COL 215 : Hardware Assignment 1

Tejas Anand, Kashish Goel

October 2022

1 Introduction

In this assignment, our objective was to implement a circuit which would take 4-digit input from the switches on the Basys3 board and display it on the 4-seven segment display on the board.

2 Design Decisions

2.1 Combinational Logic

Our first step was to figure out the expressions of logic of the 7 segment decoder. To do this, we drew the drew out the digits 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F and noted which segments are to be lighted for any particular 4-bit input. We then used our Assignment 3 code to find the minimized expression of each of the segments.

2.2 Timer Circuit

To alternate between the displayed digits in the display, we maintained a 2-bit counter which we incremented by 1 whenever there is a rising edge of the clock.
This counter served as the select input of the multiplexer, with "11" selecting the 1^{st} digit, "10" selecting the 3^{rd}, "01" selecting the 2^{nd} digit and "00" selecting the 4^{th} digit from the 4-digit input (each digit is 4 bit). Also, the digit period is equal to the time period of the clock and the refresh period is equal to 4 times the refresh period.
The default clock on Basys3 has a period of 10 ns, thus making the refresh period equal to 40 ns, which is clearly out of the acceptable range that is 1 - 16 ns.
To make a new clock, we took an n bit binary number and incremented it by 1 whenever there was a rising edge of the original clock. Then we used the 1^{st} digit of that number as the new clock. The period of the new clock is 2^n times the period of the original clock, because we go through 2^n rising edges after which the number repeats.
To meet our constraints, we required

$$1ms < 4 \cdot 2^n \cdot 10ns < 16ms \tag{1}$$

$n = 17$ satisfied our conditions, so we used a 17 bit integer to make the new clock.

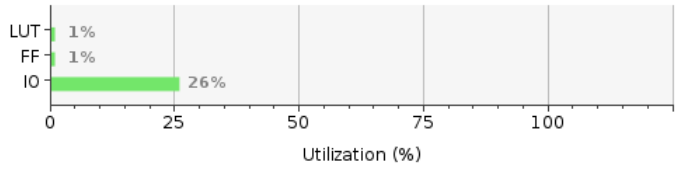
2.3 Mistakes

Initially we thought of a non-modular design, implementing the entire code in a single file. However, this didn't work out well because of excessive inputs(17) and outputs(11) the code became very long and hard to understand and had confusing expressions.
After this unsuccessful attempt, we tried to introduce modularity. We wrote the multiplexer, 7-segment decoder and the timer circuit in 3 different VHDL files. In the timer circuit file, we used the multiplexer and 7-segment decoder as components along with the clock. This made the code clear and easy to read. Readability of our code was also improved since now we used the vector data type from the *std_logic_vector* library, which we weren't utilizing previously.

3 Synthesis Report

- Hierarchy
- Summary
- Slice Logic
  - Slice LUTs (<1%)
    - LUT as Logic (<1%)
  - Slice Registers (<1%)
- Memory
- DSP
- IO and GT Specific
- Clocking
  - BUFGCTRL (3%)
- Specific Feature
- Primitives
- Black Boxes
- Instantiated Netlists

Resource	Utilization	Available	Utilization %
LUT	12	20800	0.06
FF	19	41600	0.05
IO	28	106	26.42



1. Slice Logic

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	12	0	0	20800	0.06
LUT as Logic	12	0	0	20800	0.06
LUT as Memory	0	0	0	9600	0.00
Slice Registers	19	0	0	41600	0.05
Register as Flip Flop	19	0	0	41600	0.05
Register as Latch	0	0	0	41600	0.00
F7 Muxes	0	0	0	16300	0.00
F8 Muxes	0	0	0	8150	0.00

2. Memory

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

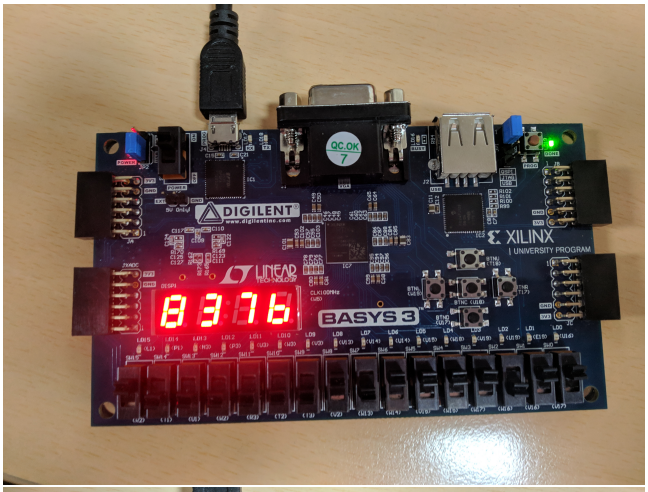
* Note: Each Block RAM Tile only has one FIFO logic available and therefore can a
or one FIFO18E1. However, if a FIFO18E1 occupies a Block RAM Tile, that tile can

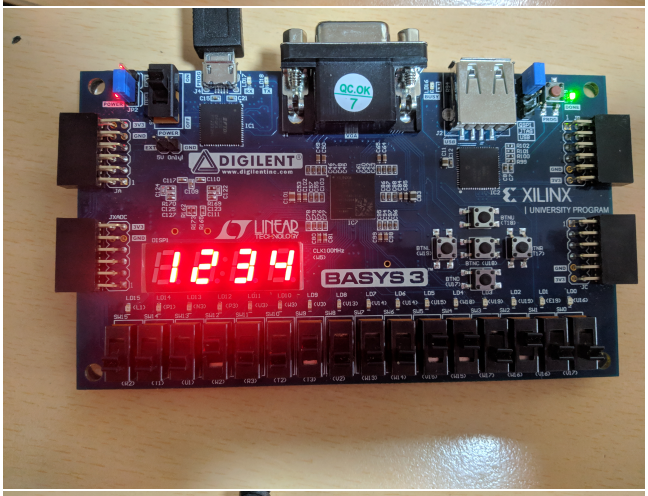
3. DSP

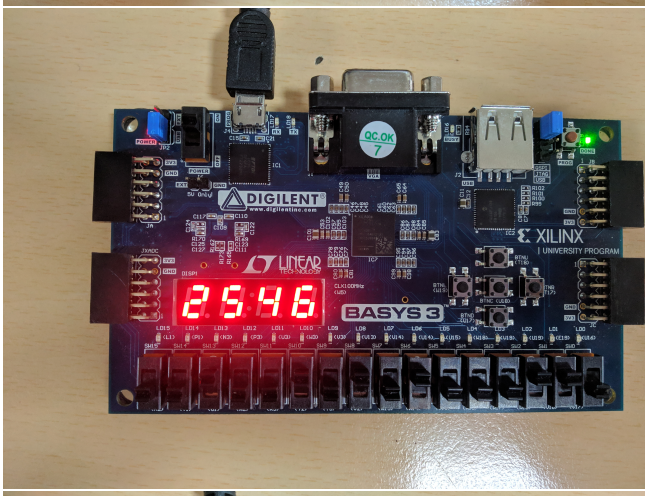
Site Type	Used	Fixed	Prohibited	Available	Util%
DSPs	0	0	0	90	0.00

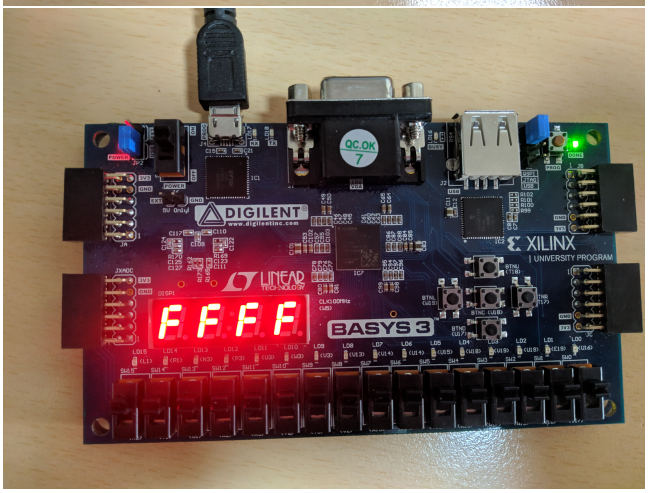
Figure 1: We can see that digit period is equal to the period of our clock and is one-fourth of refresh period

4 Some Test Inputs


8376
BASYS 3


1234
BASYS 3


2546
BASYS 3


FFFF
BASYS 3