# Assignment - 5 Report

Tanish Gupta , Divyanshu Agarwal

15th May 2022

## 1 Introduction

The objective of this assignment was to learn and use push buttons. One push button was used to enter the number to be displayed and the other push button was used to control the brightness of the digits being displayed.

## 2 Strategy and Logic

Since we already had the logic for different brightness values in Assignment 4, all we had to do was to assign each digit the appropriate brightness value as input from slider switches (while the push button for brightness control is being pressed).
For the base case, when nothing has been entered or pressed, we display "0000" with the lowest possible brightness. (The brightness for each digit can be set to 0, 1, 2 or 3 using slider switches. The 0 brightness means minimum brightness, which doesn't mean a duty cycle of 0 %).
We were facing issues while using push buttons. The optimisation method of implementing design optimized the code to an extent, that the code perceived the button inputs as clocks, and so gave the error "Cannot synthesize 3 clocks". Small tweak in constraint file was made to resolve this error,

## 3 Code and Working

The main directory contains the design.vhd file, which has the VHDL code logic and the helper.vhd, which is the same file as submitted in assignment 2 (The helper file contains the logic for single Display of LED). The directory also has the constraint file and the generated bitstream.

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_Std.all;

entity lightDisplay is
        Port (
        Number : in std_logic_vector(15 downto 0);
        Clk : in std_logic;
        btnL : in std_logic;
        btnR : in std_logic;
        LED : out std_logic_vector(6 downto 0);
        anode : out STD_LOGIC_Vector(3 downto 0));
end lightDisplay;

architecture structure of lightDisplay is
        component singleDisplay
                Port (
           A : in STD_LOGIC;
```

```
            B : in STD_LOGIC;
            C : in STD_LOGIC;
            D : in STD_LOGIC;

            light_num : in integer;
            LED : out std_logic_vector(6 downto 0);
            anode : out STD_LOGIC_Vector(3 downto 0));
end component singleDisplay;

    signal light_num        : integer;
    signal digit_A          : std_logic:='1';
    signal digit_B          : std_logic:='0';
    signal digit_C          : std_logic:='0';
    signal digit_D          : std_logic:='1';
    signal counter          : integer :=0;
    signal counter_new      : integer := 0 ;
    signal counter_2        : integer := 0 ;
    signal Sampled_number   : std_logic_vector(15 downto 0) :=x"0000";
    signal Entered_Number   : std_logic_vector(15 downto 0) := x"0000";
    signal state2           : integer := 0 ;


    begin
        process(clk)

          variable position :integer:=0 ;
          variable brightness : std_logic_vector(7 downto 0);
          type btness is array (3 downto 0) of integer  ;
          variable brightness0123 : btness := (0,0,0,0);



                begin


               if(Clk'event  and rising_Edge(clk)) then
                    counter<= ((counter+1)) ;



                    counter_new<= ((counter) / 524288) mod 4  ;
                    state2 <= ((counter)/524288) mod 1024 ;
                    counter_2 <= ((counter/16384)mod 32) ;
                    end if ;

                    position := (state2/256) ;

                    case (counter_new) is
                        when 0 =>
                            if(counter_2<brightness0123(3)) then
                                digit_A<=Sampled_Number(15);
                                digit_B<=Sampled_Number(14);
                                digit_C<=Sampled_Number(13);
                                digit_D<=Sampled_Number(12);
```

```
            light_num<=position mod 4 ;
        else
            light_num<=4;
        end if;

    when 1 =>
        if(counter_2<brightness0123(2)) then
            digit_A<=Sampled_Number(11);
            digit_B<=Sampled_Number(10);
            digit_C<=Sampled_Number(9);
            digit_D<=Sampled_Number(8);
            light_num<=((position+ 1) mod 4);
        else
            light_num<=4;
        end if;

    when 2 =>
        if(counter_2<brightness0123(1)) then
            digit_A<=Sampled_Number(7);
            digit_B<=Sampled_Number(6);
            digit_C<=Sampled_Number(5);
            digit_D<=Sampled_Number(4);
            light_num<=((position + 2) mod 4 );
        else
            light_num<=4;
        end if;

    when others =>
        if(counter_2<brightness0123(0)) then
            digit_A<=Sampled_Number(3);
            digit_B<=Sampled_Number(2);
            digit_C<=Sampled_Number(1);
            digit_D<=Sampled_Number(0);
            light_num<=((position + 3) mod 4);
        else
            light_num<=4;
        end if;

end case;

if(BtnL='1') then
    Entered_Number <= Number;
end if ;

if(state2 = 0) then
    Sampled_Number <= Entered_Number;
end if ;

if(BtnR='1') then
    brightness:=Number(7 downto 0);
end if;
```

```
                case brightness(7 downto 6) is

                   when "00" => brightness0123(3) := 2 ;
                   when  "01" => brightness0123(3) := 5 ;
                   when  "10" => brightness0123(3) := 10 ;
                   when  others => brightness0123(3) := 31 ;
                end case;


                case brightness(5 downto 4) is

                   when "00" => brightness0123(2) := 2 ;
                   when  "01" => brightness0123(2) := 5 ;
                   when  "10" => brightness0123(2) := 10 ;
                   when  others => brightness0123(2) := 31 ;
                end case;

                 case brightness(3 downto 2) is

                   when "00" => brightness0123(1) := 2 ;
                   when  "01" => brightness0123(1) := 5 ;
                   when  "10" => brightness0123(1) := 10 ;
                   when  others => brightness0123(1) := 31 ;
                end case;

                 case brightness(1 downto 0) is

                   when "00" => brightness0123(0) := 2 ;
                   when  "01" => brightness0123(0) := 5 ;
                   when  "10" => brightness0123(0) := 10 ;
                   when  others => brightness0123(0) := 31 ;
                end case;

        end process;


    Single_display: entity work.singleDisplay(Design_arch) port map(
        digit_A , digit_B, digit_C , digit_D ,light_num , LED ,anode
        );


end structure;
```

# 4 Simulation

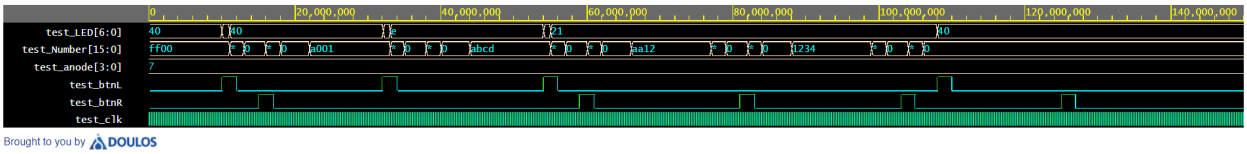


Figure 1: Generated EP Wave

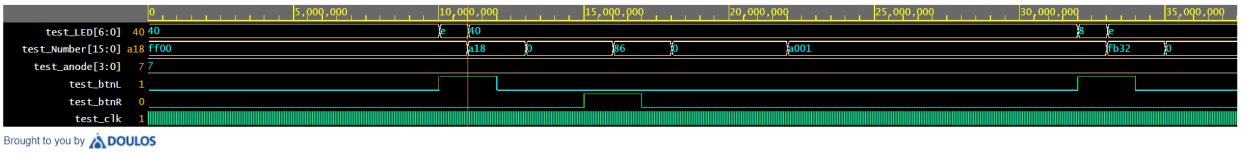


Figure 2: Generated EP Wave

The report generated on Vivado is as follows:

```
Copyright 1986-2019 Xilinx, Inc. All Rights Reserved.
----------------------------------------------------------------------------------------------------
| Tool Version : Vivado v.2019.1 (lin64) Build 2552052 Fri May 24 14:47:09 MDT 2019
| Date         : Sat May 14 22:37:37 2022
| Host         : divyanshu-HP-ENVY-x360-Convertible-13-bd0xxx
                 running 64-bit Ubuntu 20.04.2 LTS
| Command      : report_utilization -file lightDisplay_utilization_synth.rpt
                 -pb lightDisplay_utilization_synth.pb
| Design       : lightDisplay
| Device       : 7a35tcpg236-2
| Design State : Synthesized
----------------------------------------------------------------------------------------------------

Utilization Design Information

Table of Contents
-----------------

1. Slice Logic
1.1 Summary of Registers by Type
2. Memory
3. DSP
4. IO and GT Specific
```

## 1. Slice Logic

| Site Type | Used | Fixed | Available | Util% |
|---|---|---|---|---|
| Slice LUTs* | 95 | 0 | 20800 | 0.46 |
| LUT as Logic | 95 | 0 | 20800 | 0.46 |
| LUT as Memory | 0 | 0 | 9600 | 0.00 |
| Slice Registers | 91 | 0 | 41600 | 0.22 |
| Register as Flip Flop | 47 | 0 | 41600 | 0.11 |
| Register as Latch | 44 | 0 | 41600 | 0.11 |
| F7 Muxes | 0 | 0 | 16300 | 0.00 |
| F8 Muxes | 0 | 0 | 8150 | 0.00 |

### 1.1 Summary of Registers by Type

| Total | Clock Enable | Synchronous | Asynchronous |
|---|---|---|---|
| 0 | _ | – | – |
| 0 | _ | – | Set |
| 0 | _ | – | Reset |
| 0 | _ | Set | – |
| 0 | _ | Reset | – |
| 0 | Yes | – | – |
| 0 | Yes | – | Set |
| 44 | Yes | – | Reset |
| 0 | Yes | Set | – |
| 47 | Yes | Reset | – |

## 2. Memory

| Site Type | Used | Fixed | Available | Util% |
|---|---|---|---|---|
| Block RAM Tile | 0 | 0 | 50 | 0.00 |
| RAMB36/FIFO* | 0 | 0 | 50 | 0.00 |
| RAMB18 | 0 | 0 | 100 | 0.00 |

* Note: Each Block RAM Tile only has one FIFO logic available and therefore can accommodate only one FIFO36E1 or one FIFO18E1.

However, if a FIFO18E1 occupies a Block RAM Tile, that tile can still accommodate a RAMB18E1

## 3. DSP

| Site Type | Used | Fixed | Available | Util% |
|---|---|---|---|---|
| DSPs | 0 | 0 | 90 | 0.00 |

## 4. IO and GT Specific

| Site Type | Used | Fixed | Available | Util% |
|---|---|---|---|---|
| Bonded IOB | 30 | 0 | 106 | 28.30 |
| Bonded IPADs | 0 | 0 | 10 | 0.00 |
| Bonded OPADs | 0 | 0 | 4 | 0.00 |
| PHY_CONTROL | 0 | 0 | 5 | 0.00 |
| PHASER_REF | 0 | 0 | 5 | 0.00 |
| OUT_FIFO | 0 | 0 | 20 | 0.00 |
| IN_FIFO | 0 | 0 | 20 | 0.00 |
| IDELAYCTRL | 0 | 0 | 5 | 0.00 |
| IBUFDS | 0 | 0 | 104 | 0.00 |
| GTPE2_CHANNEL | 0 | 0 | 2 | 0.00 |
| PHASER_OUT/PHASER_OUT_PHY | 0 | 0 | 20 | 0.00 |
| PHASER_IN/PHASER_IN_PHY | 0 | 0 | 20 | 0.00 |
| IDELAYE2/IDELAYE2_FINEDELAY | 0 | 0 | 250 | 0.00 |
| IBUFDS_GTE2 | 0 | 0 | 2 | 0.00 |
| ILOGIC | 0 | 0 | 106 | 0.00 |
| OLOGIC | 0 | 0 | 106 | 0.00 |

## 5. Clocking

| Site Type | Used | Fixed | Available | Util% |
|---|---|---|---|---|
| BUFGCTRL | 3 | 0 | 32 | 9.38 |
| BUFIO | 0 | 0 | 20 | 0.00 |
| MMCME2_ADV | 0 | 0 | 5 | 0.00 |
| PLLE2_ADV | 0 | 0 | 5 | 0.00 |
| BUFMRCE | 0 | 0 | 10 | 0.00 |
| BUFHCE | 0 | 0 | 72 | 0.00 |
| BUFR | 0 | 0 | 20 | 0.00 |

## 6. Specific Feature

| Site Type | Used | Fixed | Available | Util% |
|---|---|---|---|---|
| BSCANE2 | 0 | 0 | 4 | 0.00 |
| CAPTUREE2 | 0 | 0 | 1 | 0.00 |
| DNA_PORT | 0 | 0 | 1 | 0.00 |
| EFUSE_USR | 0 | 0 | 1 | 0.00 |
| FRAME_ECCE2 | 0 | 0 | 1 | 0.00 |
| ICAPE2 | 0 | 0 | 2 | 0.00 |
| PCIE_2_1 | 0 | 0 | 1 | 0.00 |
| STARTUPE2 | 0 | 0 | 1 | 0.00 |
| XADC | 0 | 0 | 1 | 0.00 |

## 7. Primitives

| Ref Name | Used | Functional Category |
|---|---|---|
| FDRE | 47 | Flop & Latch |
| LDCE | 44 | Flop & Latch |
| LUT2 | 32 | LUT |
| LUT1 | 30 | LUT |
| LUT4 | 23 | LUT |
| CARRY4 | 23 | CarryLogic |
| LUT3 | 19 | LUT |
| IBUF | 19 | IO |
| LUT5 | 17 | LUT |
| LUT6 | 16 | LUT |
| OBUF | 11 | IO |
| BUFG | 3 | Clock |

## 8. Black Boxes

| Ref Name | Used |
|---|---|

## 9. Instantiated Netlists

| Ref Name | Used |
|---|---|