Simulating a Combinational Circuit with LogicWorks4

V

CD \ AB	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	0	0
10	1	1	0	0

W

CD \ AB	00	01	11	10
00	0	0	X	0
01	1	1	X	0
11	1	1	X	X
10	1	0	X	X

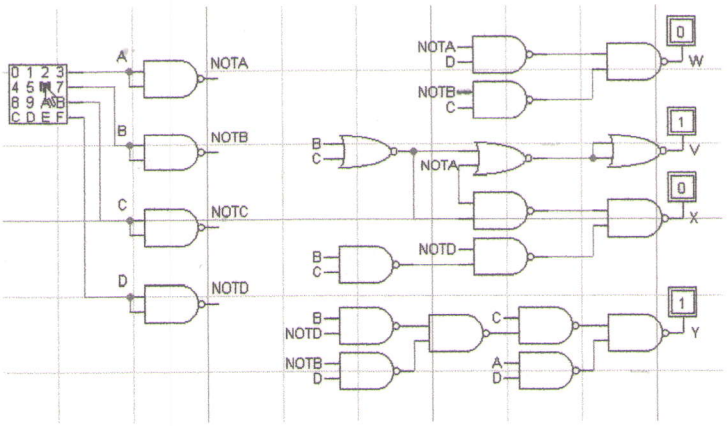
X

CD \ AB	00	01	11	10
00	1	1	X	1
01	1	0	X	0
11	0	0	X	X
10	1	0	X	X

Y

CD \ AB	00	01	11	10
00	0	0	X	0
01	0	0	X	1
11	1	0	X	X
10	0	1	X	X

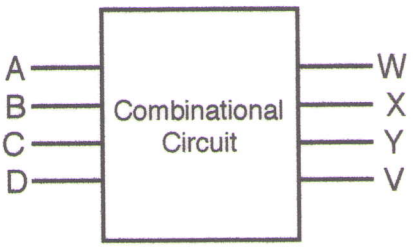
Copyright © 2000 by Robert J. Dirkman
All Rights Reserved



I. Developing the Circuit

This exercise is intended to acquaint you with the use of LogicWorks4 to simulate a simple combinational circuit. First the circuit, which uses only 2 input NAND and NOR gates, will be developed.

The circuit, as shown, has 4 inputs, A, B, C, and D, and 4 outputs, V, W, X, and Y. Each output is a function of the 4 inputs as follows:



V is 1 if the 4 input bits, ABCD, represents a binary-coded decimal digit (0-9) and is 0 otherwise. The remaining functions, W, X, and Y are defined only if ABCD is in the range 0-9; if not, the functions may have any value.

W is 1 if the binary-coded decimal digit is a prime number (1,2,3,5,7) and is 0 otherwise.

X is 1 if the binary-coded decimal digit is either zero or a power of 2 (0,1,2,4,8) and is 0 otherwise.

Y is 1 if the binary-coded decimal digit is divisible by 3 (3, 6, 9) and is 0 otherwise.

A truth table of these functions is shown to the right.

A	B	C	D	W	X	Y	V
0	0	0	0	0	1	0	1
0	0	0	1	1	1	0	1
0	0	1	0	1	1	0	1
0	0	1	1	1	0	1	1
0	1	0	0	0	1	0	1
0	1	0	1	1	0	0	1
0	1	1	0	0	0	1	1
0	1	1	1	1	0	0	1
1	0	0	0	0	1	0	1
1	0	0	1	0	0	1	1
1	0	1	0	X	X	X	0
1	0	1	1	X	X	X	0
1	1	0	0	X	X	X	0
1	1	0	1	X	X	X	0
1	1	1	0	X	X	X	0
1	1	1	1	X	X	X	0

To design the circuit, the 4 functions are written in terms of A, B, C, and D (and their complements) using Karnaugh maps. The resulting Karnaugh maps are:

V

CD \ AB	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	0	0
10	1	1	0	0

W

CD \ AB	00	01	11	10
00	0	0	X	0
01	1	1	X	0
11	1	1	X	X
10	1	0	X	X

X

CD \ AB	00	01	11	10
00	1	1	X	1
01	1	0	X	0
11	0	0	X	X
10	1	0	X	X

Y

CD \ AB	00	01	11	10
00	0	0	X	0
01	0	0	X	1
11	1	0	X	X
10	0	1	X	X

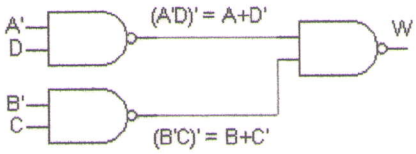
The resulting functions can be written as follows (other expressions are possible):

$$V = A' + B'C'$$
$$W = A'D + B'C$$
$$X = C'D' + A'B'C' + B'D'$$
$$Y = AD + BCD' + B'CD$$

The process of implementing the Boolean functions using NAND and NOR gates is not straightforward. It generally involves a combination of manipulating the Boolean algebraic expressions by factoring or using DeMorgan's Theorem, drawing some implementation (possibly using AND and OR gates), and using gate conversions and combinations to arrive at a circuit which contains only NAND and NOR gates.

The following implementations of the 4 functions were developed using this process. Other, possibly simpler, implementations are possible. Note that the inputs to the circuit are only A, B, C, D, and their complements. In the implementations below, intermediate variables are given. Check that these are indeed correct.

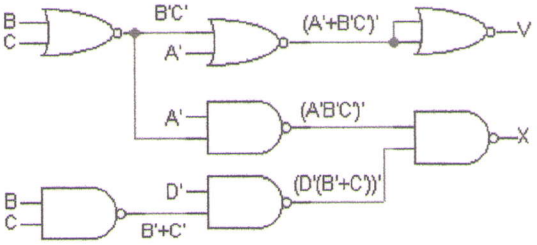
The function W can be implemented with the following circuit using only NAND gates:



$$W = A'D + B'C$$

$$\overline{W} = (A + D') \bullet (B + C')$$

The functions X and V can be implemented with the following circuit:



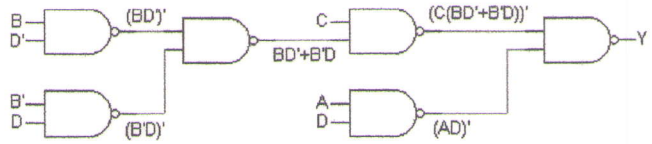
$$X = C'D' + A'B'C' + B'D'$$

$$\overline{X} = \overline{D'(B' + C')} \bullet \overline{A'B'C'}$$

$$V = A' + B'C'$$

Note that the function V can use the B'C' term which is generated for the X function, simplifying the total circuit somewhat.

Finally Y can be implemented with the following circuit which uses only NAND gates:



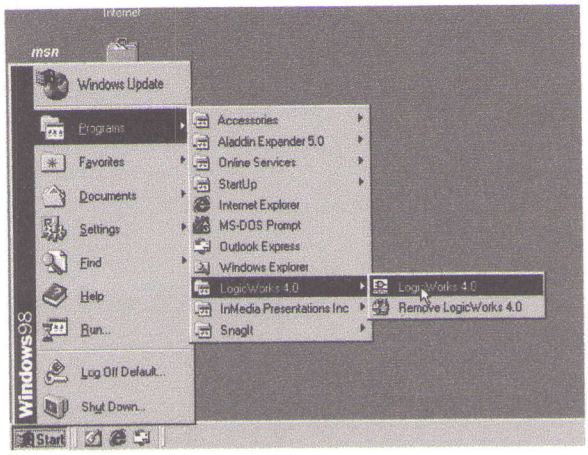
$$Y = AD + BCD' + B'CD$$

$$\overline{Y} = \overline{AD} \bullet \overline{C(BD' + B'D)}$$

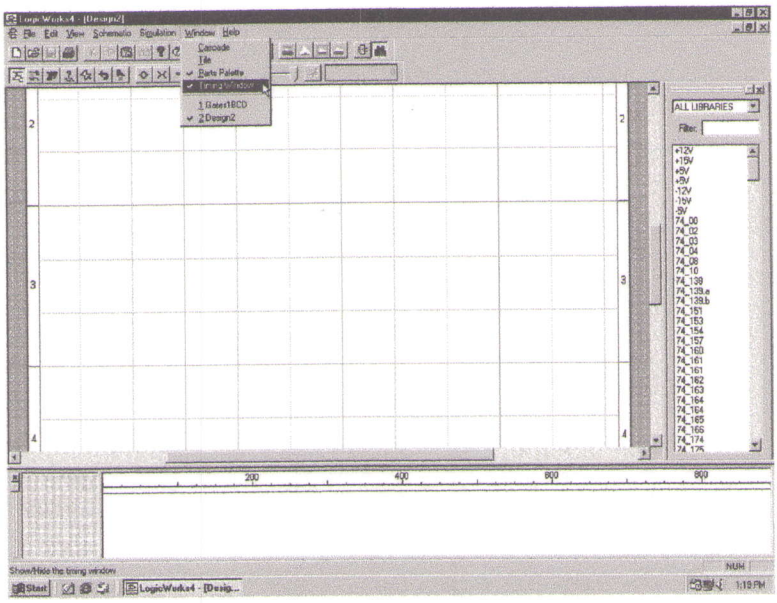
II. Simulating the Circuit

Starting LogicWorks4

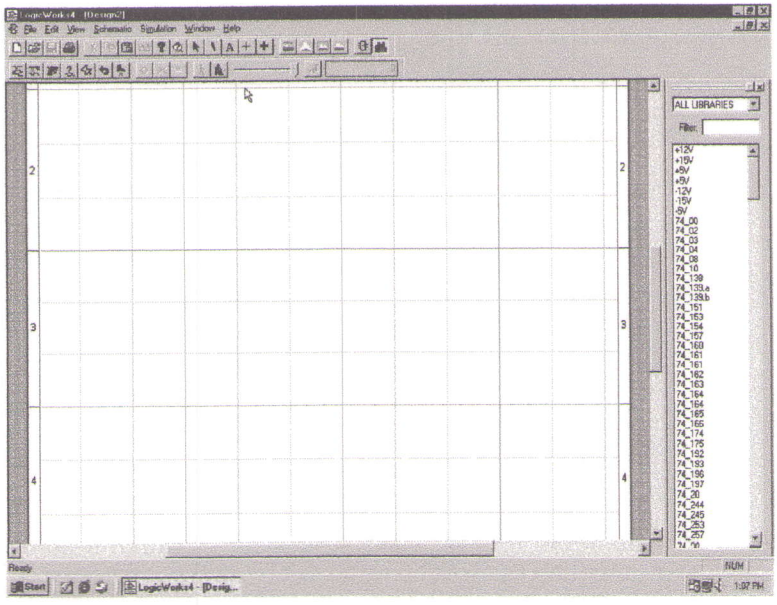
1. Start the LogicWorks4 program either by double clicking on the LogicWorks4 icon on the desktop or by using Start > Program >LogicWorks4 as shown.



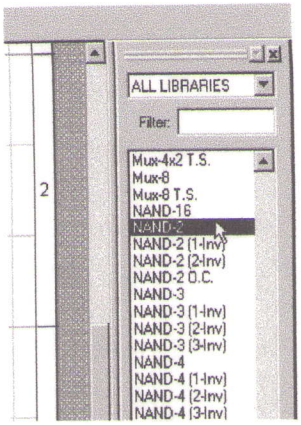
2. The LogicWorks4 window should appear on the screen. Since we will not be using the Timing Window, remove it by selecting "Timing Window" in the Window menu so that the checkmark is removed.



3. The circuit and library areas should fill the screen.



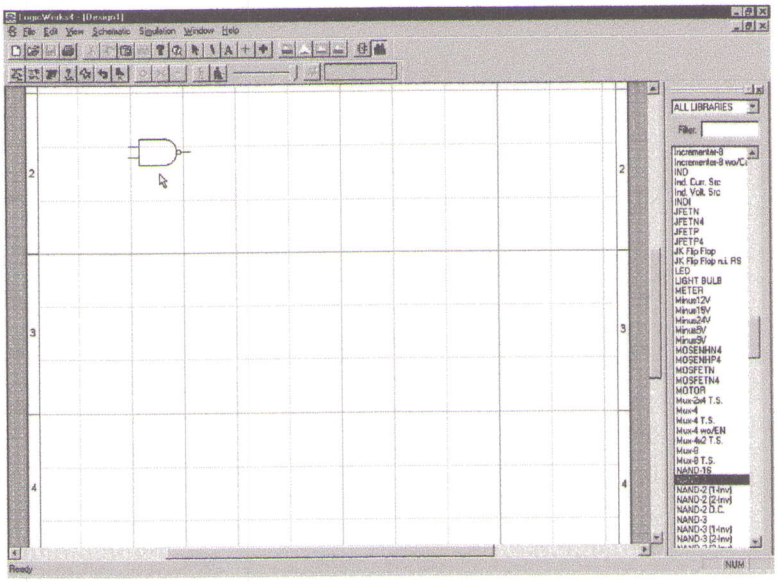
Adding and Removing Circuit Components



4. Components are selected from the scrolling libraries list on the right side of the screen. Scroll down until the NAND-2 element name appears. Scrolling is best done by dragging the scroll bar at the right of the libraries list.

Note 1: You can type "na" in the Filter Box to limit the library list.

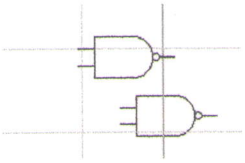
5. The cursor should be the Pointer. If not, click on the Pointer button in the top row of buttons at the top of the screen. The Pointer is an arrow pointing upwards and to the left.



6. Double click on the NAND-2 name and move the cursor into the circuit area of the screen. The cursor should now have the shape of the NAND-2 element.

7. Move the mouse so that the NAND-2 element is in the position shown. Click the left mouse button. This will place a NAND-2 element at that position.

8. Note that, as you move the mouse, the cursor still has the NAND-2 shape. We could add more NAND-2 elements to the circuit if necessary. However, we don't want to, so press the space bar and the cursor will revert back to the Pointer.

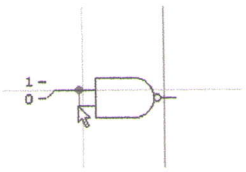


9. Scroll the library list so that the Binary Switch name appears. (Or type "bi" in the Filter Box) Double-click on the name and move the cursor into the circuit area. Note that the cursor changes to the Binary Switch symbol.

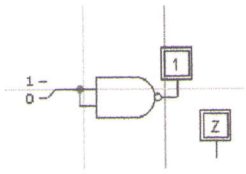
10. Move the cursor so that the right side of the switch just touches the upper input lead of the NAND-2 element, and click the left mouse button. This will place a Binary Switch into our circuit.

11. The cursor still has the shape of the Binary Switch, so press the space bar so that the cursor reverts to the Pointer.

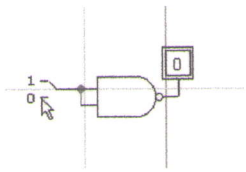
12. We want the NAND-2 element to serve as an inverter, so we need to connect the 2 input leads of the NAND-2. Move the Pointer to the tip of the lower input lead. Hold the mouse down, and drag it to the tip of the upper input lead. Let go of the mouse button. A dot should appear on the upper lead indicating that they are connected.



13. Find the Binary Probe in the library list, and double-click on it. Move the mouse so that the Binary Probe lead just touches the output lead of the NAND-2 element and click the left button of the mouse. Press the space bar to change the cursor back into the Pointer.

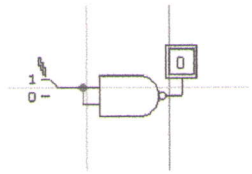


14. Click on the Binary Switch and note that the switch toggles. Note that the Binary Probe indicates that the output of the NAND-2 element is the complement of the input.

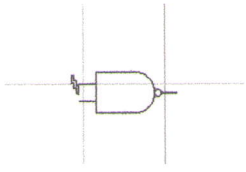


15. We don't need the Binary Switch and Binary Probe as we continue to build our circuit, so select the Zap tool from the top row of buttons. The Zap tool has the shape of a lightning bolt.

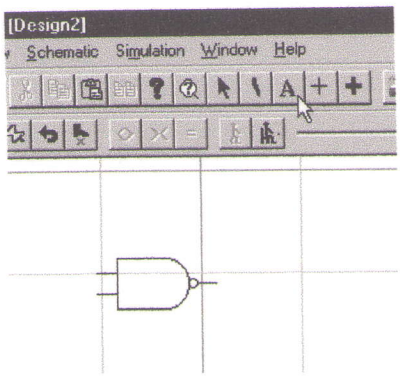
16. Make the Zap tool point to the Binary Switch, and click the left mouse button. The Binary Switch should be deleted from the circuit.



17. Use the Zap tool to delete the Binary Probe and the lead connecting the two inputs of the NAND-2 circuit. Only the NAND-2 element should remain in the circuit window.

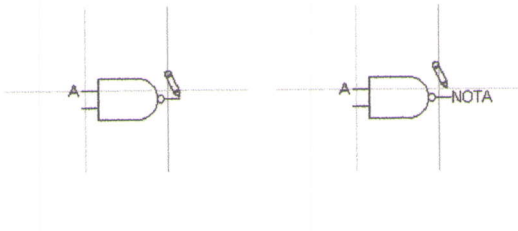


Naming Signals in the Circuit

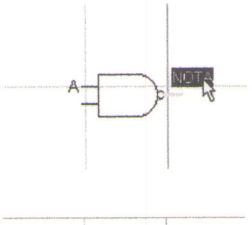


18. We will now give names to the input and output leads of the NAND-2 element. Select the Text tool which is the button with the "A" on it as shown.

19. Move the mouse into the circuit window, and the cursor changes to a pencil. Point the tip of the pencil to the tip of the upper input lead of the NAND-2 element, and click the left mouse button. A text box should appear. Type "A" followed by <enter> on the keyboard. We have just assigned the symbol "A" to the upper input lead of the NAND-2 element.

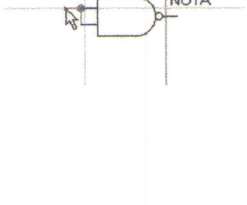


20. Point the tip of the pencil to the tip of the output lead of the NAND-2 element, and click the left mouse button. Type "NOTA" in the text box that appears, and press the <enter> key. We have just assigned the symbol "NOTA" to the output of the NAND-2 element.



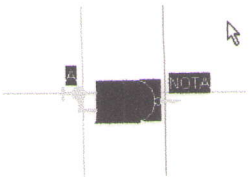
21. We want to move the location of the symbols that we just typed. Select the Pointer tool by clicking on the Pointer button in the upper row of buttons at the top of the screen. Click on the symbol "NOTA" , and, holding the mouse down, drag the symbol up as shown.

22. Likewise move the symbol "A" up as shown.

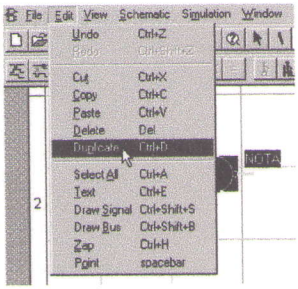


23. Connect the input leads of the NAND-2 element together by placing the Pointer on the tip of the lower input lead. Hold the mouse button down and drag it to the tip of the upper input lead. Release the mouse. Extend the upper lead of the NAND-2 element by placing the Pointer on the tip of the upper input lead and drag the mouse slightly to the left. A connection with a dot should appear as shown.

Constructing the Input Signal Generator

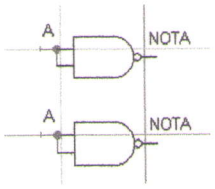


24. We now want to add three additional inverters. Select the NAND-2 element with its input names by pointing the mouse below and to the left of the circuit. Hold the mouse down and drag it above and to the right of the circuit so that a box surrounds everything in the circuit. All of the elements should become dark indicating they are selected.

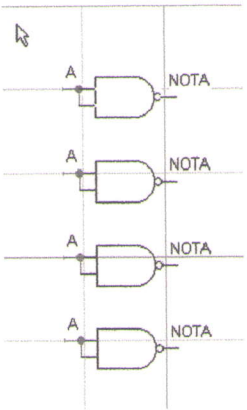


Note 2: Items can also be selected by clicking on them. If you hold the Shift key down, you can select additional items. Clicking on a blank spot on the screen will deselect the selected items.

25. Select "Duplicate" from the Edit menu as shown. This will produce a cursor which contains the items selected.



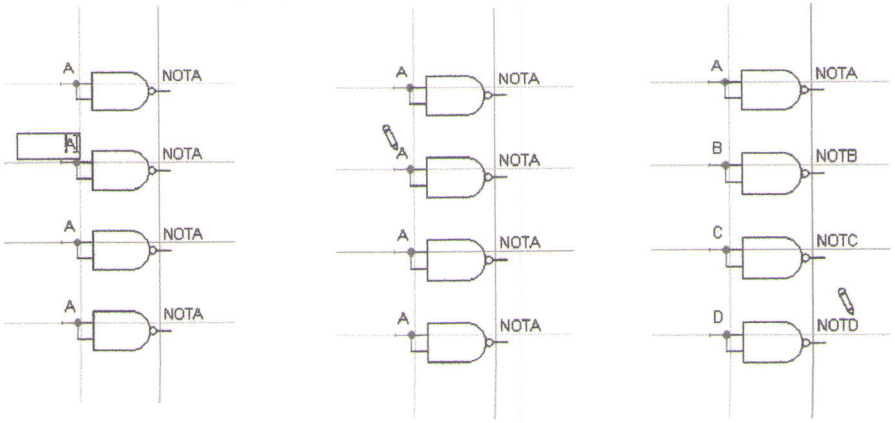
26. Position the cursor so that the NAND-2 element is directly below the original NAND-2 element and 1 guide line down. Click the left mouse. A second copy of the original circuit should appear.



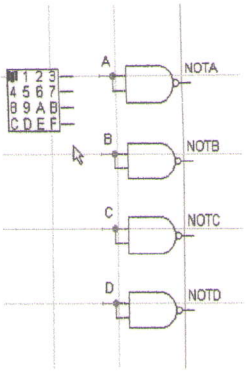
27. Add two more copies in the same manner so that the circuit window now contains four NAND-2 elements as shown.

28. We now want to change the names of the signals on the three NAND-2 elements we just added. Select the Text tool, and point the pencil to the "A" in the 2nd NAND-2 element and click the mouse. A text box should appear. Change the "A" to "B".

29. Repeat this on the 5 remaining signal names so that the names are as shown.



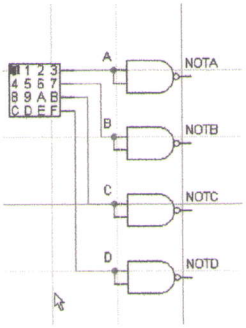
30. Type "he" in the Filter Box and select the "Hex Keypad wo/s" from the library. Place the component in the position shown. Press the space bar to return to the Pointer tool.



NOTE 3: You could use 4 Binary Switches to select values of A, B, C, and D, but the Hex Keypad is more useful for our application.

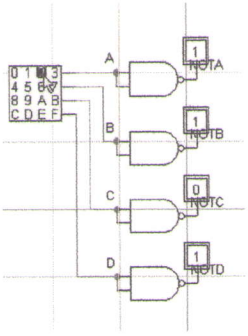
NOTE 4: Since the Binary Switch and Hex Keypad behave differently from other components when they are clicked on, it is necessary to hold down the Shift key if you want to select them by clicking on them.

31. Connect the 4 outputs of the Hex Keypad to the inputs of the NAND-2 elements with the Pointer tool. To do this, place the mouse at the tip of a lead and drag it so that the connection follows the mouse. Note that the connection will have a corner if the mouse is not moved in a straight line. If the connection has two corners, it will be necessary to lift the mouse button and press it again to form a new connection.

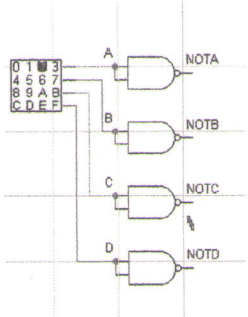


NOTE 5: When you have made a single corner, while the mouse is still down, press the Ctrl key, the Tab key, then both the Ctrl and Tab keys together and see what happens.

32. To check that the circuit we have made so far is working, place a Binary Probe at the output of each NAND-2 element as shown. Then click on the hex digits of the Hex Keypad and observe the Probes. Note that "A" is the most significant bit of the Hex digit.

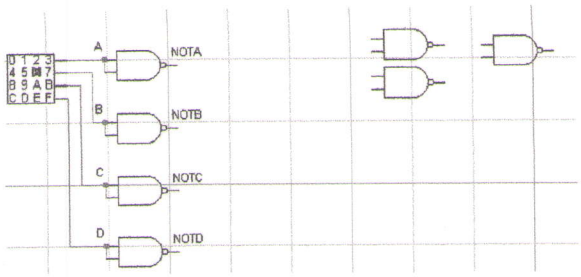


33. We won't be needing the Binary Probes that we just added since we know that this portion of our circuit works. Delete them with the Zap tool as shown.



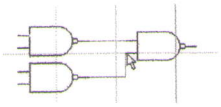
Implementing the Functions V, W, X, and Y

34. To implement function W, select the NAND-2 gate from the library and place 3 of them in the positions shown.

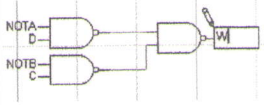


NOTE 6: You can also obtain a NAND-2 gate without using the library by selecting a NAND-2 gate in the circuit and using Duplicate from the Edit Menu.

35. Connect the 3 NAND-2 gates as shown by dragging the Pointer tool.

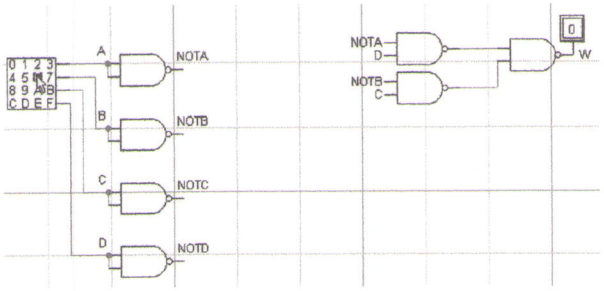


36. It is necessary to specify the variables at each input that has not been connected. Using the Text tool, name the 4 input signals as shown. Also name the output of the rightmost NAND-2 gate "W".

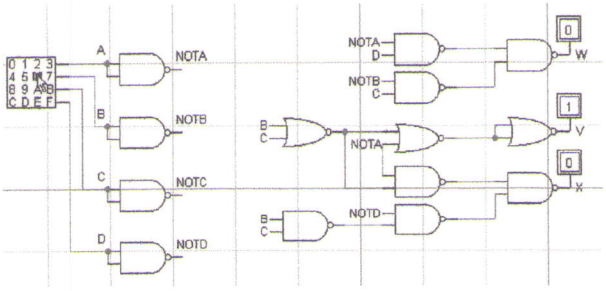


NOTE 7: If two or more points in the circuit have the same name, they are connected.

37. Add a Binary Probe to the output as shown. Check that the "W" function selects prime numbers (1,2,3,5,7).



38. Implement the functions V and X by adding the NAND-2 and NOR-2 gates and the Binary Probes in the positions shown. Connect the gates, and give the input signals the names shown.

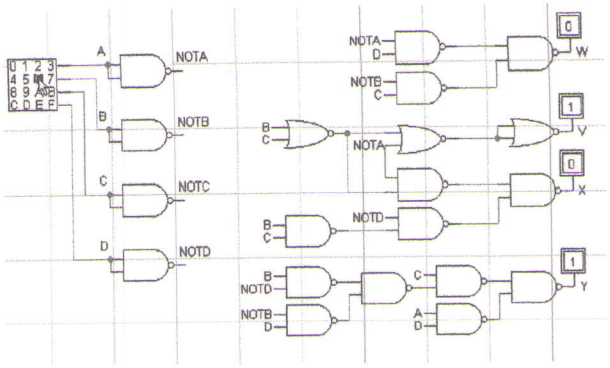


Check that the V function is 1 if the hex digit is a binary-coded decimal digit (0-9) and 0 if the hex digit is not (A-F).

Check that the X function is 1 if the binary-coded decimal digit is zero or a power of 2 (0,1,2,4,8) and 0 otherwise.

39. Implement the function Y by adding the NAND-2 gates and the Binary Probe in the positions shown. Connect the gates, and give the input signals their names.

Check that the Y function is 1 if the binary-coded decimal digit is a power of three (3,6,9) and 0 otherwise.



40. Save your circuit on a floppy disk.

Exercise 1:

Count the number of NAND-2 and NOR-2 gates in the circuit. The 7400 IC contains 4 2-input NAND gates and the 7402 IC contains 4 2-input NOR gates. Noting that the inverter functions can be accomplished using either a NAND or a NOR gate, what would you do to minimize the number of IC's used?

How many IC's do you need?

Exercise 2:

Implement the circuit in LogicWorks4 using 74_00 and 74_02 IC's instead of the NAND-2 and NOR-2 gates.

The Finished Circuit