# EEL 3701 – Digital Logic and Computer Systems

# Short Report Design Problem 2

# Greg Bolling

# UF ID 64911960

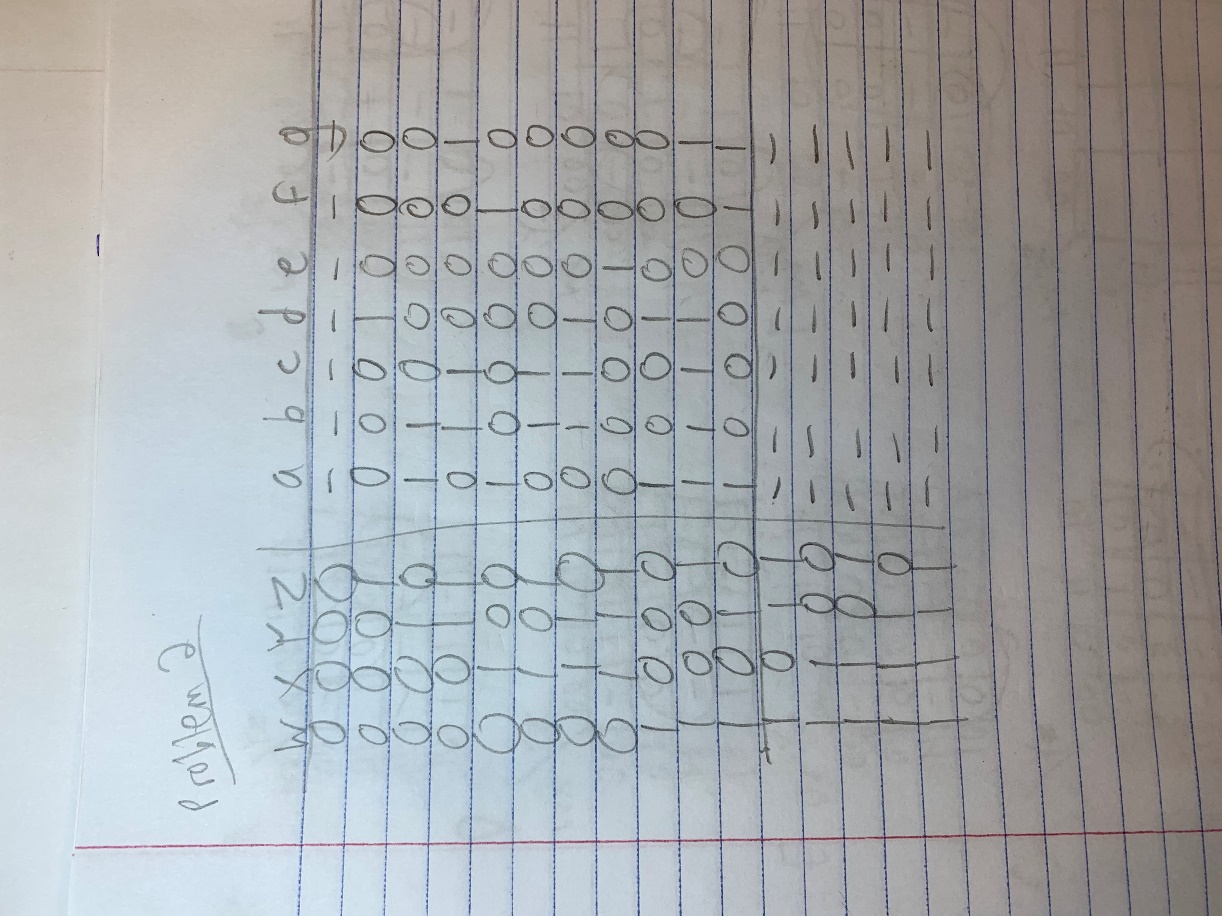
## Problem Statement

The goal of the design was to create letters A..J using the seven segment display..

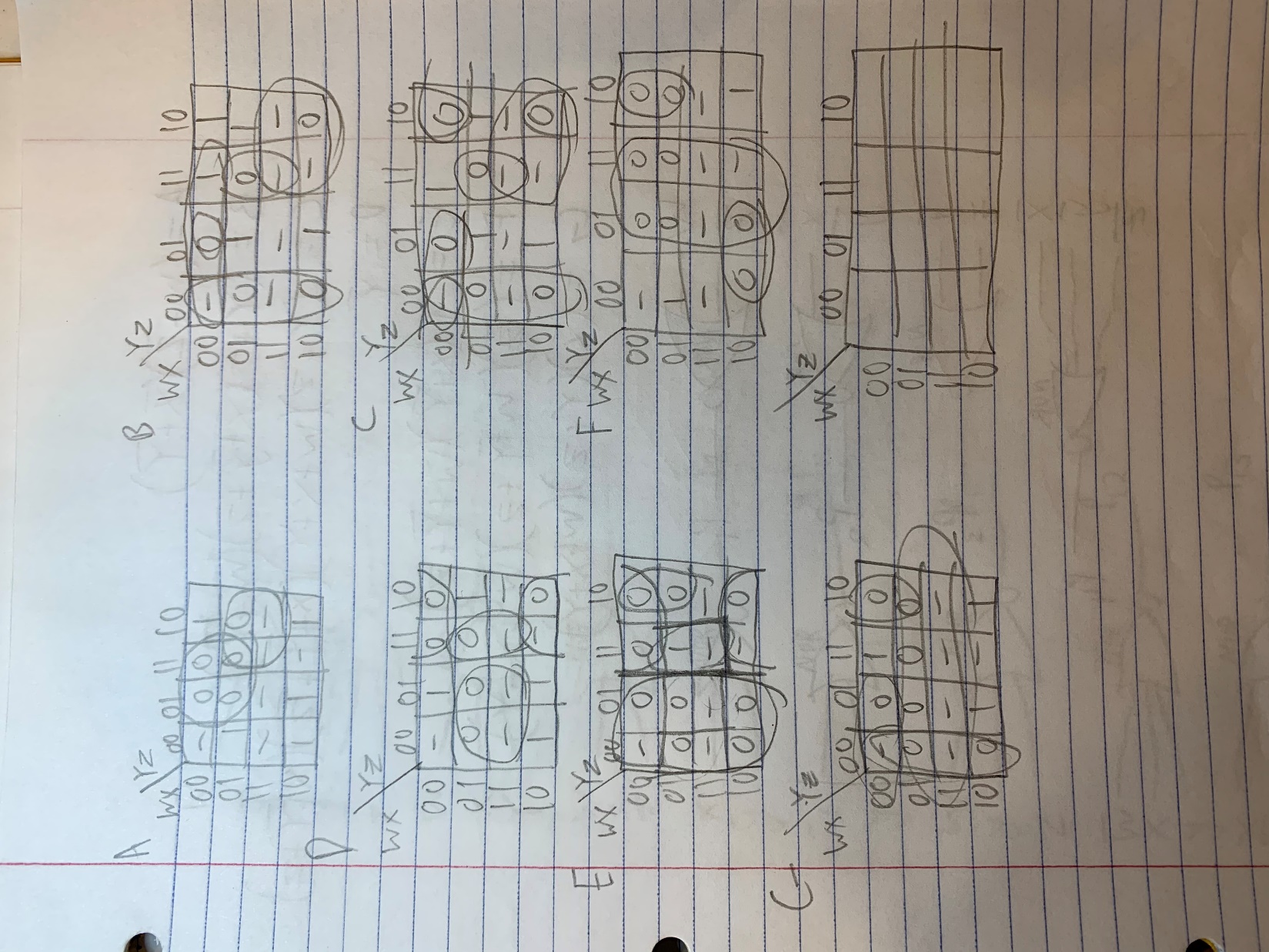
## Design

I implemented this design as a Product of Sums to make sure it was a NOR-NOR implementation. This meant using K-Maps and circling the zeros. I also tried to circle as many common terms across the seven KMaps as possible. I then wrote the equations and counted up 13 first level NOR gates and 7 more after that. There might have been a three others to create an inverter, but that depends on whether the inverted single input was available.

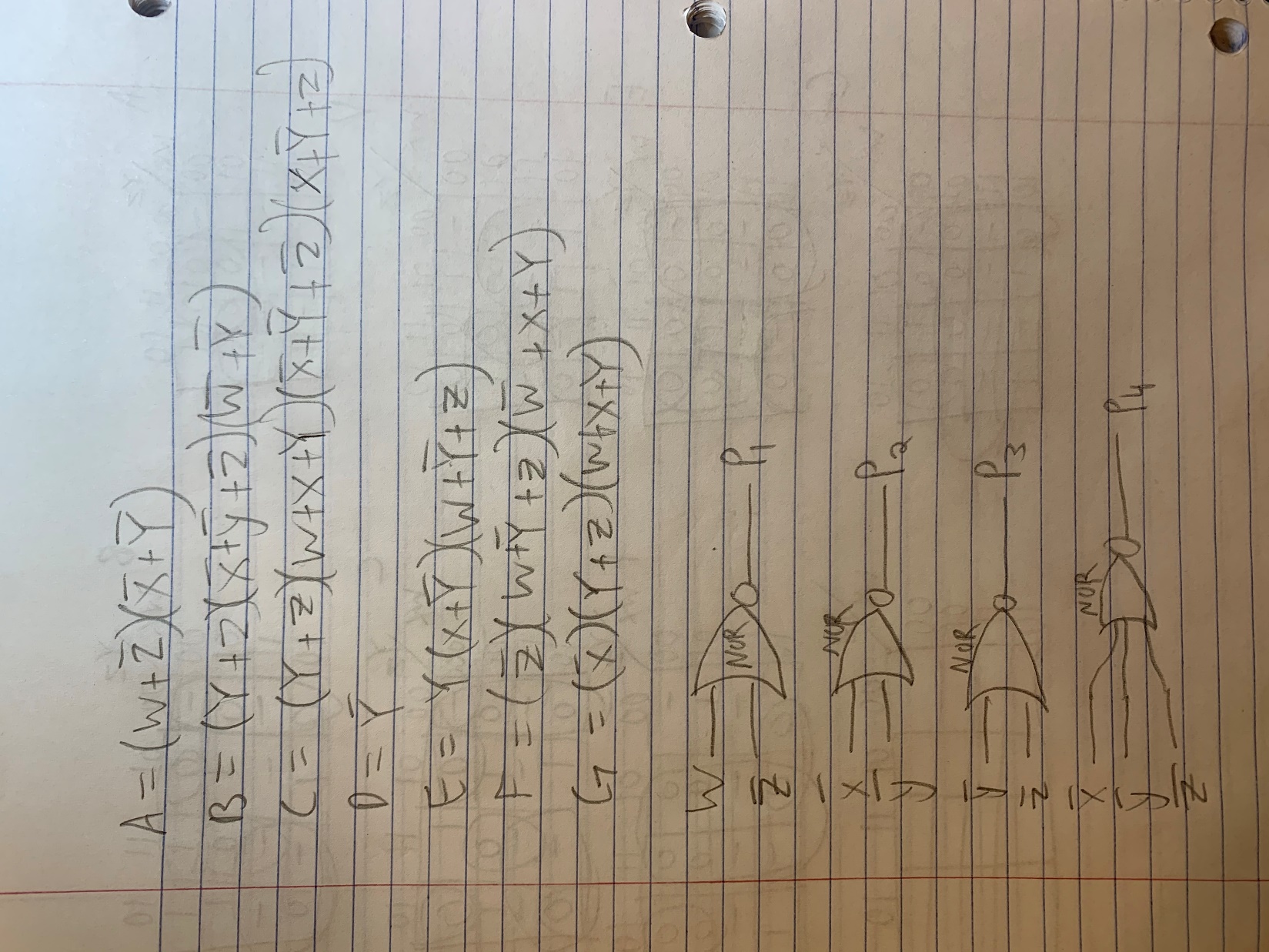
Truth Table

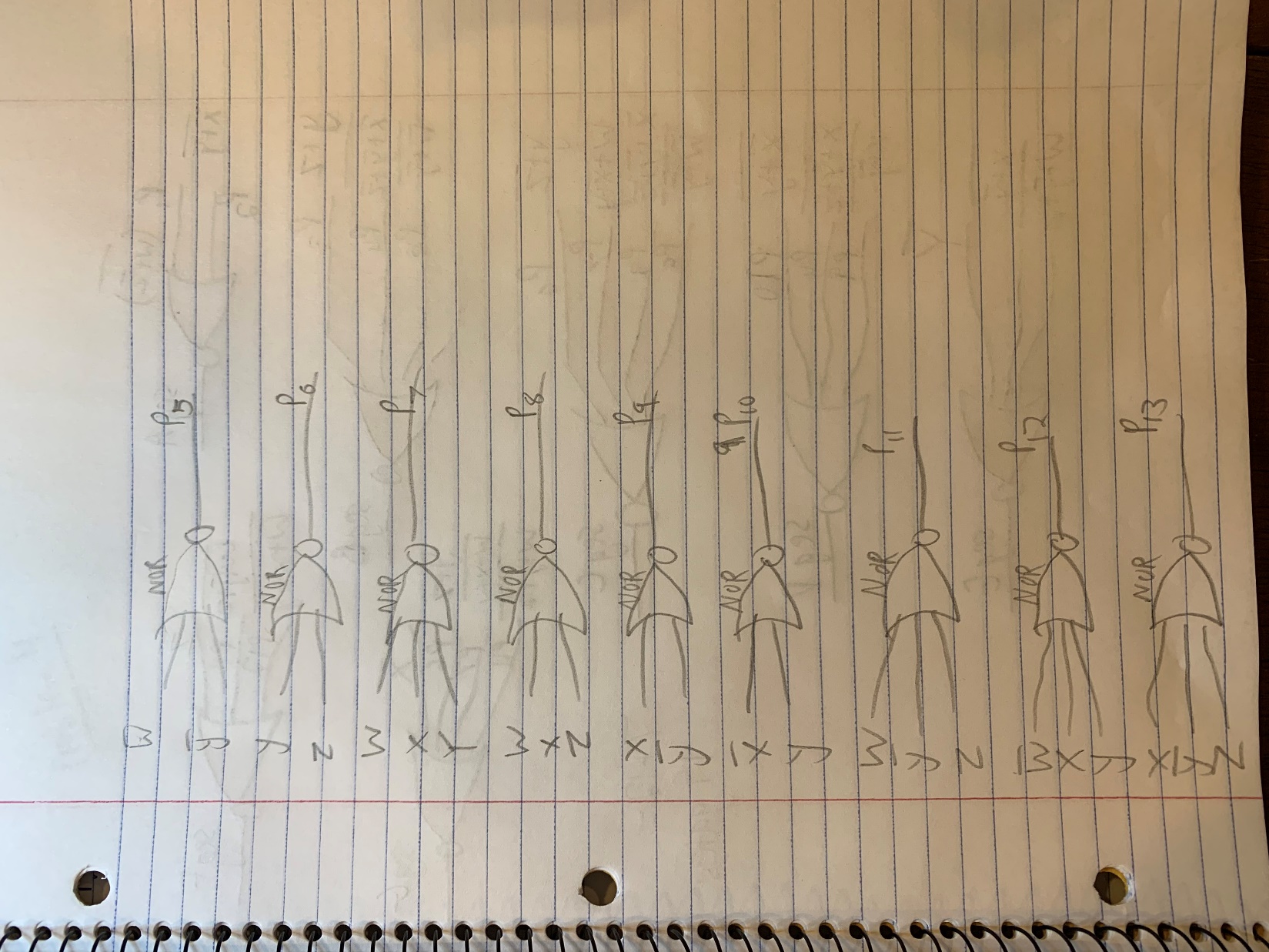


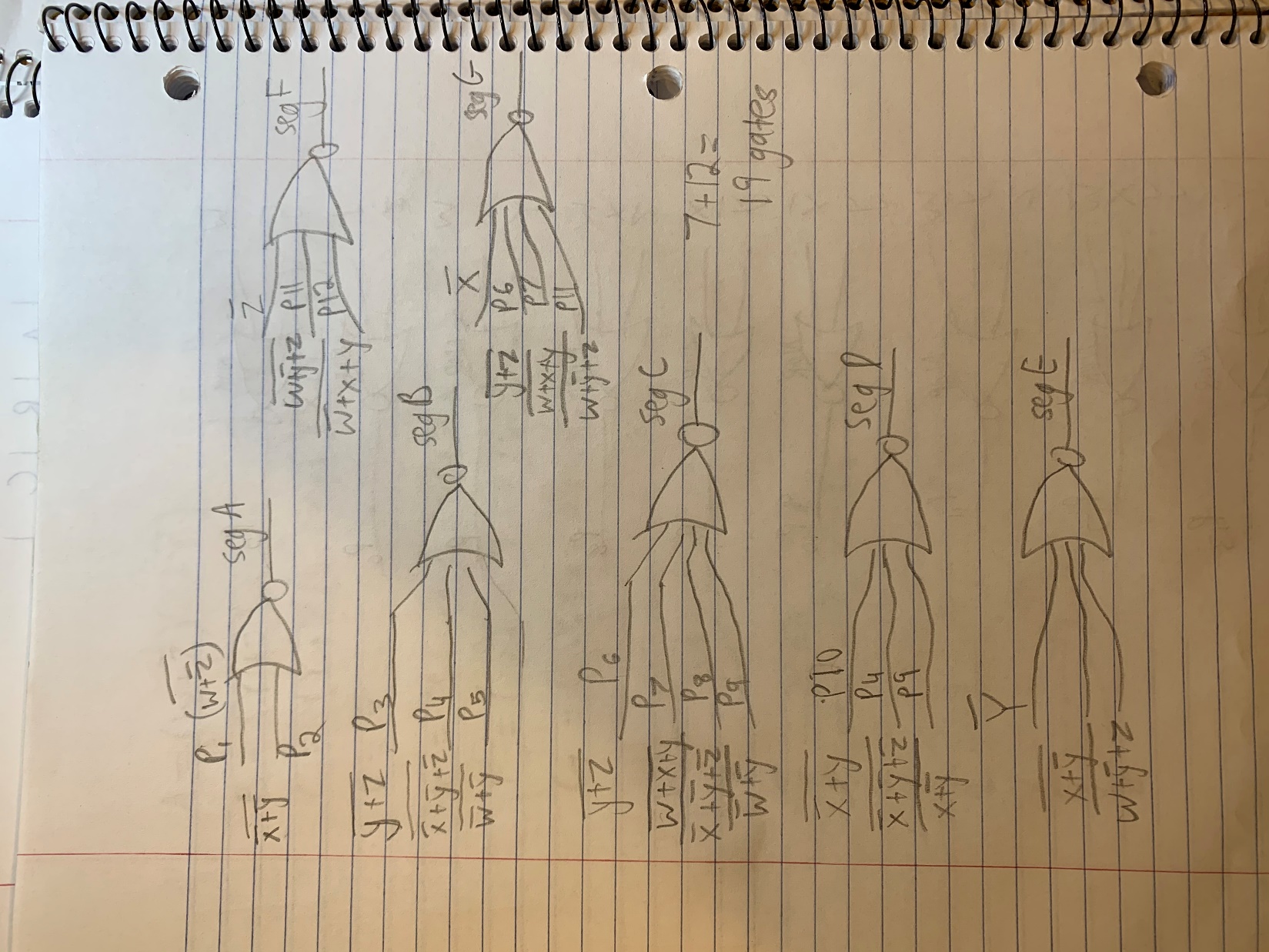
K-Maps



Equations and schematic from Max Terms (reused a lot of them so created terms individually and connected them by label)







## Implementation

I implemented this with the board we have from class. It was done in VHDL with equations as follows:

There wasn’t a 4-input NOR gate so I used an OR gate and inverted it to create a 4-input NOR gate.

-------------------------------------------------------------------------------

-- Author: Greg Bolling

-- Date: March 12, 2020

-- Class; EEL 3701

-- Assignment: DesignProblem 2 Top Level Entity

-- Class: 12368

-- Section 7441

-- PI Name: Savvas Ferekides

-------------------------------------------------------------------------------

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.numeric\_std.all;

use ieee.std\_logic\_unsigned.all;

entity DesignProblem2 is

port (

Switch1 : in std\_logic;

Switch2 : in std\_logic;

Switch3 : in std\_logic;

Switch4 : in std\_logic;

SevenSeg : out std\_logic\_vector(6 DOWNTO 0);

display : out std\_logic\_vector(3 downto 0)

);

end DesignProblem2;

architecture AttachAll of DesignProblem2 is

signal w : std\_logic;

signal x : std\_logic;

signal y : std\_logic;

signal z : std\_logic;

signal Prod1 : std\_logic;

signal Prod2 : std\_logic;

signal Prod3 : std\_logic;

signal Prod4 : std\_logic;

signal Prod5 : std\_logic;

signal Prod6 : std\_logic;

signal Prod7 : std\_logic;

signal Prod8 : std\_logic;

signal Prod9 : std\_logic;

signal Prod10: std\_logic;

signal Prod11: std\_logic;

signal Prod12: std\_logic;

signal Prod13: std\_logic;

signal SegA : std\_logic;

signal SegB : std\_logic;

signal SegC : std\_logic;

signal SegD : std\_logic;

signal SegE : std\_logic;

signal SegF : std\_logic;

signal SegG : std\_logic;

begin

w <= Switch1;

x <= Switch2;

y <= Switch3;

z <= Switch4;

display <= "0001";

Prod1 <= W nor (not z);

Prod2 <= (not x) nor (not y);

Prod3 <= not y nor not z;

Prod4 <= not( (not x or not y) or not z);

Prod5 <= not w nor not y;

Prod6 <= y nor z;

Prod7 <= not( (w or x) or Y);

Prod8 <= not( (w or x) or z);

Prod9 <= x nor not y;

Prod10 <= not x nor y;

Prod11 <= not( (w or not y) or z );

Prod12 <= not( (not w or x) or y );

Prod13 <= not( (x or not y) or z );

--SegA <= Switch1;

--SegB <= Switch2;

--SegC <= Switch3;

--SegD <= Switch4;

--SegE <= '1';

--SegF <= '1';

--SegG <= '1';

SegA <= Prod1 nor Prod2;

SegB <= not( (Prod6 or Prod4) or (Prod5 or Prod7) );

SegC <= not( (Prod6 or Prod7) or (Prod4 or Prod13) );

SegD <= not( (Prod10 or Prod4) or Prod9 );

SegE <= not( (not Y or Prod9) or Prod11 );

SegF <= not( ( z or Prod11) or Prod12 );

SegG <= not( ( x or Prod6) or (Prod7 or Prod11) );

SevenSeg <= SegG & SegF & SegE & SegD & SegC & SegB & SegA;

end AttachAll;