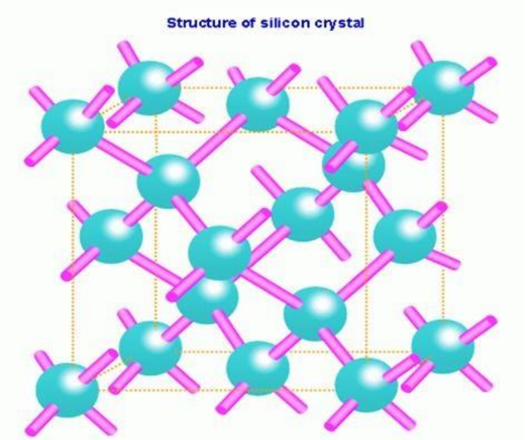
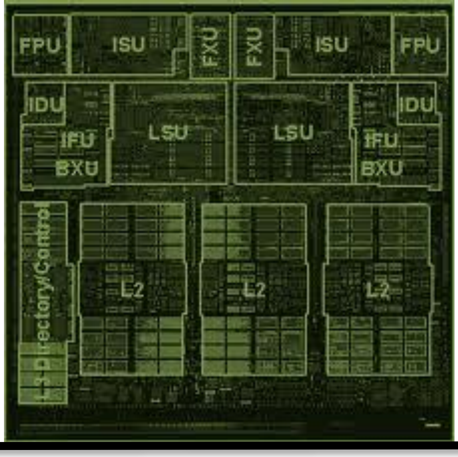
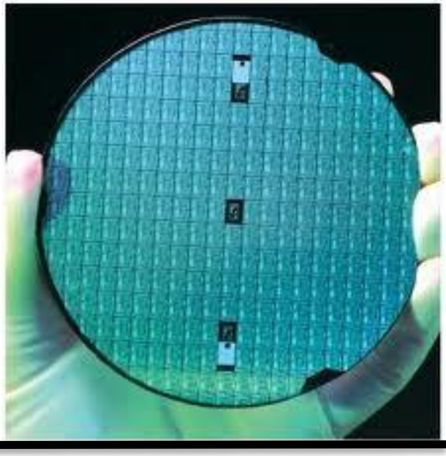






ECE 122A

VLSI Principles

Lecture 3

Prof. Kaustav Banerjee
Electrical and Computer Engineering
University of California, Santa Barbara
E-mail: kaustav@ece.ucsb.edu

NAND Gate

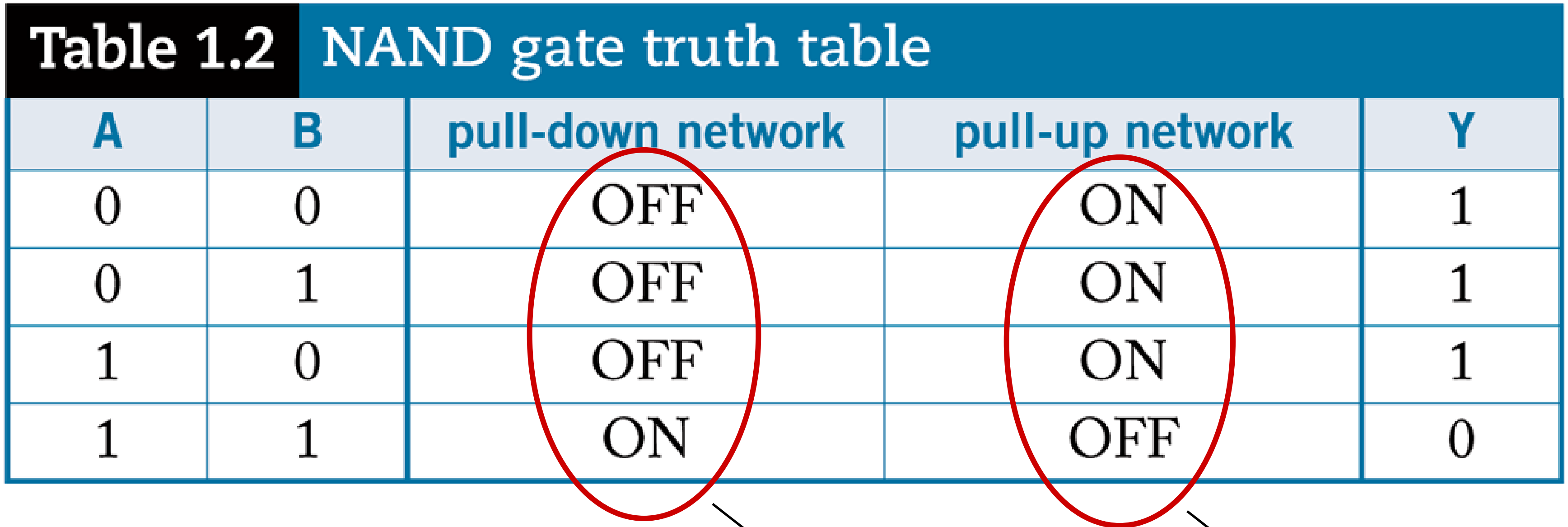
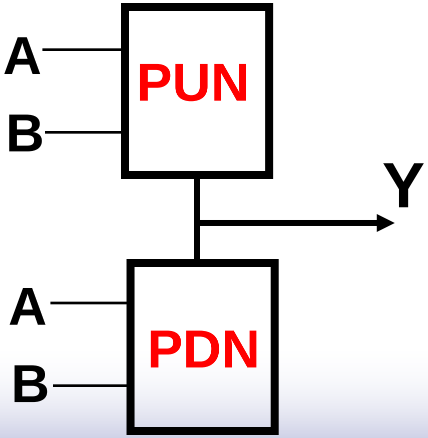


Table 1.2 NAND gate truth table

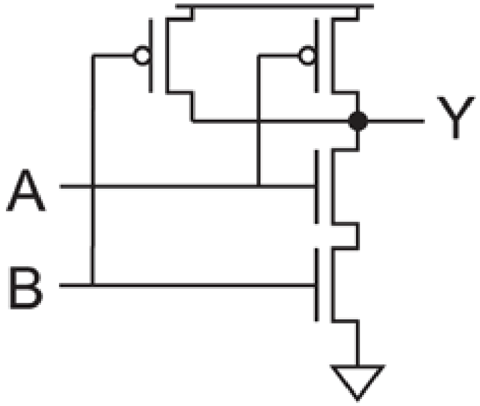
A	B	pull-down network	pull-up network	Y
0	0	OFF	ON	1
0	1	OFF	ON	1
1	0	OFF	ON	1
1	1	ON	OFF	0

2 NMOS transistors must be in series….

2 PMOS transistors must be in parallel….



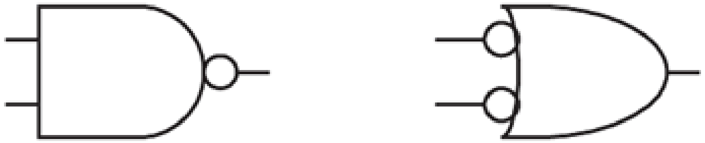
CMOS NAND Implementation



(a)

Recall De Morgan's Law…..

$$Y = \overline{A \cdot B} = \overline{A} + \overline{B}$$



(b)

Also, $Y = \overline{A + B} = \overline{A} \cdot \overline{B}$

FIG 1.11 2-input NAND gate schematic (a) and symbol (b) $Y = \overline{A \bullet B}$

CMOS 3-input NAND Implementation

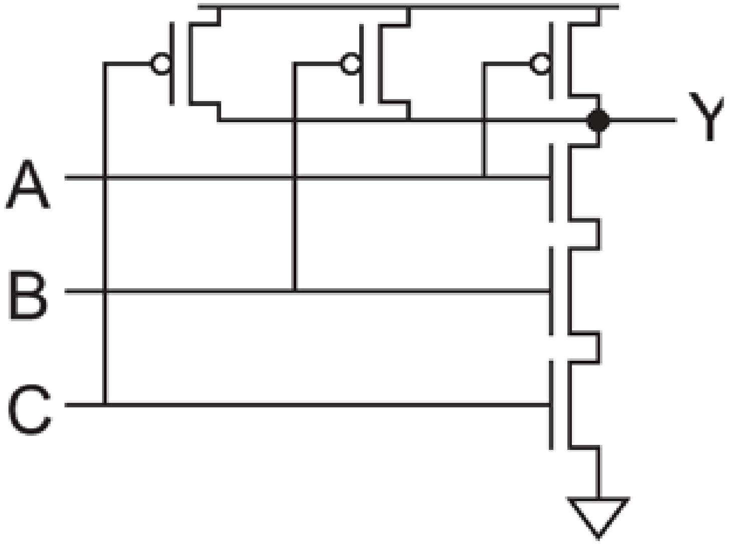


FIG 1.12 3-input NAND gate schematic $Y = \overline{A \bullet B \bullet C}$

Y=0, when A=B=C=1

Hence, A, B, C are in series for the NMOS (pull-down network)

Y=1, when A or B or C=0

Hence, A, B, C are in parallel for the PMOS (pull-up network)

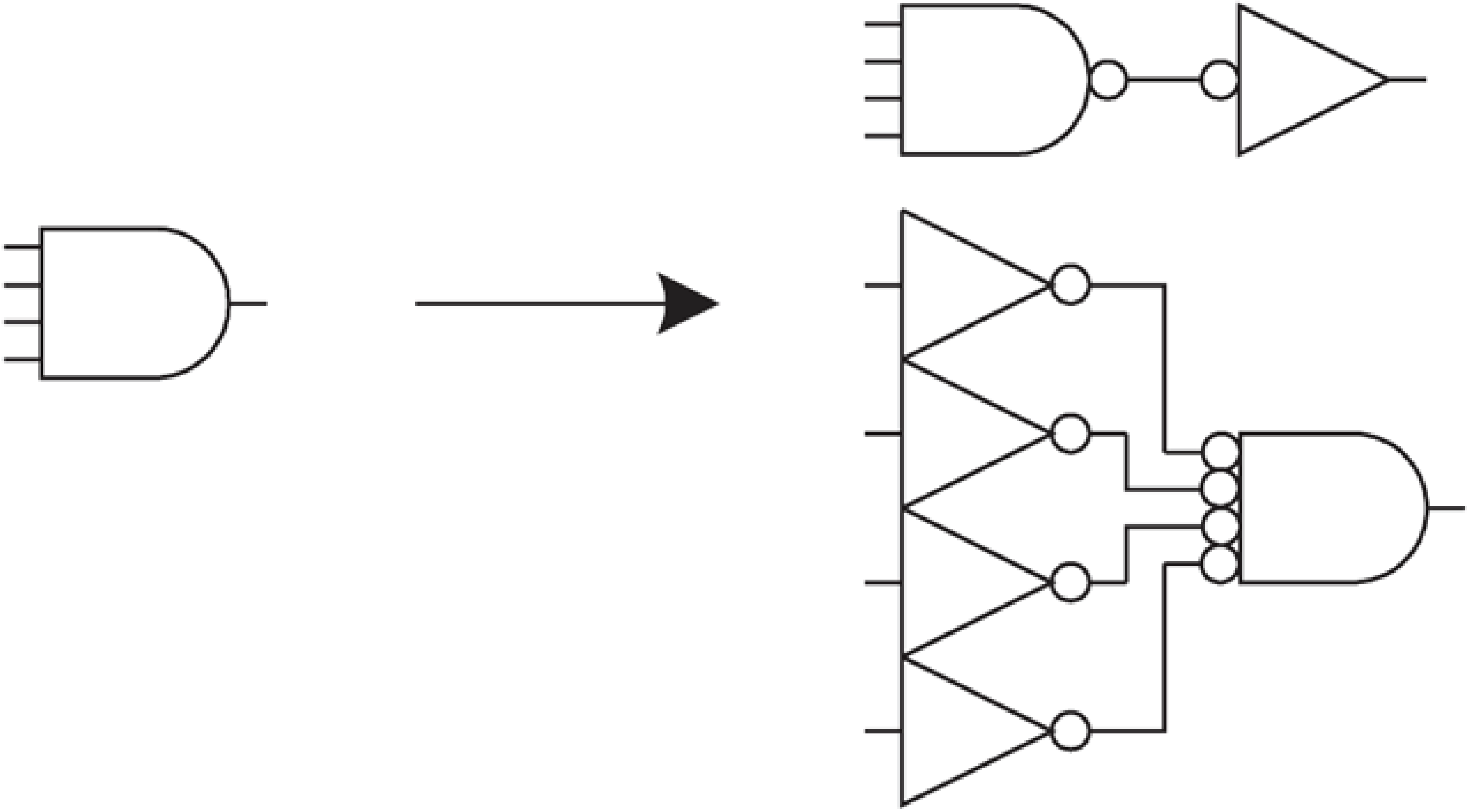


FIG 1.22 Various implementations of a CMOS 4-input AND gate

NOR Gate

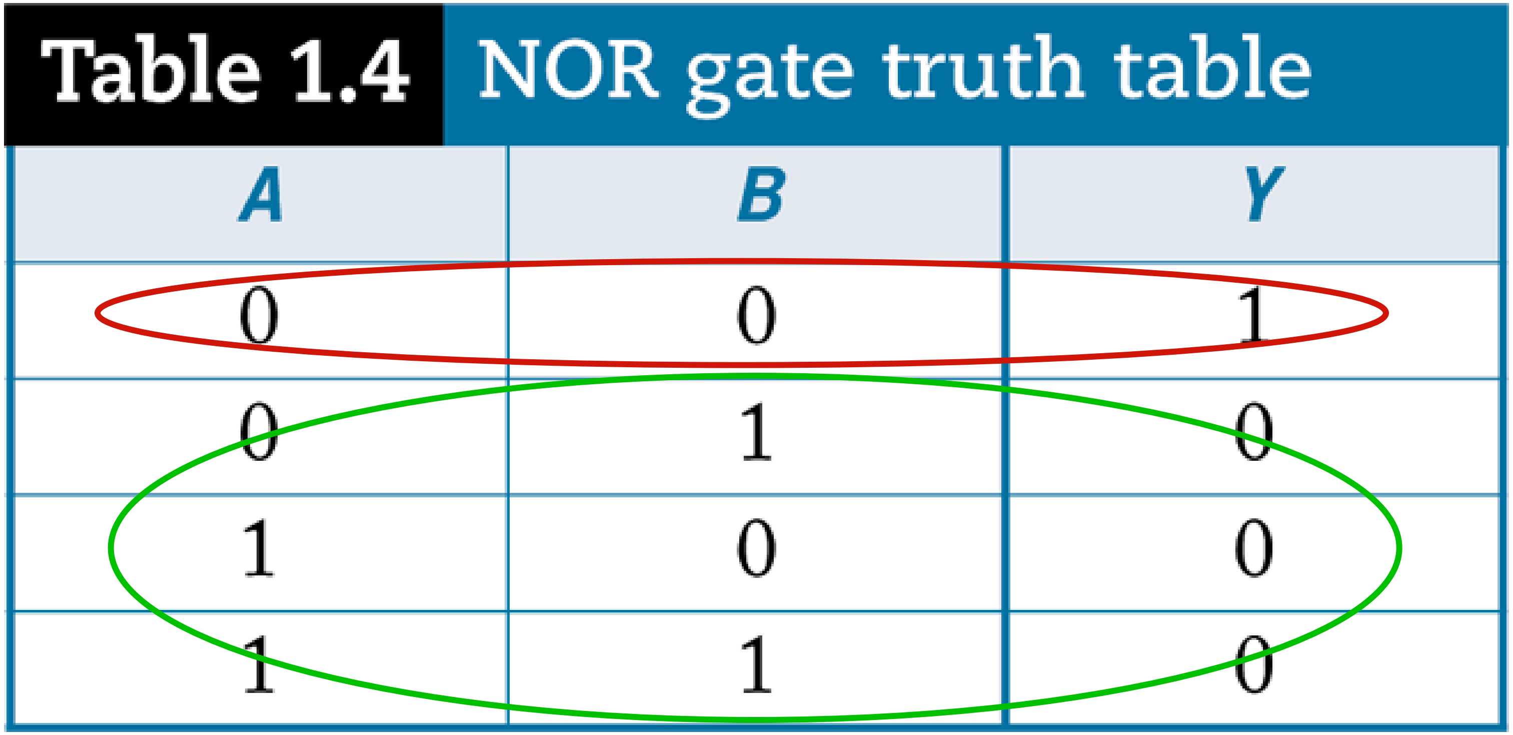


Table 1.4 NOR gate truth table

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

2 PMOS must be in series…..

2 NMOS must be in parallel….

CMOS NOR Implementation

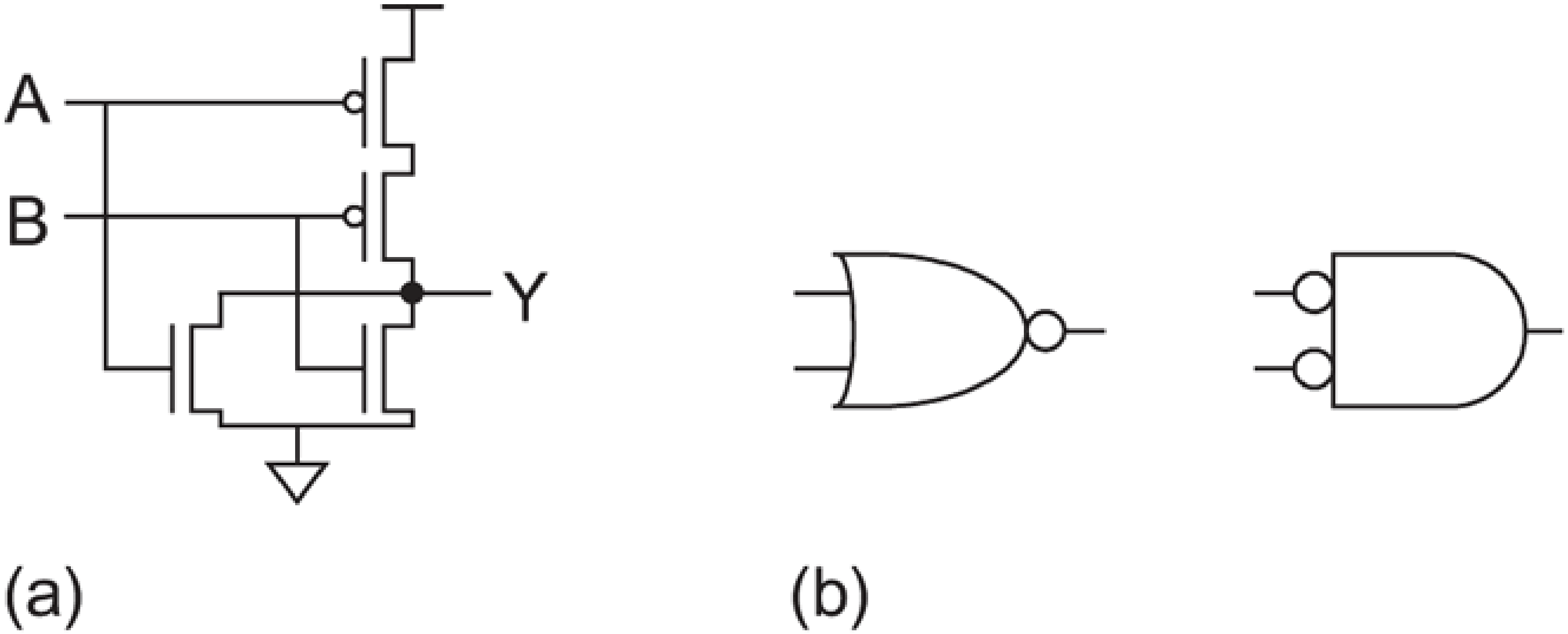


FIG 1.15 2-input NOR gate schematic (a) and symbol (b) $Y = \overline{A + B}$

CMOS 3-input NOR Implementation

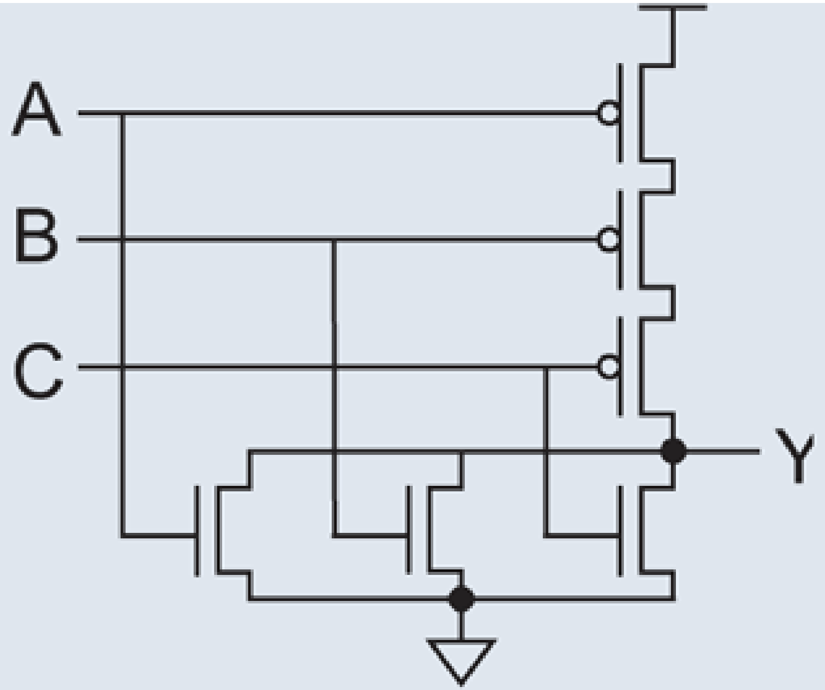


FIG 1.16 3-input NOR gate schematic $Y = \overline{A + B + C}$

Combinational Logic

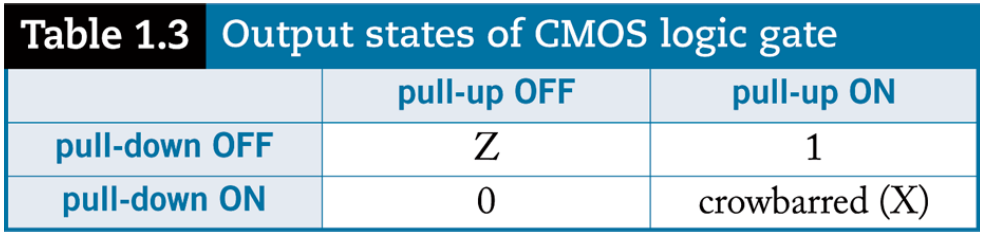


Table 1.3 Output states of CMOS logic gate

	pull-up OFF	pull-up ON
pull-down OFF	Z	1
pull-down ON	0	crowbarred (X)

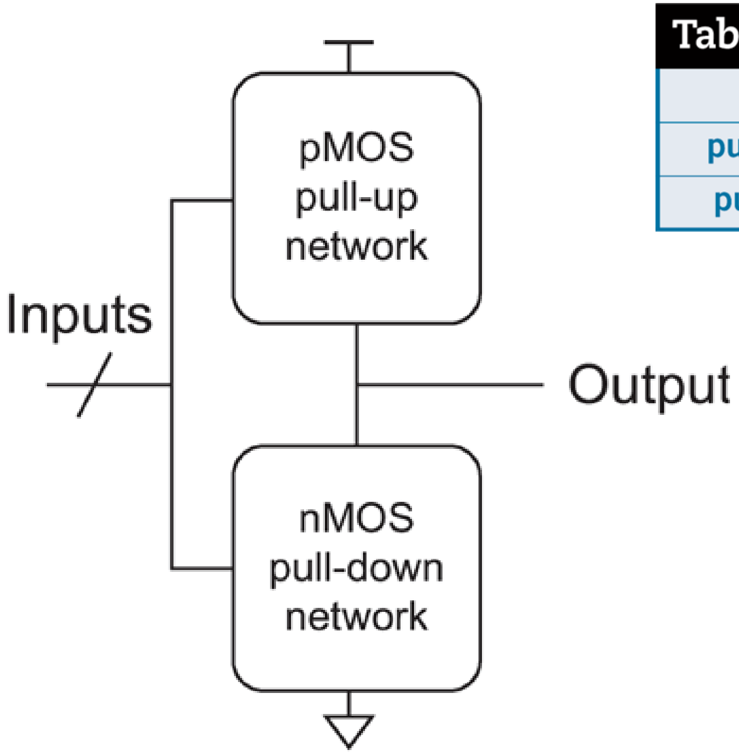


FIG 1.13 General logic gate using pull-up and pull-down networks

Output state X should always be avoided: static power

Output state Z indicates a high-impedance state. If a gate's output state = Z, that gate has no influence on the rest of the circuit

Z-state is used in tri-state logic circuits and is of relevance in certain gates such as multiplexers

Compound Gates

$Y = \overline{A.B + C.D}$

Needs 20 transistors….

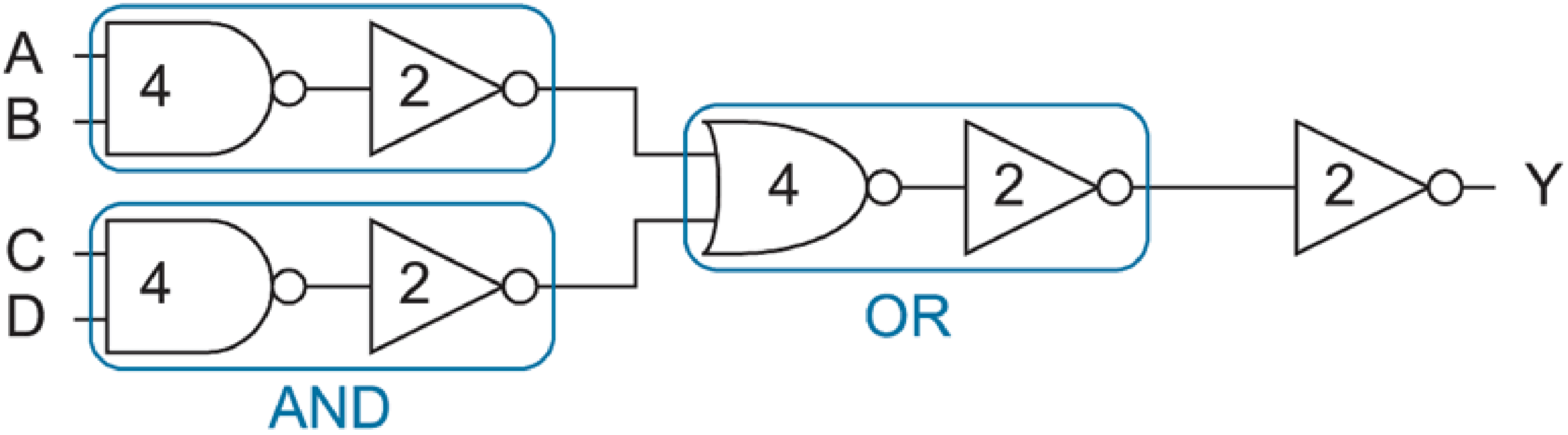
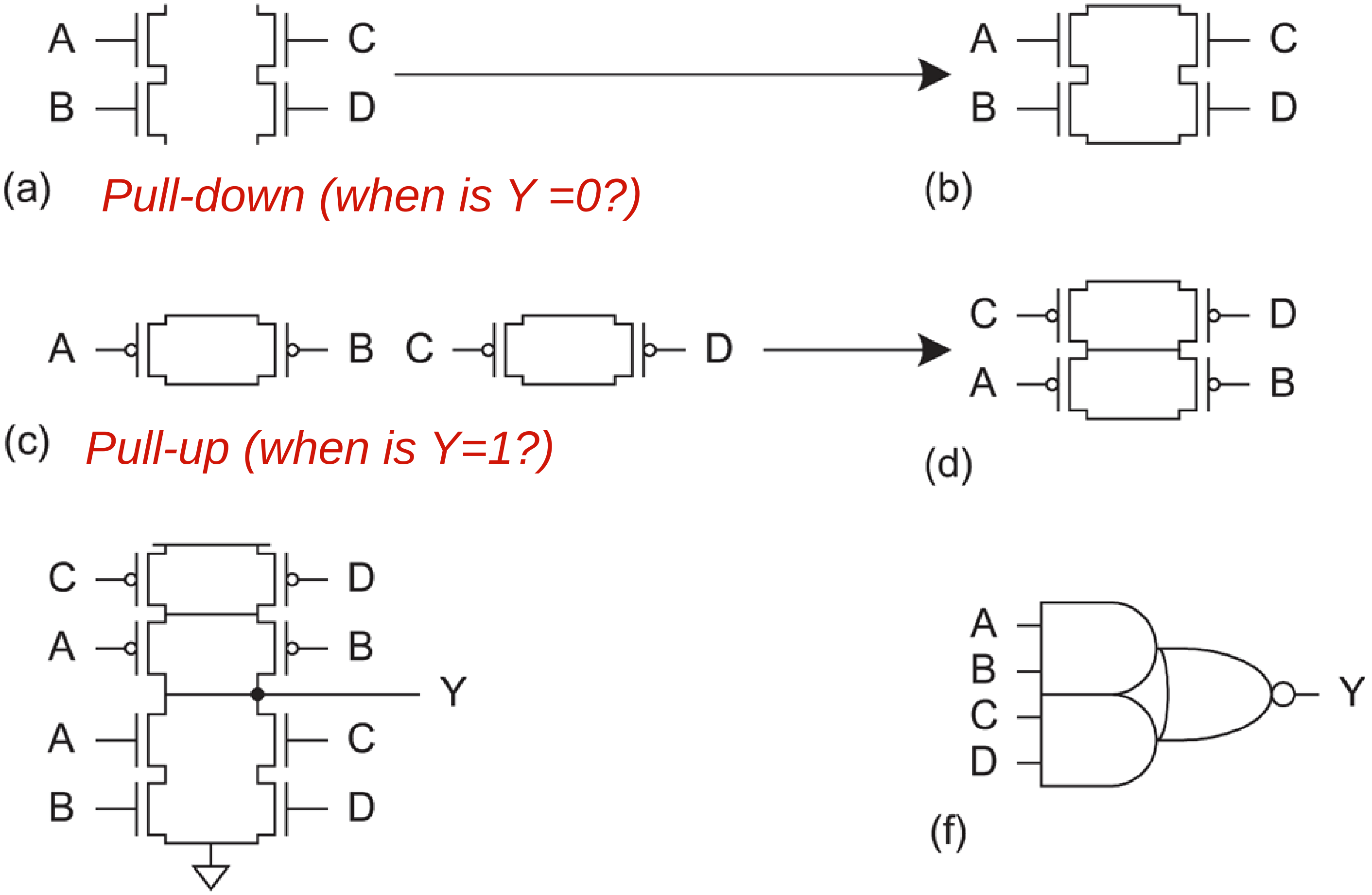


FIG 1.23 Inefficient discrete gate implementation of AOI22 indicating transistor counts

Compound Gates

$Y = \overline{A.B + C.D}$



Pull-down (when is Y =0?)

Pull-up (when is Y=1?)

Need 8 transistors

Compound Gates

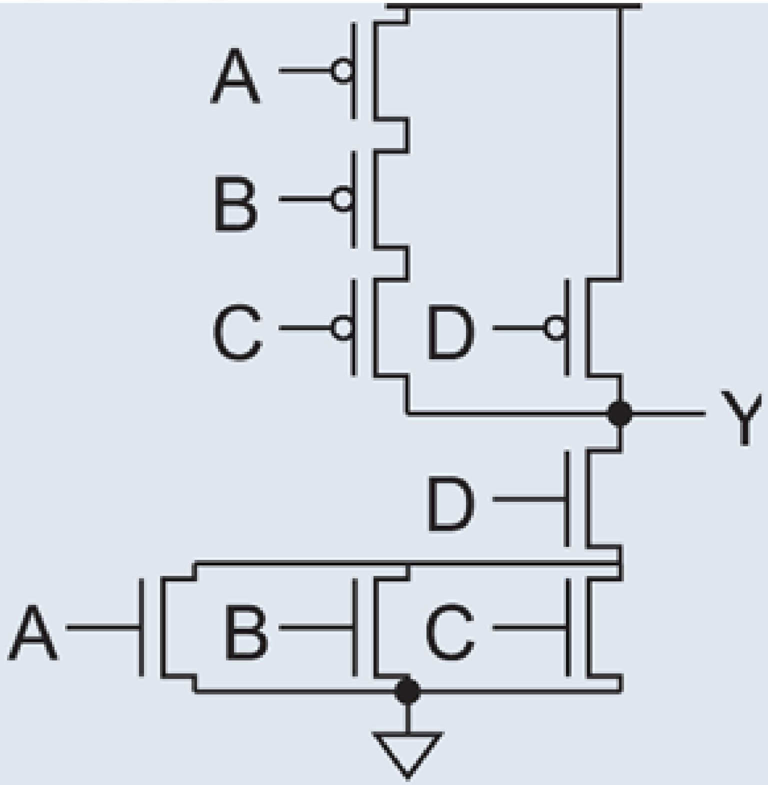


FIG 1.18 CMOS compound gate for function $Y = \overline{(A + B + C) \bullet D}$

Pass Transistors

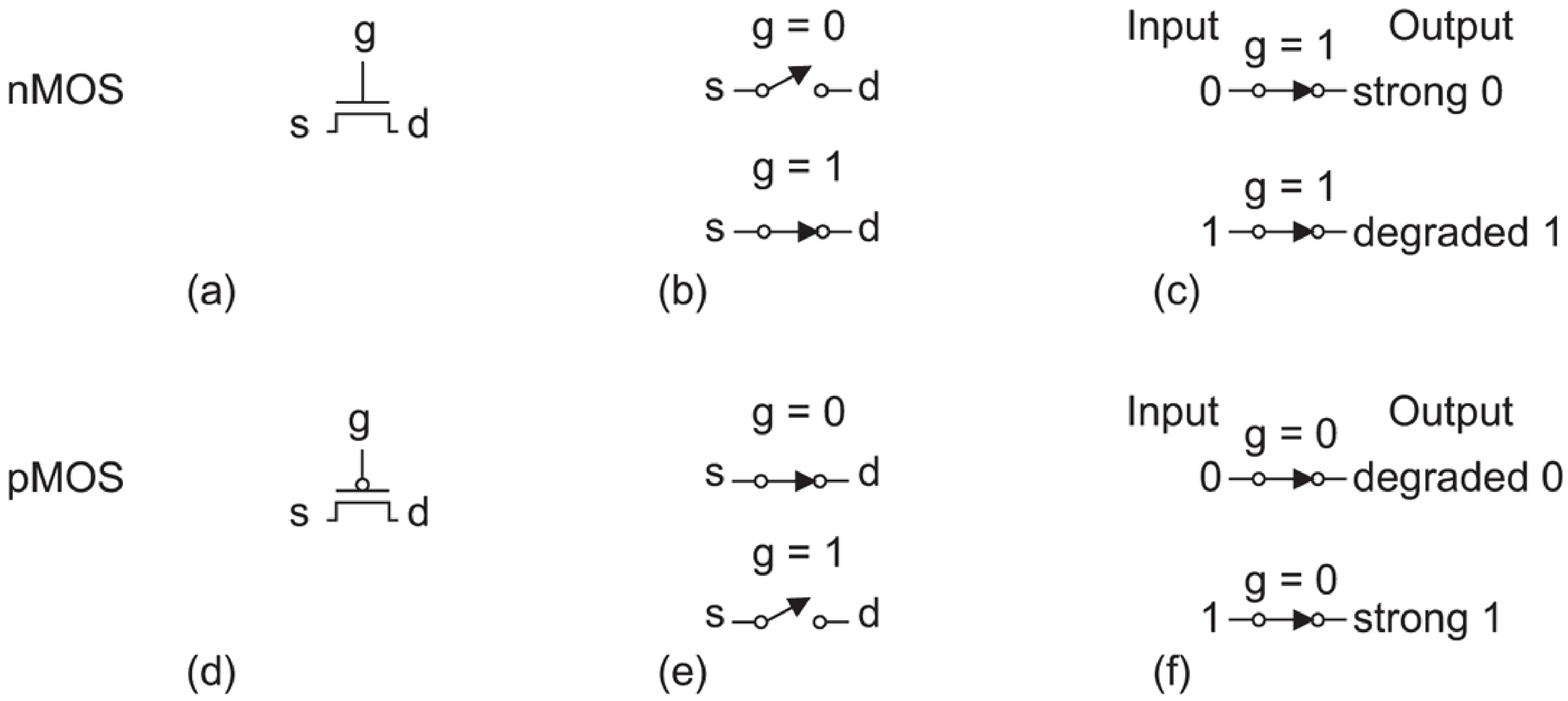
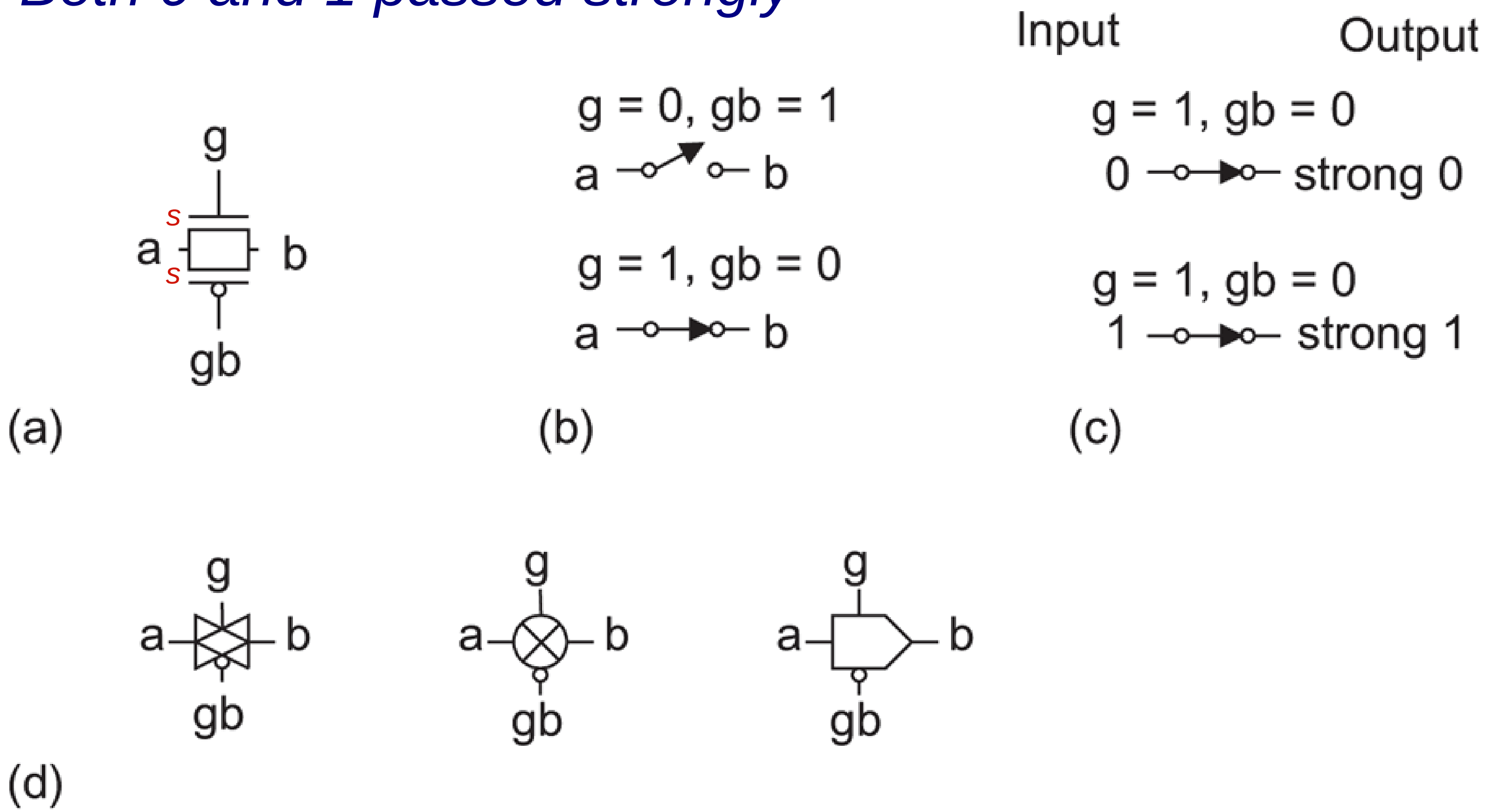


FIG 1.19 Pass transistor strong and degraded outputs

Transmission Gates: Pass Transistors in Parallel

Both 0 and 1 passed strongly



Double Rail Logic: both the control input and its complement is required

FIG 1.20 Transmission gate

Tristate Buffer

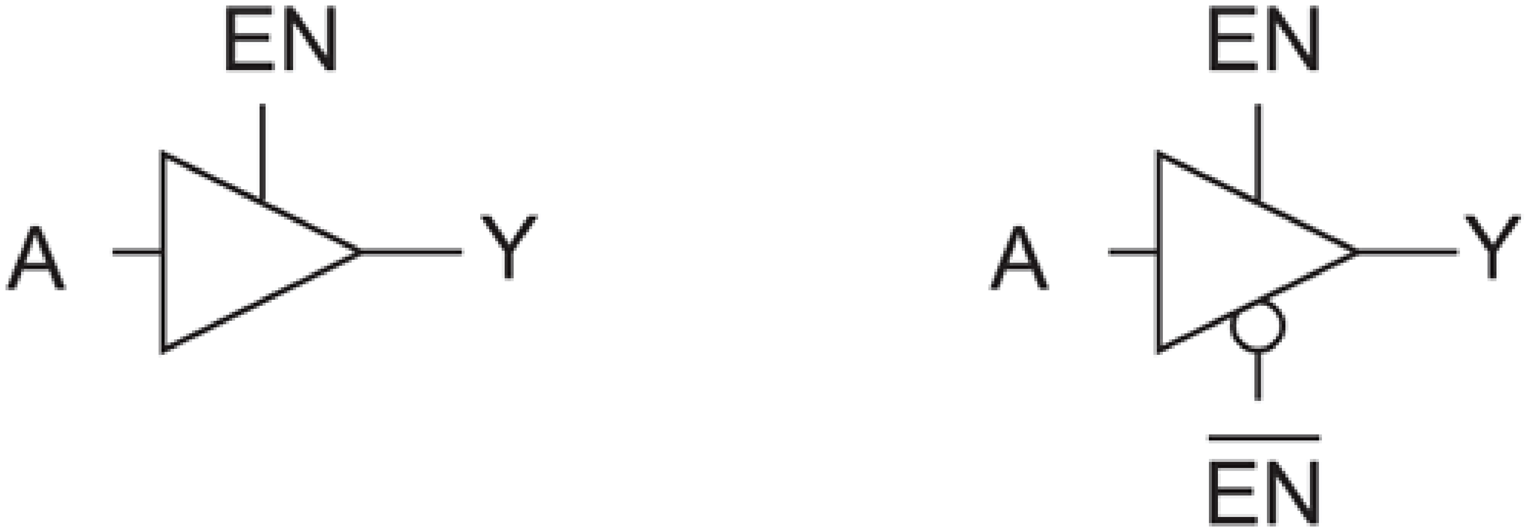


FIG 1.24 Tristate buffer symbol

Tristate Buffer

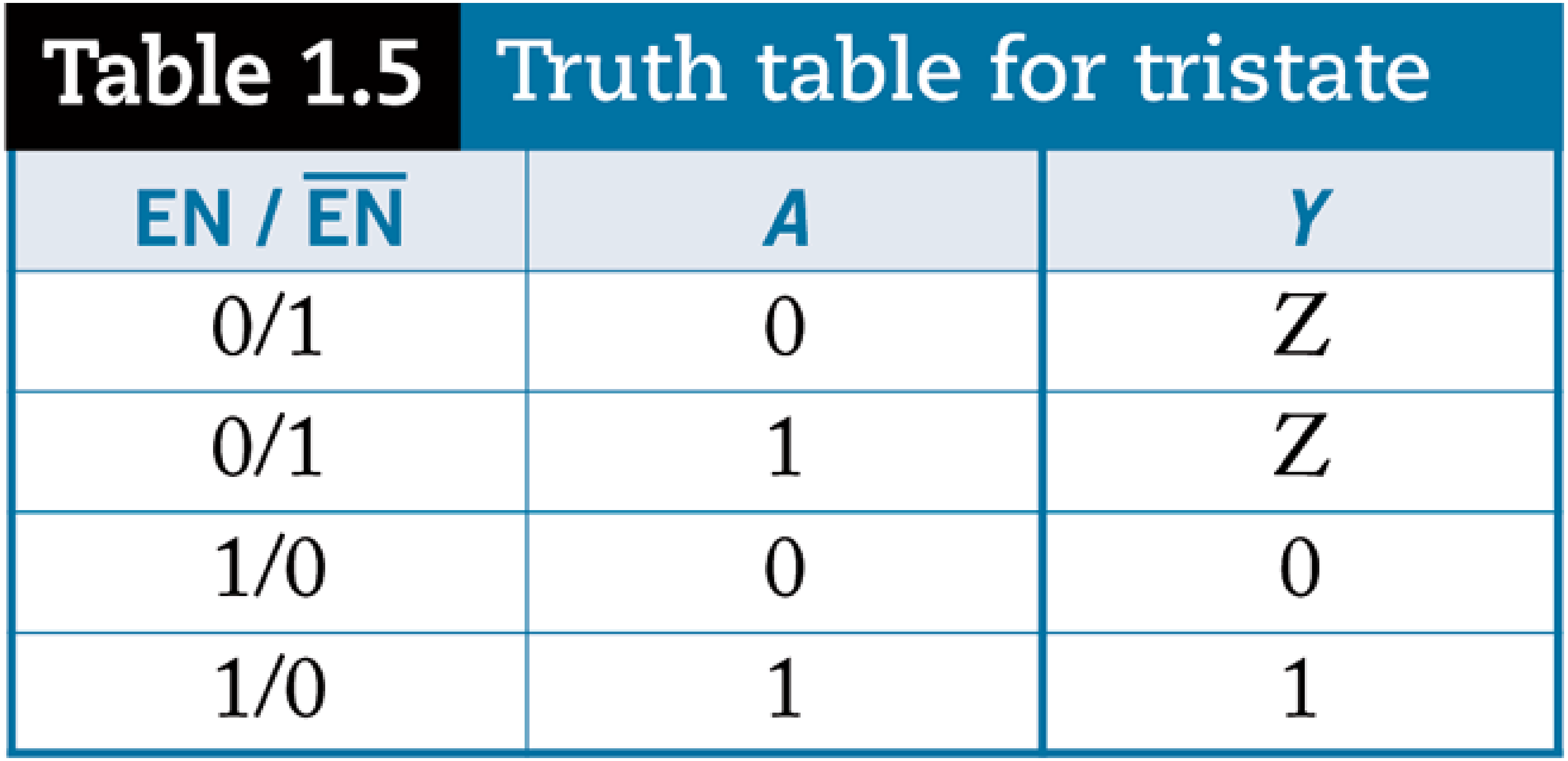


Table 1.5 Truth table for tristate

EN / $\overline{\text{EN}}$	*A*	*Y*
0/1	0	Z
0/1	1	Z
1/0	0	0
1/0	1	1

Note: Z indicates a ‘high-impedance’ third state….

Transmission Gate as Tristate Buffer

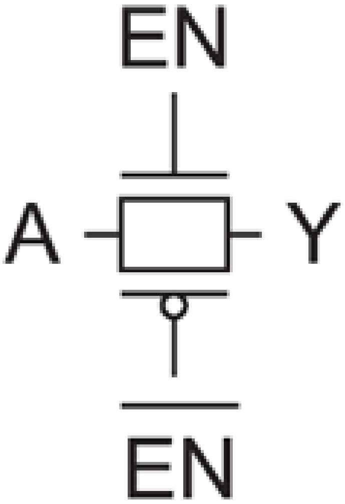


FIG 1.25 Transmission gate

Non-restoring: input-signal will slowly degrade over a ***number of stages….*** *since* ***Y*** *is not connected to* ***Vdd or Gnd***

Tristate Buffer as Inverter

Restoring:** O/P (**Y**) is directly connected to **Vdd** or **GND

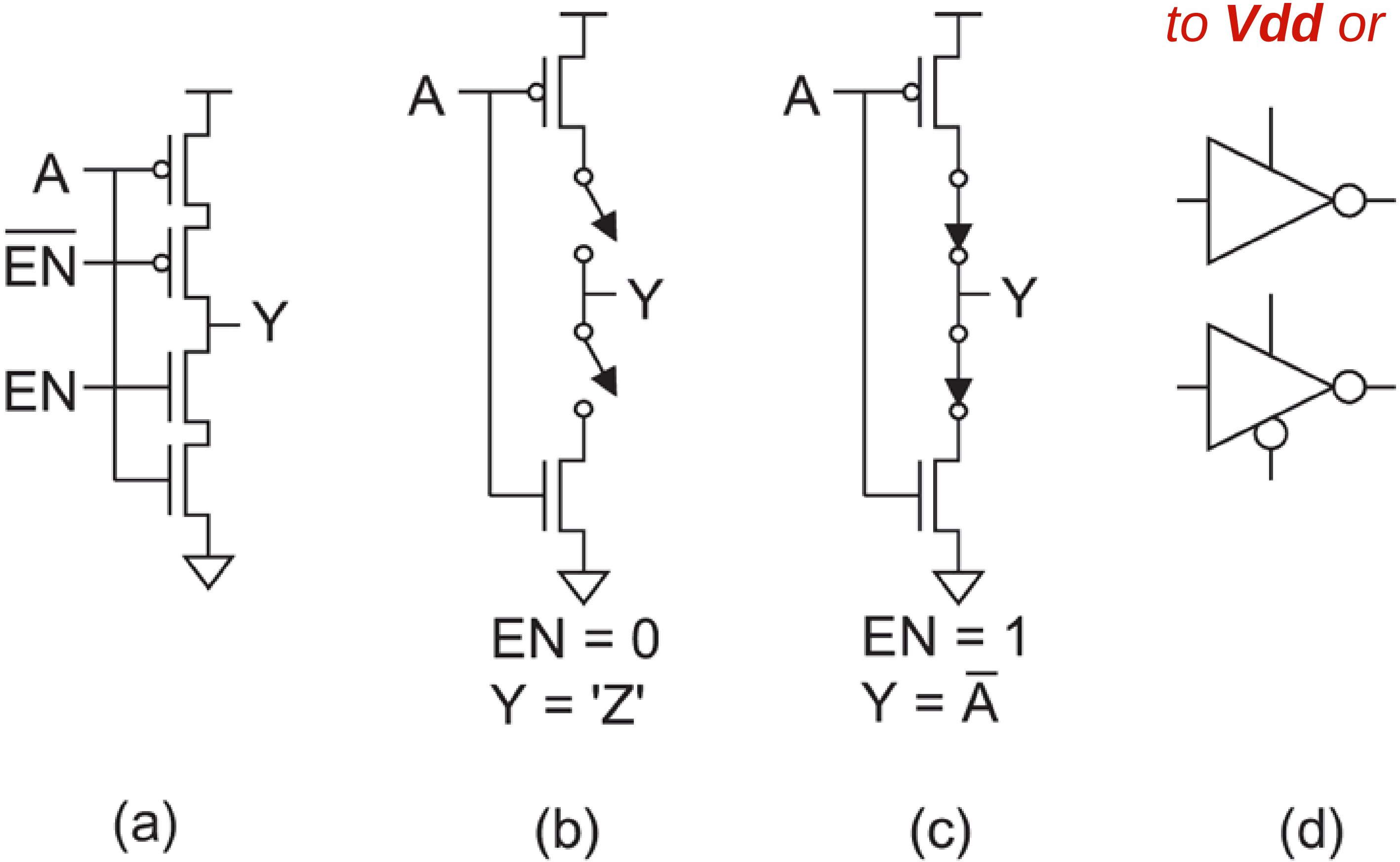
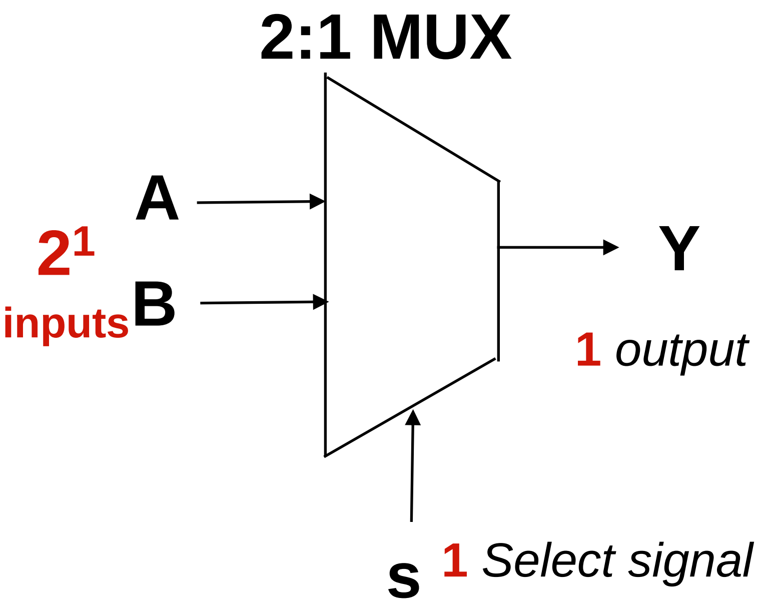


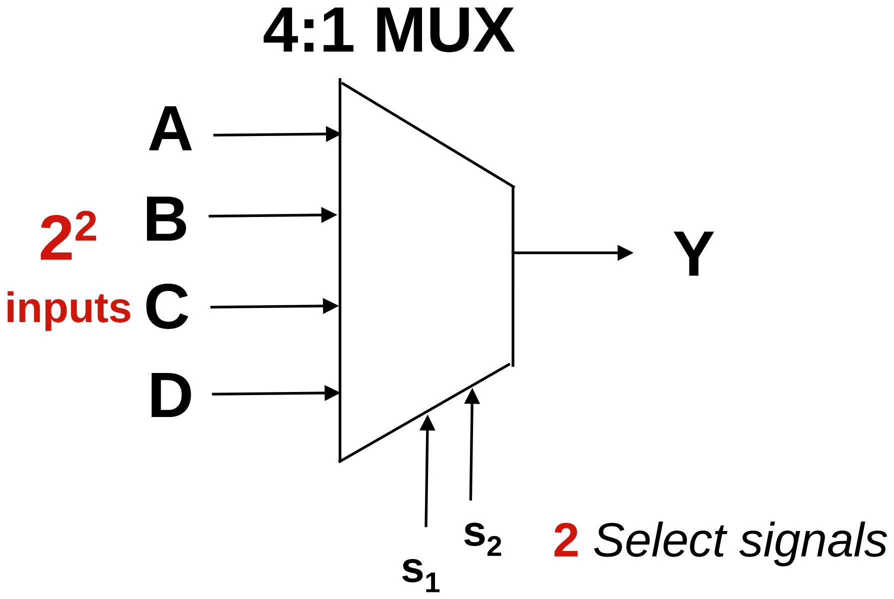
FIG 1.26 Tristate inverter

Multiplexer (MUX)

*Connects **one** of **n inputs** to the **output**….*

Used as data selectors…encoders





Y = As +Bs'

$Y = As_1s_2 + +Bs_1s_2' + Cs_1's_2 + Ds_1's_2'$

In general, 2^n inputs will have n select signals

$$Y = \sum_{k=0}^{2^n - 1} m_k I_k$$

m_k is a minterm of the n control variables and I_k is the corresponding data input

Multiplexer (MUX)

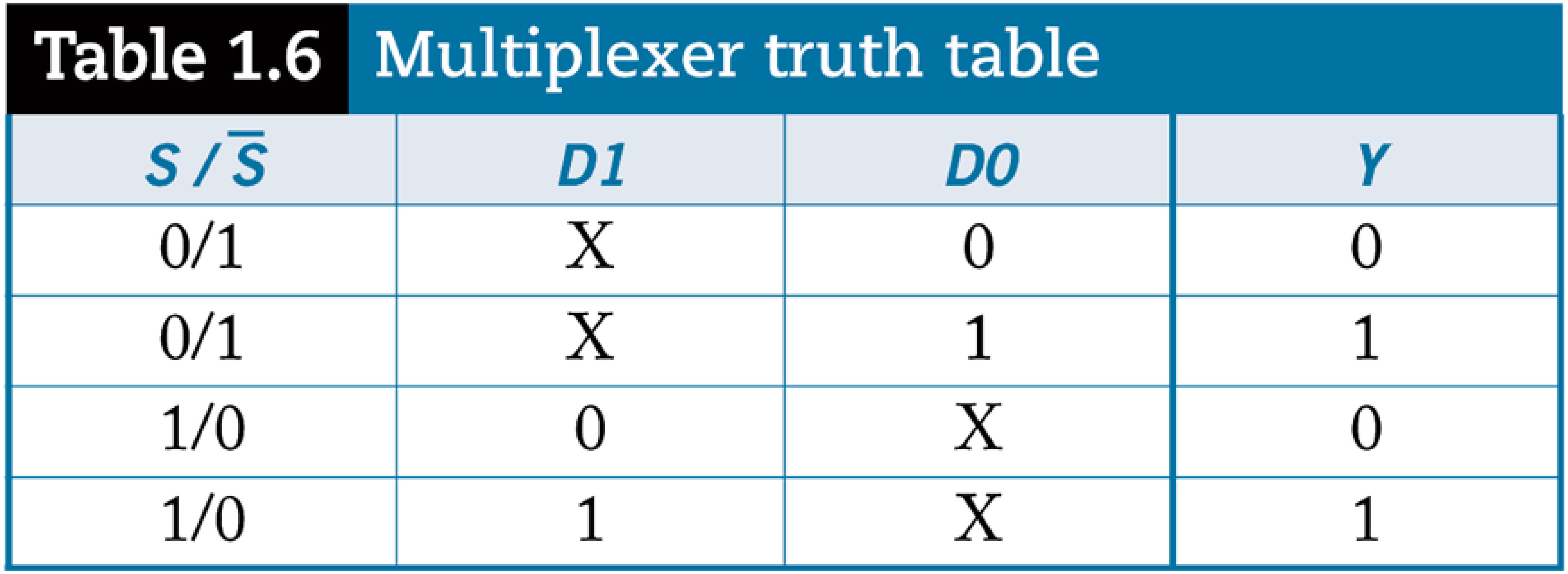


Table 1.6 Multiplexer truth table

$S / \overline{S}$	$D1$	$D0$	Y
0/1	X	0	0
0/1	X	1	1
1/0	0	X	0
1/0	1	X	1

$Y = D1.S + D0.\overline{S}$

Note: X indicates a don’t care condition

Non-restoring MUX

$Y = D1.S + D0.\bar{S}$

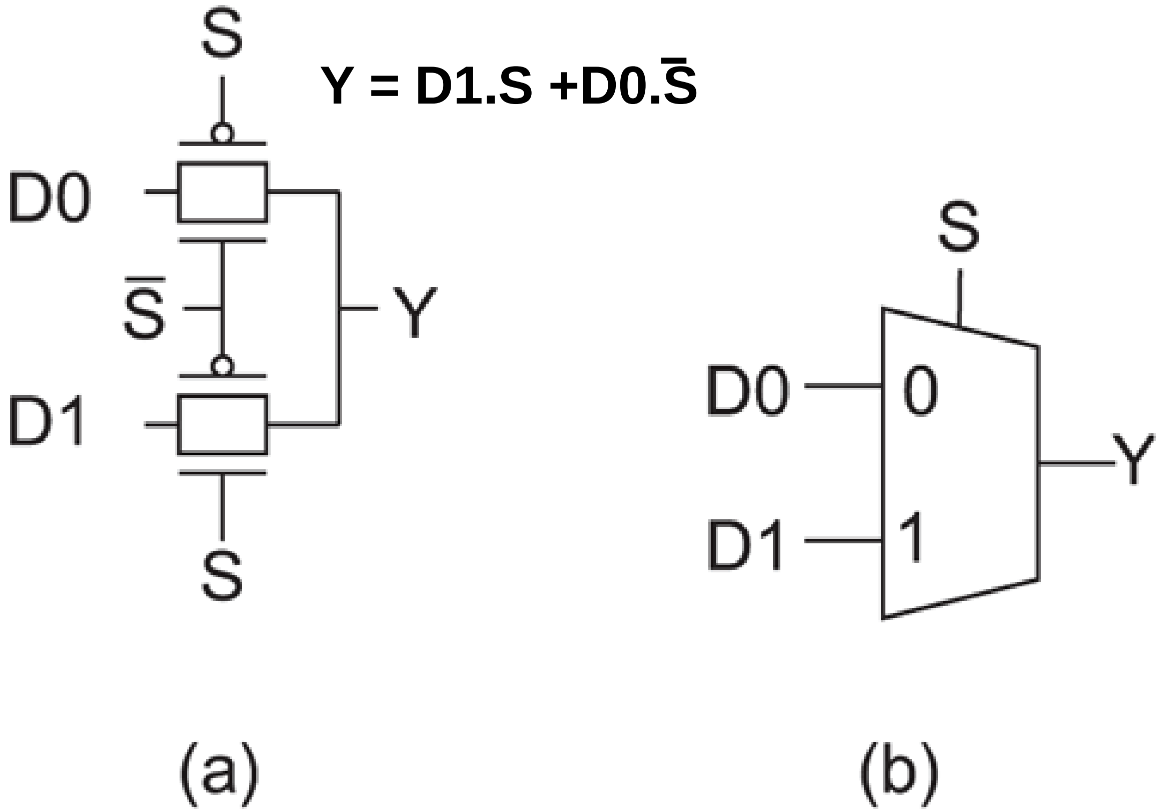


FIG 1.27 Transmission gate multiplexer

Inverting and Restoring MUX

$S/\overline{S} = 0/1$

$\overline{Y} = \mathbf{D1.S + D0.\overline{S}}$

$D0 = 0\text{: } Y = 1 = \overline{D0}$

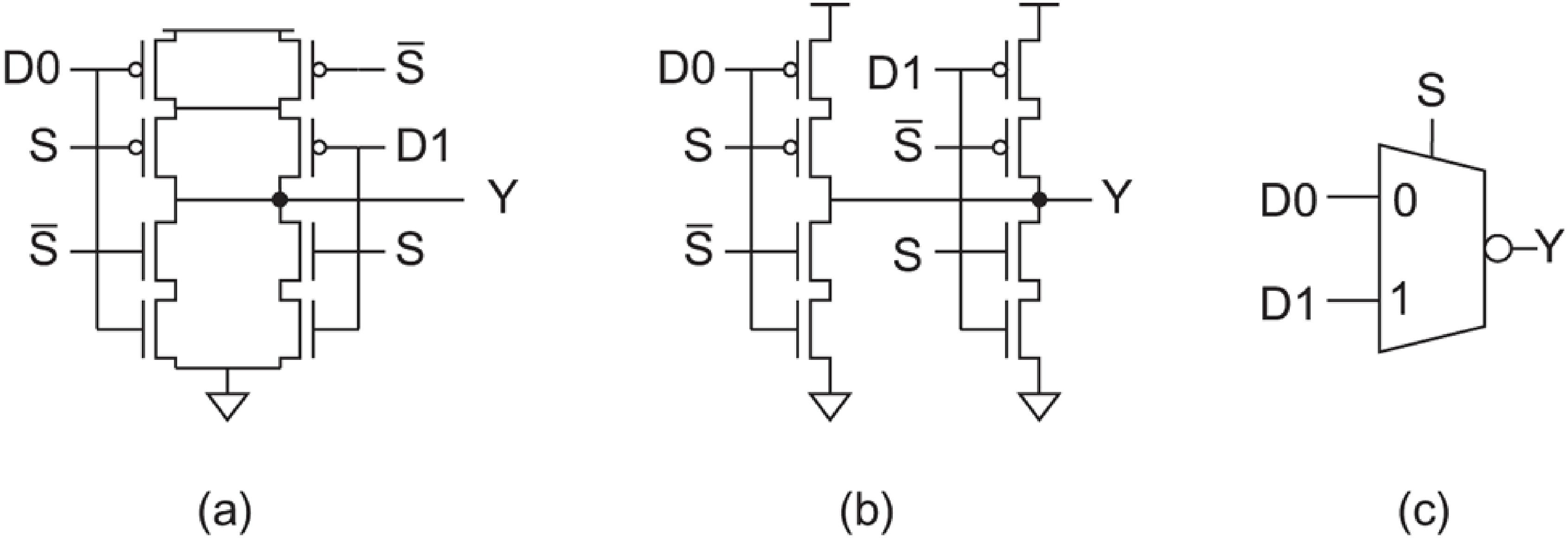
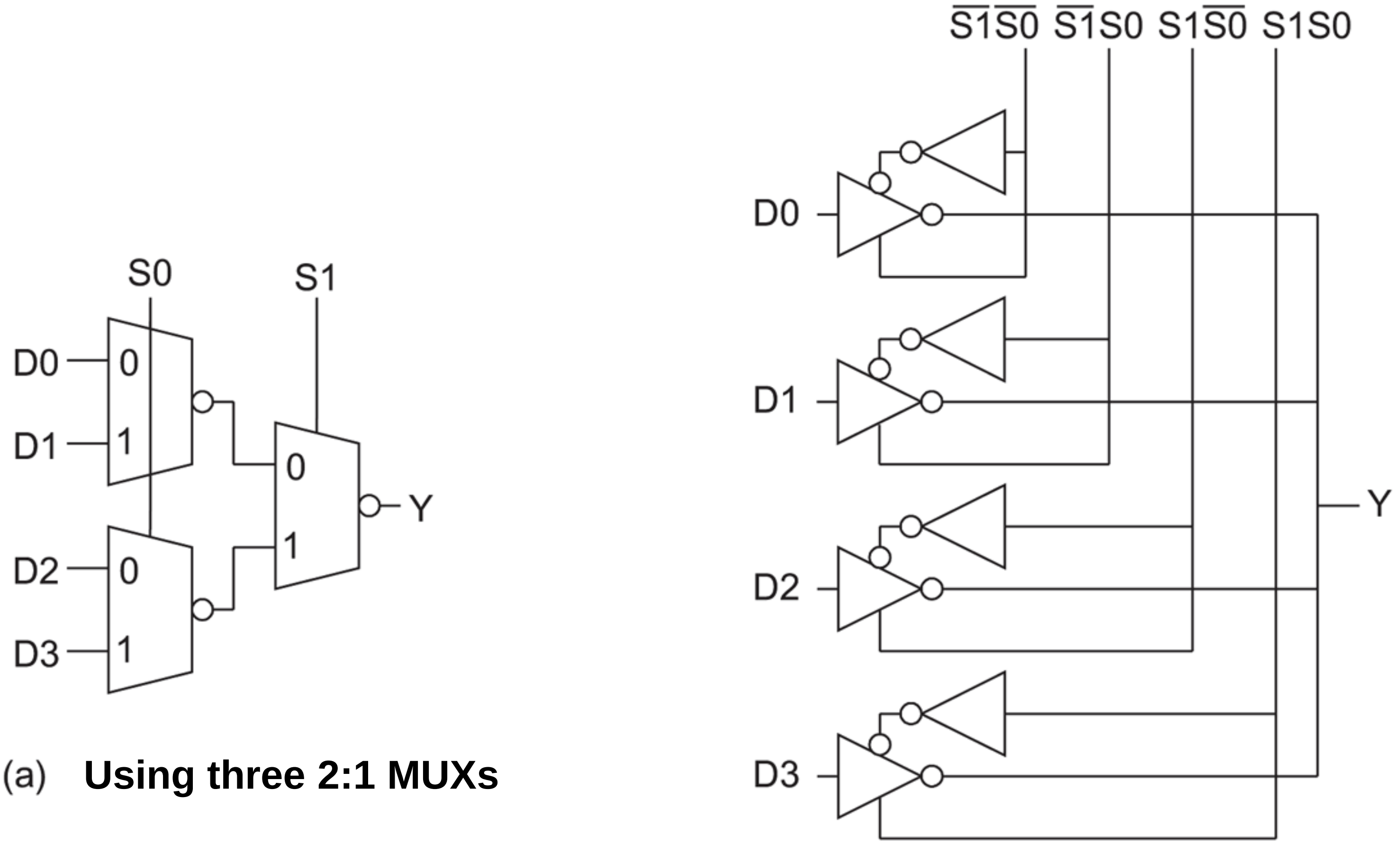


FIG 1.28 Inverting multiplexer

A 4:1 MUX


S0
S1
D0
D1
D2
D3
0
1
0
1
0
1
Y
$\overline{S1}\,\overline{S0}$ $\overline{S1}S0$ $S1\overline{S0}$ $S1S0$
D0
D1
D2
D3
Y

(a) **Using three 2:1 MUXs**

(b)

FIG 1.29 4:1 multiplexer

Static CMOS Summary

- ❑ In static circuits at every point in time (except when switching) the output is connected to either GND or V_{DD} via a low resistance path.
  - fan-in of *n* (or n-inputs) requires 2*n* (*n* N-type + *n* P-type) devices
- ❑ Non-ratioed logic: gates operate independent of PMOS or NMOS sizes (since no conflict between pull-up and pull-down networks)
- ❑ No path ever exists between Vdd and GND: low static power
- ❑ Fully-restored logic: (NMOS passes "0" only and PMOS passes "1" only
- ❑ Gates must be INVERTING: $Y = \overline{X}$, so that X=1 (NMOS pull-down network is "ON") for Y=0 (node is fully discharged)

Latches *(level sensitive device)*

CLK=1: D to Q

CLK=0:Holds state of Q

As long as CLK remains high: D to Q

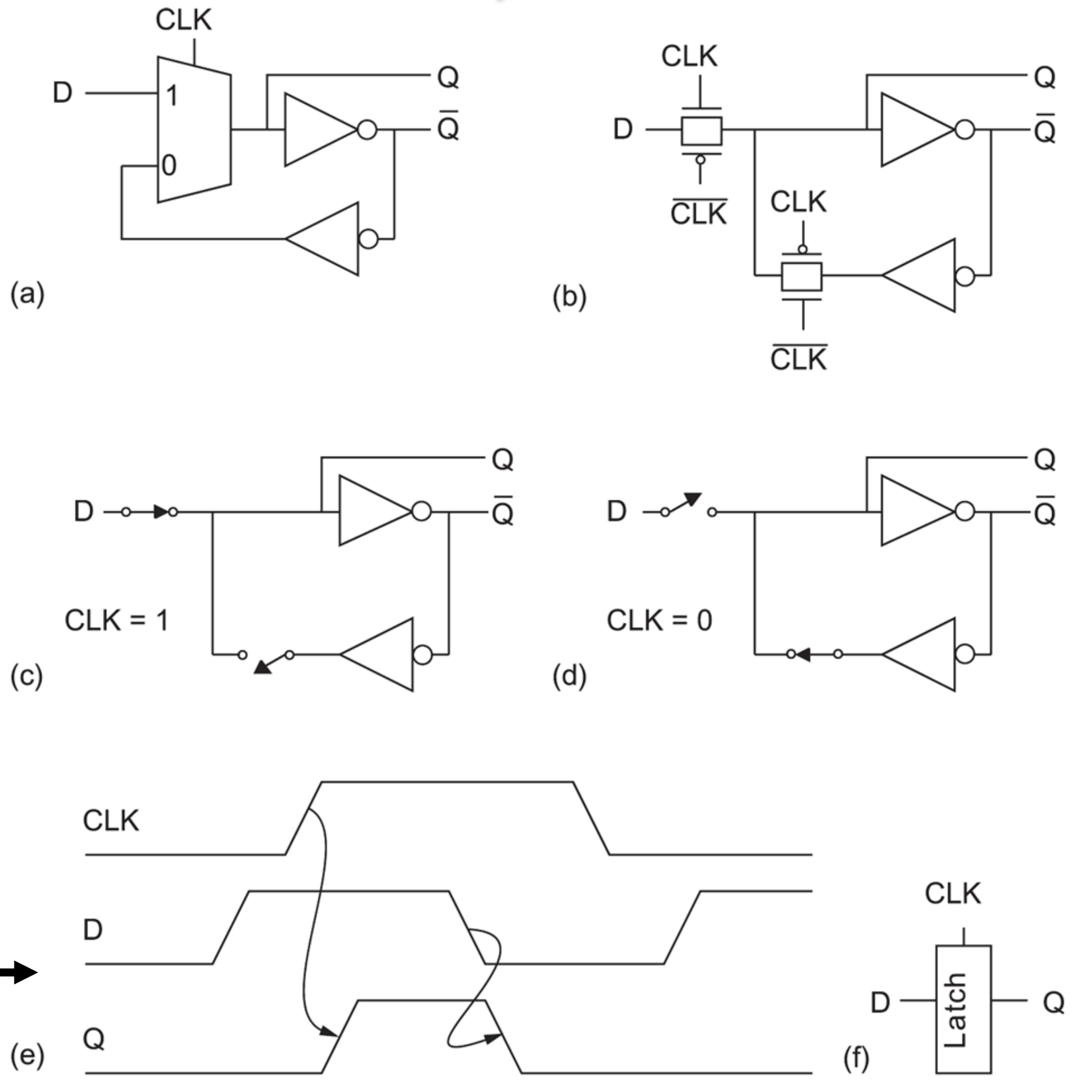


FIG 1.30 CMOS positive-level-sensitive D latch

Flip-Flops

(edge-triggered device)

Combines two latches:

One +ve sensitive (slave) and one –ve sensitive latch (master)

Edge Triggered FF or Master-Slave FF

CLK=0: D to QM

$\overline{QM} = \overline{D}$

Slave holds previous value of Q

CLK=1: master can't sample input and holds value of D

Slave opens and QM=(D) =Q

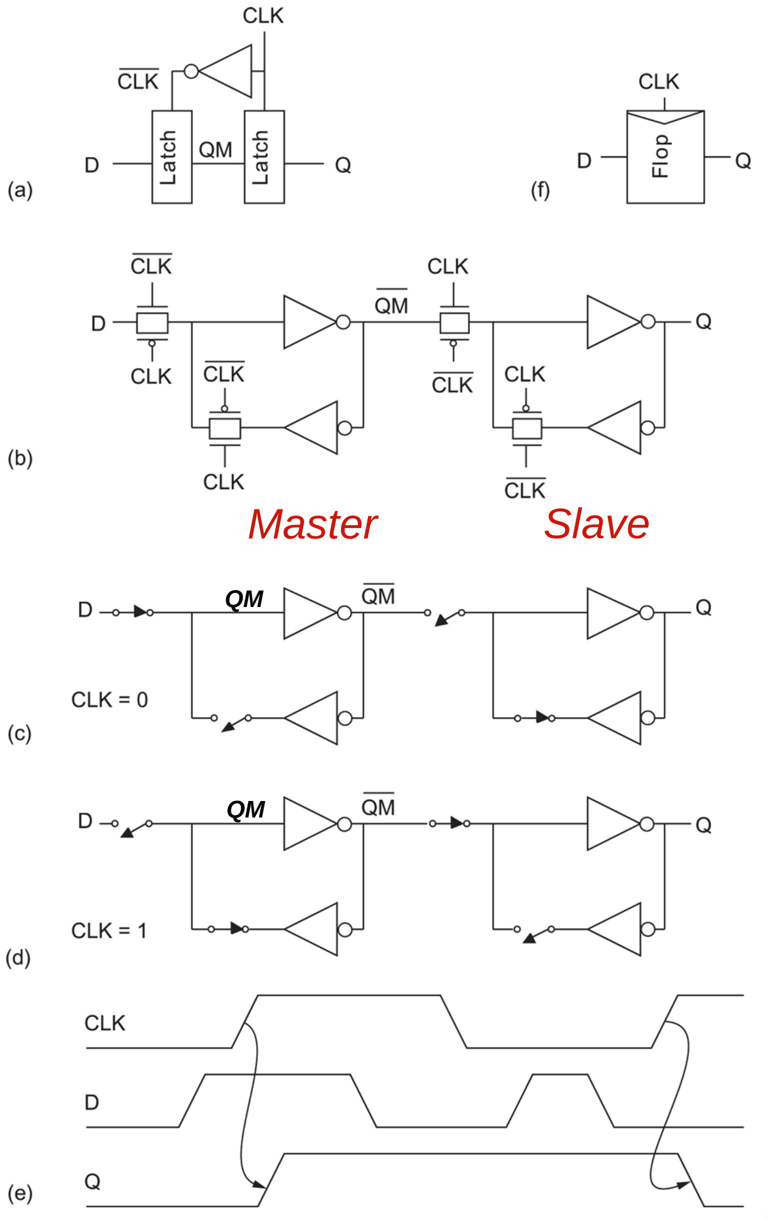
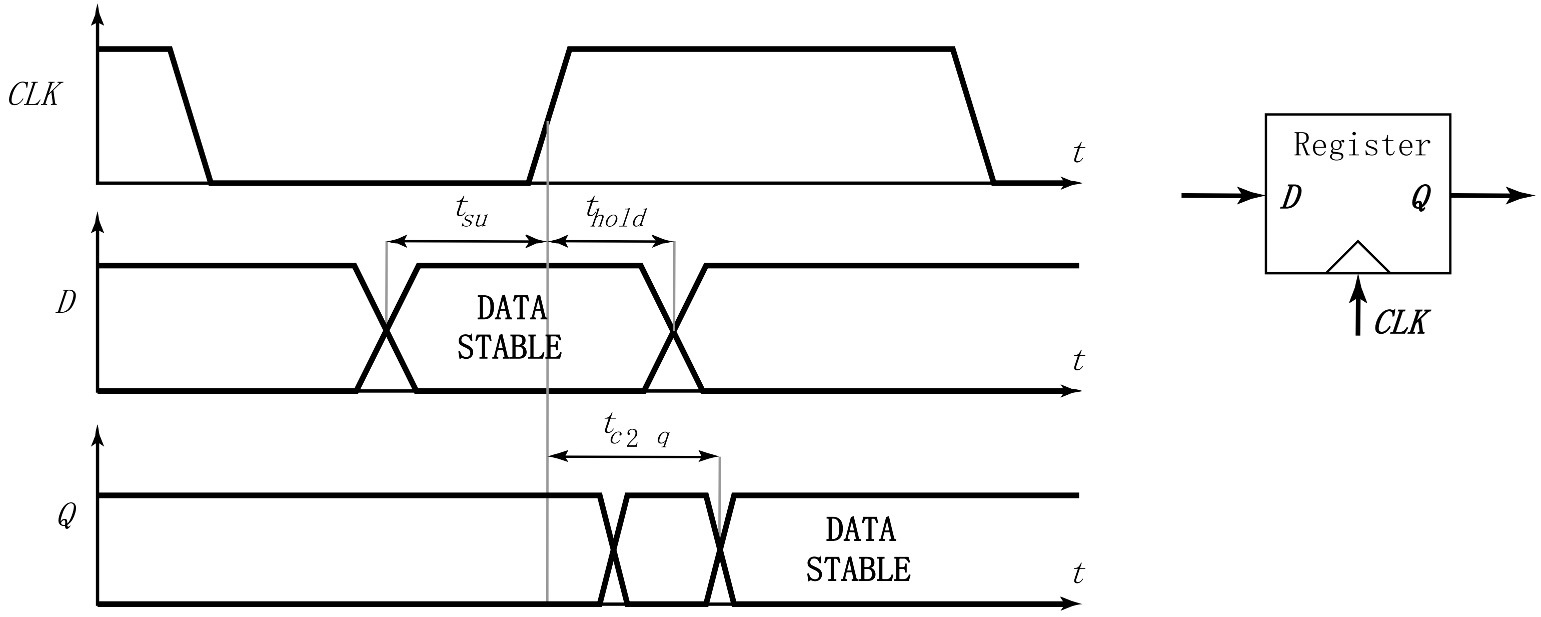


FIG 1.31 CMOS positive-edge-triggered D flip-flop

Timing Definitions



t_{su}= setup time =time for which the data inputs (D) must be valid before the CLK edge

t_{hold}= hold time =time for which data input must remain valid after the CLK edge

t_{c2q}= worst case propagation time through the Register (w.r.t the CLK edge)