Traffic Buildup Detection System

By: Tammy Luu, Yinkai Liu, Shengzhuo Liu, Patrick Pantis
278 Digital Logic – Final Project – Fall 2016
School of Engineering and Computer Science, Oakland University
Instructor: Prof. Daniel Llamocca Obregon
Introduction

- The Purpose of this project is to create a traffic buildup detection system that will better help the flow of traffic at a traffic intersection.
- By using state codes that loop repeatedly, the traffic light system could be created. Having a push button ground sensor, the code could determine if there was a build up of cars at the traffic intersection. This is done to prevent build up of traffic.
- During midnight traffic slows down significantly, so the main road is implemented to change to a flashing yellow light, as well as the side roads are changed to a flashing red. This is done to prevent cars waiting at a light when no other vehicles are driving at night.
- With all this a clock is displayed on a 7-segment display to show the timing of the lights at the traffic intersect. This makes it easier to follow the timing of the state codes as well as implementing a cross walk timer for pedestrians.
Methodology

The software used in this project is Xilinx ISE Webpack Design Software 14.7 coding with VHDL. All the code was then programmed to our NexysTM-4DDR Artix-7 FPGA board to displaying the traffic intersection lights. These are the Methodology of the coding in our project:

- Traffic Light
- Blinking Yellow
- Main Road Traffic Build Up Sensor
- 7-seg Timing display
ASM - Main and Side Traffic Light
Traffic Detected
Yellow Flickering Light
## Constraints

### Clock signal

<table>
<thead>
<tr>
<th>NET</th>
<th>LOC</th>
<th>IOSTANDARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>clk</td>
<td>T3</td>
<td>LVCMOS33</td>
</tr>
</tbody>
</table>

## BUTTONS

<table>
<thead>
<tr>
<th>NET</th>
<th>LOC</th>
<th>IOSTANDARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>day</td>
<td>N17</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>night</td>
<td>F10</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>traf</td>
<td>F17</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>btnm</td>
<td>M17</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>btnu</td>
<td>M18</td>
<td>LVCMOS33</td>
</tr>
</tbody>
</table>

## LEDs

<table>
<thead>
<tr>
<th>NET</th>
<th>LOC</th>
<th>IOSTANDARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>ld0</td>
<td>H17</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>ld1</td>
<td>K15</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>ld2</td>
<td>J13</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>ld3</td>
<td>N14</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>ld4</td>
<td>R18</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>ld5</td>
<td>V17</td>
<td>LVCMOS33</td>
</tr>
</tbody>
</table>

## 7 segment display

<table>
<thead>
<tr>
<th>NET</th>
<th>LOC</th>
<th>IOSTANDARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>led6</td>
<td>T10</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>led5</td>
<td>R10</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>led4</td>
<td>K16</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>led3</td>
<td>K13</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>led2</td>
<td>P15</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>led1</td>
<td>T11</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>led0</td>
<td>L18</td>
<td>LVCMOS33</td>
</tr>
</tbody>
</table>

## Analog inputs

<table>
<thead>
<tr>
<th>NET</th>
<th>LOC</th>
<th>IOSTANDARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>an0</td>
<td>J17</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>an1</td>
<td>J18</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>an2</td>
<td>T9</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>an3</td>
<td>J14</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>an4</td>
<td>P14</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>an5</td>
<td>T14</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>an6</td>
<td>K2</td>
<td>LVCMOS33</td>
</tr>
<tr>
<td>an7</td>
<td>U13</td>
<td>LVCMOS33</td>
</tr>
</tbody>
</table>

---

#Bank = 35, Pin name = IO_L12P_T1_MRCC_35, 5ch name = clk100mhz

TIME3SPEC T3_sys clk pin = PERIOD sys clk pin 100 MHz HIGH 50%
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity traffic is
  port (clk: in STD_LOGIC;
        clr: in STD_LOGIC;
        sensor: in STD_LOGIC;
        clear: in STD_LOGIC;
        lights: out STD_LOGIC_VECTOR(5 downto 0));
end traffic;

architecture traffic of traffic is
  type state_type is (s0, s1, s2, s3, s4, s5, s6, s7, s8, s9, s10);
  signal state: state_type;
  signal count: STD_LOGIC_VECTOR(5 downto 0) := "011110";
  constant SEC10: STD_LOGIC_VECTOR(5 downto 0) := "011110";
  constant SEC15: STD_LOGIC_VECTOR(5 downto 0) := "101110";
  constant SEC3: STD_LOGIC_VECTOR(5 downto 0) := "000010";
  constant SEC1: STD_LOGIC_VECTOR(5 downto 0) := "000011";
  constant SEC: STD_LOGIC_VECTOR(5 downto 0) := "000001";
begin
  process(clk, clr, clear, sensor)
  begin
    if clk'event and clk = '1' then
      state <= s1;
      count <= "000000";
    elsif clr = '1' then
      state <= s6;
      count <= "000000";
    elsif sensor = '1' then
      state <= s9;
      count <= "000000";
    else
      state <= s2;
      count <= "000000";
      when s2 ->
        if count < SEC3 then
          state <= s2;
          count <= count + 1;
        else
          state <= s3;
          count <= "000000";
        end if;
      when s3 ->
        if count < SEC10 then
          state <= s3;
          count <= "000000";
        else
          state <= s6;
          count <= "000000";
        end if;
    -- Flashing Yellow Light
      when s6 ->
        if count < SEC1 then
          state <= s6;
          count <= count + 1;
        else
          state <= s7;
          count <= "000000";
        end if;
      when s7 ->
        if count < SEC then
          state <= s7;
          count <= count + 1;
        else
          state <= s6;
          count <= "000000";
        end if;
      -- When the ground sensor is detected, the light is changed to red, yellow and then green, red.
      when s8 ->
        if count < SEC3 then
          state <= s8;
          count <= "000000";
        else
          state <= s9;
          count <= "000000";
        end if;
    end if;
  end process;
end traffic;
else
    state <= a6;
    count <= "000000";
end if;
--when the ground sensor is detected, the light is changed to red, yellow and then green, red.
when s9 ->
    if count < SEC3 then
        state <= a9;
        count <= count + 1;
    else
        state <= a10;
        count <= "000000";
    end if;
when a10 ->
    if count < SEC1 then
        state <= a10;
        count <= count + 1;
    else
        state <= a8;
        count <= "000000";
    end if;
when a8 ->
    if count < SEC15 then
        state <= a8;
        count <= count + 1;
    else
        state <= a4;
        count <= "000000";
    end if;
when others ->
    state <= a1;
end case;
end if;
end process;
C2: process(state)
begin
    case state is
    when s0 -> lights <= "100001";
    when s1 -> lights <= "100010";
    when s2 -> lights <= "100100";
    when s3 -> lights <= "001100";
    when s4 -> lights <= "011010";
    when s5 -> lights <= "101010";
    when s6 -> lights <= "100101";
    when s7 -> lights <= "001100";
    when s8 -> lights <= "001100";
    when s9 -> lights <= "100010";
    when s10 -> lights <= "101010";
    when others -> lights <= "100010";
end case;
end process;
end traffic:
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity digi_clk is
port (clr: in std_logic;
      clrr: in std_logic;
      sensor: in std_logic;
      clk1: in std_logic;
      seconds : out std_logic_vector(5 downto 0));
end digi_clk;

architecture Behavioral of digi_clk is
signal sec : integer range 0 to 60 :=0;
signal count : integer := 1;
signal clk : std_logic :='0';
begn
seconds <= conv_std_logic_vector(sec,6);
--clk generation. For 100 MHz clock this generates 1 Hz clock.
process(clk1,clr,clrr,sensor)
begin
if clr = '1' then
  count <= 0;
elsif sensor = '1' then
  count <= 0;
elsif clrr = '1' then
  count <= 0;
else if(clk1'event and clk1='1') then
  count <= count+1;
  if(count = 50000000) then
    clk <= not clk;
    count <= 1;
  end if;
end if;
end process;
process(clk,clr,sensor,clrr) --period of clk is 1 second.
begin
if clr = '1' then
  sec <= 0;
elsif sensor = '1' then
  sec <= 15;
elsif clrr = '1' then
  sec <= 19;
else if(clk'event and clk='1') then
  sec <= sec+1;
  if(sec = 34) then
    sec<=0;
  end if;
end if;
end process;
end Behavioral;
entity HEX is
  Port (  
    R : in  STD_LOGIC_VECTOR (5 downto 0);  
    AN : out  STD_LOGIC_VECTOR (6 downto 0));
end HEX;

architecture Behavioral of HEX is
signal temp:STD_LOGIC_VECTOR (6 downto 0);
begin
  with R select
  temp <=
  "0110000" when "000000",--1  
  "1101101" when "000001",--2  
  "1110001" when "000010",--3  
  "0112000" when "000011",--1  
  "1111110" when "000100",--0  
  "0110000" when "000101",--1  
  "1101101" when "000110",--2  
  "1111001" when "000111",--3  
  "0110011" when "001000",--4  
  "1011011" when "001001",--5  
  "1011111" when "001010",--6  
  "1110000" when "001011",--7  
  "1111111" when "001100",--8  
  "1111011" when "001101",--9  
  "1110111" when "001110",--A  
  "1111110" when "001111",--0  
  "0110000" when "010000",--1  
  "1101101" when "010001",--2  
  "1111001" when "010010",--3  
  "1111110" when "010011",--0  
  "0110000" when "010100",--1  
  "1101111" when "010101",--2  
  "1111001" when "010110",--3  
  "0110111" when "010111",--4  
  "1011011" when "011000",--5  
  "1011111" when "011001",--6  
  "1110000" when "011010",--7  
  "1111111" when "011011",--8  
  "1111011" when "011100",--9  
  "1110111" when "011101",--A  
  "0011111" when "011110",--B  
  "1001111" when "011111",--C  
  "0111101" when "100000",--D  
  "1001111" when "100001",--E  
  "1000111" when "100010",--F  
  "0000000" when Others;
  AN <= not temp;
end Behavioral;
entity traffic_lights_top is
  port(
    clk : in STD_LOGIC;
    day : in STD_LOGIC;
    night: in STD_LOGIC;
    traf: in STD_LOGIC;
    an : out STD_LOGIC_VECTOR(7 downto 0);
    seg : out STD_LOGIC_VECTOR (6 downto 0);
    ld: out STD_LOGIC_VECTOR(5 downto 0)
  );
end traffic_lights_top;

architecture traffic_lights_top of traffic_lights_top is
  component clkdiv is
    port(
      mclk : in STD_LOGIC;
      clr : in STD_LOGIC;
      clk3 : out STD_LOGIC
    );
  end component;

  component traffic is
    port (clk: in STD_LOGIC;
      Clr: in STD_LOGIC;
      clr: in STD_LOGIC;
      sensor: in STD_LOGIC;
      lights: out STD_LOGIC_VECTOR(5 downto 0));
  end component;

  component HEX is
    Port ( R : in STD_LOGIC_VECTOR (5 downto 0);
      AN : out STD_LOGIC_VECTOR (6 downto 0));
  end component;

  component digi_clk is
    port (clr :in std_logic;
      clrr: in std_logic;
      clk1 : in std_logic;
      seconds : out std_logic_vector(5 downto 0))
    );
  end component;

signal clr, clrr, sensor: STD_LOGIC;
signal lol : std_logic_vector(5 downto 0);
begin
an <= "11111110";
clr <= day;
clrr <= night;
sensor <= traf;
U1: clkdiv
  port map (mclk=>clk,
    clr=>clr,
    clk3=>clk3);

U2: traffic
  port map (clk=>clk3,
    clr=>clr,
    clrr=>clrr,
    sensor=>sensor,
    lights=>ld);

U3: digi_clk
  port map (clr=>clrr,
    clrr=>clr,
    sensor=> sensor,
    clk1=>clk,
    seconds=> lol);

U4:HEX
  port map (R=> lol,
    AN=> seg);
end traffic_lights_top;
Possible Improvements

- Count down timer could be implemented instead of a count up timer.

- Using all 2 7-seg displays to show the timing of the lights could be changed.

- More lights at the intersection, more state codes

- Improved coding the implement changing light at main intersection when build up of traffic if the button is held for 5 or more seconds. Current iteration changes the light instantly when the button is pressed.
From this project we learned how to use state codes and timing functions to create a traffic signal. This is extremely beneficial when programming things that go from one state into another depending on whether the program passes that case. If it passes it will continue onto the next case or loop its current case until the code is passed.
Any Questions about our project?

Thank you for your attention!