

MIT OpenCourseWare  
<http://ocw.mit.edu>

6.004 Computation Structures  
Spring 2009

For information about citing these materials or our Terms of Use, visit: <http://ocw.mit.edu/terms>.

# Synthesis of combinational logic

---

Problem 1. A certain function F has the following truth table:

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

- A. ★ Write a sum-of-products expression for F.
  - B. ★ Write a minimal sum-of-products expression for F. Show a combinational circuit that implements F using only INV and NAND gates.
  - C. ★ Implement F using one 4-input MUX and inverter.
  - D. ★ Write a minimal sum-of-products expression for NOT(F).
- 

Problem 2.

A. Give a minimal sum-of-products expression that is equivalent to the follow logic expressions:

(A)  $\overline{A + B}$

(B)  $A^*B^*C + \overline{A}^*B^*C + \overline{A}^*\overline{B}^*C + \overline{A}^*B^*\overline{C} + A^*\overline{B}^*C + A^*B^*\overline{C}$

B. Multiple choice: A Karnaugh map can't represent more than 2 variables along a single dimension because

1. Gray code counts can't go beyond 2 bits.
2. Each value  $v$  along a dimension must be adjacent to all values that are Hamming distance 1 from  $v$ .
3. Three is not a power of two.
4. No reason.  
You can represent 3 variables along a dimension. You couldn't make 5-variable K-maps otherwise.

- C. ★ What is the maximum number of product terms in a minimal sum-of-products expressions with three variables?
- D. ★ True or false: A boolean function of  $N$  variables with greater than  $2^{N-1}$  product terms can *always* be simplified to an expression using fewer product terms.
- E. ★ Suppose the stock room is very low on components and has only *five* NAND gates on hand. Would we be able to build an implementation of any arbitrary 2-input boolean function?
- 

Problem 3. In the Karnaugh maps below the use of "X" in a cell indicates a "don't care" situation where the value of the function for those inputs can be chosen to minimize the size of the overall expression.

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

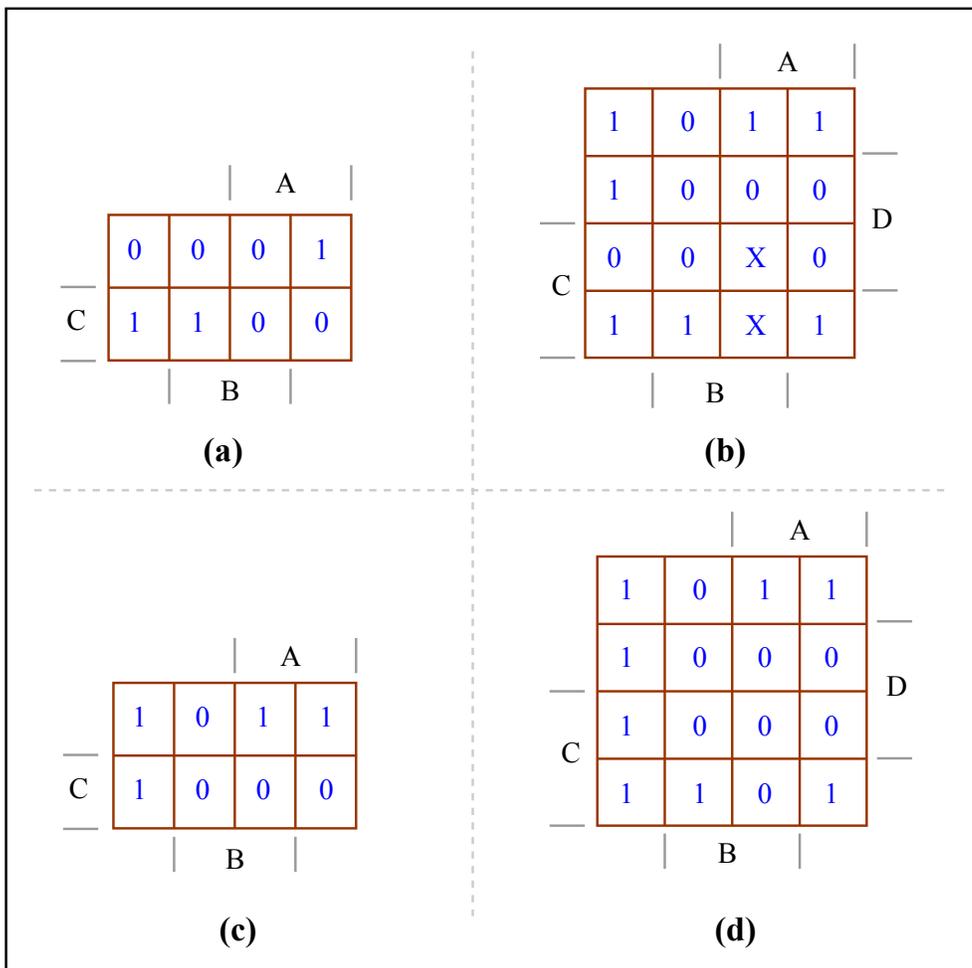
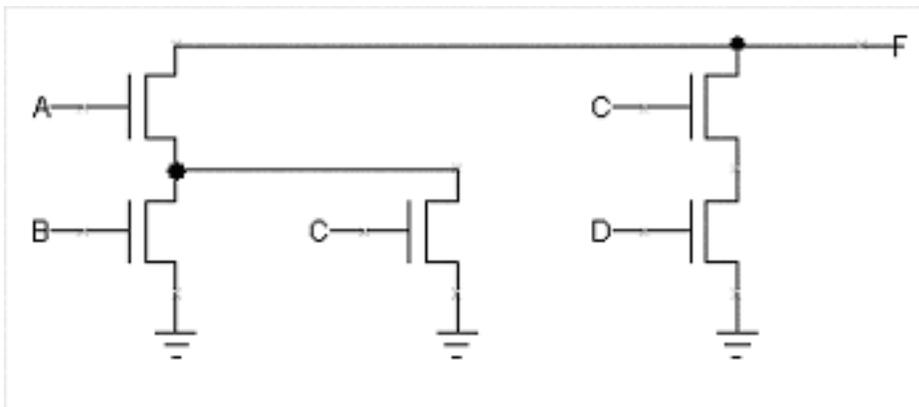


Figure by MIT OpenCourseWare.

- A. Circle the prime implicants in the Karnaugh maps and write a minimal sum-of-products expression for each of the maps.

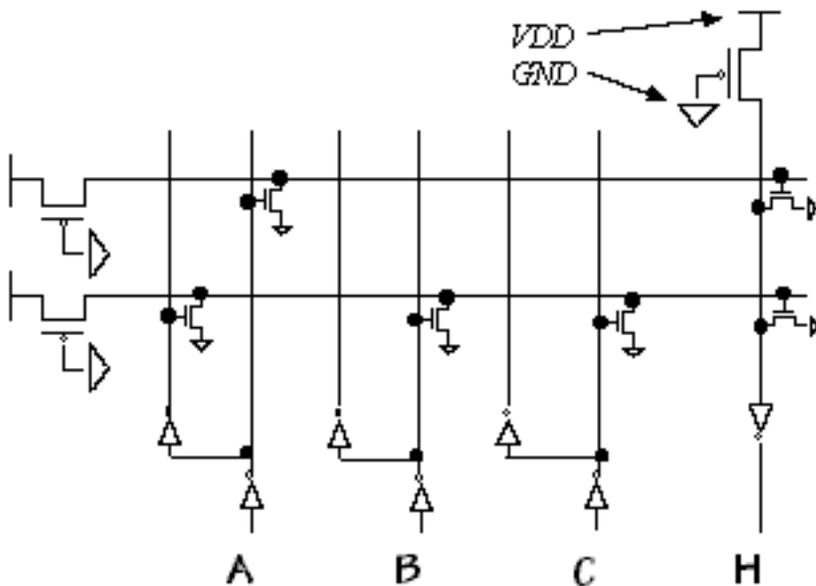
**Problem 4.** The following diagram shows the pulldown circuitry for a static CMOS logic gate that implements the function  $F(A,B,C,D)$ :



- A. Draw a circuit diagram for the pullup circuitry that would complete the static CMOS implementation of  $F(A,B,C,D)$ .

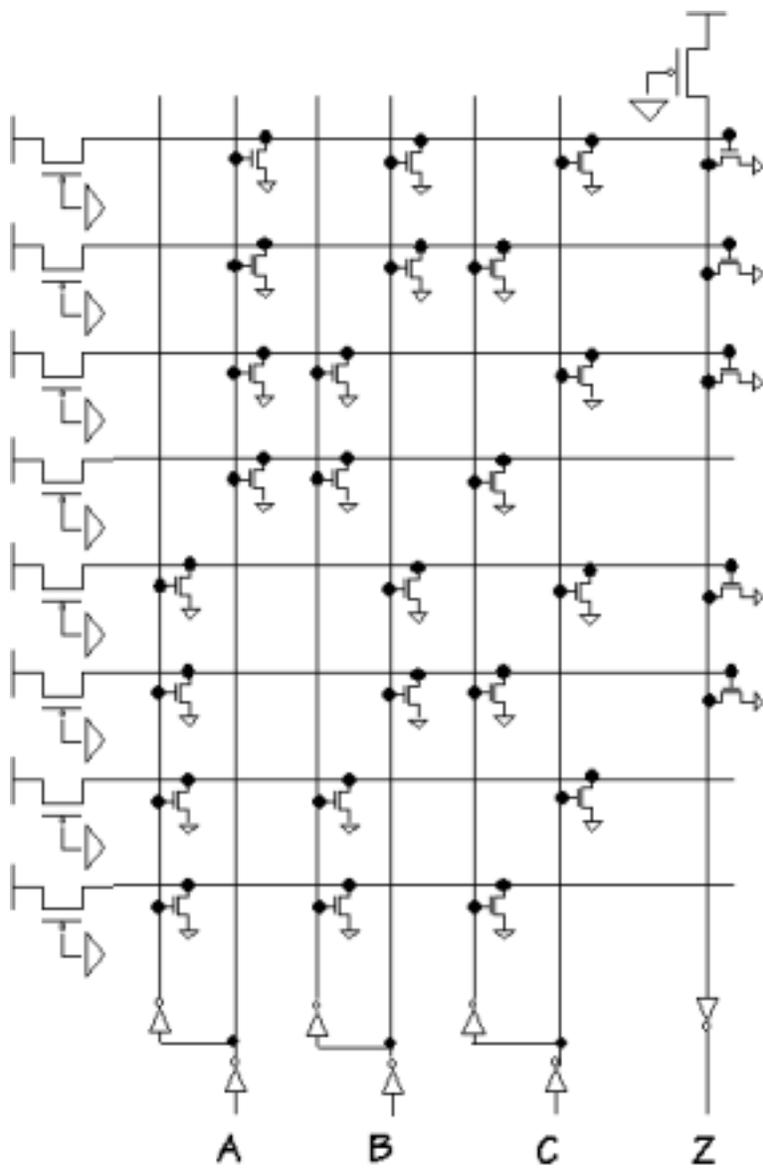
- B. Assuming the correct pullup circuitry is added to the diagram above, give a minimal sum-of-products expression for  $F(A,B,C,D)$ .
- C. Which of the following changes will decrease the propagation time of the above circuit?
- (A) decreasing the power supply voltage
  - (B) increasing the power supply voltage
  - (C) increasing the operating temperature
  - (D) redefining  $V_{OL}$  to provide increased noise margins

Problem 5. The following PLA implements  $H(A,B,C)$ :



- A. ★ Write a minimal sum-of-products expression for H.

Problem 6. The following diagram shows a ROM implementation of a 3-input Boolean function: Give the Boolean function represented at the output, Z.



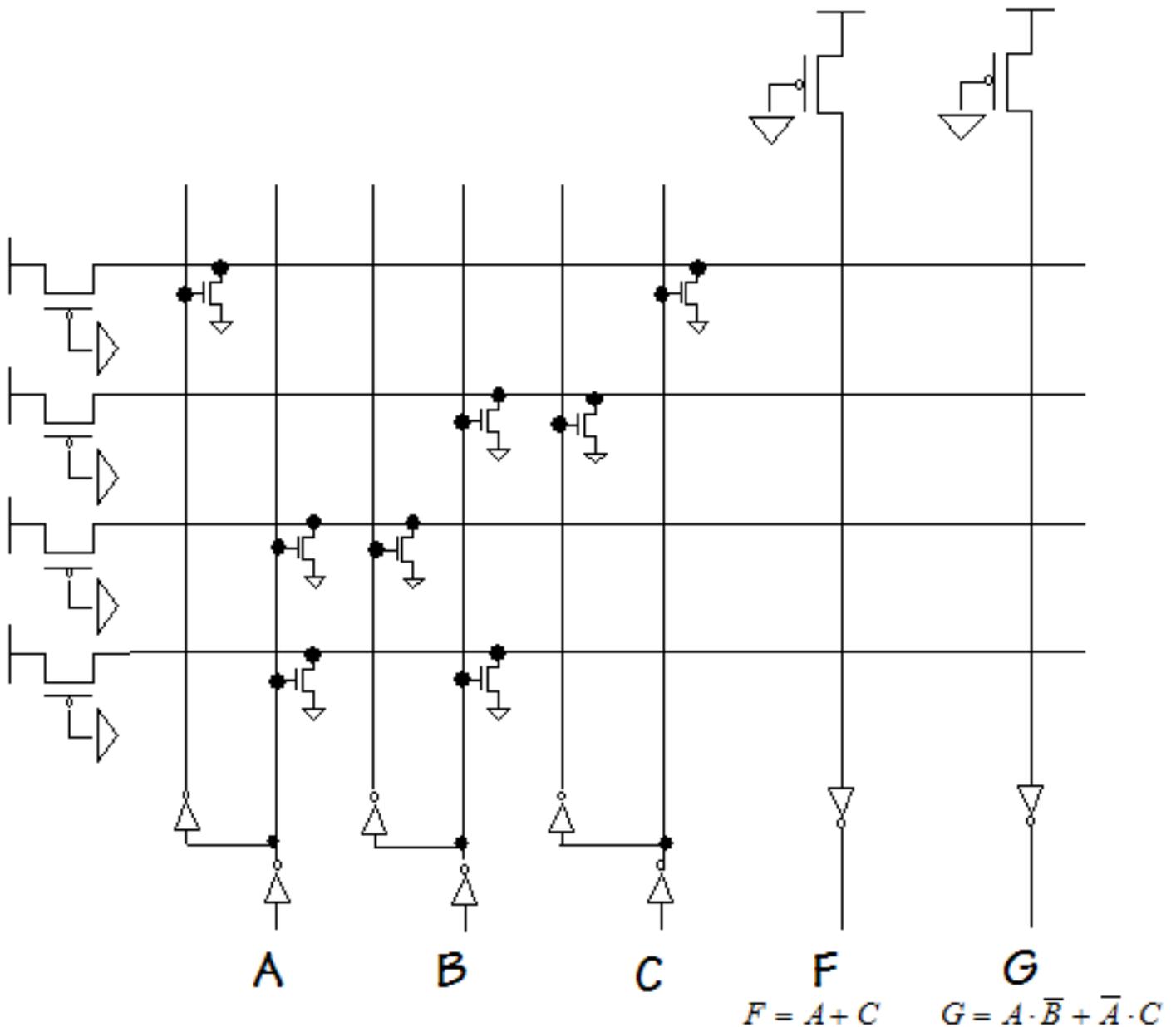
- A. For a particular set of input values, how many of the horizontal term lines carry a logical *one*?
- B. ★ Give the Boolean function represented at the output, Z.
- C. ★ Is there an example involving two valid inputs that demonstrates that the above device is not lenient?
- D. Suppose you undertake to reduce the preceding ROM implementation by eliminating any components (pulldowns, wires, and inverters) that are unused or redundant. Components are eliminated -- replaced by open circuits -- until no further components can be removed without changing the function computed by the circuit. How many word lines (horizontal lines that constitute outputs of the original decoder section of the ROM) are left after this reduction has been performed?
- E. Which of the following is the most accurate comparison of the reduced ROM circuit with a direct,

full-complementary CMOS implementation of the same function?

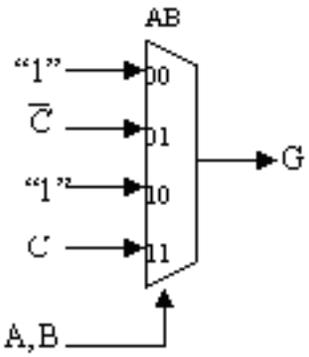
1. The reduced ROM implementation uses fewer transistors.
2. The reduced ROM implementation is faster.
3. The reduced ROM implementation has a faster output rise time.
4. The reduced ROM implementation is essentially identical to the direct CMOS implementation.

Problem 7.

A. Add the necessary pulldown NFETs to the F and G circuitry in the ROM circuit diagram below to implement the indicated logic functions for F and G.



**Problem 8.** A certain 3-input function  $G(A,B,C)$  has the following implementation:



- A. Give a minimal sum-of-products expression for  $G$ .
- B. Design a ROM implementation for  $G$ .

**Problem 9.** Digital Widgets Co. has introduced a new logic IC consisting of two comparator cells in a 14-pin package. A comparator cell, as drawn below, has four inputs and two outputs.

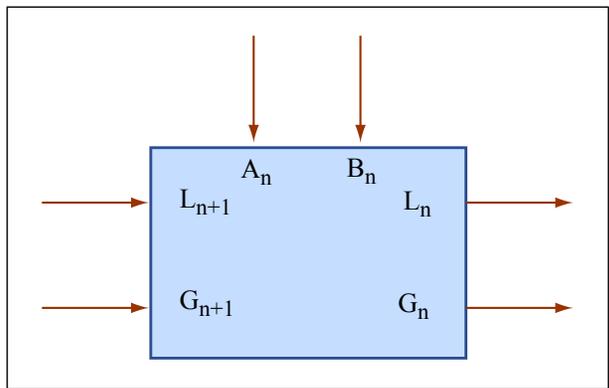


Figure by MIT OpenCourseWare.

The inputs are labeled  $A_n$ ,  $B_n$ ,  $G_{n+1}$ , and  $L_{n+1}$ , and the outputs are labeled  $G_n$  and  $L_n$ . The  $G$  and  $L$  signals have the meanings "A greater than B" and "A less than B," respectively. If both  $G$  and  $L$  are false, the meaning is  $A = B$ .  $G$  and  $L$  are never both true. Two  $k$ -bit numbers  $A$  and  $B$  may be compared using a circuit such as the following:

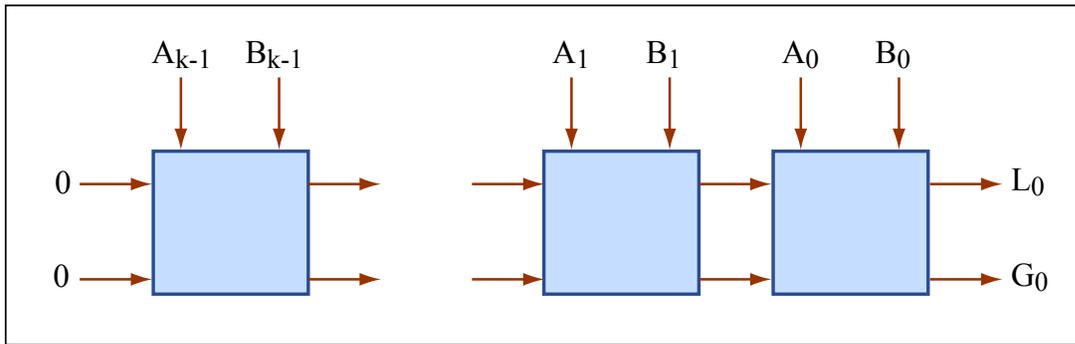


Figure by MIT OpenCourseWare.

The most significant bits are supplied as  $A_{k-1}$  and  $B_{k-1}$ , and the least significant bits are  $A_0$  and  $B_0$ .

The output of a comparison is taken from the G and L outputs of the lowest-order cell ( $G_0$  and  $L_0$ ).  $G_{n+1}$  and  $L_{n+1}$  of the highest-order cell are connected to logical 0 to indicate that the numbers are assumed to be equal until some difference is found between a pair of bits  $A_i$  and  $B_i$ .

If the  $G_{n+1}$  and  $L_{n+1}$  inputs indicate that higher-order bits have established  $A > B$  or  $A < B$ , then cell  $n$  must propagate that result to  $G_n$  and  $L_n$ . However, if  $G_{n+1}$  and  $L_{n+1}$  indicate that the higher-order bits are equal, then cell  $n$  must compare its bit of  $A$  and  $B$  to determine if  $A > B$ ,  $A < B$ , or  $A = B$  and must signal that result appropriately at  $G_n$  and  $L_n$ .

- A. Draw a logic diagram for an implementation of the Digital Widgets comparator cell.
- B. Since there is delay associated with the propagation of the G and L signals through each cell, we could make the comparator work faster by redesigning the basic cell to compare two bits at a time, halving the number of stages through which the G and L signals will need to propagate.

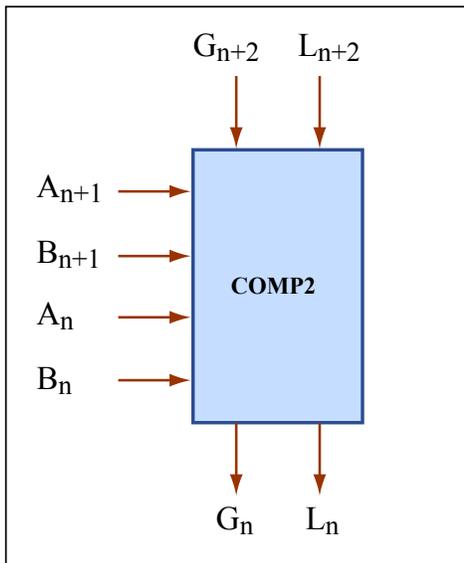


Figure by MIT OpenCourseWare.

Work out expressions for  $G_n$  and  $L_n$  as functions of  $G_{n+2}$ ,  $L_{n+2}$ ,  $A_{n+1}$ ,  $B_{n+1}$ ,  $A_n$ , and  $B_n$ . Express your answers in the form

$$G_n = G_{n+2} + \overline{L_{n+2}} \cdot f(A_{n+1}, B_{n+1}, A_n, B_n),$$

$$L_n = L_{n+2} + \overline{G_{n+2}} \cdot g(A_{n+1}, B_{n+1}, A_n, B_n).$$

- C. Given a reasonable implementation of the equations for  $G_n$  and  $L_n$  derived in part B, how does the delay from a change in  $G_{n+2}$  and  $L_{n+2}$  to the appearance of correct outputs at  $G_n$  and  $L_n$  compare with the corresponding delay for a circuit composed of a cascade of two of the cells developed in part A? Assume that all A and B inputs remain unchanged throughout.

*Note:* The reason for our interest in the propagation delay of the G and L signals, specifically, is that in a chain of N comparators, every extra gate delay in the G--L path will penalize total performance by N gate delays. The time it takes for a change in an A or B input to be reflected in the corresponding G or L output is also important, but improvements here can at best result in decreasing total delay by some constant amount.