

|  |
| --- |
| **Activity 2.2.2 NAND Logic Design** |

Introduction

The block diagram shown below represents a voting booth monitoring system. For privacy reasons, a voting booth can only be used if the booth on either side is unoccupied. The monitoring system has four inputs and two outputs. Whenever a voting booth is occupied, the corresponding input (***A, B, C, & D***) is a (1). The first output, ***Booth***, is a (1) whenever a voting booth is available. The second output, ***Alarm***, is a (1) whenever the privacy rule is violated.

Booth

A

Booth

B

Booth

C

Booth

D

Booth

Alarm

Voting Booth

Monitoring System

In this activity you will implement NAND only combinational logic circuits for the two outputs ***Booth*** and ***Alarm***. These NAND only designs will be compared with the original AOI implementations in terms of efficiency and gate/IC utilization. In a future activity, these NAND only designs will be compared to the circuits implemented using only NOR gates.

Equipment

* Paper and pencil
* Circuit Design Software (CDS)
* Digital Logic Board (DLB)
* Integrated Circuits (74LS00)
* Jumper wire

Procedure

For the sake of time, the truth table and K-Maps for the voting booth monitor systems have been completed for you. Note, for the output ***Booth*** we took advantage of several *don’t care* conditions.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | A | B | C | D | Booth | Alarm | | 0 | 0 | 0 | 0 | 1 | 0 | | 0 | 0 | 0 | 1 | 1 | 0 | | 0 | 0 | 1 | 0 | 1 | 0 | | 0 | 0 | 1 | 1 | X | 1 | | 0 | 1 | 0 | 0 | 1 | 0 | | 0 | 1 | 0 | 1 | 0 | 0 | | 0 | 1 | 1 | 0 | X | 1 | | 0 | 1 | 1 | 1 | X | 1 | | 1 | 0 | 0 | 0 | 1 | 0 | | 1 | 0 | 0 | 1 | 0 | 0 | | 1 | 0 | 1 | 0 | 0 | 0 | | 1 | 0 | 1 | 1 | X | 1 | | 1 | 1 | 0 | 0 | X | 1 | | 1 | 1 | 0 | 1 | X | 1 | | 1 | 1 | 1 | 0 | X | 1 | | 1 | 1 | 1 | 1 | X | 1 | |  |

1. In the space provided, draw the AOI circuits that implement the simplified logic expressions ***Booth*** and ***Alarm***. Limit this implementation to only 2-input AND gates (74LS08), 2-input OR gates (74LS32), and inverters (74LS04).



Booth – AOI

**GND**

**VCC**

**5V**

**S1**

**Key = A**

**S2**

**Key = B**

**S3**

**Key = C**

**S4**

**Key = D**

**U5**

**AND2**

**U7**

**AND2**

**U8**

**AND2**

**U9**

**OR2**

**U10**

**OR2**

**X1**

**2.5 V**

Alarm – AOI

1. Re-implement these circuits assuming that only 2-input NAND gates (74LS00) are available. Draw these circuits in the space provided.



Booth – NAND



Alarm – NAND

1. Using the CDS, enter and test the two logic circuits that you designed. Use switches for the inputs ***A, B, C,*** and ***D*** and a probe or LED circuit for the outputs ***Booth*** and ***Alarm***. Verify that the circuits are working as expected. Print a copy of the circuit and attach it below. Note: Even though the two circuits work independently, they are part of one design and should be simulated, tested, and prototyped together.



Booth & Alarm – CDS

1. Using the DLB, build and test the NAND logic circuits that you designed and simulated. Verify that the circuits are working as expected and the results match the results of the simulation.

**Conclusion**

1. For your AOI implementations, how many ICs (i.e., 74LS04, 74LS08, and 74LS32 chips) were required to implement your circuits? Note: You’re not just counting the number of gates used, but rather, the number of IC, in whole or part, that were required.

BOOTH

74LS04- 4/6 IC (4 out of 6 gates in IC)

74LS08- 2/4 IC (2 out of 4 gates in IC)

74LS32- ¼ IC (1 out of 4 gates in IC)

ALARM

74LS04- 0 IC

74LS08- ¾ IC (3 out of 4 gates in IC)

74LS32- 2/4 IC (2 out of 4 gates in IC)

1. For your NAND implementations, how many ICs (i.e., 74LS00 chips) were required to implement your circuits? Again, we are counting ICs, not gates.

BOOTH

74LS00- 5 ¾ ICs 5 (5 full ICs and 3 out of 4 of an IC)

1. In terms of hardware efficiency, how does the NAND implementation compare to the AOI implementation?

Using only NAND gates to make all the gates needed would make it easier to make, and chaper to buy many NAND gates to make all the circuits.

1. NAND gates are available with three inputs (74LS10) and four inputs (74LS20). Could either of these chips have been used for this design? If so, how would it have affected the efficiency of the design?

YES, it would have made it need less gates to work and would have made it more efficient.