Assignment - 6 Report

Tanish Gupta , Divyanshu Agarwal

15th May 2022

1 Introduction

The objective of this assignment was to simply design a stopwatch on the BASYS3 board, which could be controlled by 3 push buttons.
The 3 push buttons were assigned for - Start/Continue, Pause and Reset. Since there are only 4 digits that can be displayed, 1 digit for minutes, two digits for seconds and one digit for tenths of a second were used.

2 Strategy and Logic

Implemneting the counter with the help of on board clock was an easy part. The issue we faced for the most part of our lab was as follows -
We maintained a counter, that would increment on each clock edge, and then we used this counter to derive other counters, (for seconds and minutes etc.) The problem with this was that, since the time period of our clock was 10 ns, we would increment the counter every 10ns. The counter was an integer variable, so the maximum value it could take is $2^31 - 1$. So, the stopwatch would work correctly for around 21.47 seconds, after which it showed unexpected behaviour.
So, we changed the logic of the counter and reset it at every seconds, so that none of the counter takes too large value.

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 design is
    port(
        clk : in std_logic;
        startContinue : in std_logic;
        pause : in std_logic;
        reset : in std_logic;
        LED : out std_logic_vector (6 downto 0);
        anode : out std_logic_vector (3 downto 0)

    );
end design;
```

```
architecture Behavioral of design 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 digit_A : std_logic;
    signal digit_B : std_logic;
    signal digit_C : std_logic;
    signal digit_D : std_logic;
    signal light_num : integer;

begin

    process(clk)
        variable enable : std_logic := '0';
        variable counter : integer :=0 ;
        variable counter_display : integer := 0;
        variable counter_decisecond : integer := 0;
        variable counter_second_ones : integer := 0;
        variable counter_second_tens : integer := 0;
        variable counter_minutes : integer := 0;
        variable deciseconds : integer := 0;
        variable counter_seconds : integer := 0;
        variable second_ones : integer := 0;
        variable second_tens : integer := 0;
        variable minutes : integer := 0;
        variable counter2: integer :=0;
        variable counter_deciseconds_cyclic :integer:=0;
        variable past_deciseconds: integer:=0 ;
        variable Time_display : std_logic_vector(15 downto 0) := "0000000000000000";

        begin
            if(rising_edge(clk) and clk'event) then
                if(reset = '1') then
                    counter := 0;
                    Time_display := "0000000000000000";
                    past_deciseconds:=0;

                end if;

                if(startContinue = '1') then
                    enable := '1';
                elsif (pause = '1') then
                    enable := '0';
                end if;
```

```
 counter_display := (counter2/524288) mod 4;
 counter2 := counter2+1;

 if(enable = '1') then

 counter := counter + 1;

 counter_deciseconds_cyclic := counter/10000000;

if(counter_deciseconds_cyclic  = 10) then
     past_Deciseconds := past_Deciseconds+1; counter:=0;
end if;

counter_decisecond:=( (past_Deciseconds*10) + (counter/10000000)) ;



 deciseconds := counter_decisecond mod 10;
 counter_seconds := counter_decisecond/10;
 second_ones := (counter_seconds mod 60) mod 10;
 second_tens := (counter_seconds mod 60) / 10;
 minutes := ((counter_seconds/60));



 Time_display(15 downto 12) := std_logic_vector(to_unsigned(minutes,4));
 Time_display(11 downto 8) := std_logic_vector(to_unsigned(second_tens,4));
 Time_display(7 downto 4) := std_logic_vector(to_unsigned(second_ones,4));
 Time_display(3 downto 0) := std_logic_vector(to_unsigned(deciseconds,4));

 end if;

 case(counter_display) is
     when 0 =>
         digit_A <= Time_display(15);
         digit_B <= Time_display(14);
         digit_C <= Time_display(13);
         digit_D <= Time_display(12);
         light_num <= 0;

     when 1 =>
         digit_A <= Time_display(11);
         digit_B <= Time_display(10);
         digit_C <= Time_display(9);
         digit_D <= Time_display(8);
         light_num <= 1;

     when 2 =>
         digit_A <= Time_display(7);
         digit_B <= Time_display(6);
         digit_C <= Time_display(5);
         digit_D <= Time_display(4);
         light_num <= 2;
```

```
                    when others =>
                        digit_A <= Time_display(3);
                        digit_B <= Time_display(2);
                        digit_C <= Time_display(1);
                        digit_D <= Time_display(0);
                        light_num <= 3;

                end case;
            end if;

    end process;

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

4 Simulation and Synthesis

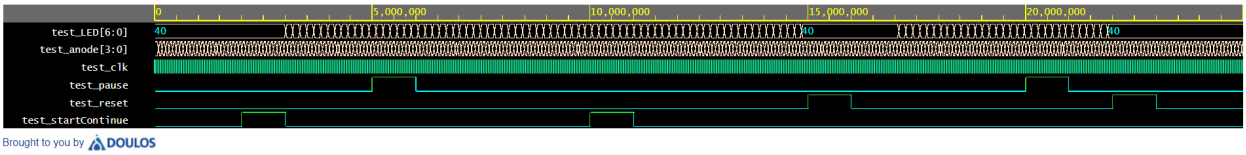

Figure 1: Generated EP Wave

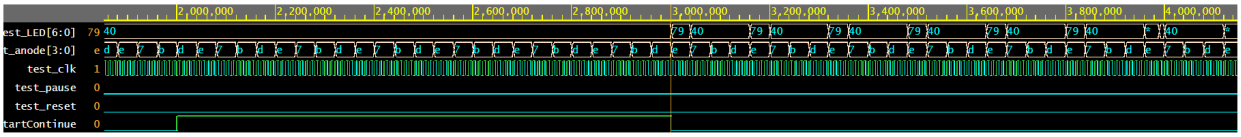

Figure 2: Generated EP Wave

These EPWaves was carefully analysed to check if all outputs were correct!
The Vivado synthesis report 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 22:20:56 2022
| Host         : divyanshu-HP-ENVY-x360-Convertible-13-bd0xxx
                 running 64-bit Ubuntu 20.04.2 LTS
| Command      : report_utilization -file design_utilization_synth.rpt
                 -pb design_utilization_synth.pb
| Design       : \design
| 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
5. Clocking
6. Specific Feature
7. Primitives
8. Black Boxes
9. Instantiated Netlists

1. Slice Logic
--------------
```

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	2170	0	20800	10.43
LUT as Logic	2170	0	20800	10.43
LUT as Memory	0	0	9600	0.00
Slice Registers	118	0	41600	0.28
Register as Flip Flop	118	0	41600	0.28
Register as Latch	0	0	41600	0.00
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
0	Yes	-	Reset
0	Yes	Set	-
118	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	15	0	106	14.15
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	1	0	32	3.13
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
LUT6	920	LUT
CARRY4	353	CarryLogic
LUT5	340	LUT
LUT4	339	LUT
LUT2	311	LUT
LUT3	269	LUT
FDRE	118	Flop & Latch
LUT1	106	LUT
OBUF	11	IO
IBUF	4	IO
BUFG	1	Clock

8. Black Boxes

Ref Name	Used

9. Instantiated Netlists

Ref Name	Used