[[1]](#footnote-1)

CmpE 124 Lab 5: Mixed-Logic Design 2

Anahit Sarao, 008435583, CmpE 124 Spring 2015, Lab Section 2

*Abstract*— This lab goes into more depth about mixed logic circuit design. It is a emphasis on lab 4 and goes into further detail about designing circuits from given equations.

# INTRODUCTION

The purpose of this lab was to create logical circuits from the seven equations that were given. From the equations a truth table was created and KMAPs were used to reduce the equations. By using the NOR and NAND gates an equivalent circuit was designed. The equations needed to be simplified to ensure efficiency and fewer hazards. This lab was done using Logicworks software.

# Design methodology

## Parts List

* Logicworks

## Truth Tables

1. f = n3 ' [n1 + (n2 xor n0)'] + n3n2n1'

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n3 | n2 | n1 | n0 | F |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

2. f = n3'n2' + n2[n1 ' n0 ' + n0(n3 xor n1)]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n3 | n2 | n1 | n0 | F |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

3. f = n3’(n1' + n0) + n2(n1' + n3)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n3 | n2 | n1 | n0 | F |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

4. f = (n3 xor n0) + n3 ' [n2 ' (n1 + n0') + n1n0']

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n3 | n2 | n1 | n0 | F |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

5. f = n0(n3 xor n2n1')

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n3 | n2 | n1 | n0 | F |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

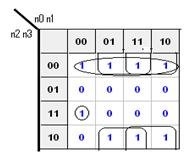
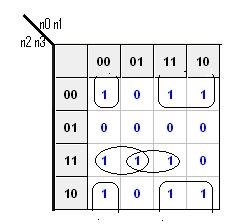
6. f = n2(n1' + n2 ' n0 ' ) + n2 ' n1 ' n0 '

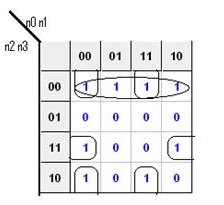
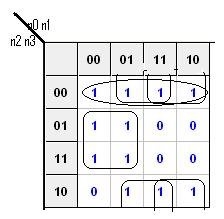
|  |  |  |  |
| --- | --- | --- | --- |
| n2 | n1 | n0 | F |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

7. f = n2n1' + n3 ' n1(n0 ' + n2 ' )

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n3 | n2 | n1 | n0 | F |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

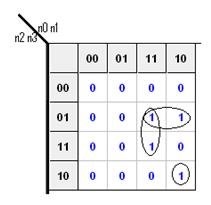
## Karnaugh Maps

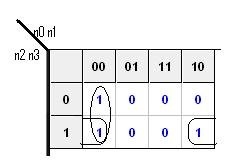
1 2

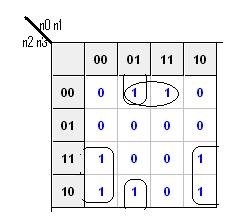
4

3

5



6



7

## Original and Derived Equations

Original Equations

1. f = n3' [n1 + (n2 xor n0)'] + n3n2n1'

2. f = n3'n2' + n2[n1'n0'+ n0(n3 xor n1)]

3. f = n3’(n1' + n0) + n2(n1' + n3)

4. f = (n3 xor n0) + n3' [n2'(n1 + n0') + n1n0']

5. f = n0(n3 xor n2n1')

6. f = n2(n1' + n2'n0') + n2’n1’n0'

7. f = n2n1' + n3'n1(n0'+ n2')

Simplified Equations

1. f=(n0`+n1+n2)(n3 + n2`+n1+n0)(n3`n1’)(n3`+n2)

2. f=n0 n1n3` + n1`n2n3` + n2``n3` + n0`n1n3`

3. f=(n3’+n2)(n3+n1’+n0)

4. f=(n3’+n0’)(n3+n2’+n1+n0)

5. f=(n1+n0)(n3+n2)(n1’+n0)(n3+n1’)(n3’+n2’+n1)

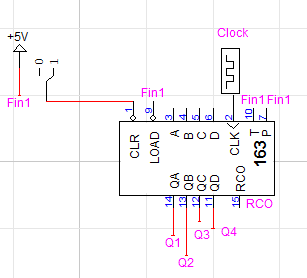
6. f=(n2+n0’)(n1)

7. f= (n3’+n1’)(n2+n1)(n2’+n1+n0)

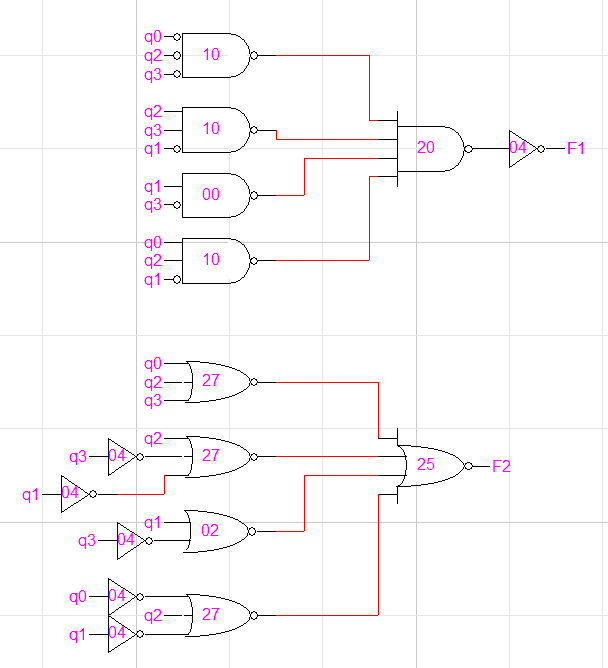
All equations used for deriving the circuit design will be noted here.

## Schematics

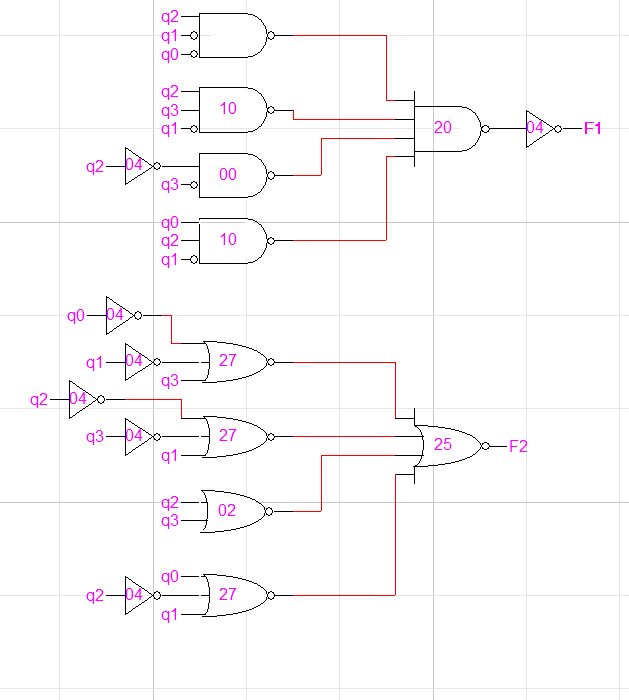
The 163 Clock was the Input for all schematics.



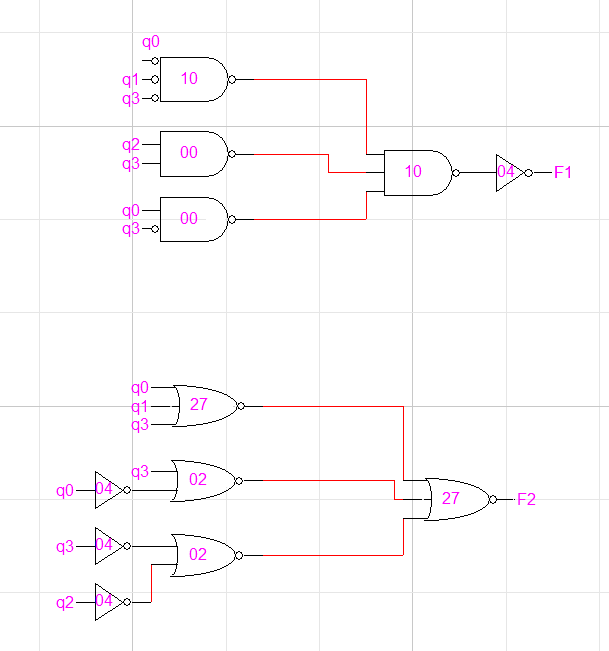
1. f = n3' [n1 + (n2 xor n0)'] + n3n2n1'



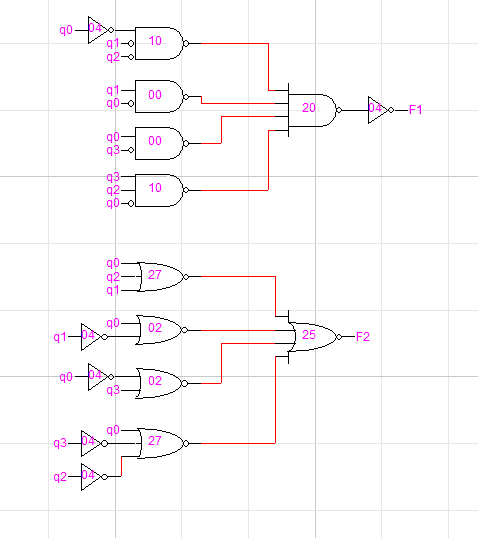
2. f = n3'n2' + n2[n1'n0'+ n0(n3 xor n1)]

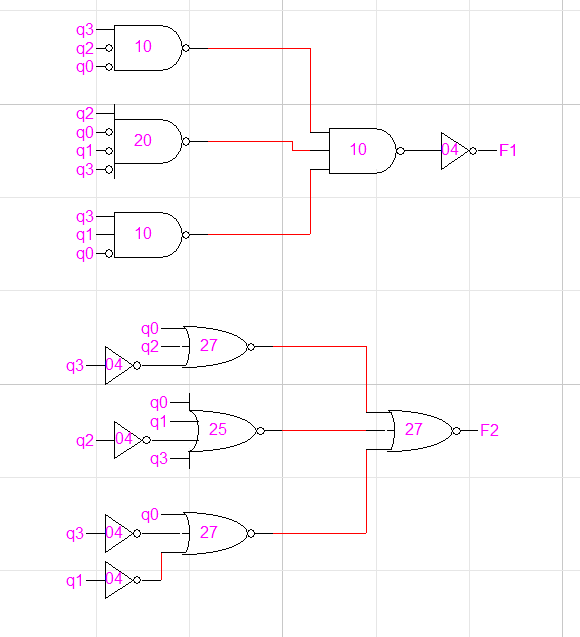


3. f = n3’(n1' + n0) + n2(n1' + n3)

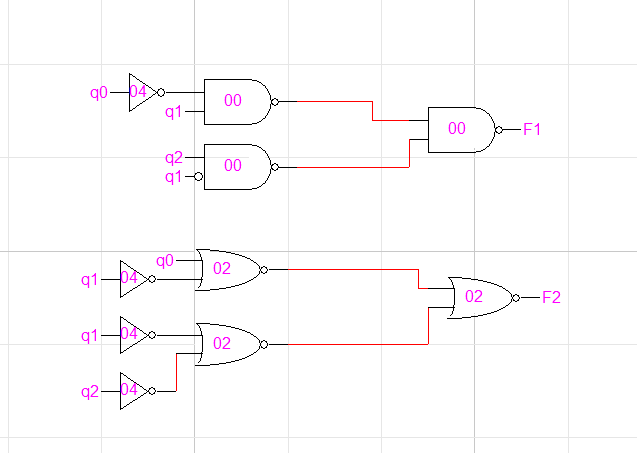


4. f = (n3 xor n0) + n3' [n2'(n1 + n0') + n1n0']

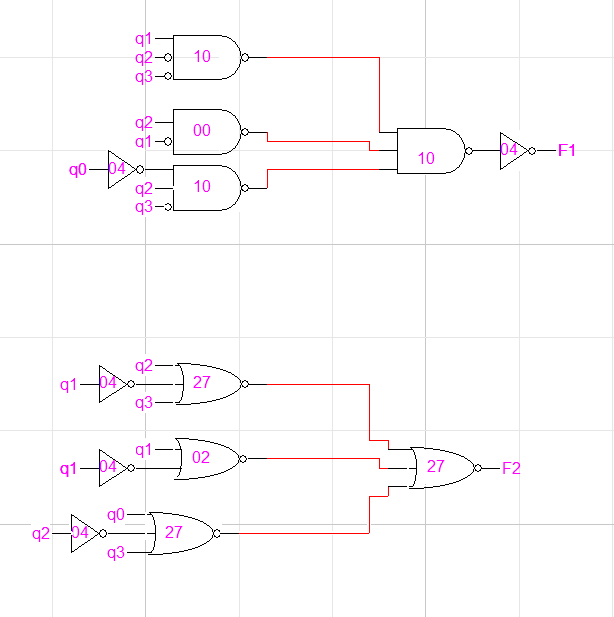


5. f = n0(n3 xor n2n1')

6. f = n2(n1' + n2'n0') + n2’n1’n0'



7. f = n2n1' + n3'n1(n0'+ n2')



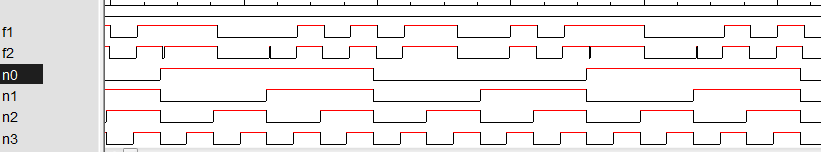
# testing procedures

1. Create a KMAP for each equation, then reduce the equation to the simplest form.
2. Make a truth and voltage table that satisfies the new equation.
3. Create the NOR and NAND of each equation.
4. Design and create the logical circuit.
5. Test the circuit using the truth table to see if the logical circuit is designed properly.

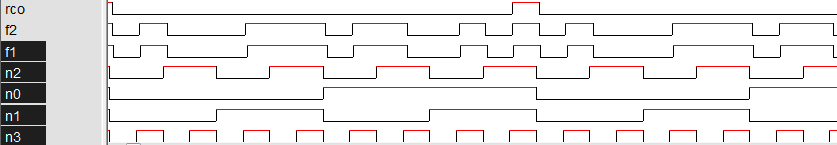
Simulation Waveforms for Each Schematic and Equation

N0,1,2,3 Represent the inputs while f1, and f2 are the outputs

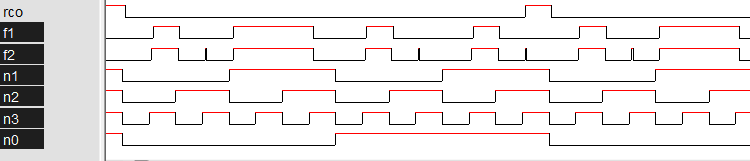
1. f = n3' [n1 + (n2 xor n0)'] + n3n2n1'



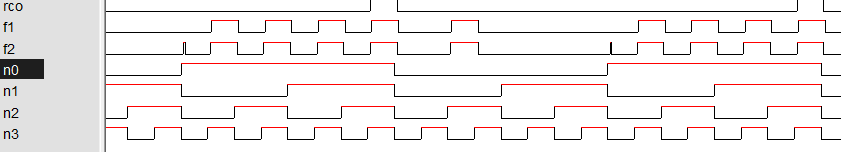
2. f = n3'n2' + n2[n1'n0'+ n0(n3 xor n1)]



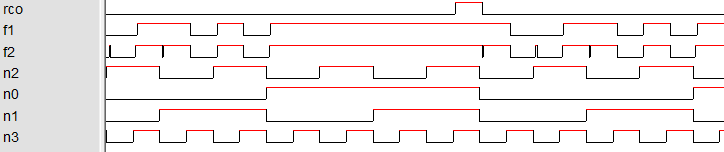
3. f = n3’(n1' + n0) + n2(n1' + n3)



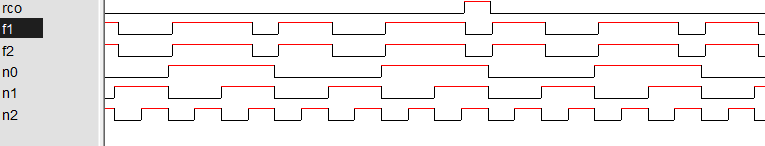
4. f = (n3 xor n0) + n3' [n2'(n1 + n0') + n1n0']



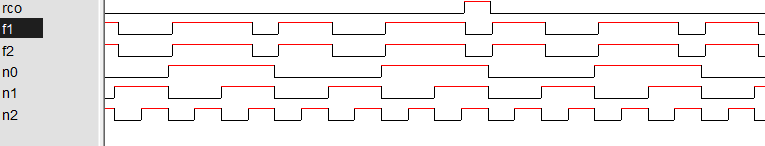
5. f = n0(n3 xor n2n1')



6. f = n2(n1' + n2'n0') + n2’n1’n0'



7. f = n2n1' + n3'n1(n0'+ n2')



# testing results

The testing results for this lab are straight forward as the NOR and NAND waveforms match each other. This means that any equation can be changed using De Morgan’s Theorem. Also it shows that any simplified version of an equation represents the original form also. Some of the waveforms contain glitches this is due to the propagation delay caused when using many gates with more than 2 inputs. Higher input gates not only increase cost to produce the circuit but they also increase the chance of glitches and hazards.

# Conclusion

The main purpose of this lab to observe how reducing equations can change the logical circuit and provide many benefits. By using Demorgans theorems in mixed logic design many equation can be reduced to reduce glitches and use less gates. This is shown in this lab as the NOR gate produced glitches making it less efficient compared to the NAND gate or NOR gate.

# appendices and references

1. Anahit Sarao, indianvip60@gmail.com [↑](#footnote-ref-1)