Lab 1 Report

Michael Cooke

October 2021

Contents

1 Part A

1.1 Draw K Maps

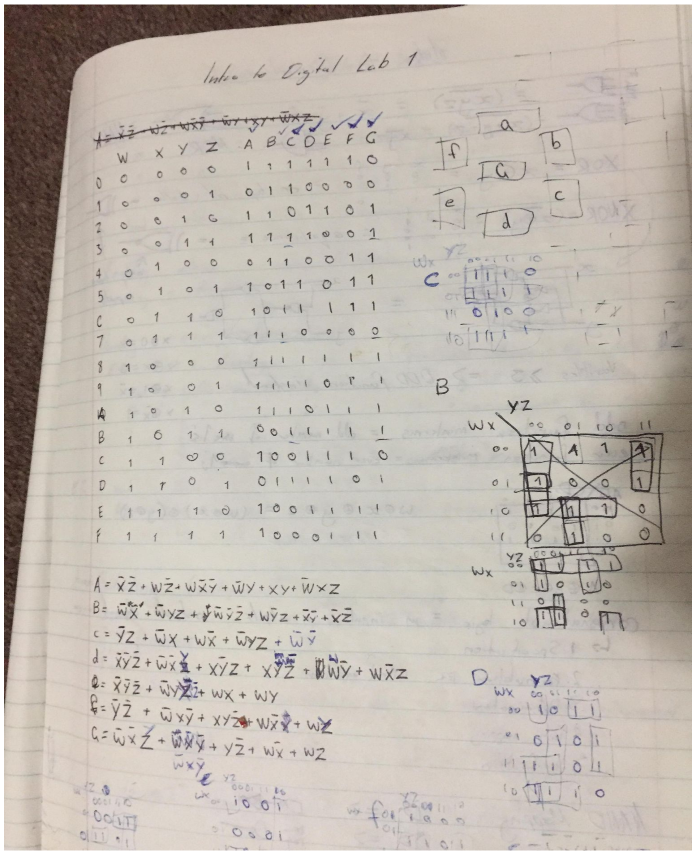
1.1.1 Summary

The purpose of this task was to create K maps to convert a binary signal into the corresponding LED's in the seven segment display to turn on. This was covered in weeks one and two of the course and is not complicated.

1.1.2 Difficulties

There were no key difficulties encountered during this process as I used `http://www.32x8.com` to verify my results before moving onto the next step.

1.1.3 Proof



Intro to Digital Lab 1

~~$A = \bar{x}\bar{z} + w\bar{z} + w\bar{x}\bar{y} + \bar{w}y + xy + \bar{w}xz$~~

	W	X	Y	Z	A	B	C	D	E	F	G
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1
A	1	0	1	0	1	1	1	0	1	1	1
B	1	0	1	1	0	0	1	1	1	1	1
C	1	1	0	0	1	0	0	1	1	1	0
D	1	1	0	1	0	1	1	1	1	0	1
E	1	1	1	0	1	0	0	1	1	1	1
F	1	1	1	1	1	0	0	0	1	1	1

$A = \bar{x}\bar{z} + w\bar{z} + w\bar{x}\bar{y} + \bar{w}y + xy + \bar{w}xz$

$B = \bar{w}\bar{x} + \bar{w}yz + \bar{w}\bar{y}\bar{z} + w\bar{y}z + \bar{x}\bar{y} + \bar{x}\bar{z}$

$c = \bar{y}z + \bar{w}x + w\bar{x} + \bar{w}yz + \bar{w}\bar{y}$

$d = \bar{x}\bar{y}\bar{z} + \bar{w}\bar{x}y + x\bar{y}z + xy\bar{z} + \bar{w}\bar{y} + w\bar{x}z$

$e = \bar{x}\bar{y}\bar{z} + \bar{w}y\bar{z} + wx + wy$

$f = \bar{y}\bar{z} + \bar{w}x\bar{y} + xy\bar{z} + w\bar{x} + wz$

$G = \bar{w}x\bar{z} + \bar{w}\bar{x}y + y\bar{z} + w\bar{x} + wz$

$\bar{w}x\bar{y}$

Figure 1: Binary Table and K-maps

1.2 Implement HexToSSD

1.2.1 Summary

For this task we had to convert the K-maps we had previously designed into Quartus files using the block diagram/schematic tool and our knowledge of logic gates. This required an understanding of how Quartus works and some thought as to the most efficient way to implement the solution.

1.2.2 Difficulties

Luckily, I had already dealt with the numerous problems with installing and running Quartus during lab 0 (which was by far the most difficult lab due to this). From there I already had an understanding of how to work within the block diagram schematic as I had completed the first part of this subject in a past semester before I had to drop the subject due to personal reasons.

1.2.3 Proof

QuartusFiles/HexToSSD

1.3 Simulate HexToSSD

1.3.1 Summary

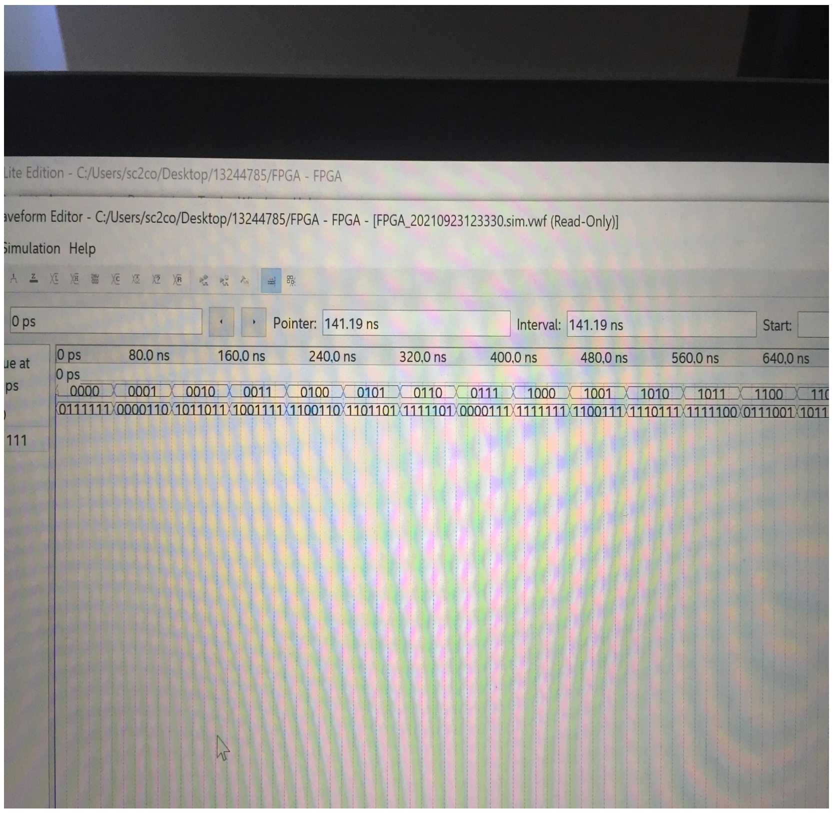
For this task we had to test the file previously created using a waveform test. This process is complicated by the very specific naming and file positioning required to have the VHDL/Verilog compile. In my partition the .vwf files are not placed in the correct directory by default and this had to be remedied at the same time as -novopt was removed.

1.3.2 Difficulties

I failed the first waveform test and it ended up looking like gibberish. When I reviewed the underlying file I noticed that instead of lining up $WXYZ\bar{W}\bar{X}\bar{Y}\bar{Z}$ I had instead done $WXYZ\bar{Z}\bar{Y}\bar{X}\bar{W}$ and this was quickly fixed.

1.3.3 Proof

The waveform worked perfectly as you can test is you read across the rows in my kmap, note I included the waveform from after bus implementation:


Lite Edition - C:/Users/sc2co/Desktop/13244785/FPGA - FPGA
aveform Editor - C:/Users/sc2co/Desktop/13244785/FPGA - FPGA - [FPGA_20210923123330.sim.vwf (Read-Only)]
Simulation Help
Pointer:
Interval:
Start:
ue at
ps

```
0 ps                 Pointer: 141.19 ns   Interval: 141.19 ns
0 ps     80.0 ns   160.0 ns   240.0 ns   320.0 ns   400.0 ns   480.0 ns   560.0 ns   640.0 ns
0 ps
0000    0001    0010    0011    0100    0101    0110    0111    1000    1001    1010    1011    1100    110
0111111 0000110 1011011 1001111 1100110 1101101 1111101 0000111 1111111 1100111 1110111 1111100 0111001 1011
111
```

Figure 2: Waveform

2 Part B

2.1 Create Top Level

2.1.1 Summary

Take the previous work and covert it into something that would run on the FPGA board. To do this I assigned WXYZ to the buttons and activated the tube on pin_133. The segments were then connected to their corresponding pins and everything was adjusted to be active low.

2.1.2 Difficulties

Initially I had the wrong board set on this project (I created a separate on for Lab 0) as I had clearly not read the whole name string and just matched based on the last three characters. This was an easy fix.

2.1.3 Proof

I actually overwrote this one and had to go back and redo it for this report: QuartusFiles/HexToSSDTopLevel

2.2 Program Board

2.2.1 Summary

To conclude this lab we had to successfully run the top level entry on the FPGA. It had taken me a long time to get the USB blaster working due to having to use the part of the patch to update the driver without installing the entire patch as it would cause Quartus to crash on start.

2.2.2 Difficulties

Due to the previous work done during lab 0 everything worked the first try.

2.2.3 Proof

In this video iterate through a couple numbers to check functionality: SupportingVideo/HexWorking

3 System I/O

Parameter	Min	Typical	Max	Units	Description
VDD	4.5	5	5.5	V	Power
In0	-	0	-	V/logic	Input Button
In1	-	0	-	V/logic	Input Button
In2	-	0	-	V/logic	Input Button
In3	-	0	-	V/logic	Input Button
LED[0..6]	-	0	-	V/logic	Output LED Segments

Table 1: System Inputs and Outputs

4 Use Case

This system changes the state of an LED seven segment display based on the state of the four input buttons. The number displayed being the hexadecimal conversion of the binary inputs given by the button state, with the most significant bit situated on the left. The system needs to be powered by 5V via the USB connection or a battery. The default state is for the LED display to show 0.