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)

May 7, 2019

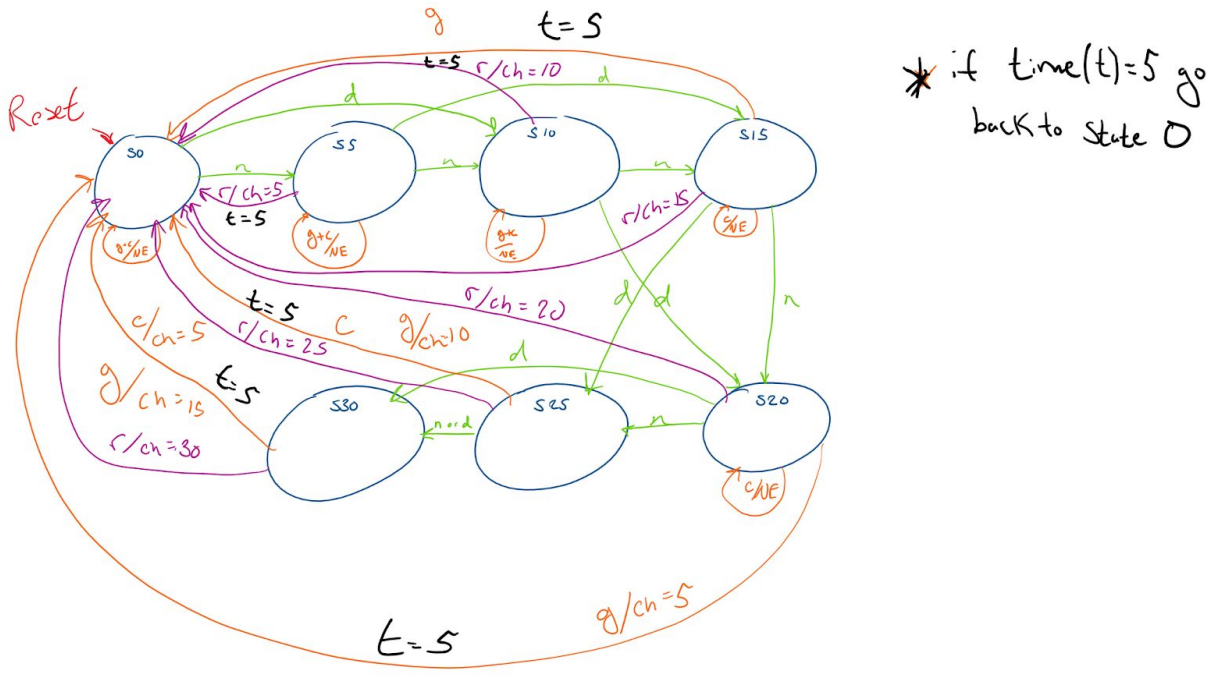
Lab #11

Introduction: In Lab 11 we continue to cover Vending Machines. The Vending Machine we will design will dispense either gum (0.15) or candy (0.25). It will also give the user a option to refund all the money they have entered. The Vending Machine will accept nickels(0.05) and dimes(0.10). In Lab 11 we build upon Lab 10 further by adding a countdown timer that keeps the money that the user has entered after a 5 seconds. On the FPGA board there will be a number of leds as well as a seven segment display to give a visual representation of the amount of money that has been inserted by the user, the number of gum or candy that has been added, or whether the user has enough funds.

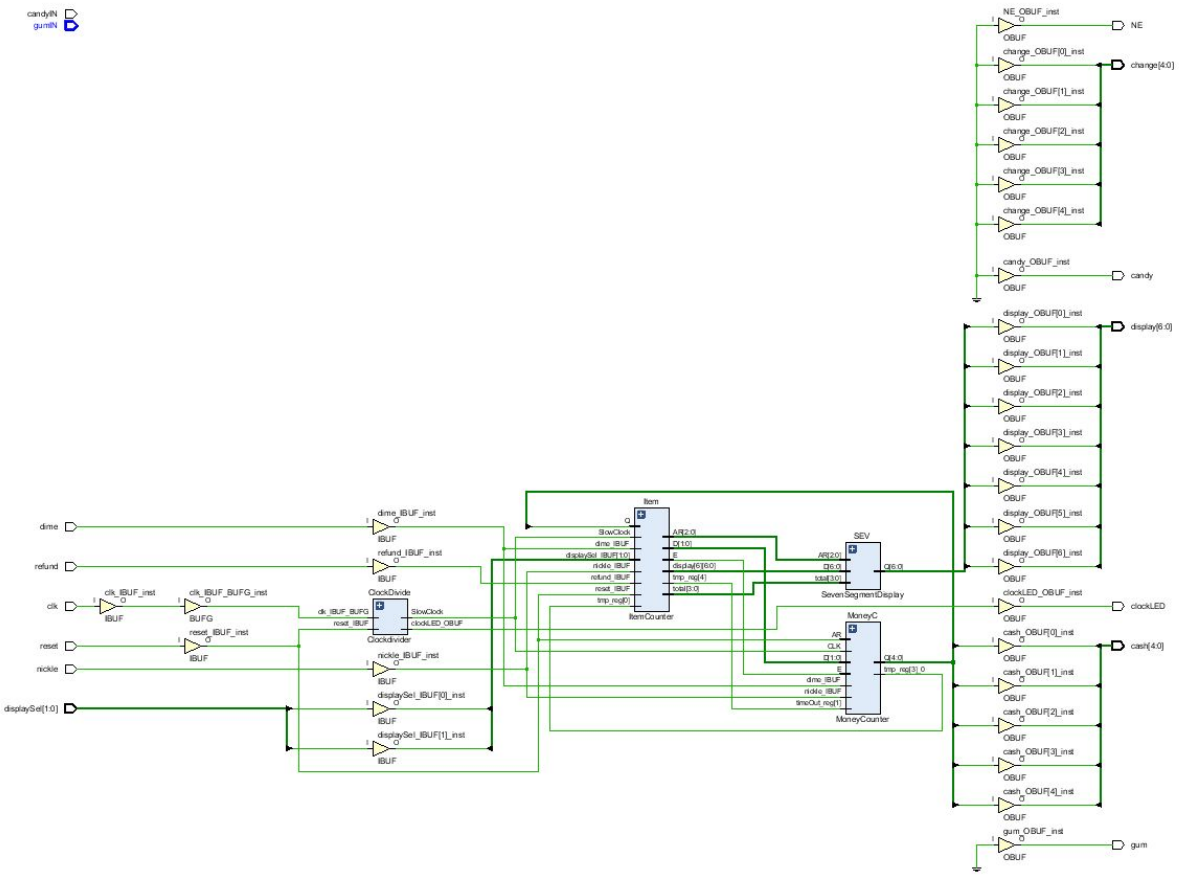
Background: Vending Machines are found in everyday life, from schools, office buildings, or any place where people need a quick snack. In this lab we will be using a Finite State Machine design a Vending Machine which will simulate a real-life one.

Demo: https://youtu.be/e9JhHbQZsiA

State Diagram



# Schematic


candyIN
gumIN
NE_OBUF_inst
NE
change_OBUF[0]_inst
change_OBUF[1]_inst
change_OBUF[2]_inst
change_OBUF[3]_inst
change_OBUF[4]_inst
change[4:0]
candy_OBUF_inst
candy
display_OBUF[0]_inst
display_OBUF[1]_inst
display_OBUF[2]_inst
display_OBUF[3]_inst
display_OBUF[4]_inst
display_OBUF[5]_inst
display_OBUF[6]_inst
display[6:0]
clockLED_OBUF_inst
clockLED
cash_OBUF[0]_inst
cash_OBUF[1]_inst
cash_OBUF[2]_inst
cash_OBUF[3]_inst
cash_OBUF[4]_inst
cash[4:0]
gum_OBUF_inst
gum
OBUF
dime
refund
clk
reset
nickle
displaySel[1:0]
dime_IBUF_inst
refund_IBUF_inst
clk_IBUF_inst
clk_IBUF_BUFG_inst
reset_IBUF_inst
nickle_IBUF_inst
displaySel_IBUF[0]_inst
displaySel_IBUF[1]_inst
IBUF
IBUFG
ClockDivide
ClockDivider
Item
ItemCounter
SEV
SevenSegmentDisplay
MoneyC
MoneyCounter

PIN ASSIGNMENT

| | | | | | |
|---|---|---|---|---|---|
| All ports (30) | | | | | |
| cash (5) | OUT | | ✓ | (Multiple) | LVCMOS33* |
| cash[4] | OUT | L1 | ✓ | 35 | LVCMOS33* |
| cash[3] | OUT | P1 | ✓ | 35 | LVCMOS33* |
| cash[2] | OUT | N3 | ✓ | 35 | LVCMOS33* |
| cash[1] | OUT | P3 | ✓ | 35 | LVCMOS33* |
| cash[0] | OUT | U3 | ✓ | 34 | LVCMOS33* |
| change (5) | OUT | | ✓ | 14 | LVCMOS33* |
| change[4] | OUT | W18 | ✓ | 14 | LVCMOS33* |
| change[3] | OUT | V19 | ✓ | 14 | LVCMOS33* |
| change[2] | OUT | U19 | ✓ | 14 | LVCMOS33* |
| change[1] | OUT | E19 | ✓ | 14 | LVCMOS33* |
| change[0] | OUT | U16 | ✓ | 14 | LVCMOS33* |
| display (7) | OUT | | ✓ | 34 | LVCMOS33* |
| display[6] | OUT | W7 | ✓ | 34 | LVCMOS33* |
| display[5] | OUT | W6 | ✓ | 34 | LVCMOS33* |
| display[4] | OUT | U8 | ✓ | 34 | LVCMOS33* |
| display[3] | OUT | V8 | ✓ | 34 | LVCMOS33* |
| display[2] | OUT | U5 | ✓ | 34 | LVCMOS33* |
| display[1] | OUT | V5 | ✓ | 34 | LVCMOS33* |
| display[0] | OUT | U7 | ✓ | 34 | LVCMOS33* |
| displaySel (2) | IN | | ✓ | 14 | LVCMOS33* |
| displaySel[1] | IN | V16 | ✓ | 14 | LVCMOS33* |
| displaySel[0] | IN | V17 | ✓ | 14 | LVCMOS33* |
| Scalar ports (11) | | | | | |
| candy | OUT | V14 | ✓ | 14 | LVCMOS33* |
| candyIN | IN | R2 | ✓ | 34 | LVCMOS33* |
| clk | IN | W5 | ✓ | 34 | LVCMOS33* |
| clockLED | OUT | V13 | ✓ | 14 | LVCMOS33* |
| dime | IN | R3 | ✓ | 34 | LVCMOS33* |
| gum | OUT | U14 | ✓ | 14 | LVCMOS33* |
| gumIN | IN | T1 | ✓ | 34 | LVCMOS33* |
| NE | OUT | W3 | ✓ | 34 | LVCMOS33* |
| nickle | IN | W2 | ✓ | 34 | LVCMOS33* |
| refund | IN | T2 | ✓ | 34 | LVCMOS33* |
| reset | IN | U1 | ✓ | 34 | LVCMOS33* |

Results

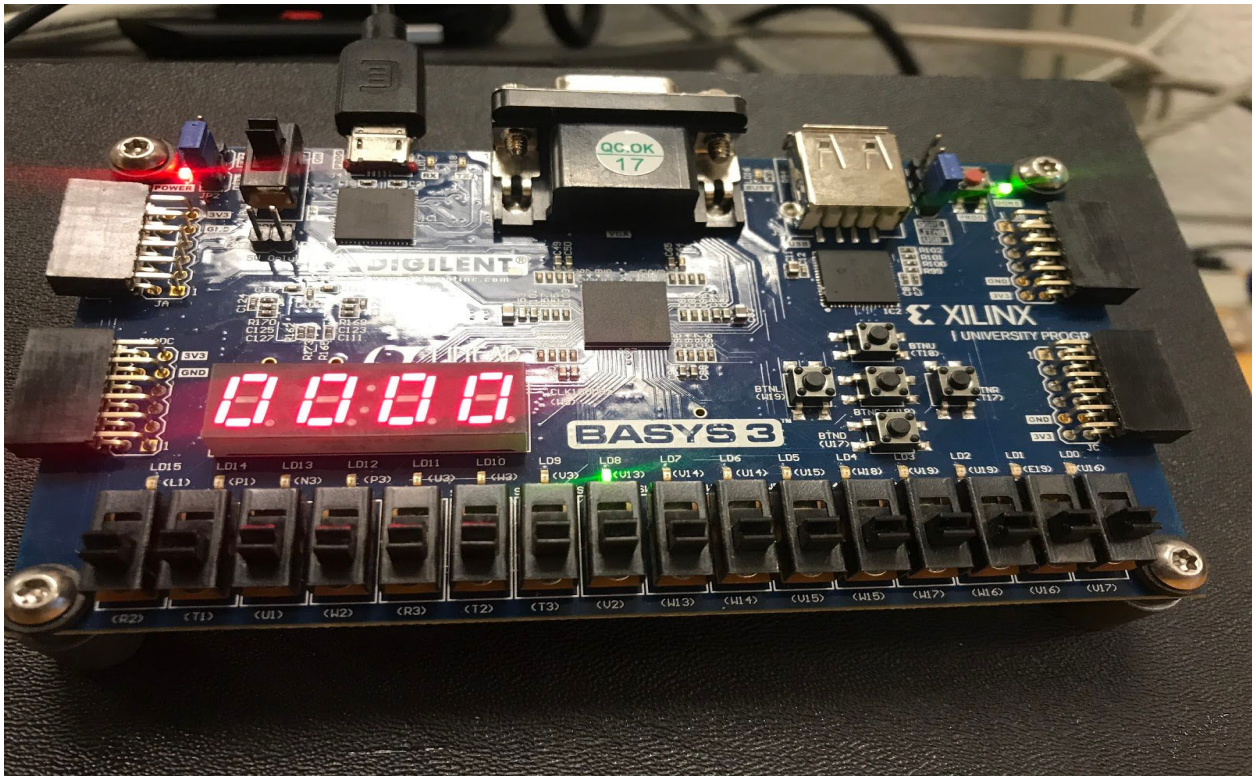



Figure 1.1) System initial state, no money has been added. Nothing selected

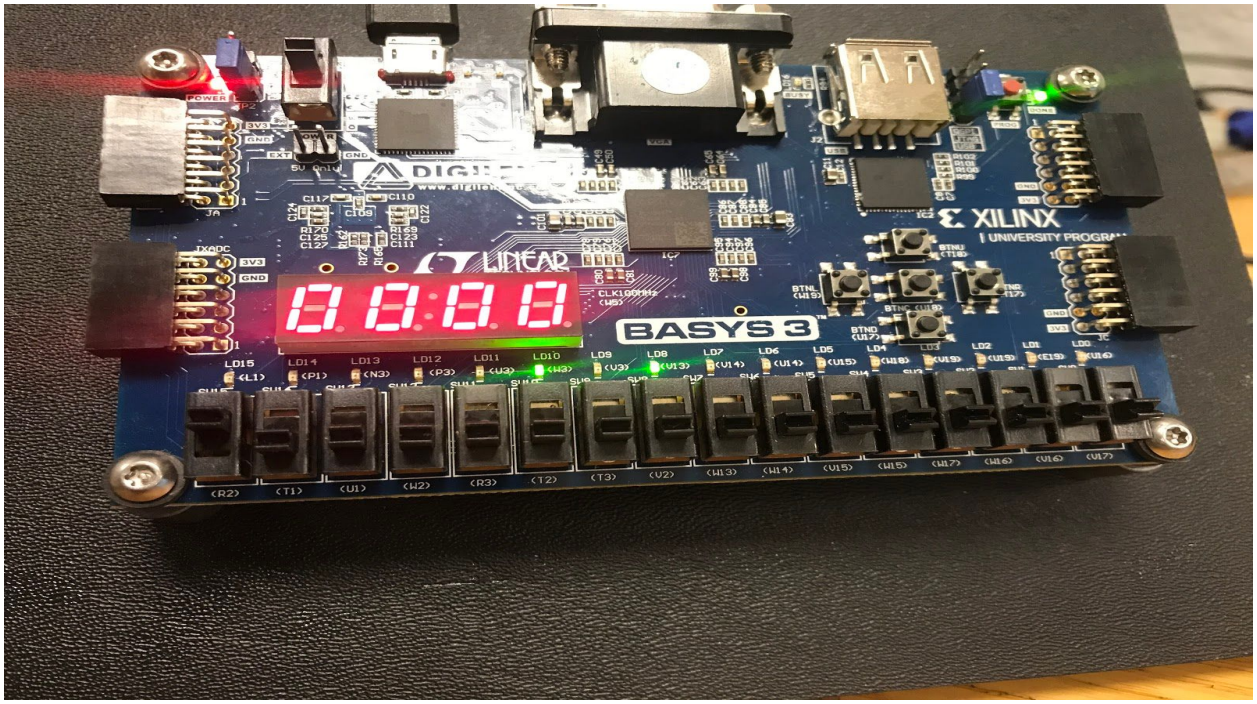



Figure 1.2)System initial state, no money has been added. Candy Selected, shows not enough money for candy

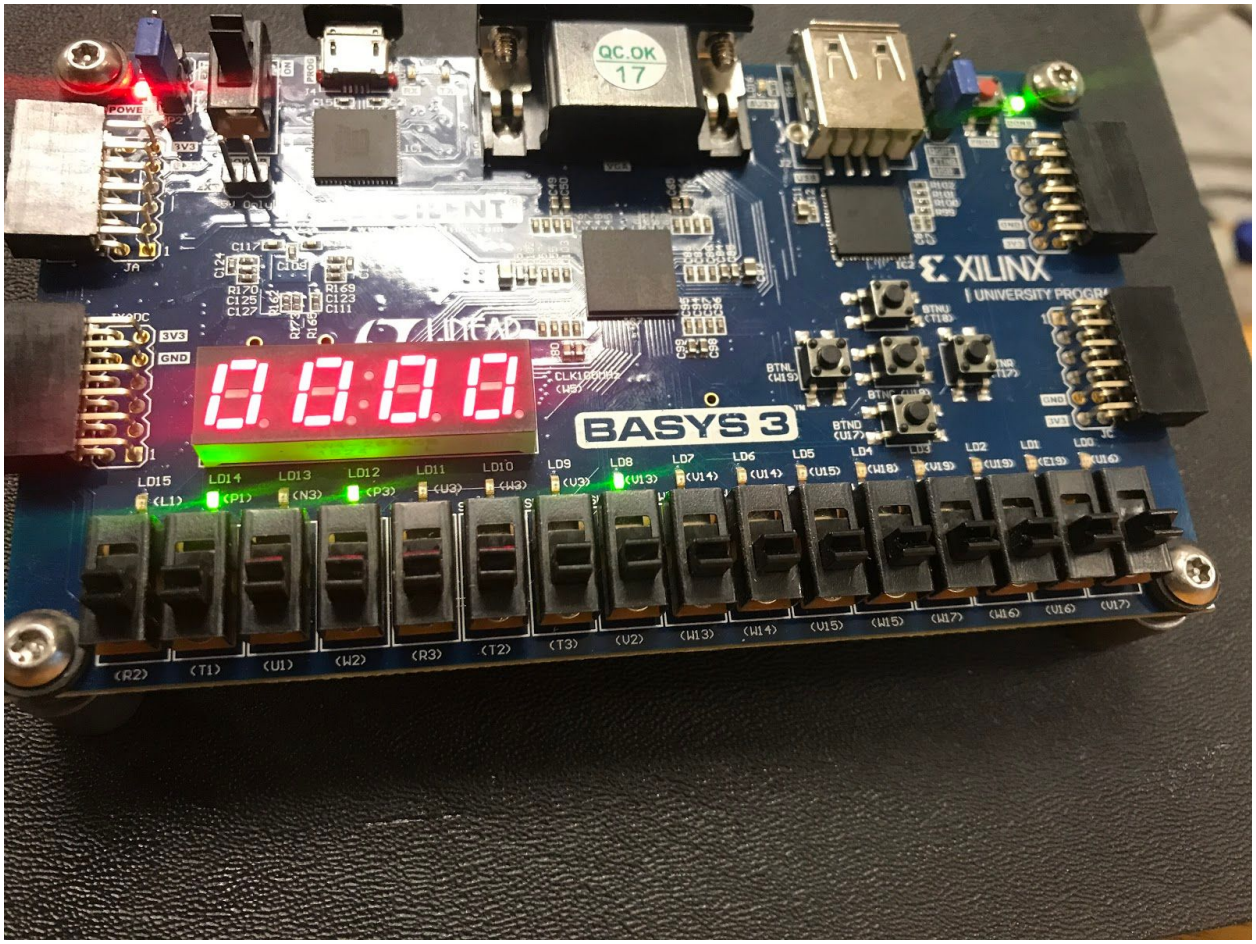


Figure 1.3) dime has been added

Conclusion: This lab gave us more hands on experience with FPGA boards and helped us to understand more complex logic systems including a timer. The amount decisions that can be made in each state at first seemed to be complex but as we progressed through the design it became much more clear. This lab put to the test our understanding of everything that we have designed before this, building on top of our understanding of finite state machines. A fitting final lab.

# Appendices

## VHDL

### SevenSegmentDisplay

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity SevenSegmentDisplay is
 Port (ip : in std_logic_vector(3 downto 0);
       display : out std_logic_vector(6 downto 0));
end SevenSegmentDisplay;

architecture Behavioral of SevenSegmentDisplay is

begin
   process(ip)
      begin
         if(ip = "0000") then display <= "0000001";--0
         elsif(ip = "0001") then display <= "1001111";
         elsif(ip = "0010") then display <= "0010010";
         elsif(ip = "0011") then display <= "0000110";
         elsif(ip = "0100") then display <= "1001100";
         elsif(ip = "0101") then display <= "0100100";
         elsif(ip = "0110") then display <= "0100000";
         elsif(ip = "0111") then display <= "0001111";
         elsif(ip = "1000") then display <= "0000000";
         elsif(ip = "1001") then display <= "0001100";
         elsif(ip = "1010") then display <= "0001000";
         elsif(ip = "1011") then display <= "1000110";
         elsif(ip = "1100") then display <= "0110001";
         elsif(ip = "1101") then display <= "1000110";
         elsif(ip = "1110") then display <= "0110000";
         elsif(ip = "1111") then display <= "0110000";--F
         end if;

   end process;

end Behavioral;
```

### ClockDivider

```
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):= "00000000000000000000000000";
        begin                          -- calculations
          wait until ((clk'EVENT) AND (clk = '1'));
          if (reset = '1') then
               cnt := "00000000000000000000000000";
           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;
```

TopLevel

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity TopLevel is
  Port (clk,reset,gumIN,candyIN,refund,nickle,dime: in std_logic;
       displaySel:in std_logic_vector(1 downto 0);--what needs to be displayed
       gum,candy:out std_logic;
       display:out std_logic_vector(6 downto 0);
       cash:out std_logic_vector(4 downto 0);
       change: out std_logic_vector(4 downto 0);
       clockLED,NE: out std_logic);
end TopLevel;

architecture Behavioral of TopLevel is
   signal nickleSig,dimeSig,gumOut,candyOut,clockOut,refundtmp,tmpTime: std_logic;
   signal total,outTimetmp: std_logic_vector(3 downto 0);
```

```
component counddown port (clk,reset,coin:in std_logic;
                    countTime:out std_logic_vector(3 downto 0));
end component;

component clockDivider port( clk : in  STD_LOGIC;
                     reset : in  STD_LOGIC;
                     SlowClock,ledO : out  STD_LOGIC);
end component;

component VM   port (clk,reset,gumSel,candySel,refund,nickle,dime: in std_logic;
                timeIN: in std_logic_vector(3 downto 0);
                nickleCount,dimeCount,NotEnough,gum,candy,startTime:out std_logic;
                change:out std_logic_vector(4 downto 0));
end component;

component MoneyCounter port (clk,reset,nickleIN,dimeIN,gumOut,candyOut,refund:in std_logic;
                     timeIN:in std_logic_vector(3 downto 0);
                     moneyCount:out std_logic_vector(4 downto 0));
end component;

component ItemCounter  Port (clk,reset,candyIN,gumIN,startTimer:in std_logic;
                     sel : in std_logic_vector(1 downto 0);--chose which total you want to see displayed
                     timeOut: out std_logic_vector(3 downto 0);
                     total : out std_logic_vector(3 downto 0));
end component;

component SevenSegmentDisplay port ( ip : in std_logic_vector(3 downto 0);
                          display : out std_logic_vector(6 downto 0));
end component;




begin

refundtmp<=refund;
gum<=gumOut;
candy<=candyOut;

ClockDivide: ClockDivider port map(clk=>clk,
                          reset=>reset,
                          SlowClock=>clockOut,
                          ledO=>clockLED);
Machine: VM port map( clk=>clockOut,
                reset=>reset,
                gumSel=> gumIN,
                candySel=>candyIN,
```

```
                refund=>refundtmp,
                nickle=>nickle,
                dime=>dime,
                timeIN=>outTimetmp,
                nickleCount=>nickleSig,
                dimeCount=>dimeSig,
                NotEnough=>NE,
                gum=>gumOut,
                candy=>candyOut,
                startTime=>tmpTime,
                change=>change);


MoneyC: MoneyCounter port map(clk=>clockOut,
                        reset=>reset,
                        nickleIN=>nickleSig,
                        dimeIN=>dimeSig,
                        gumOut=>gumOut,
                        candyOut=>candyOut,
                        refund =>refundtmp,
                        timeIN=>outTimetmp,
                        moneyCount=>cash);

Item: ItemCounter port map(clk=>clockOut,
                      reset=>reset,
                      candyIN=>candyOut,
                      gumIN=>gumOut,
                      startTimer=>tmpTime,
                      sel=>displaySel,
                      timeOut=>outTimetmp,
                      total=>total);

SEV:SevenSegmentDisplay port map(ip=>total,display=>display);


end Behavioral;
```

Vending Machine

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity VM is
  Port (clk,reset,gumSel,candySel,refund,nickle,dime: in std_logic;
        timeIN: in std_logic_vector(3 downto 0);
        nickleCount,dimeCount,NotEnough,gum,candy,startTime:out std_logic;
        change:out std_logic_vector(4 downto 0));
```

```
end VM;

architecture Behavioral of VM is
   type state_type is (S0,S5,S10,S15,S20,S25,S30);
   signal cs,ns: state_type;
   begin
   process(clk,reset)
      begin
         if (reset='1')then
            cs<=S0;
         elsif(clk'event AND clk='1')then
           cs<=ns;
         end if;
   end process;

   process(cs,nickle,dime,refund,gumSel,candySel)
   begin
      --initilization
      startTime<='1';
      NotEnough<='0';
      gum<='0';
      candy<='0';
      change<="00000";
      nickleCount<=nickle;
      dimeCount<=dime;

   case cs is
   when S0=>
          if(timeIN< "0101" and (refund='1' or gumSel='1' or candySel='1' or nickle='1' or dime='1'))then
               if(refund<='1')then
                  change<="00000";
                  startTime<='0';
                  ns<=S0;
               elsif(gumSel='1' or candySel='1')then
                 NotEnough<='1';
                 startTime<='0';
                 ns<=S0;
               elsif(nickle='1')then
                    startTime<='0';
                    ns<=S5;
               elsif(dime='1')then
                    startTime<='0';
                    ns<=S10;
                end if;
           elsif(timeIN>="0101" and(refund='0' or gumSel='0' or candySel='0' or nickle='0' or dime='0')) then
               ns<=s0;
          end if;
               startTime<='1';
```

```
when S5=>
  if(timeIN< "0101" and (refund='1' or gumSel='1' or candySel='1' or nickle='1' or dime='1'))then
    if(refund<='1')then
       change<="00101";
       startTime<='0';
       ns<=S0;
    elsif(gumSel='1' or candySel='1')then
     NotEnough<='1';
     startTime<='0';
     ns<=S5;
    elsif(nickle='1')then
         startTime<='0';
         ns<=S10;
    elsif(dime='1')then
         startTime<='0';
         ns<=S15;
     end if;
 elsif(timeIN>="0101" and(refund='0' or gumSel='0' or candySel='0' or nickle='0' or dime='0')) then
    ns<=s0;
    startTime<='0';
end if;
     startTime<='1';
when S10=>
  if(timeIN< "0101" and (refund='1' or gumSel='1' or candySel='1' or nickle='1' or dime='1'))then
     if(refund<='1')then
        change<="01010";
        startTime<='0';
        ns<=S0;
     elsif(gumSel='1' or candySel='1')then
      NotEnough<='1';
      startTime<='0';
      ns<=S10;
     elsif(nickle='1')then
          startTime<='0';
          ns<=S15;
     elsif(dime='1')then
          startTime<='0';
          ns<=S20;
      end if;
  elsif(timeIN>="0101" and(refund='0' or gumSel='0' or candySel='0' or nickle='0' or dime='0')) then
     ns<=s0;
     startTime<='0';
  end if;
      startTime<='1';
when S15=>
   if(timeIN< "0101" and (refund='1' or gumSel='1' or candySel='1' or nickle='1' or dime='1'))then
      if(refund<='1')then
         change<="01111";
```

```
            startTime<='0';
            ns<=S0;
        elsif(candySel='1')then
            NotEnough<='1';
            startTime<='0';
            ns<=S15;
        elsif(gumSel='1')then
            gum<='1';
            startTime<='0';
            ns<=S0;
        elsif(nickle='1')then
             startTime<='0';
             ns<=S20;
        elsif(dime='1')then
             startTime<='0';
             ns<=S25;
         end if;
    elsif(timeIN>="0101" and(refund='0' or gumSel='0' or candySel='0' or nickle='0' or dime='0')) then
        ns<=s0;
        startTime<='0';
    end if;
        startTime<='1';
when S20=>
    if(timeIN< "0101" and (refund='1' or gumSel='1' or candySel='1' or nickle='1' or dime='1'))then
        if(refund<='1')then
            change<="10100";
            startTime<='0';
            ns<=S0;
        elsif(candySel='1')then
            NotEnough<='1';
            startTime<='0';
            ns<=S20;
        elsif(gumSel='1')then
            gum<='1';
            startTime<='0';
            ns<=S0;
            change<="00101";
        elsif(nickle='1')then
             startTime<='0';
             ns<=S25;
        elsif(dime='1')then
             startTime<='0';
             ns<=S30;
         end if;
    elsif(timeIN>="0101" and(refund='0' or gumSel='0' or candySel='0' or nickle='0' or dime='0')) then
        ns<=s0;
        startTime<='0';
    end if;
```

```
        startTime<='1';
when S25=>
    if(timeIN< "0101" and (refund='1' or gumSel='1' or candySel='1' or nickle='1' or dime='1'))then
        if(refund<='1')then
            change<="11001";
            startTime<='0';
            ns<=S0;
        elsif(candySel='1')then
            candy<='1';
            startTime<='0';
            ns<=S0;
        elsif(gumSel='1')then
            change<="01010";
            gum<='1';
            startTime<='0';
            ns<=S0;
        elsif(nickle='1')then
             startTime<='0';
             ns<=S30;
        elsif(dime='1')then
             startTime<='0';
             ns<=S30;
         end if;
    elsif(timeIN>="0101" and(refund='0' or gumSel='0' or candySel='0' or nickle='0' or dime='0')) then
        ns<=s0;
        startTime<='0';
    end if;
        startTime<='1';
when S30=>
    if(timeIN< "0101" and (refund='1' or gumSel='1' or candySel='1' or nickle='1' or dime='1'))then
        if(refund<='1')then
            change<="11110";
            startTime<='0';
            ns<=S0;
        elsif(candySel='1')then
            change<="00101";
            candy<='1';
            startTime<='0';
            ns<=S0;
        elsif(gumSel='1')then
            change<="01111";
            gum<='1';
            startTime<='0';
            ns<=S0;
        elsif(nickle='1')then
             startTime<='0';
             ns<=S30;
             change<="00101";
```

```
        elsif(dime='1')then
            startTime<='0';
            ns<=S30;
            change<="01010";
          end if;
      elsif(timeIN>="0101" and(refund='0' or gumSel='0' or candySel='0' or nickle='0' or dime='0')) then
         ns<=s0;
         startTime<='0';
      end if;
      startTime<='1';
   end case;
 end process;
end Behavioral;
```

Item Counter

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity ItemCounter is
 Port (clk,reset,candyIN,gumIN,startTimer:in std_logic;
       sel : in std_logic_vector(1 downto 0);--chose which total you want to see displayed
       timeOut: out std_logic_vector(3 downto 0);
       total : out std_logic_vector(3 downto 0));
end ItemCounter;

architecture Behavioral of ItemCounter is
signal tmpG,tmpC,tmpTime :std_logic_vector(3 downto 0);--gum and candy
begin

process(clk,reset)
begin
   if(reset='1')then
      tmpG<="0000";
      tmpC<="0000";
      tmpTime<="0000";
   elsif(clk'event and clk='1')then
      if(gumIN='1')then
         tmpG<=tmpG + "0001"; --increment
      end if;
      if(candyIN='1')then
         tmpC<=tmpC + "0001"; --increment
      end if;

    --for the timer
    if(startTimer='1')then
        if(tmpTime<"0101")then
```

```
            tmpTime<=tmpTime + "0001";   --increments timer
            timeOut<=tmpTime;
        end if;
        if(tmpTime>="0101")then
             tmpTime<="0000";
             timeOut<=tmpTime;
        end if;
    end if;
    --if the timer aint start
    if(startTimer='0')then
       tmpTime<="0000";
    end if;

       --based on selector...
       if(sel="00")then
          total<=tmpG;--displau the total gum
       elsif(sel="01")then
          total<=tmpC;--display the total candy
       elsif(sel="01")then
          total<=tmpTime;--display the total candy
       end if;
    end if;
end process;
end Behavioral;
```

Money Counter

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;


entity MoneyCounter is
 Port (clk,reset,nickleIN,dimeIN,gumOut,candyOut,refund:in std_logic;
       timeIN:in std_logic_vector(3 downto 0);
       moneyCount:out std_logic_vector(4 downto 0));
end MoneyCounter;

architecture Behavioral of MoneyCounter is
 signal tmp: std_logic_vector(4 downto 0);--amount of money insterted
begin
process(clk,reset)
begin
    if (reset='1')then
       tmp<="00000";--restock
    elsif(clk'event and clk='1')then
       if(nickleIN='1')then--if nickle is inserted
          tmp<=tmp + "0101";  --add nickel(5 cents)
       elsif(dimeIN='1')then--if dime is inserted
```

```
        tmp<=tmp + "1010";--add dime(10 cents)
      elsif(gumOut='1' or candyOut='1')then--if gum or candy is selected
        tmp<="00000";--revert to no money entered
      elsif(refund='1')then--refund is high
        tmp<="00000";
      end if;

      if(timeIN="0101")then
        tmp<="00000";
      end if;
  end if;
      moneyCount<=tmp;
   end process;
end Behavioral;
```

Countdown

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity countdown is
 Port (clk,reset,coin:in std_logic;
       countTime:out std_logic_vector(3 downto 0));
end countdown;

architecture Behavioral of countdown is
signal tmp: std_logic_vector(3 downto 0);
begin

tmp<="0000";  --setting these to 15 so the counter can show how many are left in machine
process(clk,reset)
begin
   if (reset='1')then
   tmp<="0000";
   elsif(clk'event and clk='1')then
    if(coin='1')then
      if(tmp<"0101")then
      tmp<=tmp + "0001";   --increments countdown
      end if;
    end if;
    end if;
    countTime<=tmp;
   end process;

end Behavioral;
```

Constraints

```
set_property IOSTANDARD LVCMOS33 [get_ports {change[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {change[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {change[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {change[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {change[0]}]
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 PACKAGE_PIN W18 [get_ports {change[4]}]
set_property PACKAGE_PIN V19 [get_ports {change[3]}]
set_property PACKAGE_PIN U19 [get_ports {change[2]}]
set_property PACKAGE_PIN E19 [get_ports {change[1]}]
set_property PACKAGE_PIN U16 [get_ports {change[0]}]
set_property PACKAGE_PIN W7 [get_ports {display[6]}]
set_property PACKAGE_PIN W6 [get_ports {display[5]}]
set_property PACKAGE_PIN U8 [get_ports {display[4]}]
set_property PACKAGE_PIN V8 [get_ports {display[3]}]
set_property PACKAGE_PIN U5 [get_ports {display[2]}]
set_property PACKAGE_PIN V5 [get_ports {display[1]}]
set_property PACKAGE_PIN U7 [get_ports {display[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports clockLED]
set_property IOSTANDARD LVCMOS33 [get_ports dime]
set_property IOSTANDARD LVCMOS33 [get_ports refund]
set_property IOSTANDARD LVCMOS33 [get_ports reset]
set_property PACKAGE_PIN V13 [get_ports clockLED]
set_property PACKAGE_PIN R3 [get_ports dime]
set_property PACKAGE_PIN T2 [get_ports refund]
set_property PACKAGE_PIN U1 [get_ports reset]


set_property PACKAGE_PIN W5 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk]


set_property PACKAGE_PIN V14 [get_ports candy]
set_property IOSTANDARD LVCMOS33 [get_ports candy]
set_property IOSTANDARD LVCMOS33 [get_ports {displaySel[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {displaySel[1]}]
set_property PACKAGE_PIN U14 [get_ports gum]
set_property PACKAGE_PIN W3 [get_ports NE]
set_property PACKAGE_PIN W2 [get_ports nickle]
set_property IOSTANDARD LVCMOS33 [get_ports gum]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports NE]
set_property IOSTANDARD LVCMOS33 [get_ports nickle]


set_property IOSTANDARD LVCMOS33 [get_ports {cash[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {cash[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {cash[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {cash[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {cash[0]}]
set_property PACKAGE_PIN N3 [get_ports {cash[2]}]
set_property PACKAGE_PIN L1 [get_ports {cash[4]}]
set_property PACKAGE_PIN P3 [get_ports {cash[1]}]
set_property PACKAGE_PIN P1 [get_ports {cash[3]}]
set_property PACKAGE_PIN U3 [get_ports {cash[0]}]


set_property PACKAGE_PIN V17 [get_ports {displaySel[0]}]
set_property PACKAGE_PIN V16 [get_ports {displaySel[1]}]


set_property IOSTANDARD LVCMOS33 [get_ports candyIN]
set_property IOSTANDARD LVCMOS33 [get_ports gumIN]
set_property PACKAGE_PIN R2 [get_ports candyIN]
set_property PACKAGE_PIN T1 [get_ports gumIN]
```