Library File:

AND2, 2, i1&i2, 200
OR2, 2, i1|i2, 200
NAND2, 2, ~(i1&i2), 150
NOT, 1, ~i1, 50
XOR2, 2, (i1&~i2)|(~i1&i2), 300
NOR2, 2, ~(i1|i2), 150
AND3, 3, i1&i2&i3, 300
OR3, 3, i1|i2|i3, 300
NAND3, 3, ~(i1&i2&i3), 250
NOR3, 3, ~(i1|i2|i3), 250
AND4, 4, i1&i2&i3&i4, 400
OR4, 4, i1|i2|i3|i4, 400
NAND4, 4, ~(i1&i2&i3&i4), 350
NOR4, 4, ~(i1|i2|i3|i4), 350

Circuit Files:

Circuit 1 File:

INPUTS:
A
B
C
COMPONENTS:
G0, XOR2, W0, A, B
G1, NAND2, W0, W1, C
G2, NOR4, W2, A, B, C, W1

Circuit 2 File:

INPUTS:
A
B
C
D
COMPONENTS:
G0, NOT, W0, B
G1, NOT, W1, C
G2, AND2, W2, A, B
G3, AND2, W3, W0, W1
G4, OR2, W4, C, D
G5, NOR2, W5, W2, W3
G6, NAND2, W6, W4, W5

Circuit 3 File:

INPUTS:
A
B
COMPONENTS:
G0, NOT, w0, A
G1, NAND2, w1, w0,B
G2, NAND2, w2, B, w0
G3, NAND2, w3, w1, w2

Circuit 4 File:
INPUTS:
A
B
C
D
COMPONENTS:
G0, NAND2, W0, A, B
G1, OR2, W1, C, D
G2, NAND2, W2, W0, W1

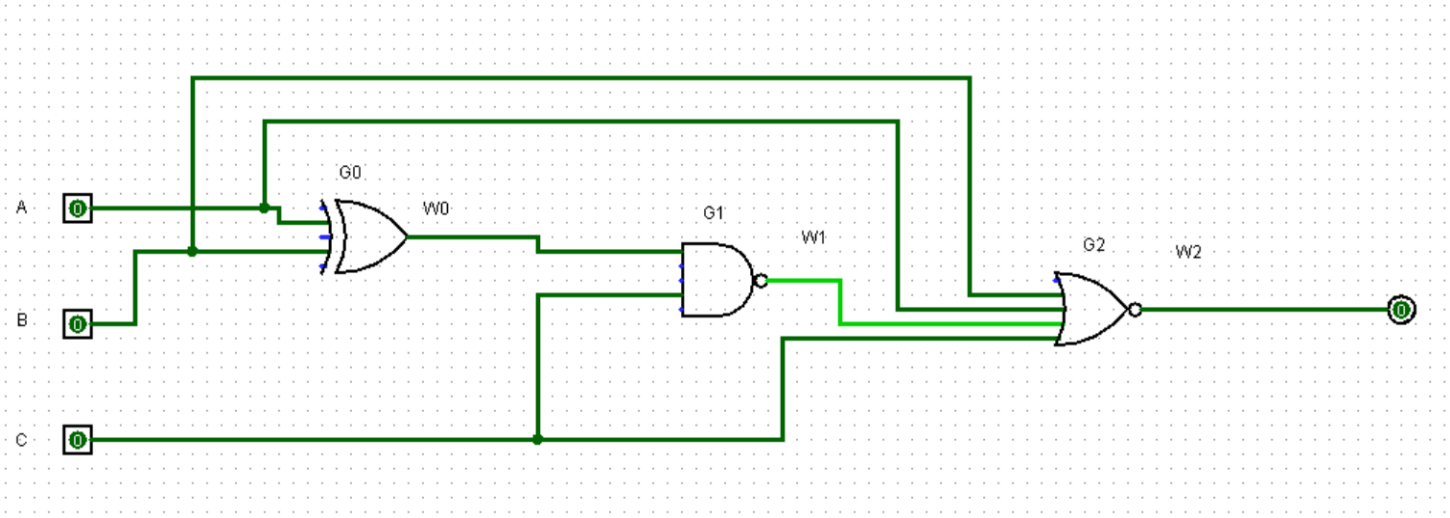
Circuit 5 File:
INPUTS:
A
B
C
COMPONENTS:
G0, NOR2, W0, A, B
G1, NOT, W1, C
G2, NAND2, W2, W0, W1

Circuit 6 File:
INPUTS:
A
B
C
D
COMPONENTS:
G0, NAND2, W0, A
G1, NAND3, W1, A, B, C
G2, NOR2, W2, C, D
G3, NAND4, W3, W0, W1, W2, D

Truth Tables & Diagrams:

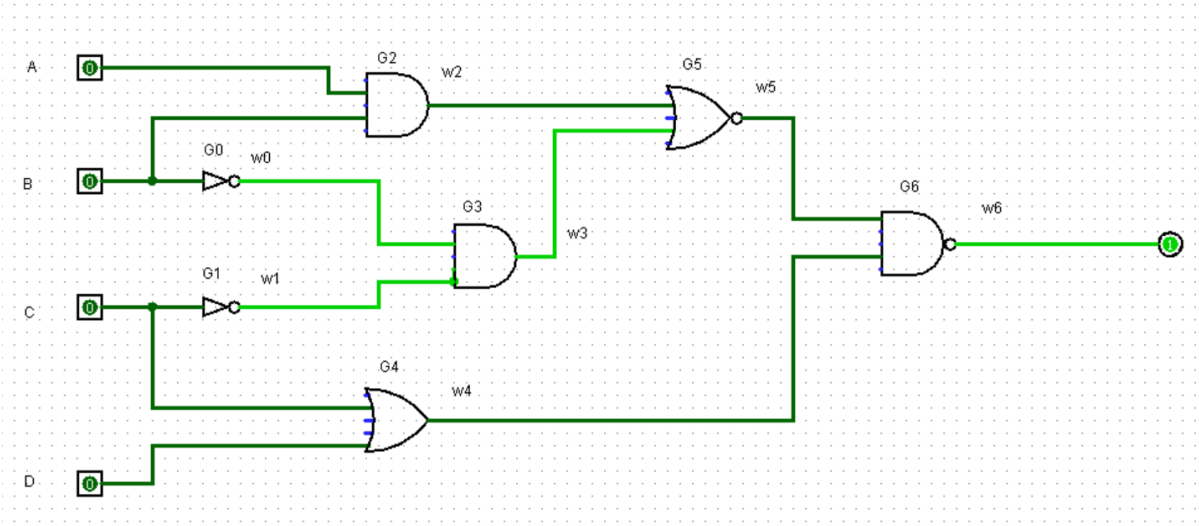
1.

A	B	C	W0	W1	W2
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	1	1	0
0	1	1	1	0	0
1	0	0	1	1	0
1	0	1	1	0	0
1	1	0	0	1	0
1	1	1	0	1	0



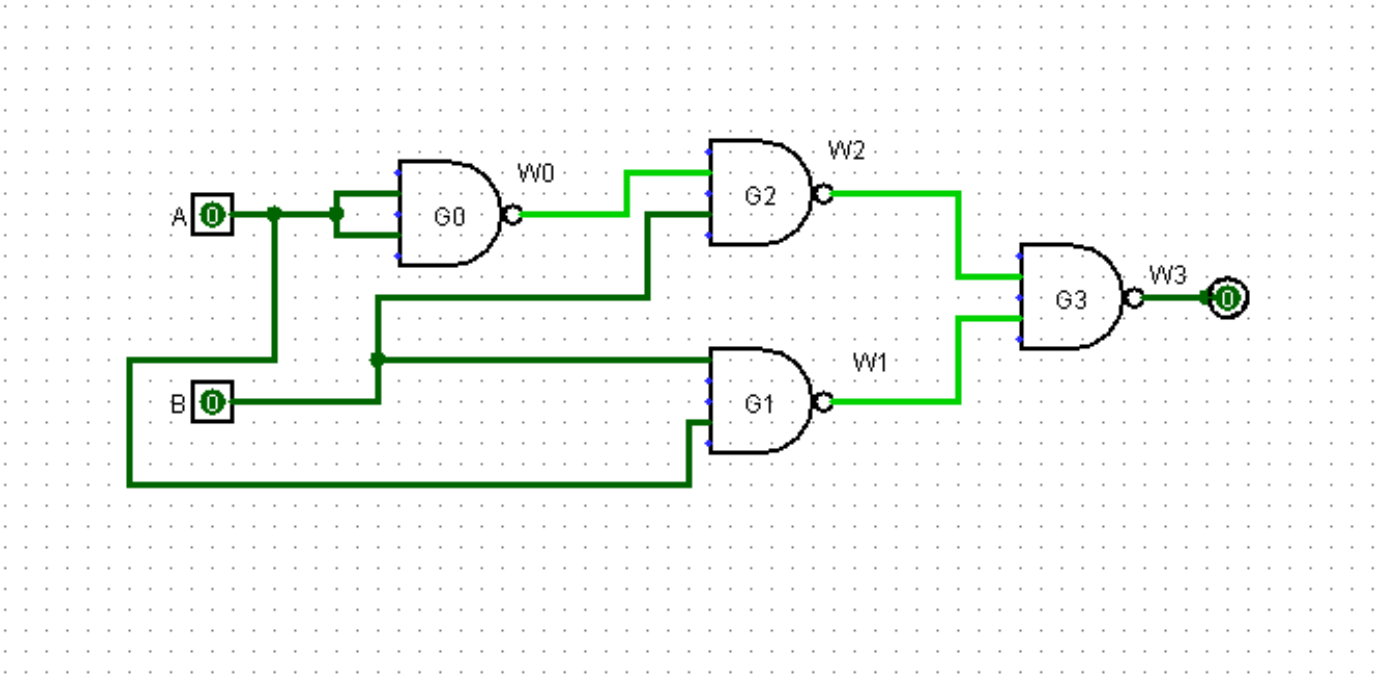
2.

A	B	C	D	W0	W1	W2	W3	W4	W5	W6
0	0	0	0	1	1	0	1	0	0	1
0	0	0	1	1	1	0	1	1	0	1
0	0	1	0	1	0	0	0	1	1	0
0	0	1	1	1	0	0	0	1	1	0
0	1	0	0	0	1	0	0	0	1	1
0	1	0	1	0	1	0	0	1	1	0
0	1	1	0	0	0	0	0	1	1	0
0	1	1	1	0	0	0	0	1	1	0
1	0	0	0	1	1	0	1	0	0	1
1	0	0	1	1	1	0	1	1	0	1
1	0	1	0	1	0	0	0	1	1	0
1	0	1	1	1	0	0	0	1	1	0
1	1	0	0	0	1	1	0	0	0	1
1	1	0	1	0	1	1	0	1	0	1
1	1	1	0	0	0	1	0	1	0	1
1	1	1	1	0	0	1	0	1	0	1



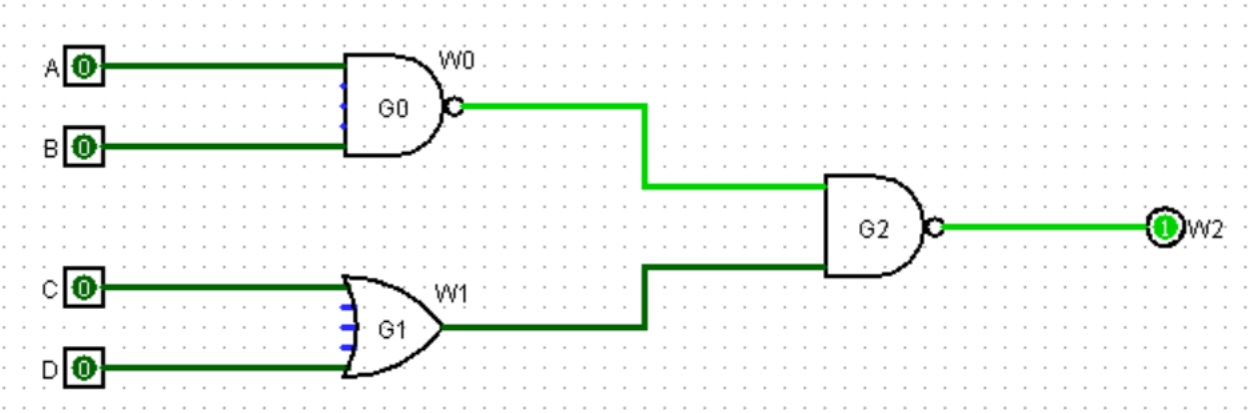
3.

A	B	W0	W1	W2	W3
0	0	1	1	1	0
0	1	1	1	0	1
1	0	0	1	1	0
1	1	0	0	1	1



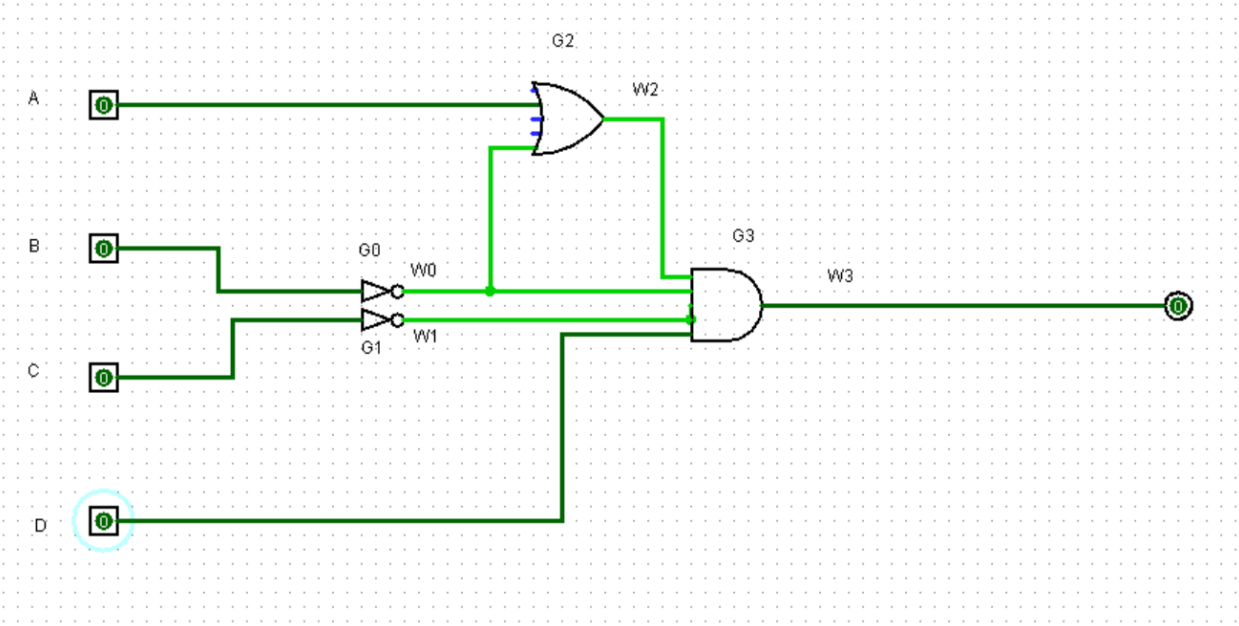
4.

A	B	C	D	W0	W1	W2
0	0	0	0	1	0	1
0	0	0	1	1	1	0
0	0	1	0	1	1	0
0	0	1	1	1	1	0
0	1	0	0	1	0	1
0	1	0	1	1	1	0
0	1	1	0	1	1	0
0	1	1	1	1	1	0
1	0	0	0	1	0	1
1	0	0	1	1	1	0
1	0	1	0	1	1	0
1	0	1	1	1	1	0
1	1	0	0	0	0	1
1	1	0	1	0	1	1
1	1	1	0	0	1	1
1	1	1	1	0	1	1



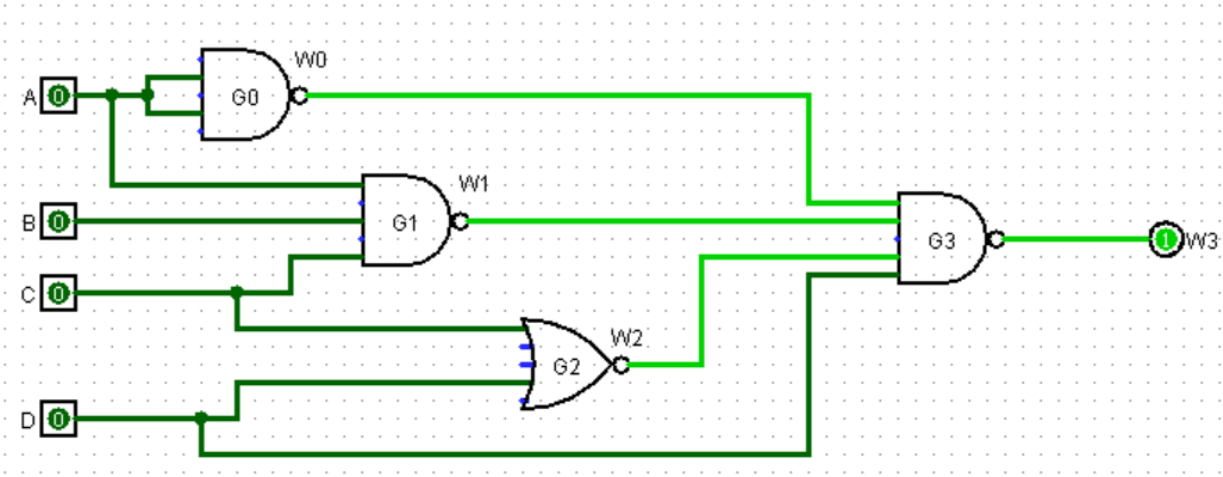
5.

A	B	C	W0	W1	W2
0	0	0	1	1	0
0	0	1	1	0	1
0	1	0	0	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	0	0	1
1	1	0	0	1	1
1	1	1	0	0	1



6.

A	B	C	D	W0	W1	W2	W3
0	0	0	0	1	1	1	1
0	0	0	1	1	1	0	1
0	0	1	0	1	1	0	1
0	0	1	1	1	1	0	1
0	1	0	0	1	1	1	1
0	1	0	1	1	1	0	1
0	1	1	0	1	1	0	1
0	1	1	1	1	1	0	1
1	0	0	0	0	1	1	1
1	0	0	1	0	1	0	1
1	0	1	0	0	1	0	1
1	0	1	1	0	1	0	1
1	1	0	0	0	1	1	1
1	1	0	1	0	1	0	1
1	1	1	0	0	0	0	1
1	1	1	1	0	0	0	1



Program Outline:

```
# Input Library Path
# Read the library file
# Store contents in Library "as 2D list"
#   Library has: gateName, inputsNum, outputExpressions, delay

# FIGURE OUT # # # # TO DO: function to translate the expression # # # # #

# Create a class/ struct called 'Components'
#   Name, Type, Output, Inputs [list of dictionaries]

# A = {
#    "Value": 0
# }
# A["Value"]=1
# print(A["Value"])


# Create a class/ struct called 'Circuit'
#    it has inputs [list], Gates [list], function that gets a log and returns
it to main program
#       log file of what output go into which GATES (to look up when we modify
the wires nd delay)
#       use the list.append() method (as if it's a vector)

# Input Circuit Path
# Read Circuit from File and update the data and send the log to the main file
# Create gate instances

# Log file:
#  Inputs, components

# # # # KEEP TRACK OF TIME # # # #
# # # At the beginning all wires and inputs are set to zero
# Time = 0
# Input Stimuli Path


# Read stimuli file and Create a list of objects of the format [time, signal,
value]
# Increment by 100 ps
#    until reaches time in the first input
#    checks the log for the value of the input
#    if the value of the input doesn't change, nothing
#    if it changes,
#       Loop through the gates
#           We compare componentType to gateName (library)
#           And them figure out the change that will happen to which outputs
```

```
#           and put it in a similar object [time, signal, value]
#           Do a depth first update

# POSSIBLE ISSUE # # # Check if this might put issues with chronological order

#       Create a minheap of the time objects

#       Output it to file

#  GUI

# Note to self: always try to use data structure we learned

#    compare the currentTime+Delay and the time of the next gate



# Extra feature: Real time simulation
```