ECE M16 and CS M51A Winter 2019 Section 2

Logic Design of Digital Systems

Dr. Yutao He

Verilog Lab #2 - Design of Combinational Systems

Due: February 21, 2019

Team ID: P26

1. **Name: Huynh Derrick**
   Last First

   Student ID: 705101279

   Signature: *Derrick Huynh*

2. **Name: Xia Lucas**
   Last First

   Student ID: 005099109

   Signature: *Lucas Xia*

Date: 2/21/2019

Result	
Correctness	
Creativity	
Report	
Total Score	

Project 2 Report

Abstract

The circuit to be designed is a BCD to 7-segment display decoder. The circuit must be able to convert a given BCD number number to a set of seven outputs corresponding to the segments of the display. The display shows a decimal representation of the outputs as shown:

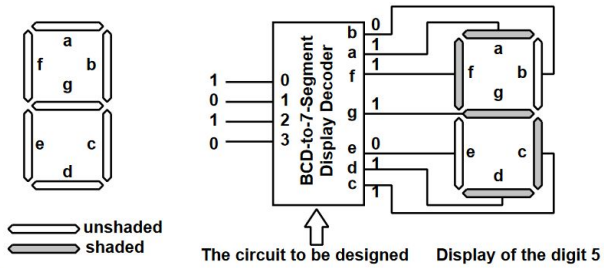


Figure 1: The Specification of how the Circuit would be Used

Decimal No.	a	b	c	d	e	f	g
0	ON	ON	ON	ON	ON	ON	OFF
1	OFF	ON	ON	OFF	OFF	OFF	OFF
2	ON	ON	OFF	ON	ON	OFF	ON
3	ON	ON	ON	ON	OFF	OFF	ON
4	OFF	ON	ON	OFF	OFF	ON	ON
5	ON	OFF	ON	ON	OFF	ON	ON
6	ON	OFF	ON	ON	ON	ON	ON
7	ON	ON	ON	OFF	OFF	OFF	OFF
8	ON	ON	ON	ON	ON	ON	ON
9	ON	ON	ON	ON	OFF	ON	ON

Table 1: The Seven-Segment Display Logic Specification

Figure 2: The Truth Table from Decimal Input to the Seven-segment Outputs.

Decimal No.	BCD Code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Table 2: The Input Encoding Scheme

Segment State	Binary Code
ON	1
OFF	0

Table 3: The Output Encoding Scheme

Figure 3: The Encoding Scheme of Inputs and Outputs.

The truth table above (Figure 2) shows a more in-depth explanation of the circuit with the encoding scheme as shown in Figure 3.

The Switching Functions of the Circuit

Minimal Switching Expressions in NAND-NAND form for each output:
a = (x1' && x3' && ' && (x0' && x2')')'
b = (x3' && (x3' && x2')' && (x0' && x1')' && (x0 && x1)')'
c = (x1' && x0' && x2' && x3')'
d = (x3' && (x2' && x0')' && (x1 && x0')' && (x2' && x1)' && (x2 && x1' && x0)')'
e = ((x2' && x0')' && (x1 && x0')')'
f = ((x2' && x0')' && (x2 && x1')' && (x1' && x0')' && x3')'
g = (x3' && (x2' && x1)' && (x2 && x1')' && (x1 && x0'))'

In this schematic, to reduce the number of gates, some prime implicants that included don't cares were replaced with smaller implicants that were used in other functions so they were shared. This was done in a, f, and g. In a, (x0 && x2) was changed to (x0 && 'x1 && x2) and in f and g, (x2 && x1') was changed to (x0 && 'x1 && x2).

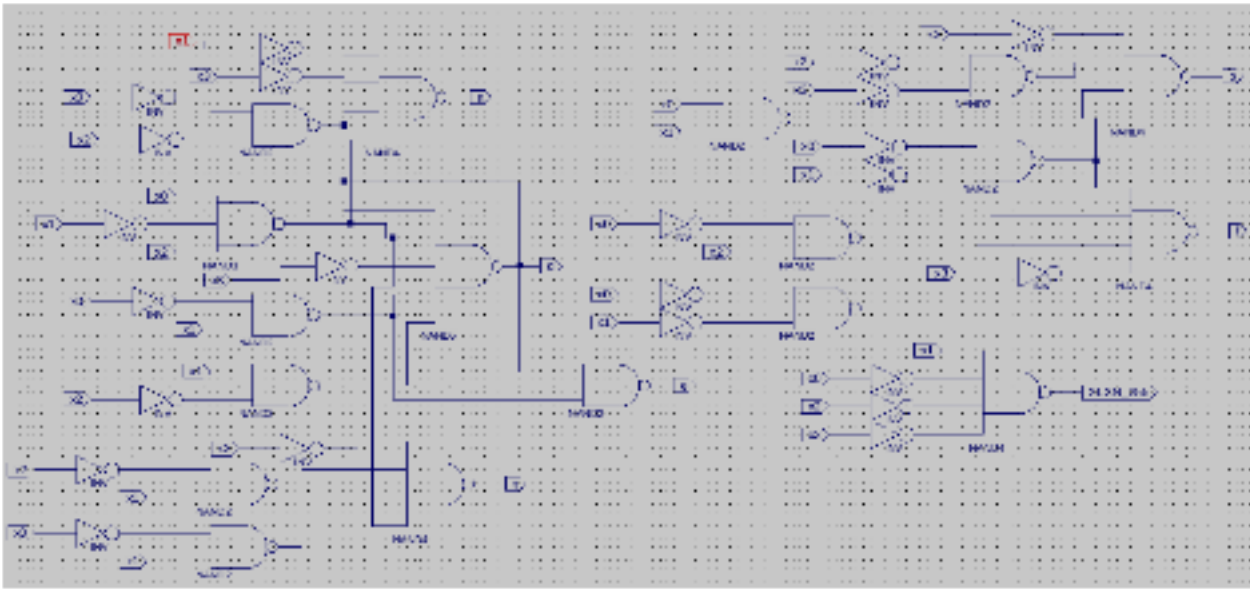


Figure 4: The Schematic of the Circuit

The final minimal switching expressions of the circuit can be found in the Switching Expression section.

Switching Expressions in NAND-NAND form used to minimize network as whole:
a = (x1' && x3' && (x0 && 'x1 && x2)' && (x0' && x2')')'
b = (x3' && (x3' && x2')' && (x0' && x1')' && (x0 && x1)')'
c = (x1' && x0' && x2' && x3')'
d = (x3' && (x2' && x0')' && (x1 && x0')' && (x2' && x1)' && (x2 && x1' && x0)')'

e = ((x2’ && x0’)’ && (x1 && x0’)’)’
f = ((x2’ && x0’)’ && (x0 && ‘x1 && x2)’ && (x1’ && x0’)’ && x3’)’
g = (x3’ && (x2’ && x1)’ && (x0 && ‘x1 && x2)’ && (x1 && x0’))’

Cost Analysis: The minimal amount of 16 NAND gates were used in the schematic, as opposed to original 18 NAND gates used in the individual minimal switching expressions.

The Simulation Result

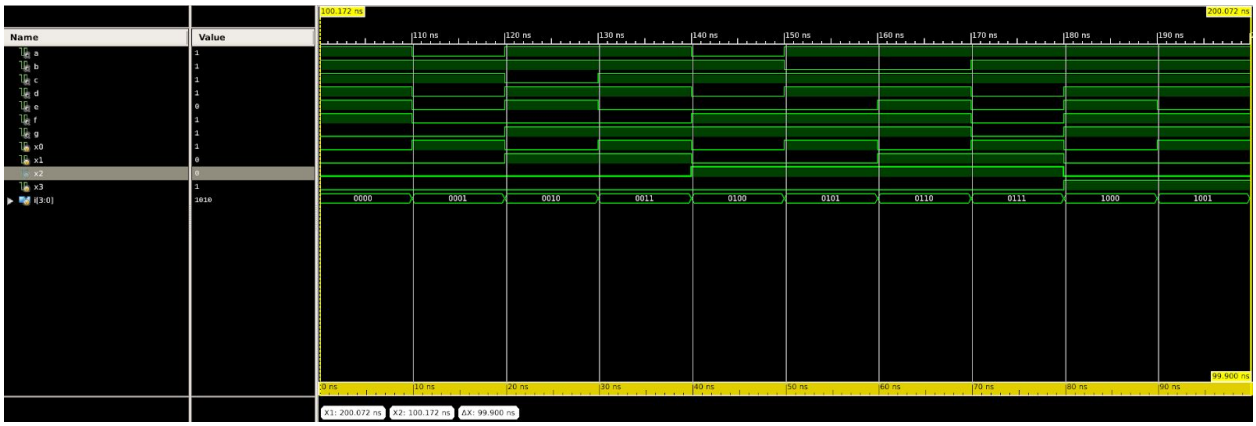


Figure 5: The Timing Diagram of the circuit

Shown in the timing diagram above are the different outputs of the network based on the different inputs. The first 7 lines represent a-g respectively, with the next 4 lines representing the different BCD digits of the decimal input. The last line shows the combined BCD input. Each line of the output (a-g) shows whether that segment of the display is lit. A high bar represents a 1, and means the LED is turned on. A low bar represents 0, or when the LED is off. The ON and OFFs shown in the timing diagram match the encoding scheme given to us in the project outline.

The Design Review

In this project, many of the skills we learned in class and practiced on homework were applied directly to satisfying the project specification. Thus, tasks such as finding the minimal switching expressions for individual outputs using K-maps was not a difficult assignment (despite a couple easily fixed errors in the process). The new feature of using a NAND-NAND network instead of an AND-OR network also was not a challenge as we had done this for class.

Coding in verilog was also not as hard as it was previously, as we could use what we learned from Project 1 and apply it here. On the testbench file, we changed our approach for the test cases and used a for loop instead of manually changing the inputs for more efficiency and bette -written code.

What was more difficult in this project was creating the schematic. More specifically, it was problematic in trying to minimize the number of gates used in the entire circuit, as we had to

consider all seven outputs instead of each one individually. The hardest step of this process was figuring out that using nonessential, non-prime implicants actually could decrease the number of gates (more detail in the schematic, Figure ----).

The extra credit part of the project also was very difficult and we spent most of our time trying to figure it out. We mainly struggled with understanding how to upload code into Mojo FPGA board and changing the LEDs based on an input. We had to experiment with the testbench file to figure out that it did not influence the FPGA at all. Then, we decided to use physical inputs in the form of buttons as the inputs to the circuit. After some hardware errors, we successfully managed to light up the LEDs. The final step was to then make it more visually appealing, so we changed the color scheme to red, white, blue LEDs and physically formed a 7-segment display by moving the LEDs around. A picture of this can be found in the appendix (Figure **----**).

The most important aspects of the project was implementing and creating the schematic for the whole circuit. Creating this two-level implementation network from nothing but a table of possible inputs and the proper outputs was proof that our skills could be applied to designing useful circuits. The extra credit also helped cement this perspective, as we could physically see the circuit work.

Team Member Contributions

Approximate Percentages: Lucas- 47%, Derrick- 53% (roughly equal)

Lucas, Responsibilities and Contributions:
Found switching expressions for outputs d, e, f, g with Karnaugh maps; coded the corresponding parts of the verilog file; coded the constraints file for the extra credit; suggested ideas for hardware design of display; minimized number of gates of output in schematic; helped solve issues that came up in finding switching expressions, coding, and hardware; wrote large parts of the report (abstract, design review, team member contributions, figure descriptions).

Derrick, Responsibilities and Contributions:
Found switching expressions for outputs a, b, c with Karnaugh maps; coded the corresponding parts of the verilog file; coded testbench file; screen-dumped waveform timing diagram; did hardware design of the 7-segment display; created circuit schematic using the Xilinx ISE; created cover page on the report, and worked on the switching expressions and timing diagram sections of the report.

Appendix - The Detailed Design Worksheet

All the Figures used are shown below:

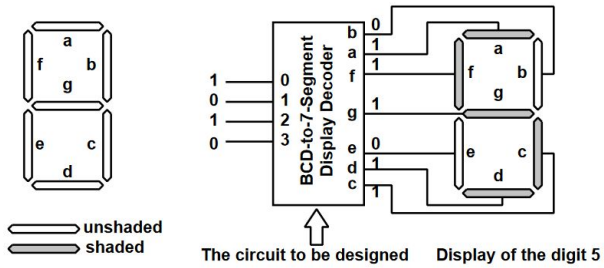


Figure 1: The Specification of how the Circuit would be used

Decimal No.	a	b	c	d	e	f	g
0	ON	ON	ON	ON	ON	ON	OFF
1	OFF	ON	ON	OFF	OFF	OFF	OFF
2	ON	ON	OFF	ON	ON	OFF	ON
3	ON	ON	ON	ON	OFF	OFF	ON
4	OFF	ON	ON	OFF	OFF	ON	ON
5	ON	OFF	ON	ON	OFF	ON	ON
6	ON	OFF	ON	ON	ON	ON	ON
7	ON	ON	ON	OFF	OFF	OFF	OFF
8	ON	ON	ON	ON	ON	ON	ON
9	ON	ON	ON	ON	OFF	ON	ON

Table 1: The Seven-Segment Display Logic Specification

Figure 2: The Truth Table from Decimal Input to the Seven-segment Outputs.

Decimal No.	BCD Code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Table 2: The Input Encoding Scheme

Segment State	Binary Code
ON	1
OFF	0

Table 3: The Output Encoding Scheme

Figure 3: The Encoding Scheme of Inputs and Outputs.

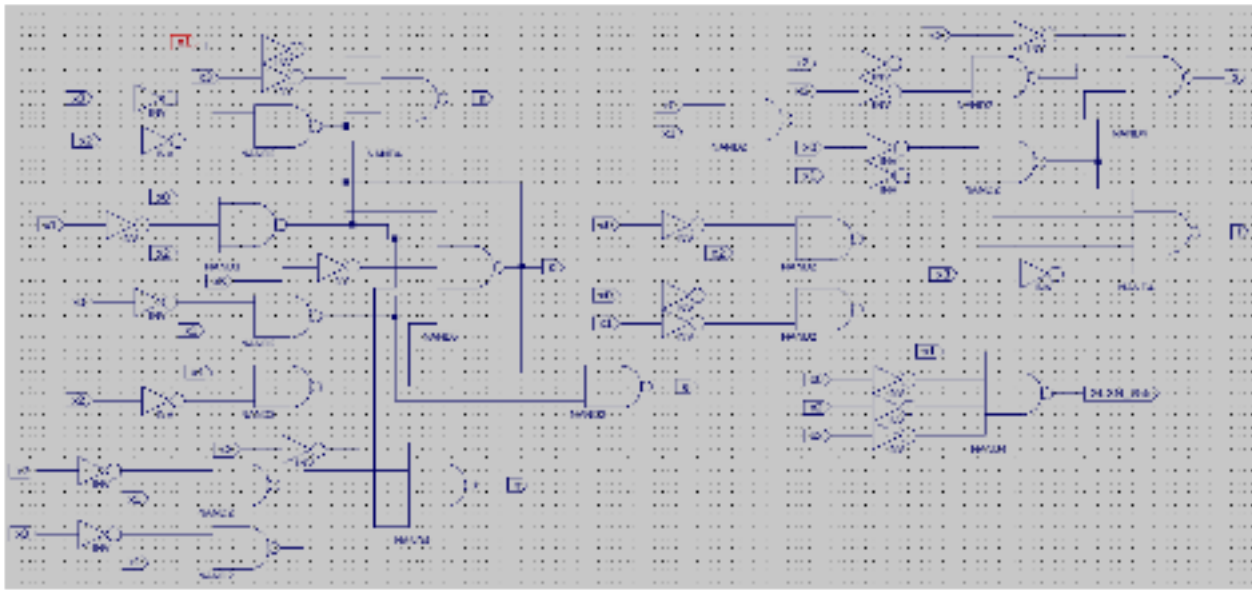


Figure 4: The Schematic of the Circuit

The final minimal switching expressions of the circuit can be found in the Switching Expression section.

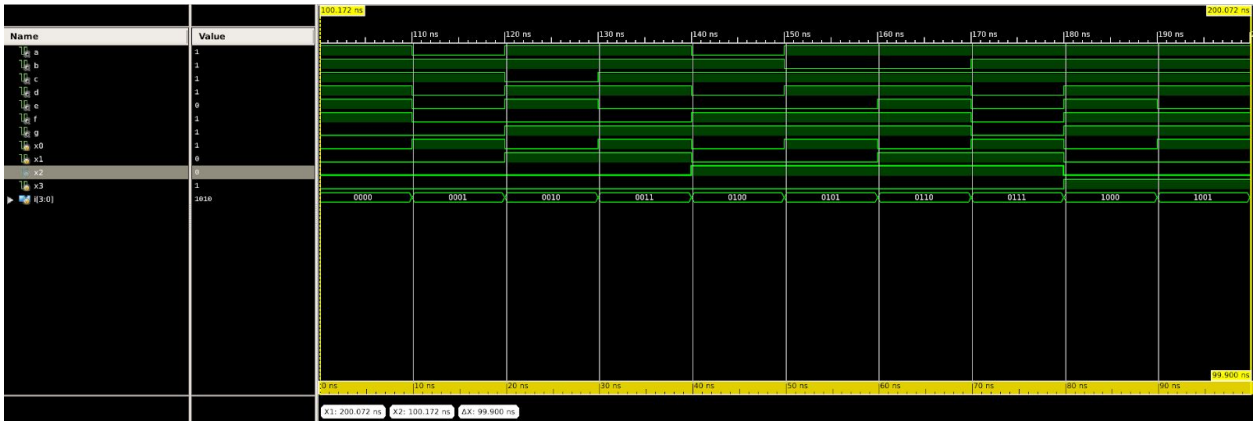


Figure 5: The Timing Diagram of the circuit

4 bit input — 1 bit output: (1, 0)

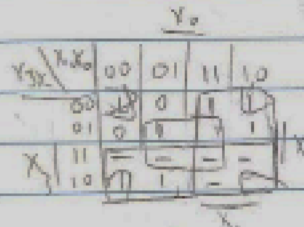
Inputs: X_3, X_2, X_1, X_0 — Outputs: a, b, c, d, e, f, g (ON = 1, OFF = 0)

1 digit input (decimal) — 4 bit input $X_3X_2X_1X_0$

1.

Decimal	BCD	a	b	c	d	e	f	g
0	0000	1	1	1	1	1	1	0
1	0001	0	1	1	0	0	0	0
2	0010	1	1	0	1	1	0	1
3	0011	1	1	1	1	0	0	1
4	0100	0	1	1	0	0	1	1
5	0101	1	0	1	1	0	1	1
6	0110	1	0	1	1	1	1	1
7	0111	1	1	1	0	0	0	0
8	1000	1	1	1	1	1	1	1
9	1001	1	1	1	1	0	1	1

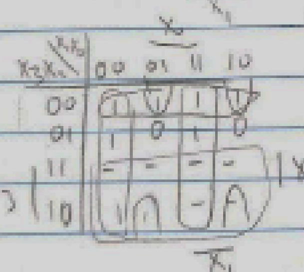
for A)



$A = X_1 + X_3 + X_0X_2 + X_0'X_2'$ AND-OR

$= \overline{(X_1' \cdot X_3' \cdot (\overline{X_0X_2}) \cdot (\overline{X_0'X_2'}))}$ NAND-NAND

B:

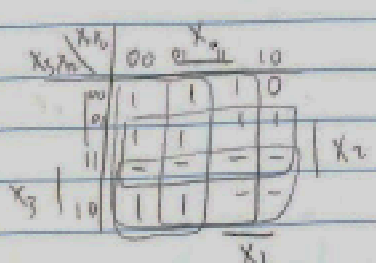


$B = X_3 + X_3'X_2' + X_0X_1' + X_0X_1$ AND OR

$(X_2' X_0X_1 + X_2'X_0'X_1')$

$= \overline{(X_3' (\overline{X_3'X_2'}) (\overline{X_0X_1'}) (\overline{X_0X_1}))}$ NAND-NAND

C:



$C: X_1' + X_3 + X_2 + X_0$ AND OR

$= \overline{X_1 X_3' X_2' X_0'}$ NAND-NAND

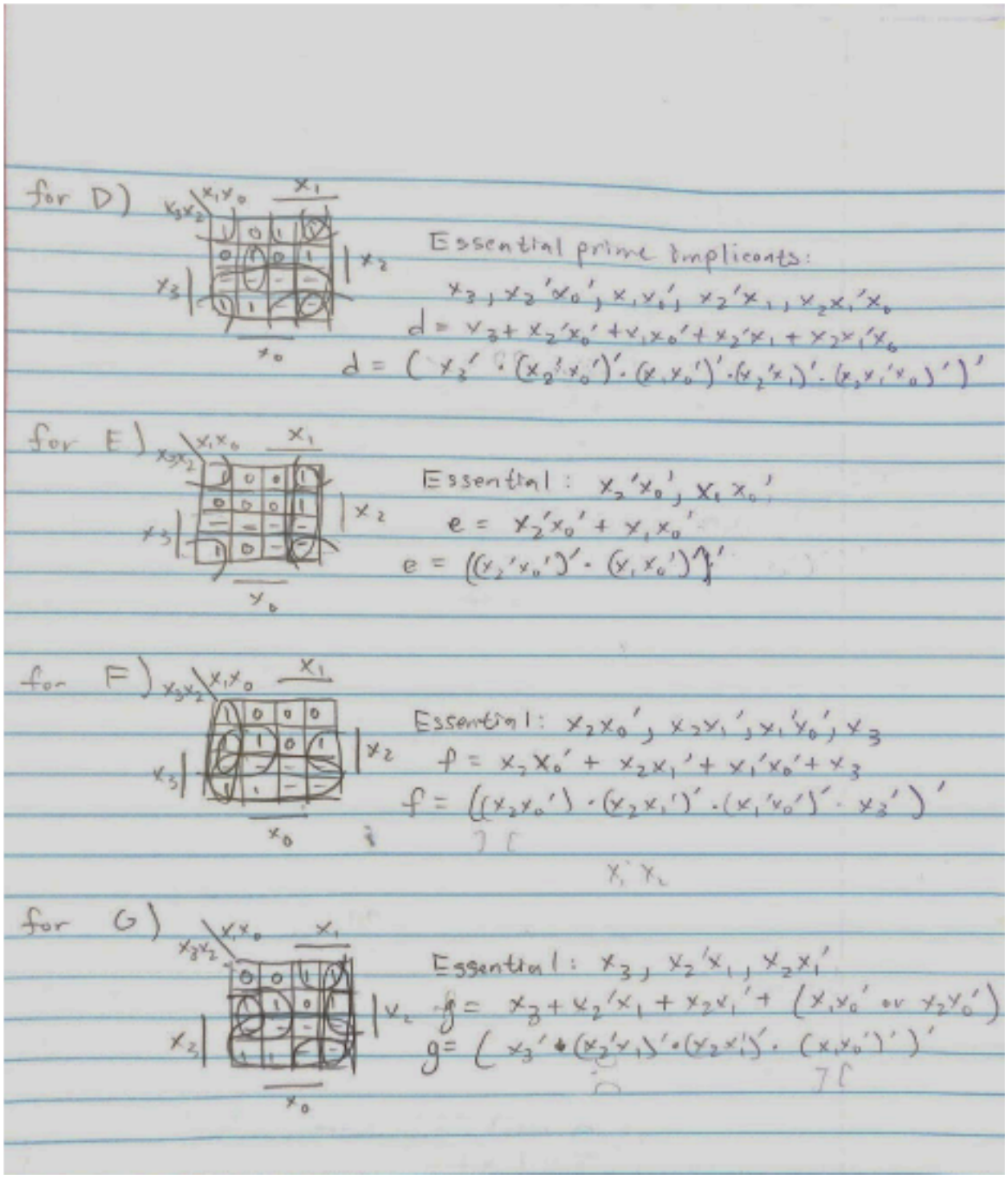


Figure 6: Handwritten work of Truth Table, Encoding Schemes of Input/Output, K-maps, Essential Prime Implicants, Minimal Switching Expressions, Conversion from AND-OR to NAND-NAND.

Conversion from AND-OR to NAND-NAND consisted of changing all ORs to ANDs and then complementing all the gates.

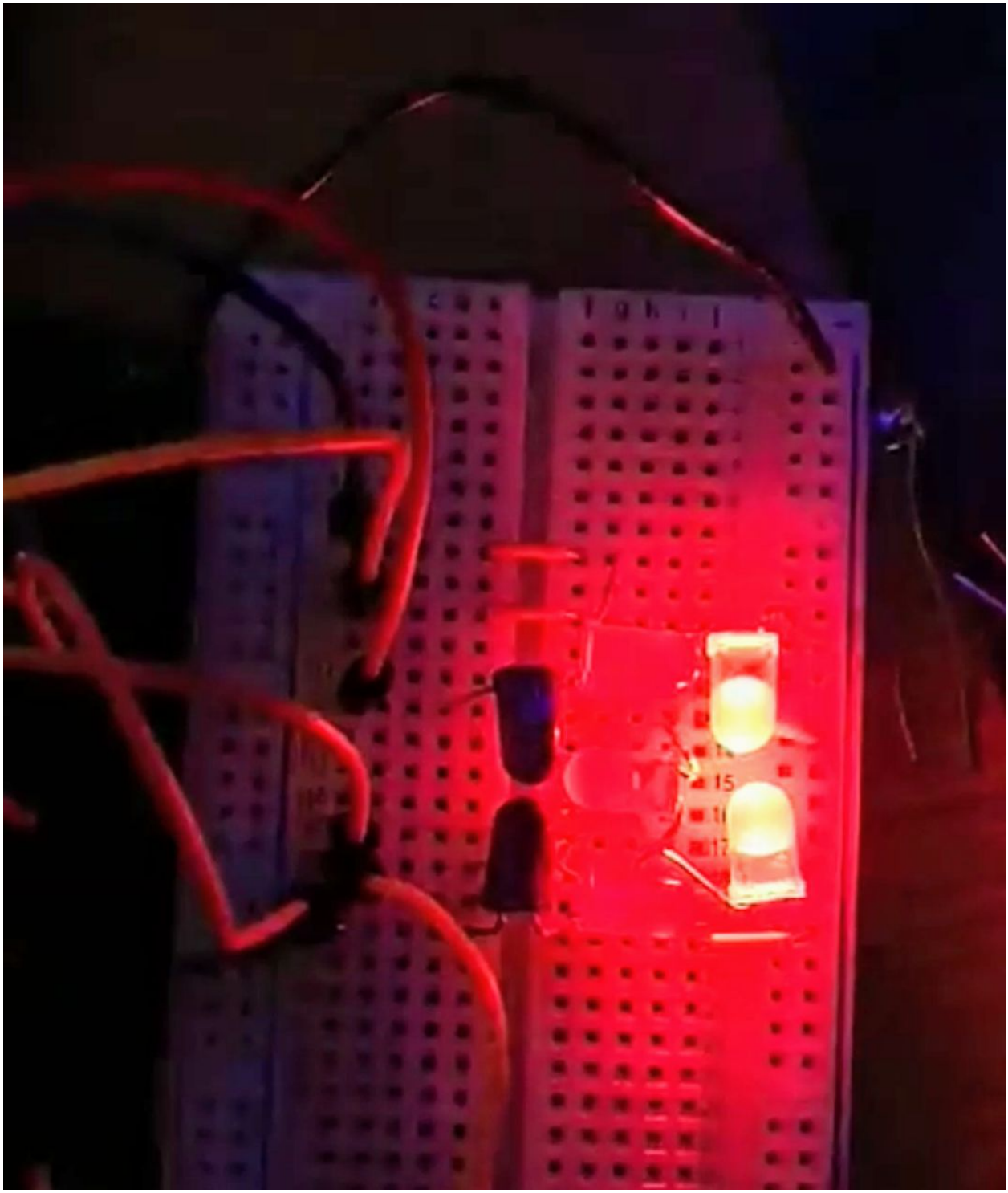


Figure 7: Display of Mojo FPGA Development Board with the circuit programmed with our code, with input of 0001 (or 1 in decimal)

The above is part of our extra credit, which we demo-ed, but for completeness, a picture of it working is included.