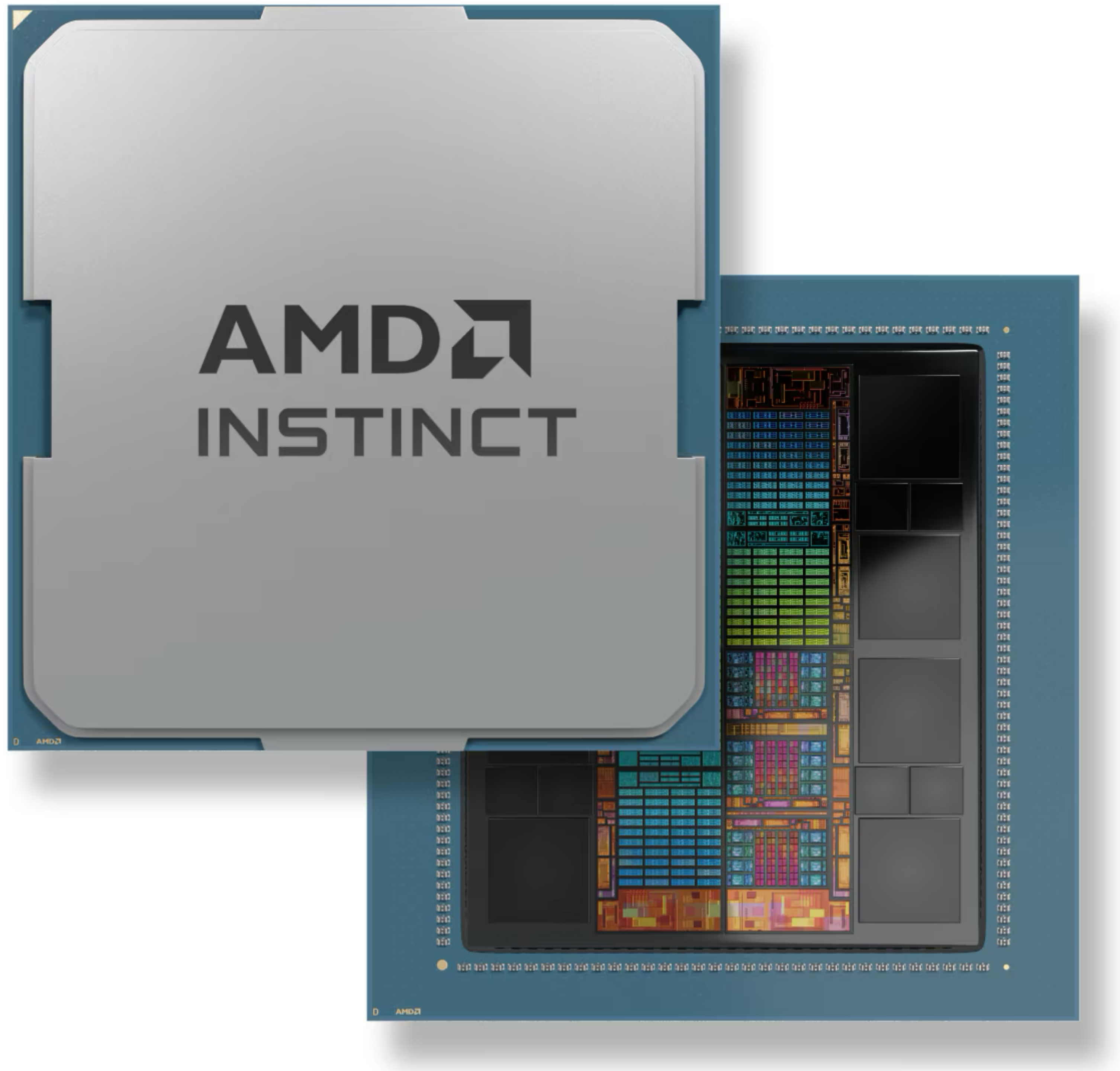
Digital Logic Design + Computer Architecture

Sayandeep Saha

Assistant Professor
Department of Computer Science and Engineering
Indian Institute of Technology Bombay

Combinational Circuits

Do You Want to Design Some Day?



Design with Gates

- **Logic gates**: perform logical operations on input signals
- **Positive (negative) logic polarity**: constant 1 (0) denotes a high voltage and constant 0 a low (high) voltage
- **Combinational circuits:** No memorization
- **Synchronous sequential circuits**: have memory; driven by a clock that produces a train of equally spaced pulses
- **Propagation delay**: time to propagate a signal through a gate
- **Asynchronous circuits**: are almost free-running and do not depend on a clock; controlled by initiation and completion signals

Exclusive-OR (XOR) and XNOR

Exclusive-OR: modulo-2 addition, i.e., $A \oplus B = 1$ if either A or B is 1, but not both.

Commutativity: $A \oplus B = B \oplus A$

Associativity: $(A \oplus B) \oplus C = A \oplus (B \oplus C) = A \oplus B \oplus C$

Distributivity: $(AB) \oplus (AC) = A(B \oplus C)$

If $A \oplus B = C$, then

$$A \oplus C = B$$
$$B \oplus C = A$$
$$A \oplus B \oplus C = 0$$

Exclusive-OR of an even (odd) number of elements, whose value is 1, is 0 (1)

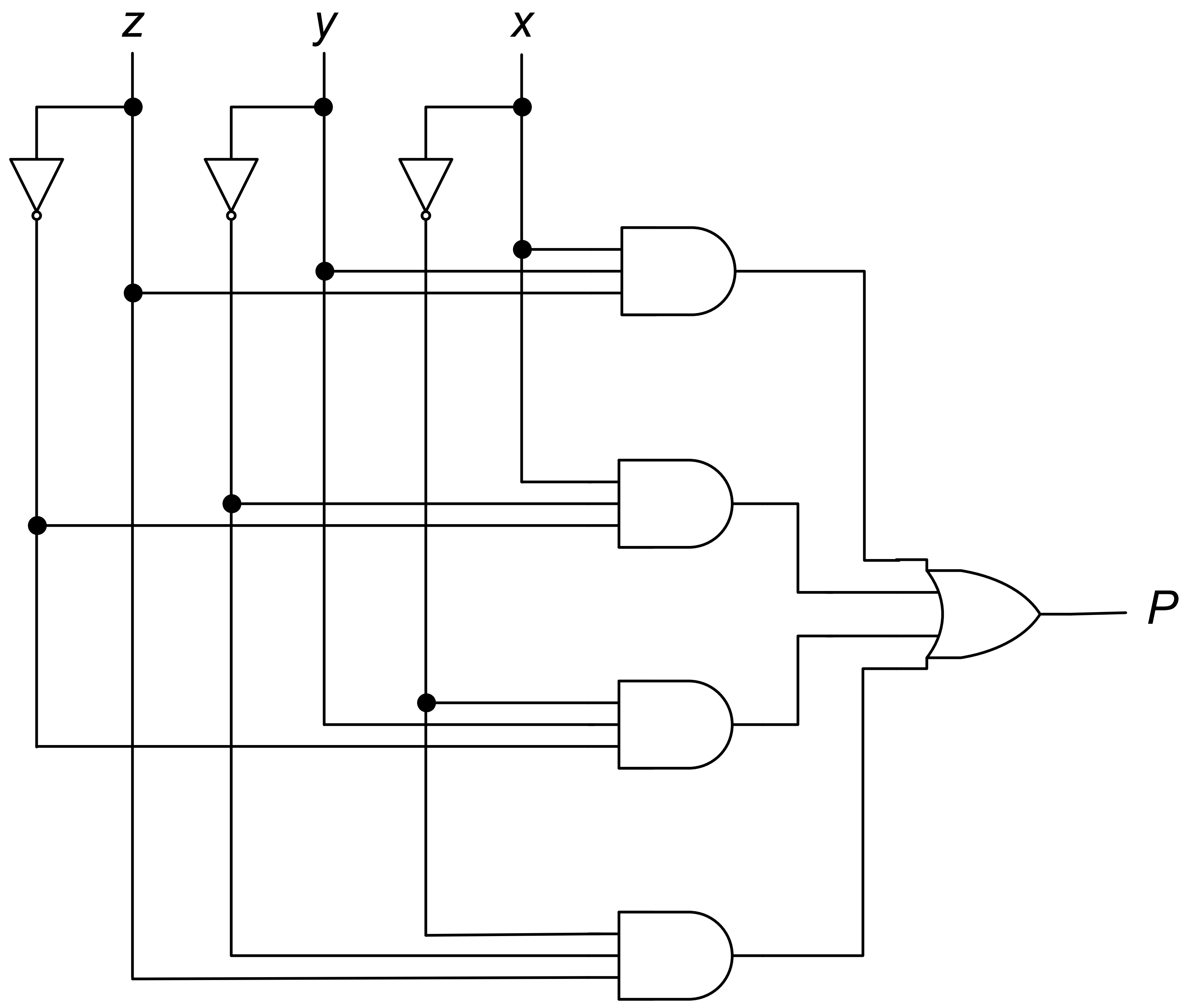
XNOR: Complement of XOR

Combinational Circuits: Parity-bit Generator

Parity-bit generator: produces output value 1 if and only if an odd number of its inputs have value 1

z \ xy	00	01	11	10
0	0	1	0	1
1	1	0	1	0

(*a*) Map.



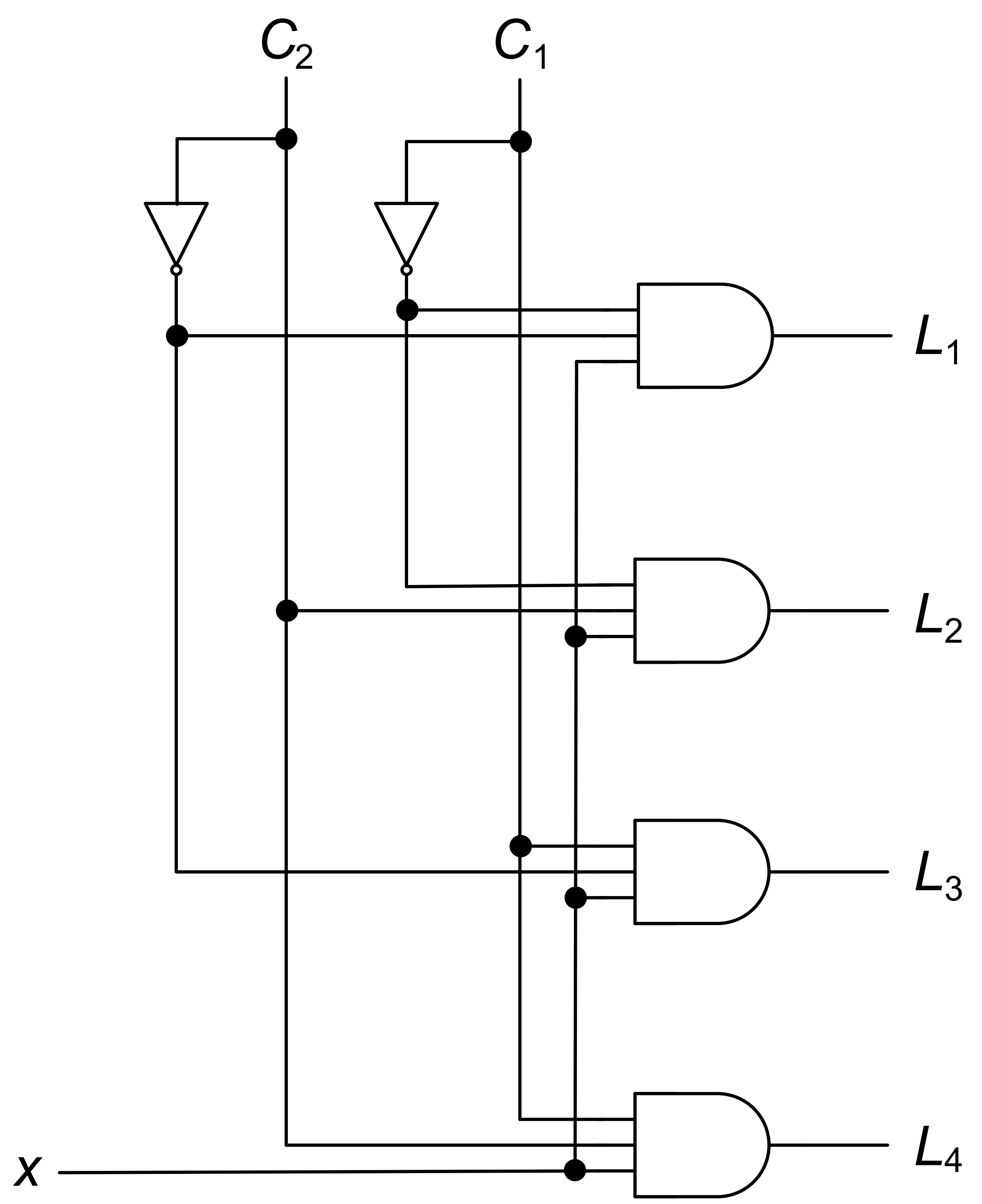
(*b*) Implementation.

$$P = x'y'z + x'yz' + xy'z' + xyz = x \oplus y \oplus z$$

Combinational Circuits: Serial to Parallel

Serial-to-parallel converter: distributes a sequence of binary digits on a serial input to a set of different outputs, as specified by external control signals

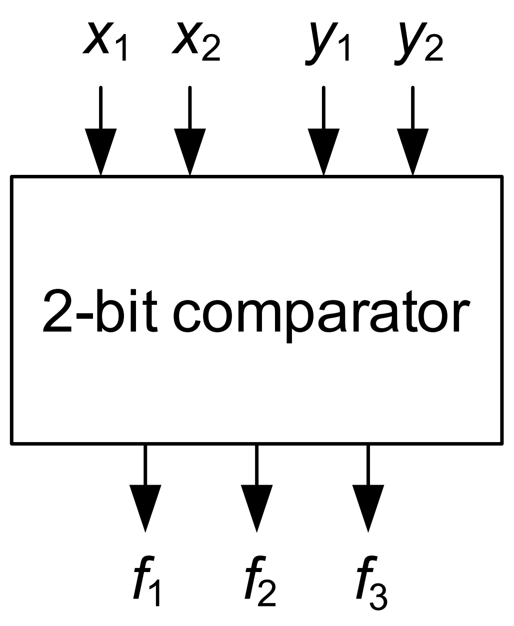
Control		*Output lines*				*Logic equations*
C_1	C_2	L_1	L_2	L_3	L_4	
0	0	x	0	0	0	$L_1 = xC_1'C_2'$
0	1	0	x	0	0	$L_2 = xC_1'C_2$
1	0	0	0	x	0	$L_3 = xC_1C_2'$
1	1	0	0	0	x	$L_4 = xC_1C_2$



Combinational Circuits: Comparators

***n*-bit comparator**: compares the magnitude of two numbers X and Y, and has three outputs f_1, f_2, and f_3

- $f_1 = 1$ iff $X > Y$
- $f_2 = 1$ iff $X = Y$
- $f_3 = 1$ iff $X < Y$



(*a*) Block diagram.

$f_1 = ?$

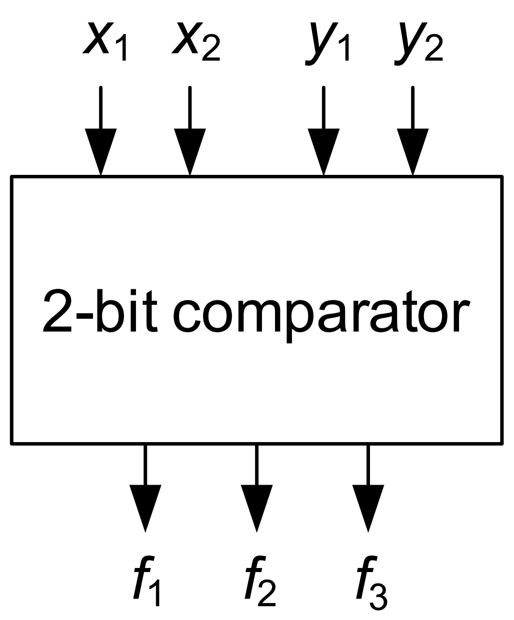
$f_2 = ?$

$f_3 = ?$

Combinational Circuits: Comparators

***n*-bit comparator**: compares the magnitude of two numbers X and Y, and has three outputs f_1, f_2, and f_3

- $f_1 = 1$ iff $X > Y$
- $f_2 = 1$ iff $X = Y$
- $f_3 = 1$ iff $X < Y$



(*a*) Block diagram.

y_1y_2 \ x_1x_2	00	01	11	10
00	2	1	1	1
01	3	2	1	1
11	3	3	2	3
10	3	3	1	2

(*b*) Map for f_1, f_2, and f_3.

$f_1 = ?$

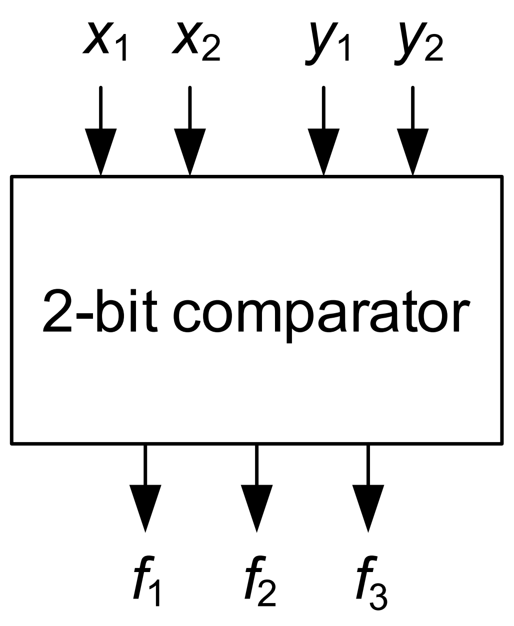
$f_2 = ?$

$f_3 = ?$

Combinational Circuits: Comparators

***n*-bit comparator**: compares the magnitude of two numbers X and Y, and has three outputs f_1, f_2, and f_3

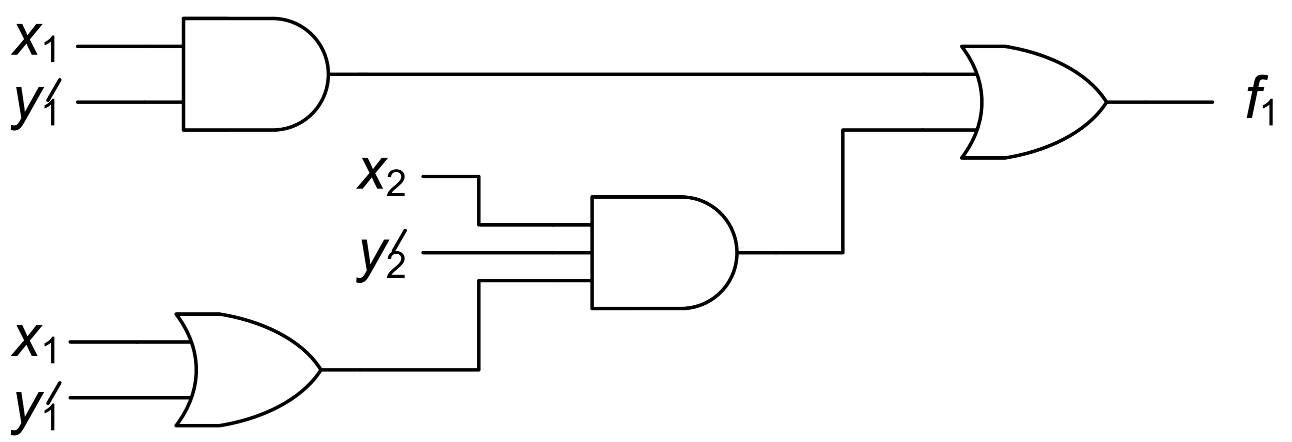
- $f_1 = 1$ iff $X > Y$
- $f_2 = 1$ iff $X = Y$
- $f_3 = 1$ iff $X < Y$



(*a*) Block diagram.

y_1y_2 \ x_1x_2	00	01	11	10
00	2	1	1	1
01	3	2	1	1
11	3	3	2	3
10	3	3	1	2

(*b*) Map for f_1, f_2, and f_3.



(*c*) Circuit for f_1.

$$f_1 = x_1x_2y_2' + x_2y_1'y_2' + x_1y_1' = (x_1 + y_1')x_2y_2' + x_1y_1'$$

$$\begin{aligned} f_2 &= x_1'x_2'y_1'y_2' + x_1'x_2y_1'y_2 + x_1x_2'y_1y_2' + x_1x_2y_1y_2 \\ &= x_1'y_1'(x_2'y_2' + x_2y_2) + x_1y_1(x_2'y_2' + x_2y_2) \\ &= (x_1'y_1' + x_1y_1)(x_2'y_2' + x_2y_2) \end{aligned}$$

$$f_3 = x_2'y_1y_2 + x_1'x_2'y_2 + x_1'y_1 = x_2'y_2(y_1 + x_1') + x_1'y_1$$

Combinational Circuits: Comparators

Four-bit comparator: 8 inputs (four for *A*, four for *B*, and three outputs *A*>*B*, *A*<*B* and *A*=*B*

$$x_i = A_i B_i + A_i' B_i' \qquad i = 0, 1, 2, 3$$

$$(A = B) = x_3 x_2 x_1 x_0$$

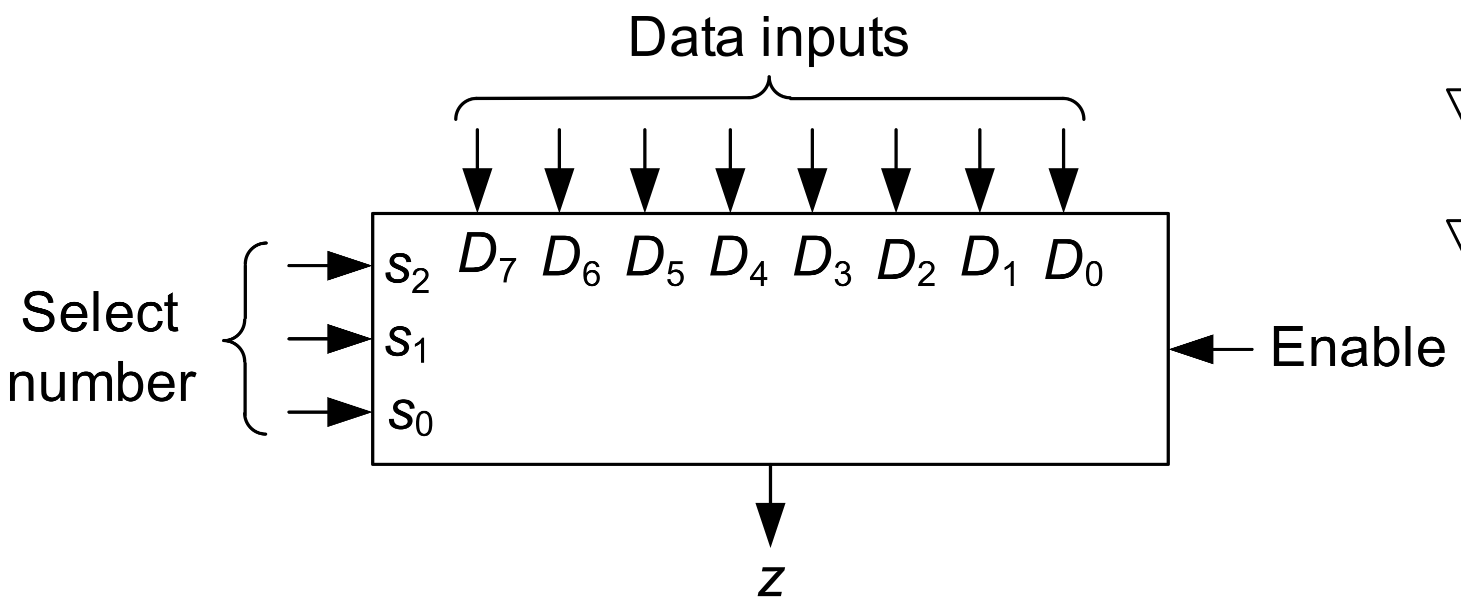
$$(A > B) = A_3 B_3' + x_3 A_2 B_2' + x_3 x_2 A_1 B_1' + x_3 x_2 x_1 A_0 B_0'$$

$$(A < B) = A_3' B_3 + x_3 A_2' B_2 + x_3 x_2 A_1' B_1 + x_3 x_2 x_1 A_0' B_0$$

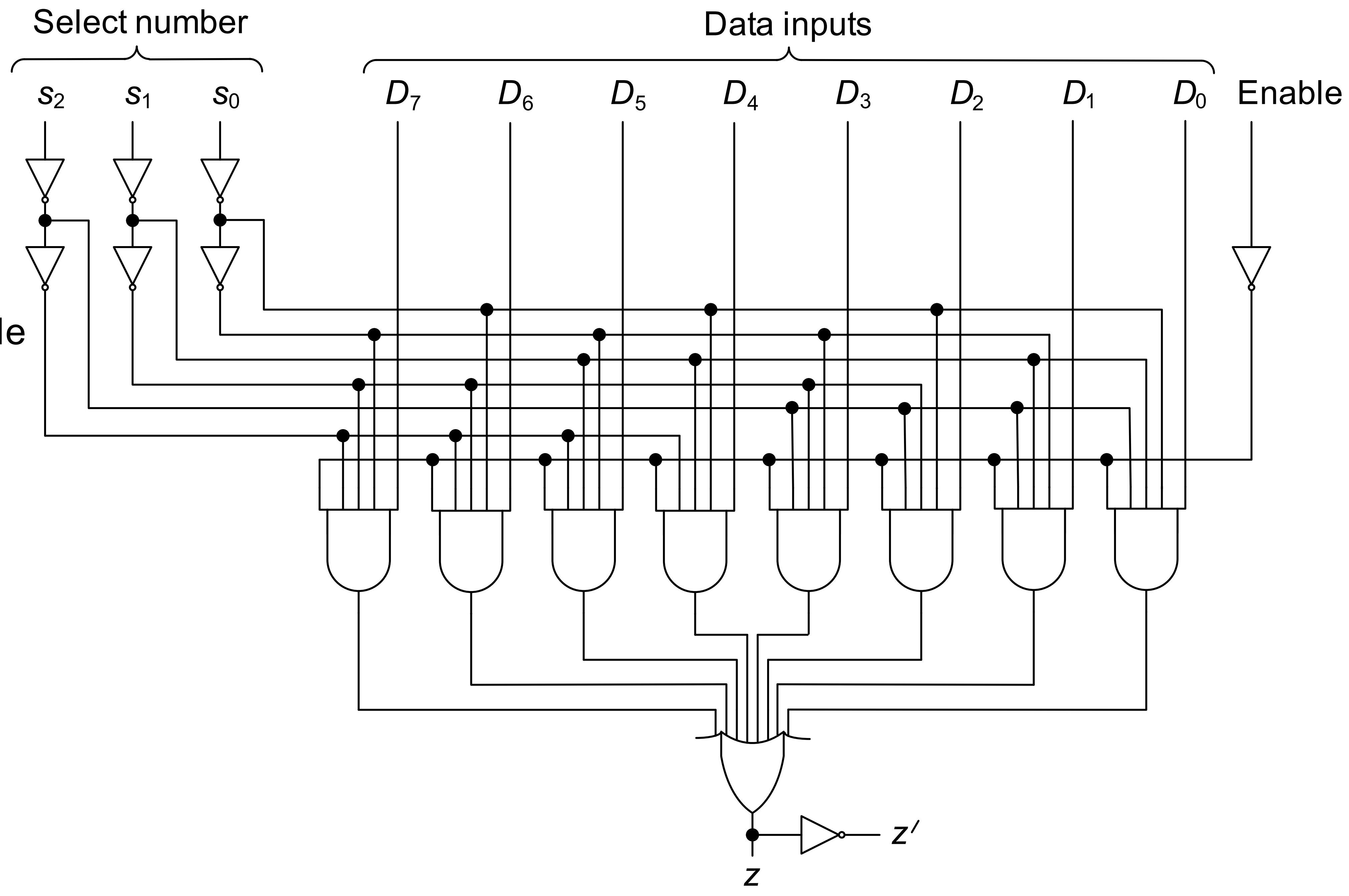
Combinational Circuits: Multiplexers

Multiplexer: electronic switch that connects one of n inputs to the output
Data selector: application of multiplexer

- n data input lines, D_0, D_1, …, D_{n-1}
- m select digit inputs s_0, s_1, …, s_{m-1}
- 1 output
- **Can you design a simple data selectors with 2 input data lines?**



(*a*) Block diagram.

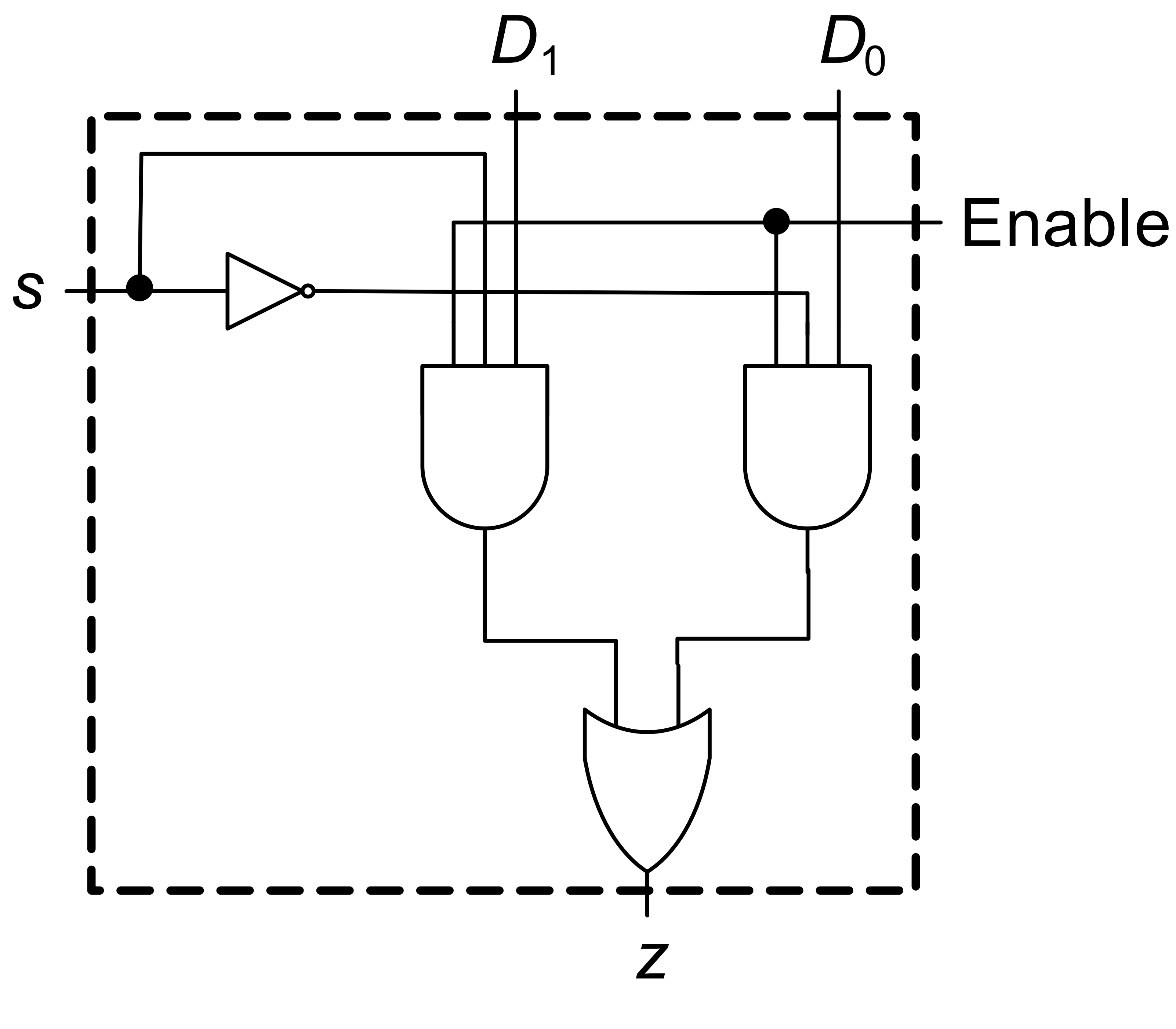


(*b*) Logic diagram.

Combinational Circuits: Multiplexers

Data selectors: can implement arbitrary switching functions

Example: implementing two-variable functions



$z = sD_1 + s'D_0$

If $s = A$, $B = D_0$, and $B' = D_1$, then $z = A \oplus B$.

Implementing Switching Function with Mux

To implement an *n*-variable function: a data selector with n-1 select inputs and 2^{n-1} data inputs

Implementing three-variable functions:

$$z = s_2's_1'D_0 + s_2's_1D_1 + s_2s_1'D_2 + s_2s_1D_3$$

Example: $s_1 = A$, $s_2 = B$, $D_0 = C$, $D_1 = 1$, $D_2 = 0$, $D_3 = C'$

$$z = A'B'C + AB' + ABC'$$

$$= AC' + B'C$$

General case: Assign n-1 variables to the select inputs and last variable and constants 0 and 1 to the data inputs such that desired function results