

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)

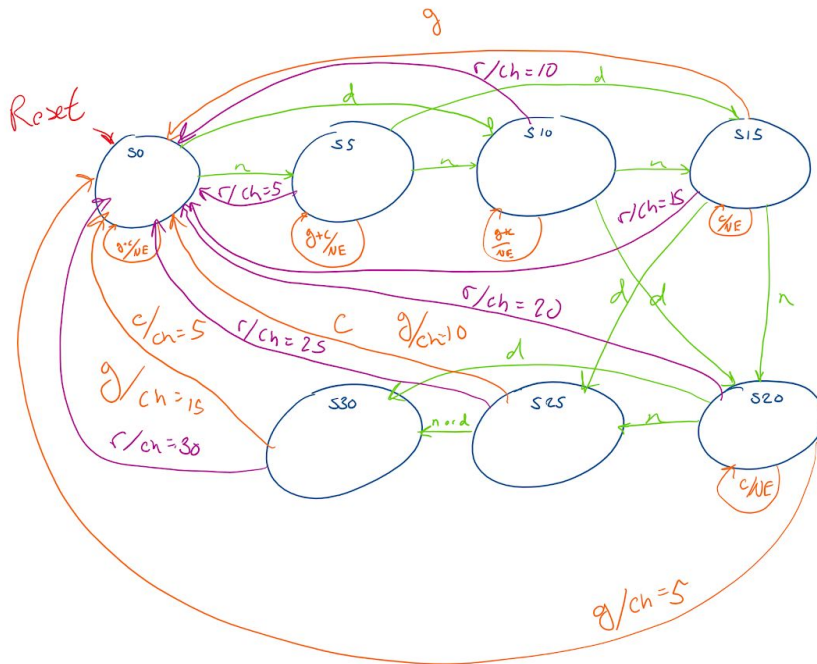
May 2, 2019

Lab #10

Introduction: In Lab 10 we 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). 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.

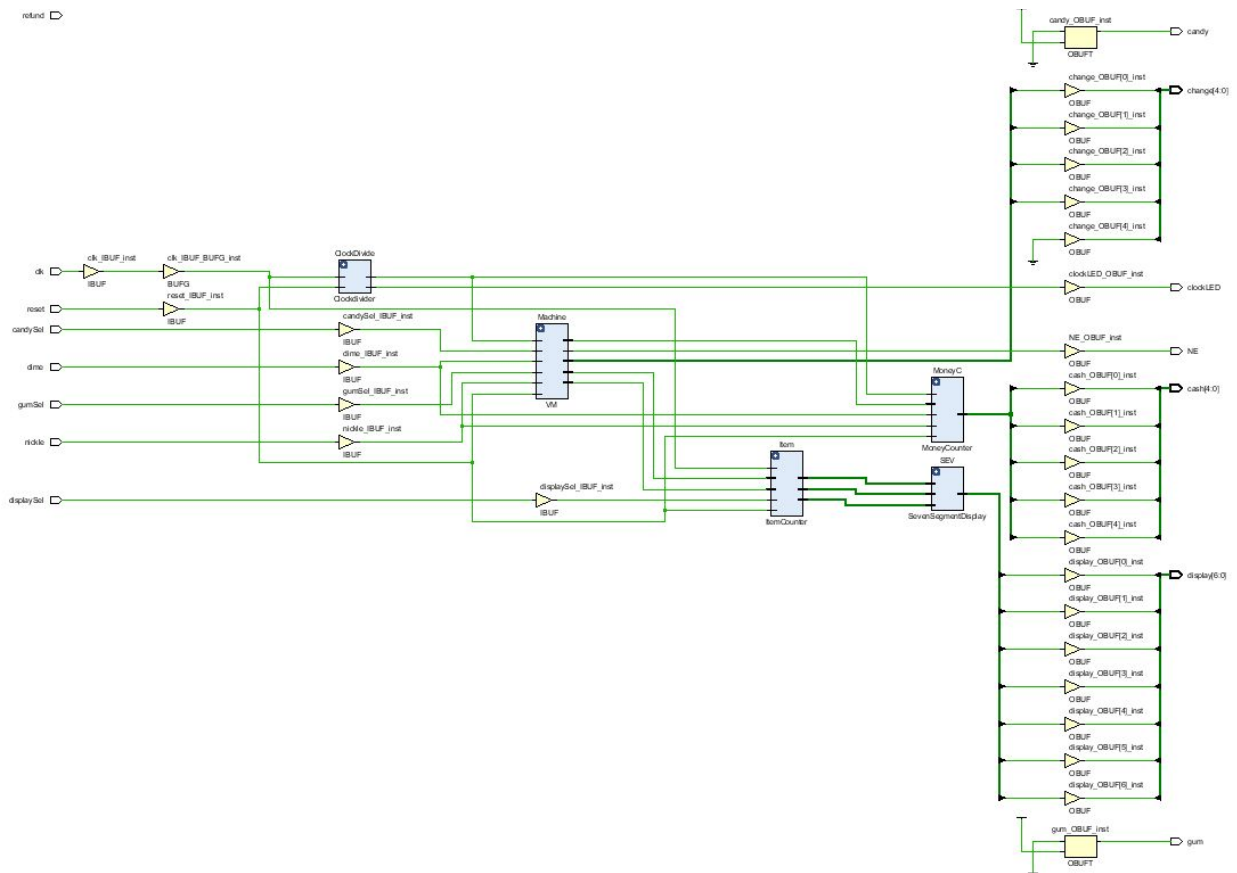
Background: Vending Machines are found in everyday life, from schools, office buildings, or any place where people need a quick snack. In this lab we will be using a Finite State Machine design a Vending Machine which will simulate a real-life one.

State Diagram





































* if no input
stay in current state

Schematic



PIN ASSIGNMENT

Name	Direction	Package Pin	
▼  All ports (29)			
▼  cash (5)	OUT		
 cash[4]	OUT	L1	▼
 cash[3]	OUT	P1	▼
 cash[2]	OUT	N3	▼
 cash[1]	OUT	P3	▼
 cash[0]	OUT	U3	▼
▼  change (5)	OUT		
 change[4]	OUT	W18	▼
 change[3]	OUT	V19	▼
 change[2]	OUT	U19	▼
 change[1]	OUT	E19	▼
 change[0]	OUT	U16	▼
▼  display (7)	OUT		
 display[6]	OUT	W7	▼
 display[5]	OUT	W6	▼
 display[4]	OUT	U8	▼
 display[3]	OUT	V8	▼
 display[2]	OUT	U5	▼
 display[1]	OUT	V5	▼
 display[0]	OUT	U7	▼
▼  Scalar ports (12)			
 candy	OUT	V14	▼
 candySel	IN	R2	▼
 clk	IN	W5	▼
 clockLED	OUT	V13	▼
 dime	IN	R3	▼
 displaySel	IN	V17	▼
 gum	OUT	U14	▼
 gumSel	IN	T1	▼
 NE	OUT	W3	▼
 nickle	IN	W2	▼
 refund	IN	T2	▼
 reset	IN	U1	▼

Results

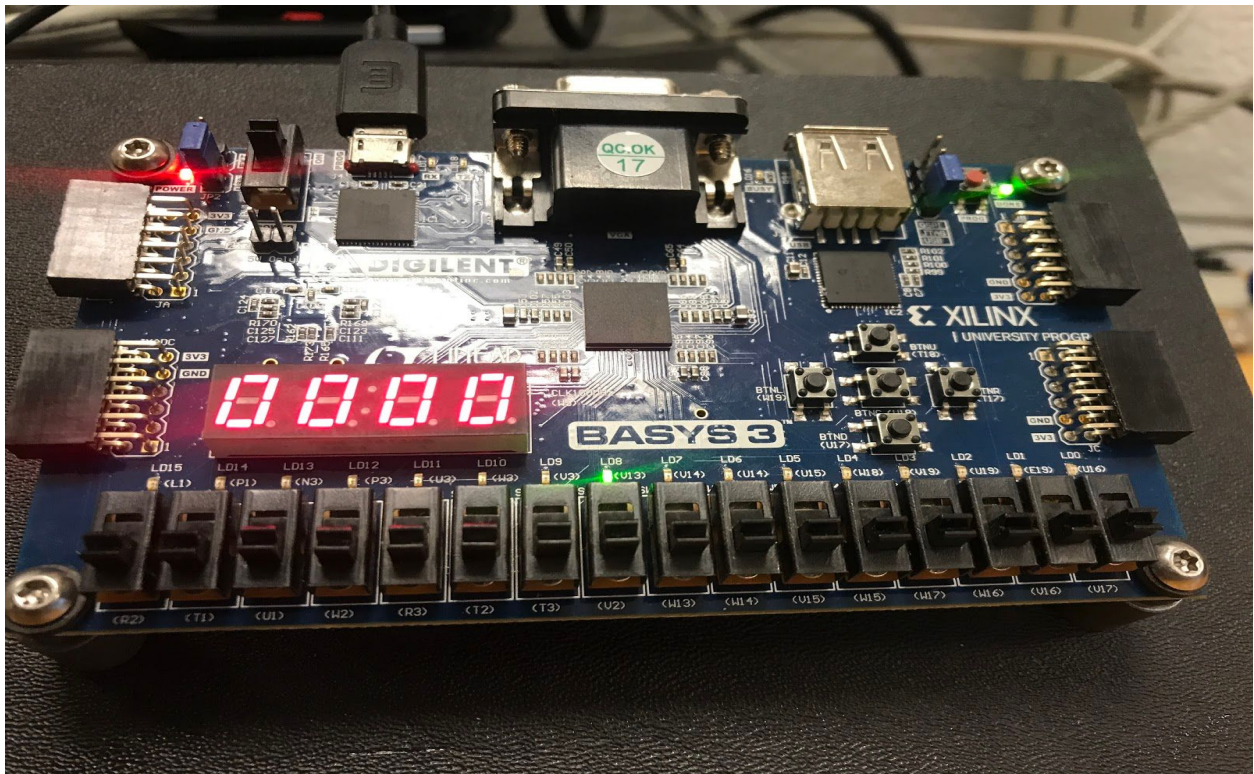


Figure 1.1) System initial state, no money has been added. Nothing selected

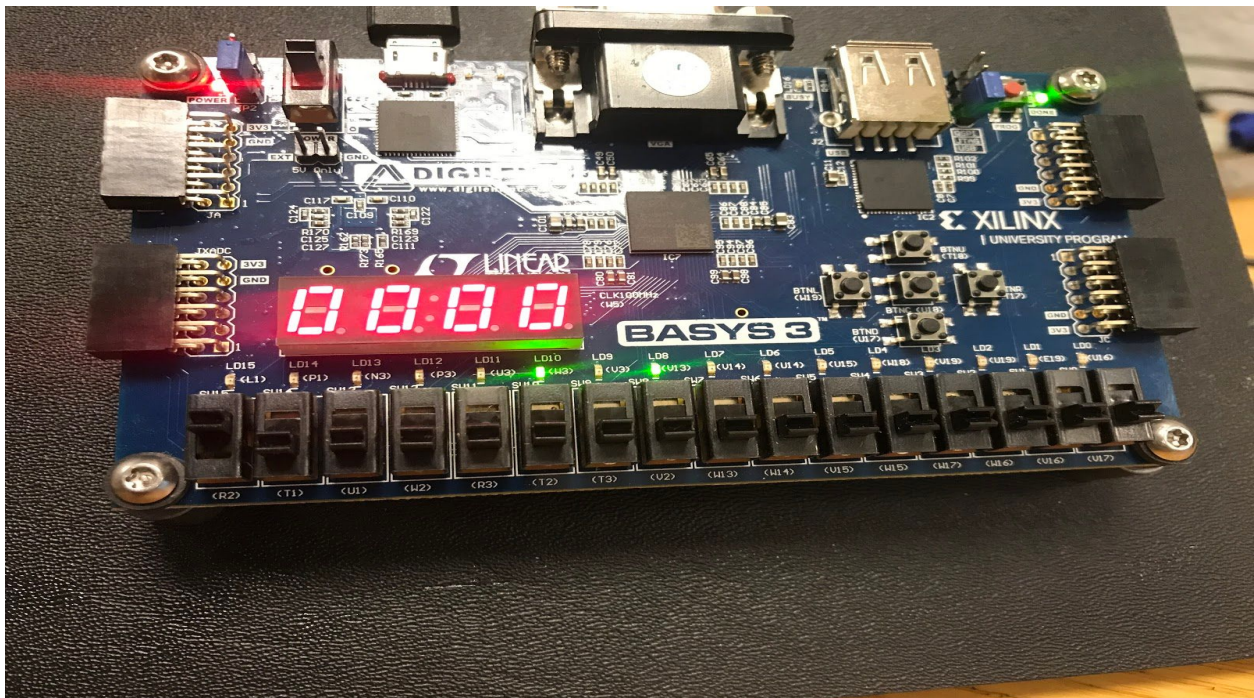


Figure 1.2) System initial state, no money has been added. Candy Selected, shows not enough money for candy

Appendices

VHDL

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

entity TopLevel is
  Port (clk,reset,gumSel,candySel,refund,nickle,dime: in std_logic;
        displaySel:in std_logic;--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: std_logic;
  signal total: std_logic_vector(3 downto 0);

  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;
                    nickleCount,dimeCount,NotEnough,gum,candy:out std_logic;
                    change: out std_logic_vector(4 downto 0));
end component;

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

  component ItemCounter port (clk,reset,candyIN,gumIN:in std_logic;
                              sel : in std_logic;
                              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

ClockDivide: ClockDivider port map(clk=>clk,
reset=>reset,
SlowClock=>clockOut,
ledO=>clockLED);

Machine: VM port map(clk=>clockOut,
reset=>reset,
gumSel=> gumSel,
candySel=>candySel,
refund=>refund,
nickle=>nickle,
dime=>dime,
nickleCount=>nickleSig,
dimeCount=>dimeSig,
NotEnough=>NE,
gum=>gumOut,
candy=>candyOut,
change=>change);

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

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

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

end Behavioral;


```

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):= "0000000000000000000000000000";
        begin
            -- calculations
            wait until ((clk'EVENT) AND (clk = '1'));
            if (reset = '1') then
                cnt := "0000000000000000000000000000";
            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;

```

```

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;
          nickleCount,dimeCount,NotEnough,gum,candy: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
            NotEnough<='0';
            gum<='0';
            candy<='0';
            change<="00000";
            nickleCount<=nickle;
            dimeCount<=dime;
            case cs is
            when S0=>
                if(gumSel='1' or candySel='1')then
                    NotEnough<='1';
                    ns<=S0;
                else
                    if(nickle='1')then
                        ns<=S5;
                    elsif(dime='1')then

```

```

        ns<=S10;
    elseif((nickle='0' AND dime='0') AND (gumSel='0' AND candySel='0'))then
        ns<=cs;
    elseif(refund<='1')then
        ns<=S0;
    end if;
end if;

```

when S5=>

```

    if(gumSel='1' or candySel='1')then
        NotEnough<='1';
        ns<=S5;
    else
        if(nickle='1')then
            ns<=S10;
        elseif(dime='1')then
            ns<=S15;
        elseif((nickle='0' AND dime='0') AND (gumSel='0' AND candySel='0'))then
            ns<=cs;
        elseif(refund='1')then
            change<="00101";
            ns<=S0;
        end if;
    end if;
end if;

```

when S10=>

```

    if(gumSel='1' or candySel='1')then
        NotEnough<='1';
        ns<=S10;
    else
        if(nickle='1')then
            ns<=S15;
        elseif(dime='1')then
            ns<=S20;
        elseif((nickle='0' AND dime='0') AND (gumSel='0' AND candySel='0'))then
            ns<=cs;
        elseif(refund='1')then
            change<="01010";
            ns<=S0;
        end if;
    end if;
end if;

```

when S15=>

```

if(candySel='1')then
    NotEnough<='1';
    ns<=S15;
elseif(gumSel='1')then
    gum<='1';
    ns<=S0;
else
    if(nickle='1')then
        ns<=S20;
    elseif(dime='1')then
        ns<=S25;
    elseif((nickle='0' AND dime='0') AND (gumSel='0' AND candySel='0'))then
        ns<=cs;
    elseif(refund='1')then
        change<="01111";
        ns<=S0;
    end if;
end if;

```

when S20=>

```

if(candySel='1')then
    NotEnough<='1';
    ns<=S20;
elseif(gumSel='1')then
    gum<='1';
    change<="00101";
    ns<=S0;
else
    if(nickle='1')then
        ns<=S25;
    elseif(dime='1')then
        ns<=S30;
    elseif((nickle='0' AND dime='0') AND (gumSel='0' AND candySel='0'))then
        ns<=cs;
    elseif(refund='1')then
        change<="10100";
        ns<=S0;
    end if;
end if;

```

when S25=>

```

if(candySel='1')then
    candy<='1';
    ns<=S0;

```



```

        elsif(gumSel='1')then
            gum<='1';
            change<="01010";
            ns<=S0;
        else
            if(nickle='1' or dime='1')then
                ns<=S30;
            elsif((nickle='0' AND dime='0') AND (gumSel='0' AND candySel='0'))then
                ns<=cs;
            elsif(refund='1')then
                change<="11001";
                ns<=S0;
            end if;
        end if;
    when S30=>
        if(candySel='1')then
            candy<='1';
            change<="00101";
            ns<=S0;
        elsif(gumSel='1')then
            gum<='1';
            change<="01111";
            ns<=S0;
        else
            if(nickle='1')then
                change<="00101";
                ns<=S30;
            elsif(dime='1')then
                change<="01010";
                ns<=S30;
            elsif((nickle='0' AND dime='0') AND (gumSel='0' AND candySel='0'))then
                ns<=cs;
            elsif(refund='1')then
                change<="11110";
                ns<=S0;
            end if;
        end if;
    end case;
end process;
end Behavioral;

```

```

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

entity MoneyCounter is
  Port (clk,reset,nickleIN,dimeIN,gumOut,candyOut:in std_logic;
        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
      end if;
    end if;
    moneyCount<=tmp;
  end process;
end Behavioral;

```

```

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

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

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

  process(clk,reset)
  begin
    if(reset='1')then
      tmpG<="0000";
      tmpC<="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;
      --based on selector...
      if(sel='0')then
        total<=tmpG;--displau the total gum
      elsif(sel='1')then
        total<=tmpC;--display the total candy
      end if;
    end if;
  end process;
end Behavioral;

```

```

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;

```


VHDL

```
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 candySel]
set_property IOSTANDARD LVCMOS33 [get_ports clockLED]
set_property IOSTANDARD LVCMOS33 [get_ports dime]
set_property IOSTANDARD LVCMOS33 [get_ports gumSel]
set_property IOSTANDARD LVCMOS33 [get_ports refund]
set_property IOSTANDARD LVCMOS33 [get_ports reset]
set_property PACKAGE_PIN R2 [get_ports candySel]
set_property PACKAGE_PIN V13 [get_ports clockLED]
set_property PACKAGE_PIN R3 [get_ports dime]
set_property PACKAGE_PIN T1 [get_ports gumSel]
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 IOSTANDARD LVCMOS33 [get_ports {amount[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {amount[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {amount[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {amount[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {amount[0]}]
set_property PACKAGE_PIN L1 [get_ports {amount[4]}]
set_property PACKAGE_PIN P1 [get_ports {amount[3]}]
set_property PACKAGE_PIN U3 [get_ports {amount[0]}]
set_property PACKAGE_PIN P3 [get_ports {amount[1]}]
set_property PACKAGE_PIN N3 [get_ports {amount[2]}]
set_property PACKAGE_PIN V14 [get_ports candy]
set_property IOSTANDARD LVCMOS33 [get_ports candy]
set_property IOSTANDARD LVCMOS33 [get_ports displaySel]
set_property PACKAGE_PIN V17 [get_ports displaySel]
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]
```