```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 11/27/2018 10:27:30 AM
-- Design Name:
-- Module Name: project_384 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity project_384 is
--I/O declaration
    Port ( push_button : in STD_LOGIC_VECTOR (1 downto 0);
           CLK : in STD_LOGIC;
           anode : out STD_LOGIC_VECTOR (7 downto 0);
           cathode : out STD_LOGIC_VECTOR (7 downto 0));
end project_384;

architecture Behavioral of project_384 is

--signal declaration
signal enable : std_logic:= '0';
signal start_button, sig1_button : integer:= 0;
```

```
signal count_a, count_b, count_c, count_d, disp_1, disp_2, disp_3, disp_4 : integer
:= 0;
signal count_1: integer :=0;
signal count_2: integer :=0;
signal temp_1: STD_LOGIC :='0';
signal temp_2: STD_LOGIC :='0';
signal disp_refresh, timer : STD_LOGIC;

begin process(CLK)
begin
    if (CLK='1' and CLK'event) then
        count_1 <= count_1 + 1;
        count_2 <= count_2 + 1;
        --1 MHz / 240 Hz = 416,666.67
        --50% duty cycle = 416,666.67 / 2 = 208,333.33
        if (count_1 = 208334) then
            temp_1 <= NOT temp_1;
            count_1 <= 0;
        end if;
        --1 MHz / 100 Hz = 1,000,000
        --50% duty cycle = 1,000,000 / 2 = 500,000
        if (count_2 = 500000) then
           temp_2 <= NOT temp_2;
           count_2 <= 0;
        end if;
    end if;
end process;

disp_refresh <= temp_1; --240 Hz
timer <= temp_2; --100 Hz

process (disp_refresh, timer, push_button, enable) is
begin
    --reset
    if (push_button(1) = '1' and enable = '0') then
        --display
        disp_1 <= 0;
        disp_2 <= 0;
        disp_3 <= 0;
        disp_4 <= 0;
        count_a <= 0;
        count_b <= 0;
        count_c <= 0;
        count_d <= 0;
    end if;
```

```
    --rising edge
    if (timer='1' and timer'event) then
        if (enable = '1') then
            disp_1 <= count_a;
            disp_2 <= count_b;
            disp_3 <= count_c;
            disp_4 <= count_d;

            --if count reaches 9, add 1 to the next display
            count_a <= count_a + 1;
            if (count_a = 9) then
                count_a <= 0;
                count_b <= count_b + 1;
                if (count_b = 9) then
                    count_b <= 0;
                    count_c <= count_c + 1;
                        if (count_c = 9) then
                            count_c <= 0;
                           count_d <=count_d + 1;
                                if (count_d = 9) then
                                   count_d <= 0;
                                end if;
                        end if;
                end if;
            end if;
        end if;
    end if;
end process;


process (disp_refresh)
variable digit : unsigned (1 downto 0) := "00"; --keeps track of 7 segment displays
begin
    if (disp_refresh='1' and disp_refresh'event) then
        case digit is
            when "00" =>
                case (disp_1) is
                    when 0 =>
                        anode <= "11111110";
                        cathode <= "11000000";
                    when 1 =>
                        anode <= "11111110";
                        cathode <= "11111001";
                    when 2 =>
                        anode <= "11111110";
                        cathode <= "10100100";
```

```
            when 3 =>
                anode <= "11111110";
                cathode <= "10110000";
            when 4 =>
                anode <= "11111110";
                cathode <= "10011001";
            when 5 =>
                anode <= "11111110";
                cathode <= "10010010";
            when 6 =>
                anode <= "11111110";
                cathode <= "10000010";
            when 7 =>
                anode <= "11111110";
                cathode <= "11111000";
            when 8 =>
                anode <= "11111110";
                cathode <= "10000000";
            when 9 =>
                anode <= "11111110";
                cathode <= "10010000";
            when others =>
                anode <= "11111110";
                cathode <= "11111111";
        end case;

    when "01" =>
        case (disp_2) is
            when 0 =>
                anode <= "11111101";
                cathode <= "11000000";
            when 1 =>
                anode <= "11111101";
                cathode <= "11111001";
            when 2 =>
                anode <= "11111101";
                cathode <= "10100100";
            when 3 =>
                anode <= "11111101";
                cathode <= "10110000";
            when 4 =>
                anode <= "11111101";
                cathode <= "10011001";
            when 5 =>
                anode <= "11111101";
                cathode <= "10010010";
```

```
            when 6 =>
                anode <= "11111101";
                cathode <= "10000010";
            when 7 =>
                anode <= "11111101";
                cathode <= "11111000";
            when 8 =>
                anode <= "11111101";
                cathode <= "10000000";
            when 9 =>
                anode <= "11111101";
                cathode <= "10010000";
            when others =>
                anode <= "11111101";
                cathode <= "11111111";
        end case;

    when "10" =>
        case (disp_3) is
            when 0 =>
                anode <= "11111011";
                cathode <= "01000000";
            when 1 =>
                anode <= "11111011";
                cathode <= "01111001";
            when 2 =>
                anode <= "11111011";
                cathode <= "00100100";
            when 3 =>
                anode <= "11111011";
                cathode <= "00110000";
            when 4 =>
                anode <= "11111011";
                cathode <= "00011001";
            when 5 =>
                anode <= "11111011";
                cathode <= "00010010";
            when 6 =>
                anode <= "11111011";
                cathode <= "00000010";
            when 7 =>
                anode <= "11111011";
                cathode <= "01111000";
            when 8 =>
                anode <= "11111011";
                cathode <= "00000000";
```

```
            when 9 =>
                anode <= "11111011";
                cathode <= "00010000";
            when others =>
                anode <= "11111011";
                cathode <= "01111111";
        end case;

    when "11" =>
        case (disp_4) is
            when 0 =>
                anode <= "11110111";
                cathode <= "11000000";
            when 1 =>
                anode <= "11110111";
                cathode <= "11111001";
            when 2 =>
                anode <= "11110111";
                cathode <= "10100100";
            when 3 =>
                anode <= "11110111";
                cathode <= "10110000";
            when 4 =>
                anode <= "11110111";
                cathode <= "10011001";
            when 5 =>
                anode <= "11110111";
                cathode <= "10010010";
            when 6 =>
                anode <= "11110111";
                cathode <= "10000010";
            when 7 =>
                anode <= "11110111";
                cathode <= "11111000";
            when 8 =>
                anode <= "11110111";
                cathode <= "10000000";
            when 9 =>
                anode <= "11110111";
                cathode <= "10010000";
            when others =>
                anode <= "11110111";
                cathode <= "11111111";
        end case;
end case;
digit := digit + 1;
```

```
    end if;
end process;

process(count_a)
begin

    --start/stop
    --at every rising edge, check the status of the push button
   if (timer='1' and timer'event) then
        if (push_button(0) = '1') then
            start_button <= 1;
        elsif (push_button(0) = '0') then
            start_button <= 0;
        end if;
        sig1_button <= start_button;
        if (sig1_button = 0 and start_button = 1) then
            enable <= not enable;
        end if;
    end if;
end process;

end Behavioral;
```