)
begin
case input is
when "0000" =>
    output <= "1000000"; --0
when "0001" =>
    output <= "1111001"; --1
when "0010" =>
    output <= "0100100";--2
when "0011" =>
    output <= "0110000"; --3
when "0100" =>
    output <= "0011001"; --4
when "0101" =>
    output <= "0010010";--5
```

```
when "0110" =>
    output <= "0000010";--6
when "0111" =>
    output <= "1111000";--7
when "1000" =>
    output <= "0000000";--8
when "1001" =>
    output <= "0010000";--9
when "1010" =>
     output <= "0100000";--10 A
when "1011" =>
     output <= "0000011";--11 B
when "1100" =>
     output <= "1000110";--12 C
when "1101" =>
      output <= "0100001";--13 D
when "1110" =>
      output <= "0000110";--14 E
when "1111" =>
      output <= "0001110";--15 F
end case;
end process;

end Behavioral;
```

Counter VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity counter is
  Port (CLK,R,ENABLE,CLEAR: in std_logic;
        COUNTER :out std_logic_vector(3 downto 0));
end counter;

architecture Behavioral of counter is
signal tmp: std_logic_vector(3 downto 0);--temporary signal for counter
```

```
begin
process(R,CLK)
begin
    if(R='1')then
    tmp<="0000";--reset the count
    elsif(CLK'event and (CLK='1'))then
        if(ENABLE='1') then
        tmp <= tmp + "0001";--start the count
        end if;
    end if;
    COUNTER <= tmp;
    end process;
end Behavioral;
```

FSM VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity FSM is
 Port (CLK,R,START: in std_logic;
        DONE,ENABLE,CLEAR: out std_logic);
end FSM;

architecture Behavioral of fsm is
signal CS,NS: std_logic;
    begin
    process(CLK, R)
    begin
        if(R='1') then
        CS<='0';
        CLEAR<='1';
        elsif(CLK' event and (CLK='1'))then
        CS<=NS;
      end if;
      end process;
    process(CS,START)
    begin
        case CS is
        when '0' => --State 0 handles both the DONE state and the INITIAL state of the FSM
                  if(START ='1') then
                   NS<='1';
                   ENABLE<='1';
                   DONE<='0';
                   elsif(START='0')then
                   ENABLE<='0';
                   NS<= '0';
                   DONE<='1';
                   end if;
        when '1' => --count state
                  if(START ='1') then
                  NS<='1';
                  ENABLE<='1';
                  DONE<='0';
                 elsif(START='0')then
                  ENABLE<='0';
                  NS<= '0';
                  DONE<='1';
                 end if;
        end case;
    end process;
end Behavioral;
```

TopLevel VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity TopLevel is
 Port (CLK,START,RESET: in std_logic;
       CLKoutput,DONE: out std_logic;
       DISPLAY: out std_logic_vector(6 downto 0) );
end TopLevel;
```

```
architecture Behavioral of TopLevel is

signal tmpEnable,tmpCLEAR,tmpCLK: std_logic;
signal tmpDisplay: std_logic_vector( 3 downto 0);

   component FSM Port (CLK,R,START: in std_logic;
                  DONE,ENABLE,CLEAR: out std_logic);
   end component;
   --ClockDivider Component Declaration
   component ClockDivider Port ( clk : in  STD_LOGIC;
                           reset : in  STD_LOGIC;
                           SlowClock,ledO : out  STD_LOGIC);
   end component;
   --Counter Component Declaration
   component Counter Port (CLK,R,ENABLE,CLEAR: in std_logic;
                     COUNTER :out std_logic_vector(3 downto 0));
   end component;
   --SevenSegmentDisplay Component Declaration
   component SevenSegmentDisplay  Port (input: in std_logic_vector (3 downto 0);
                     output:out std_logic_vector(6 downto 0));
   end component;

      begin
      --Component Instantiation
      Clock: ClockDivider port
map(CLK=>CLK,RESET=>RESET,SlowClock=>tmpCLK,ledO=>CLKoutput);
      Machine: FSM port
map(CLK=>tmpCLK,R=>RESET,START=>START,DONE=>DONE,ENABLE=>tmpEnable,CLE
AR=>tmpCLEAR);
      COUNT : COUNTER port
map(CLK=>tmpClk,R=>RESET,ENABLE=>tmpEnable,CLEAR=>tmpCLEAR,COUNTER=>tmp
Display);
      SevSeg: SevenSegmentDisplay port map(input=>tmpDisplay,output=>DISPLAY);

end Behavioral;
```

CONSTRAINTS

```
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {DISPLAY[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports CLK]
set_property IOSTANDARD LVCMOS33 [get_ports CLKoutput]
set_property IOSTANDARD LVCMOS33 [get_ports DONE]
set_property IOSTANDARD LVCMOS33 [get_ports RESET]
set_property IOSTANDARD LVCMOS33 [get_ports START]
set_property PACKAGE_PIN U7 [get_ports {DISPLAY[6]}]
set_property PACKAGE_PIN V5 [get_ports {DISPLAY[5]}]
set_property PACKAGE_PIN U5 [get_ports {DISPLAY[4]}]
set_property PACKAGE_PIN V8 [get_ports {DISPLAY[3]}]
set_property PACKAGE_PIN U8 [get_ports {DISPLAY[2]}]
set_property PACKAGE_PIN W6 [get_ports {DISPLAY[1]}]
set_property PACKAGE_PIN W7 [get_ports {DISPLAY[0]}]
set_property PACKAGE_PIN W5 [get_ports CLK]
set_property PACKAGE_PIN L1 [get_ports DONE]
set_property PACKAGE_PIN T1 [get_ports RESET]
set_property PACKAGE_PIN R2 [get_ports START]
set_property PACKAGE_PIN V3 [get_ports CLKoutput]
```