COMP3222/9222 Digital Circuits & Systems

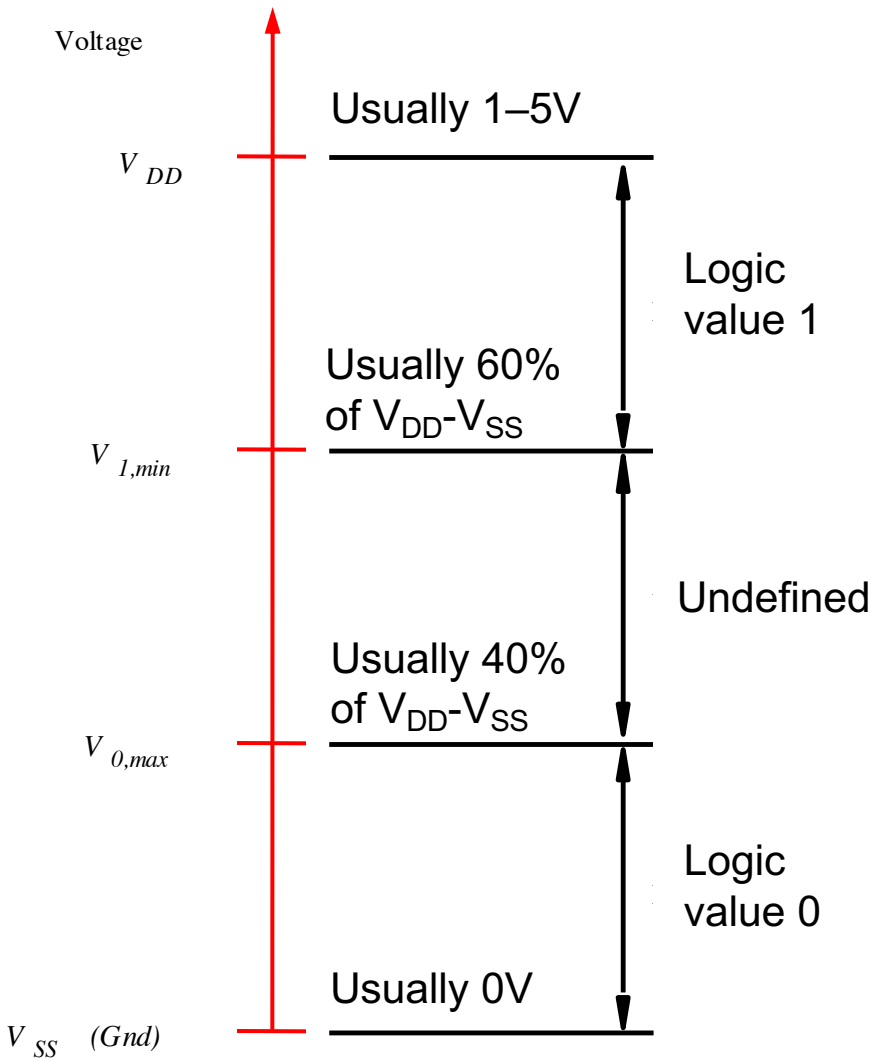
10. Implementation Technology

Objectives

- Learn about integrated circuit implementation technologies
- Understand the structure and operation of CMOS logic gates
- Know about the variety of digital implementation technologies and their application
- Review the structure of field-programmable gate arrays and other programmable logic devices

Logic values as voltage levels

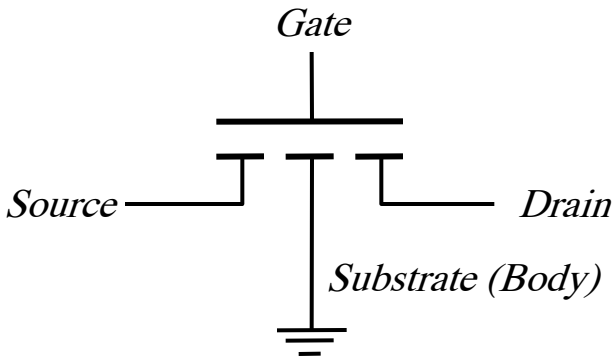
- Logic variables are physically represented using voltage or current levels
  - Use of voltage levels more common
- Usually, logic 0 is represented by a low voltage level and logic 1 by a high V level
  - Known as a *positive logic* system



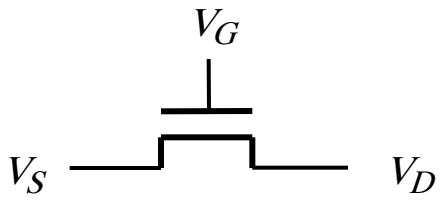
B&V3, Figure 3.1

nMOS transistor as a switch

(a) A simple switch controlled by the input *x*



(b) nMOS transistor



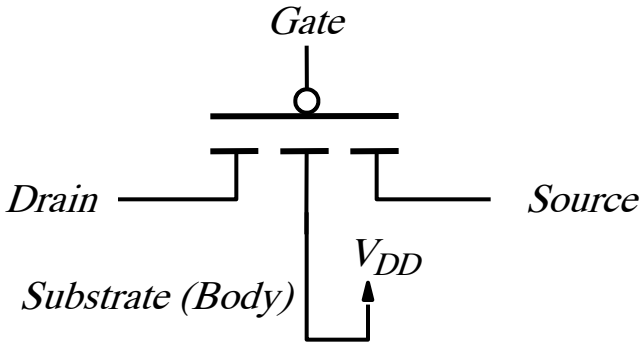
(c) Simplified symbol for an nMOS transistor

B&V3, Figure 3.2

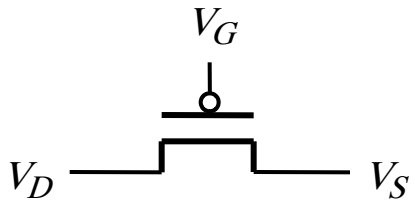
- Logic circuits are built with transistors operated in *cutoff* or *saturation* modes
- The most popular type of transistor for implementing a switch is the *metal oxide semiconductor field-effect transistor (MOSFET)*
  - Two types: n-channel or nMOS and p-channel or pMOS
- In nMOS, the terminal with the lower voltage level is deemed to be the source (of electrons)
  - When V_G is below a so-called threshold voltage, v_t, there is no connection between the source and drain – the device is turned off
  - When $V_G > v_t$, the device is switched on and $V_D \cong V_S$

pMOS transistor as a switch

(a) A switch with the opposite behavior of Figure 3.2 *a*



(b) pMOS transistor



(c) Simplified symbol for a pMOS transistor

- pMOS transistors have the opposite behaviour of nMOS transistors
  - The substrate is connected to V_{DD} rather than ground
  - The source terminal has the higher voltage (sources holes, here)

B&V3, Figure 3.3

nMOS operation – cutoff → subthreshold ($v_{GS} < v_t$)

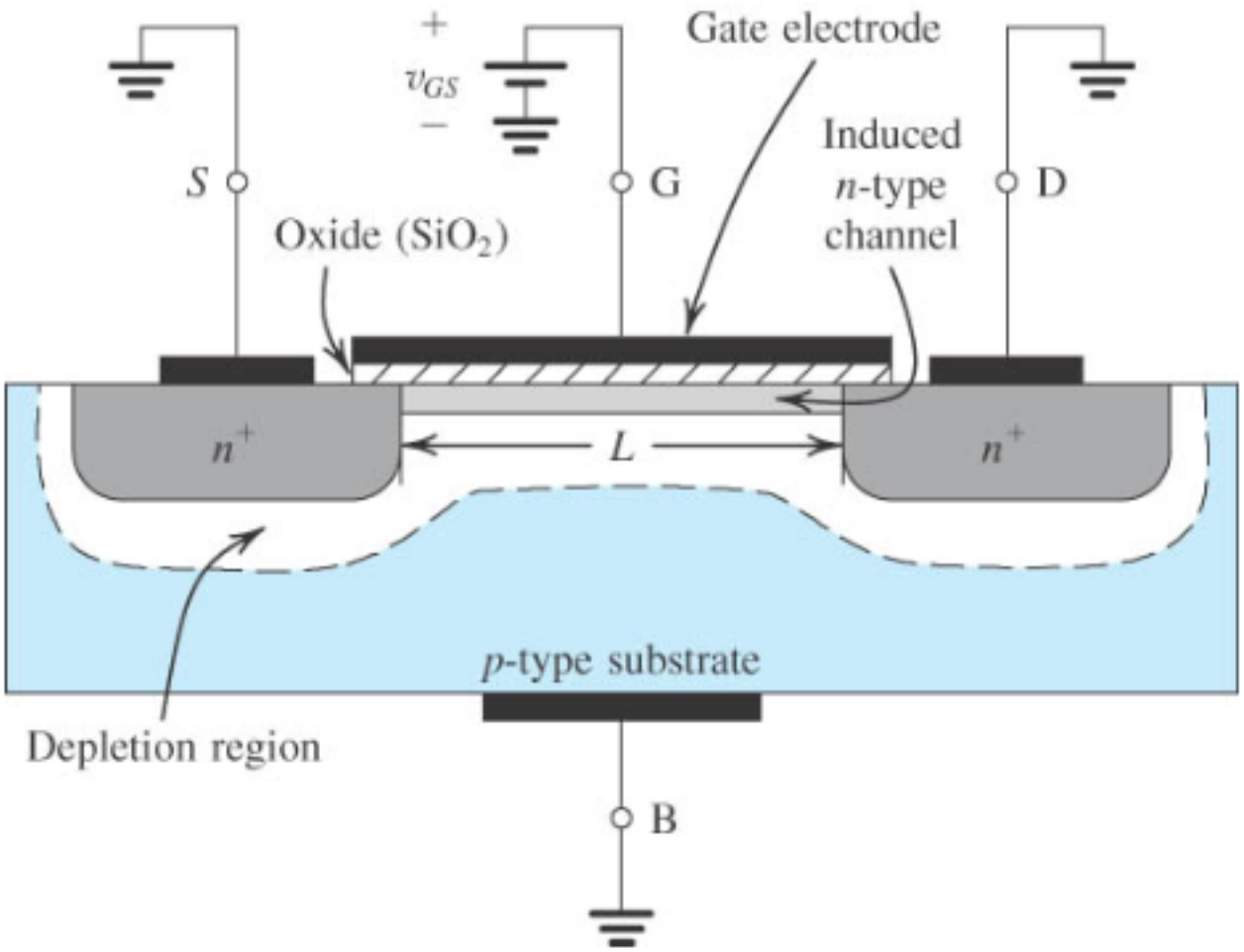


Image due to: F. Najmabadi, UCSD
See 3222 website for his lecture notes

nMOS operation – triode mode

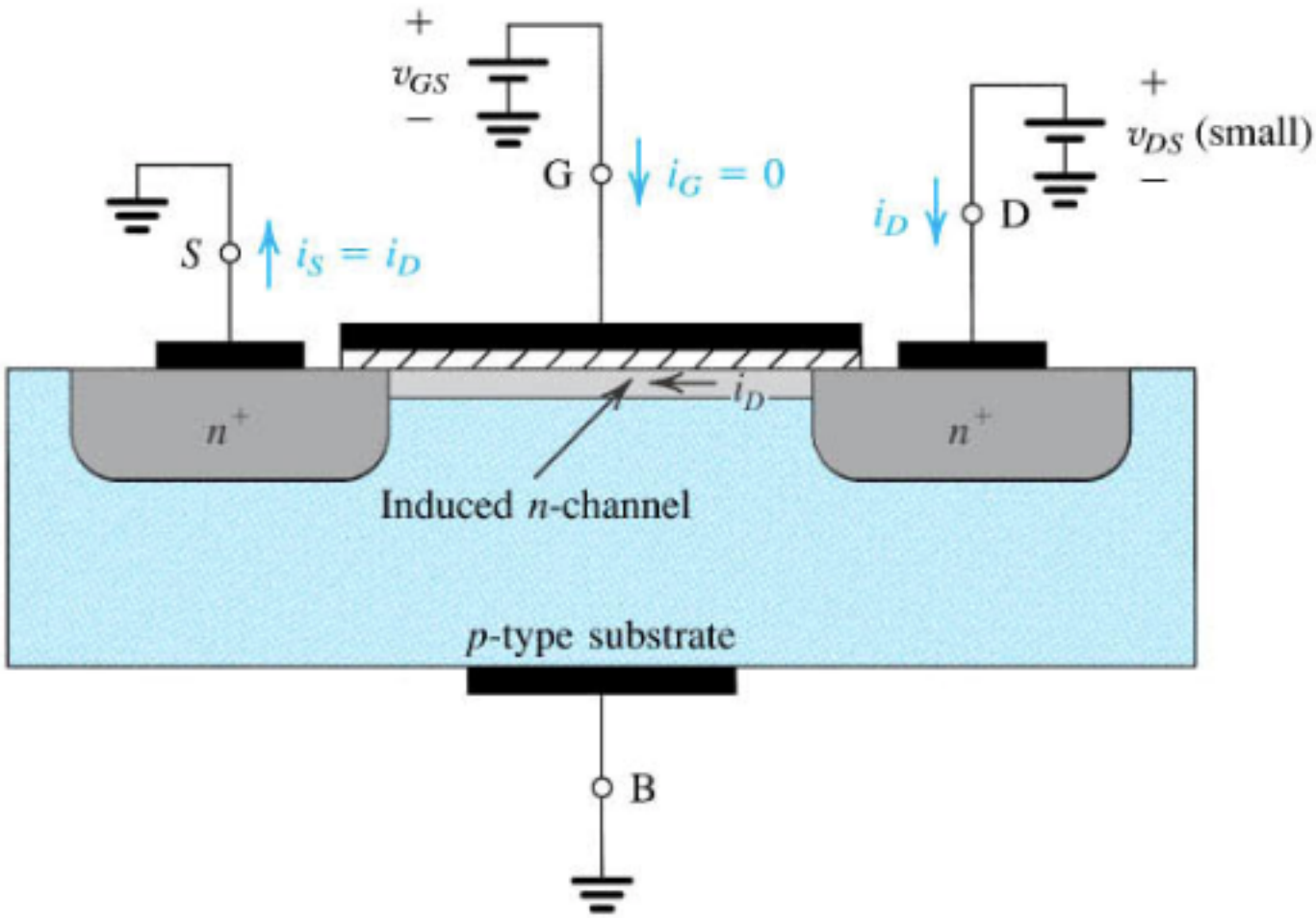


Image due to: F. Najmabadi, UCSD
See 3222 website for his lecture notes

nMOS operation – pinchoff

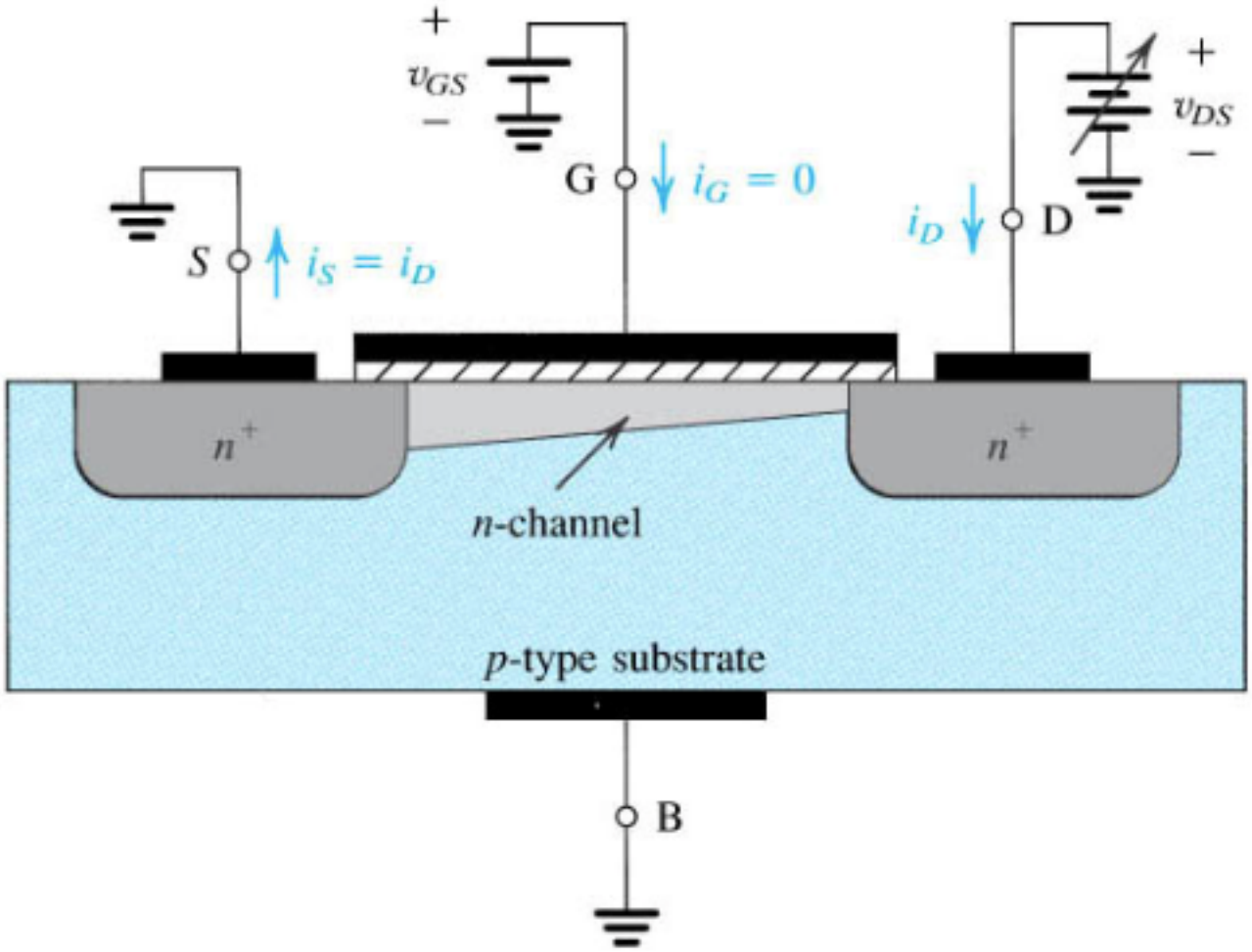


Image due to: F. Najmabadi, UCSD
See 3222 website for his lecture notes

nMOS operation – saturation

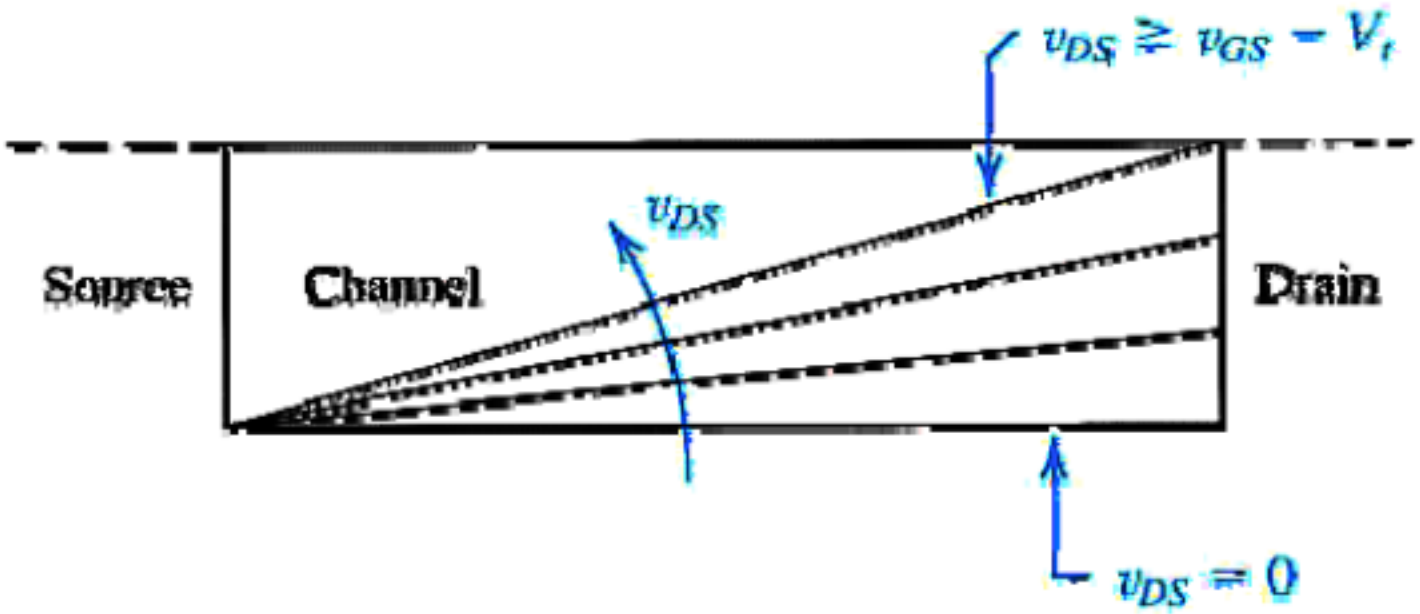


Image due to: F. Najmabadi, UCSD
See 3222 website for his lecture notes

nMOS operation – V-I characteristic

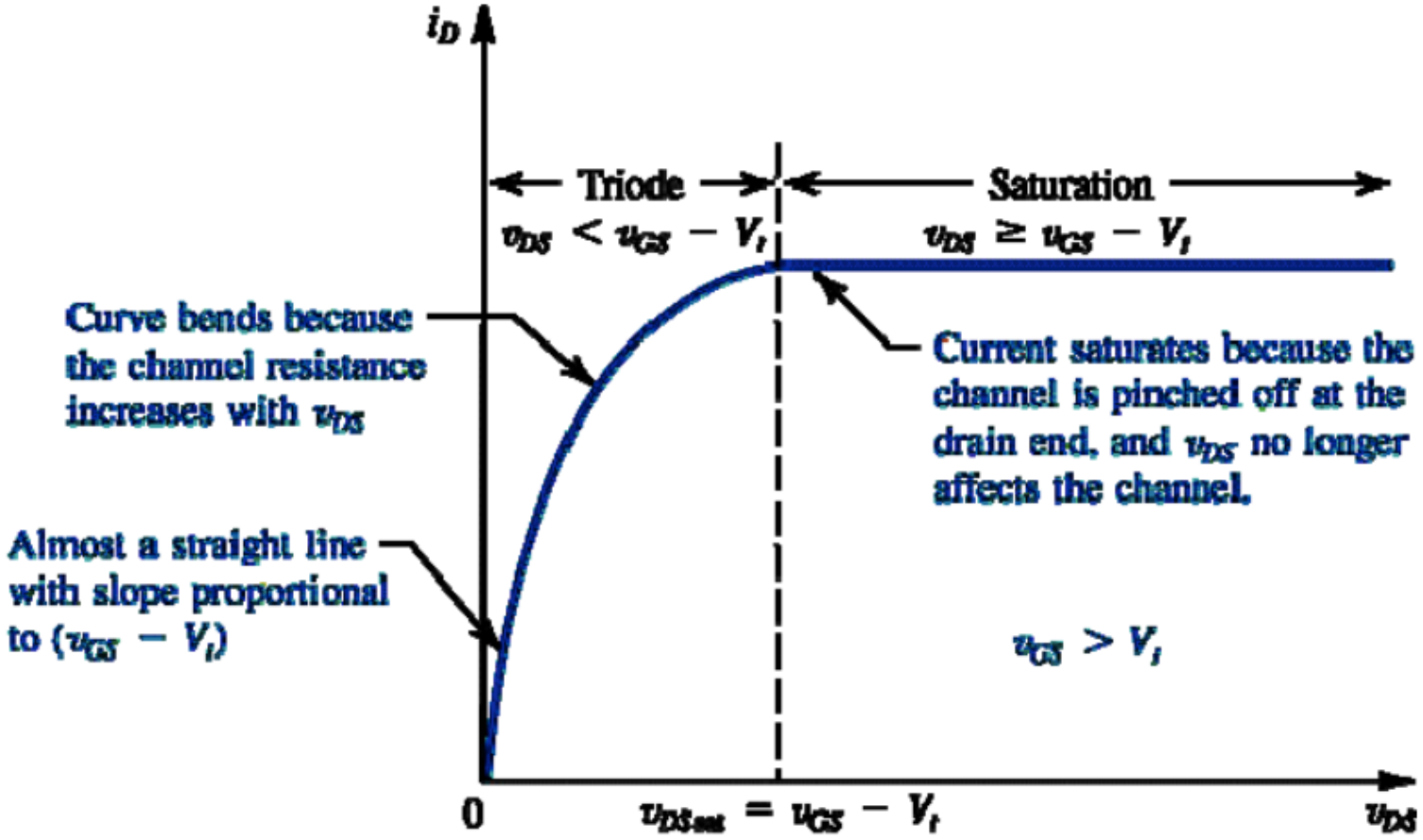
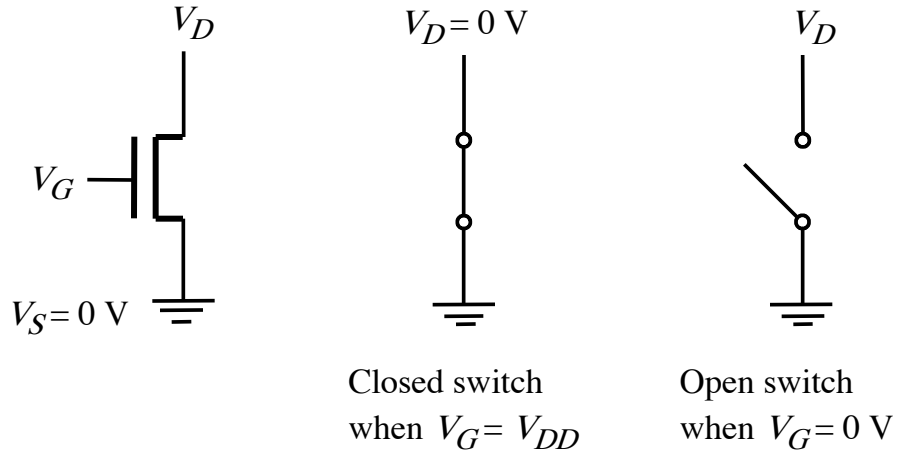


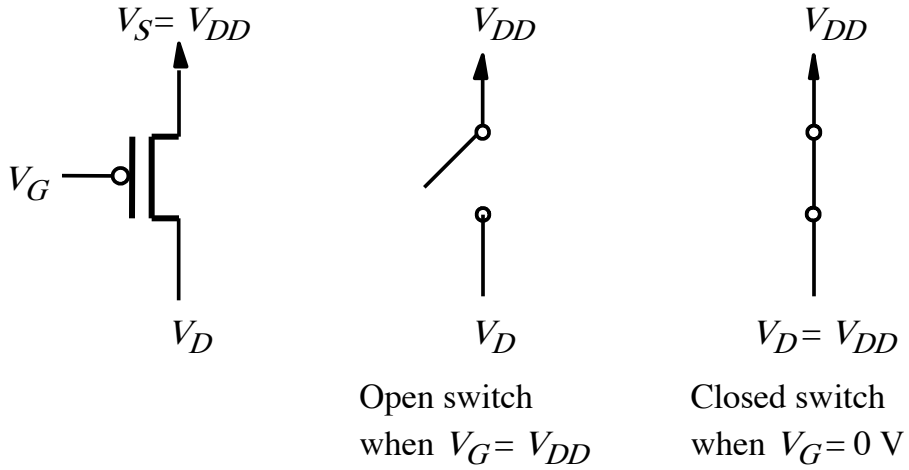
Image due to: F. Najmabadi, UCSD
See 3222 website for his lecture notes

nMOS and pMOS transistors in logic circuits



(a) nMOS transistor

When switched on, the nMOS drain is pulled down to ground

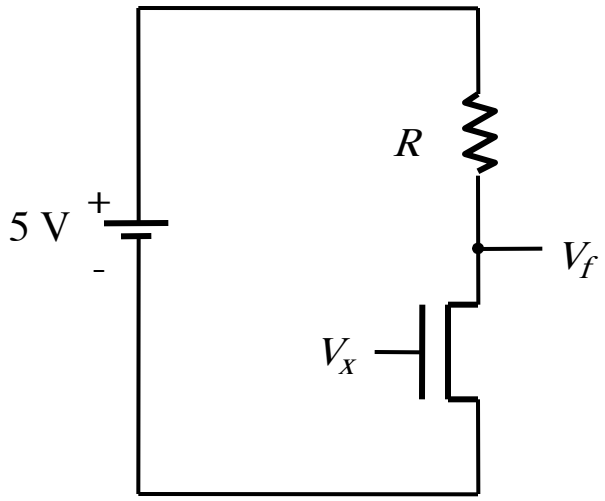


(b) pMOS transistor

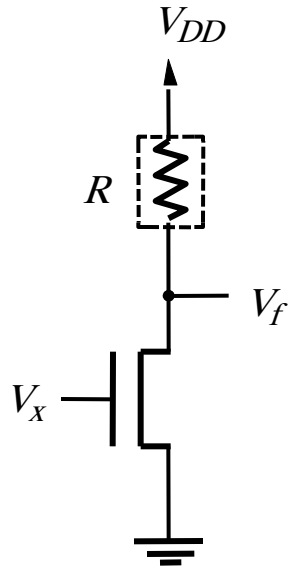
When switched on, the pMOS drain is pulled up to V_{DD}

B&V3, Figure 3.4

A NOT gate built using nMOS technology

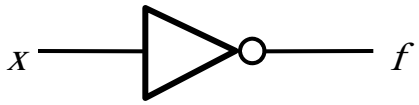


(a) Circuit diagram



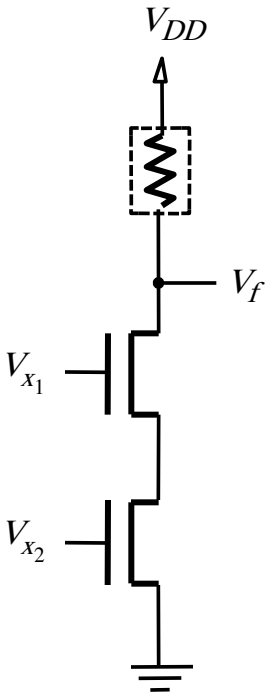
(b) Simplified circuit diagram

The resistor limits current flow through the transistor



(c) Graphical symbol

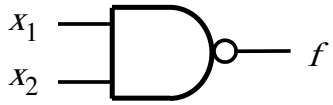
nMOS realization of a NAND gate



(a) Circuit

x_1	x_2	f
0	0	1
0	1	1
1	0	1
1	1	0

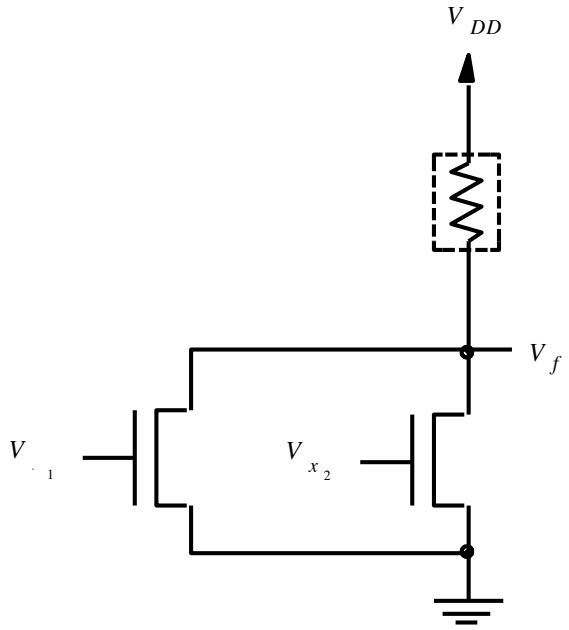
(b) Truth table



(c) Graphical symbol

B&V3, Figure 3.6

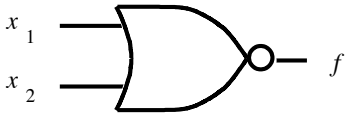
nMOS realization of a NOR gate



(a) Circuit

x_1	x_2	f
0	0	1
0	1	0
1	0	0
1	1	0

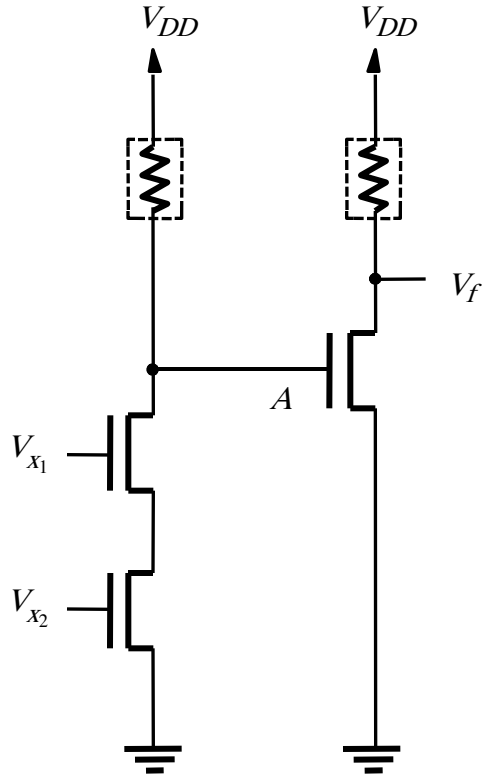
(b) Truth table



(c) Graphical symbol

B&V3, Figure 3.7

nMOS realization of an AND gate

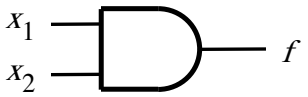


(a) Circuit

Constructed from a NAND gate followed by an INVERTER

x_1	x_2	f
0	0	0
0	1	0
1	0	0
1	1	1

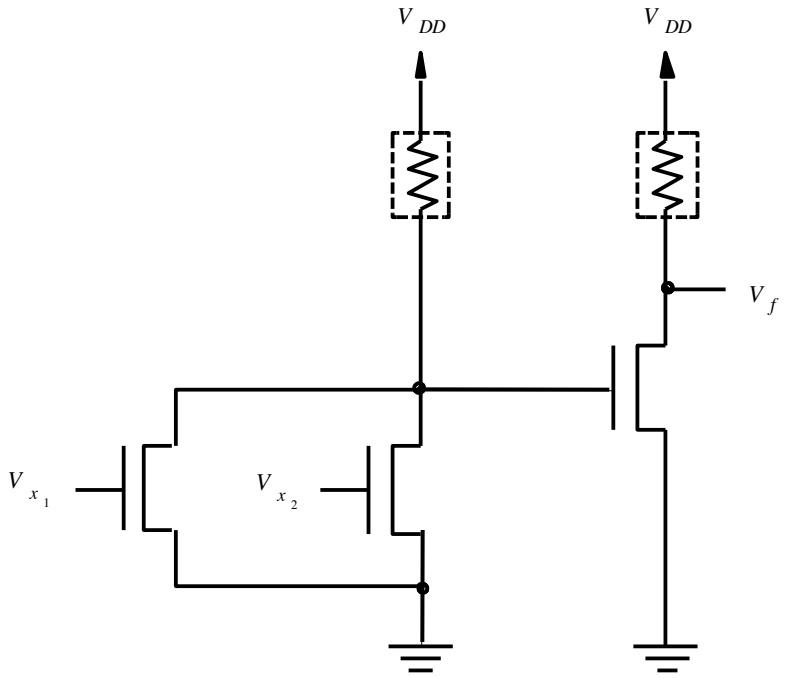
(b) Truth table



(c) Graphical symbol

B&V3, Figure 3.8

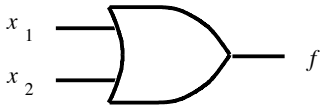
nMOS realization of an OR gate



(a) Circuit

x_1	x_2	f
0	0	0
0	1	1
1	0	1
1	1	1

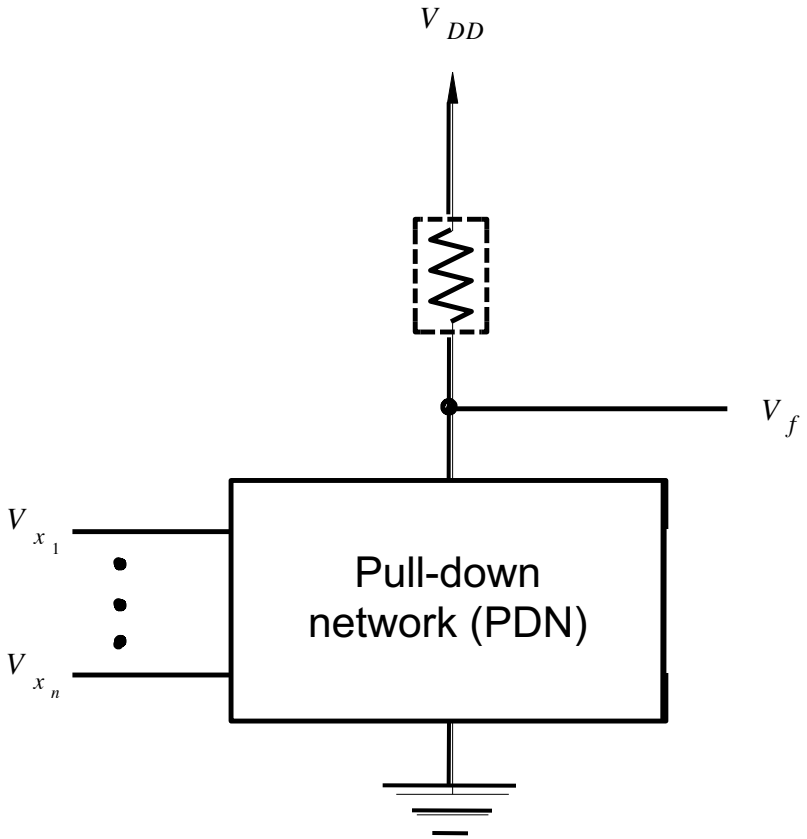
(b) Truth table



(c) Graphical symbol

B&V3, Figure 3.9

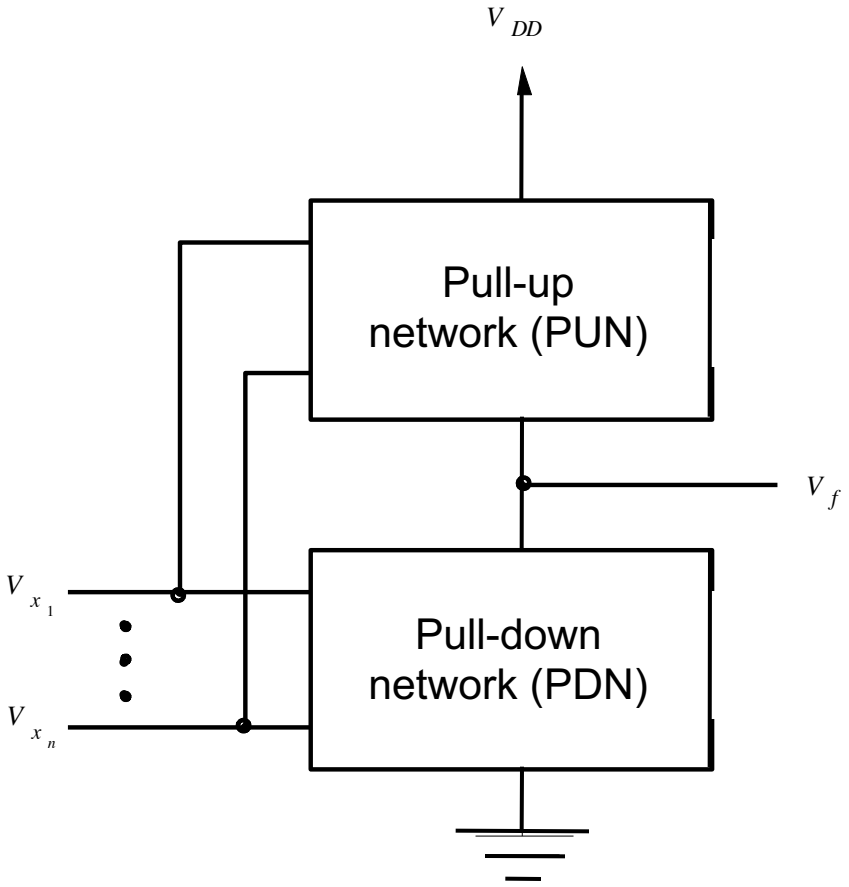
Structure of an nMOS circuit



B&V3, Figure 3.10

- Generalizing the forgoing structures, note that the nMOS transistor network acts to pull the drain to ground
- We can derive an equivalent pMOS circuit for each of the previous circuits
- We can also combine both nMOS and pMOS networks together in what is known as *complementary MOS (CMOS)* technology
  - Involves replacing the pull-up resistor with a pull-up network that complements the structure of the nMOS-based pull-down network
  - Eliminates the power dissipation in the current-limiting resistor & simplifies fabrication

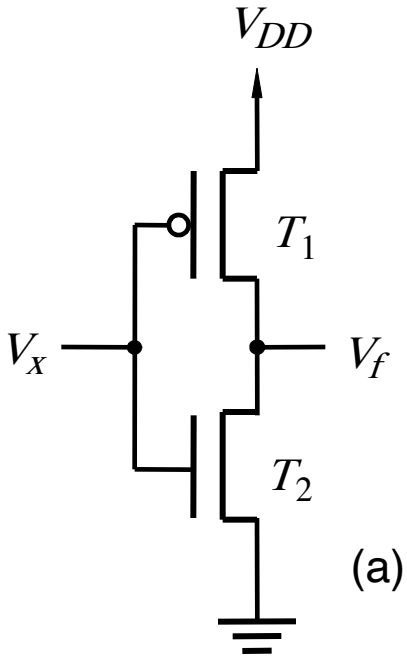
Structure of a CMOS circuit



- The functions realized by the PUN and PDN are complements of each other
  - For any valuation of the inputs, either the PDN pulls V_f down to ground or the PUN pulls V_f up to V_{DD}
- The PUN & PDN have equal numbers of transistors and are arranged such that the networks are duals of each other
  - Wherever the PDN has nMOS transistors in series, the PUN has pMOS transistors in parallel and vice versa

B&V3, Figure 3.11

CMOS realization of a NOT gate



(a) Circuit

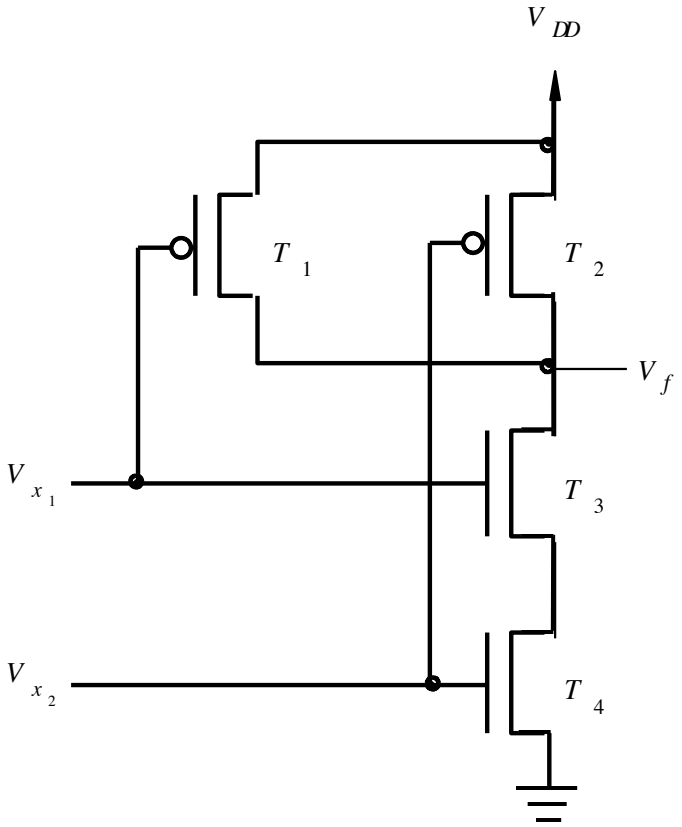
x	T_1	T_2	f
0	on	off	1
1	off	on	0

(b) Truth table and transistor states

B&V3, Figure 3.12

- Note that ALMOST no current flows in a CMOS inverter when the input is either low or high.
- This is true for all CMOS circuits; ALMOST no current flows, and hence ALMOST no power is dissipated under steady state conditions.
- However, such *leakage current* (through the gate) mounts as the transistor feature sizes and threshold voltages (gate voltage at which current just begins to flow) decrease
- So called *dynamic power* or *switching power* is transitionally dissipated when the gate changes state – the amount dissipated is *proportional to the clock frequency and the square of the supply voltage V_{DD}*

CMOS realization of a NAND gate

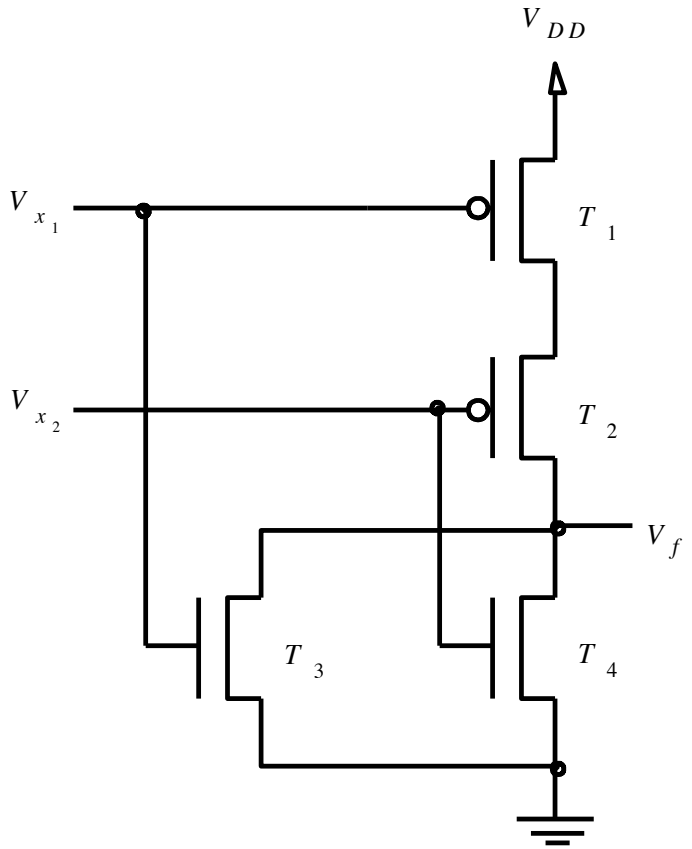


x_1	x_2	T_1	T_2	T_3	T_4	f
0	0	on	on	off	off	1
0	1	on	off	off	on	1
1	0	off	on	on	off	1
1	1	off	off	on	on	0

- The circuit can be derived from the NAND function $f = \overline{x_1 x_2}$
- This expression specifies the conditions for which $f = 1$; hence it defines the PUN:
- From DeMorgan, $f = \overline{x_1 x_2} = \overline{x}_1 + \overline{x}_2$
  - Thus $f = 1$ when either x_1 or x_2 have the value 0, which means the PUN must have two pMOS transistors connected in parallel
- The PDN must implement the complement of f which is $\overline{f} = x_1 x_2$
  - Since $\overline{f} = 1$ when both x_1 and x_2 are 1, the PDN must have two nMOS transistors in series

B&V3, Figure 3.13

CMOS realization of a NOR gate



(a) Circuit

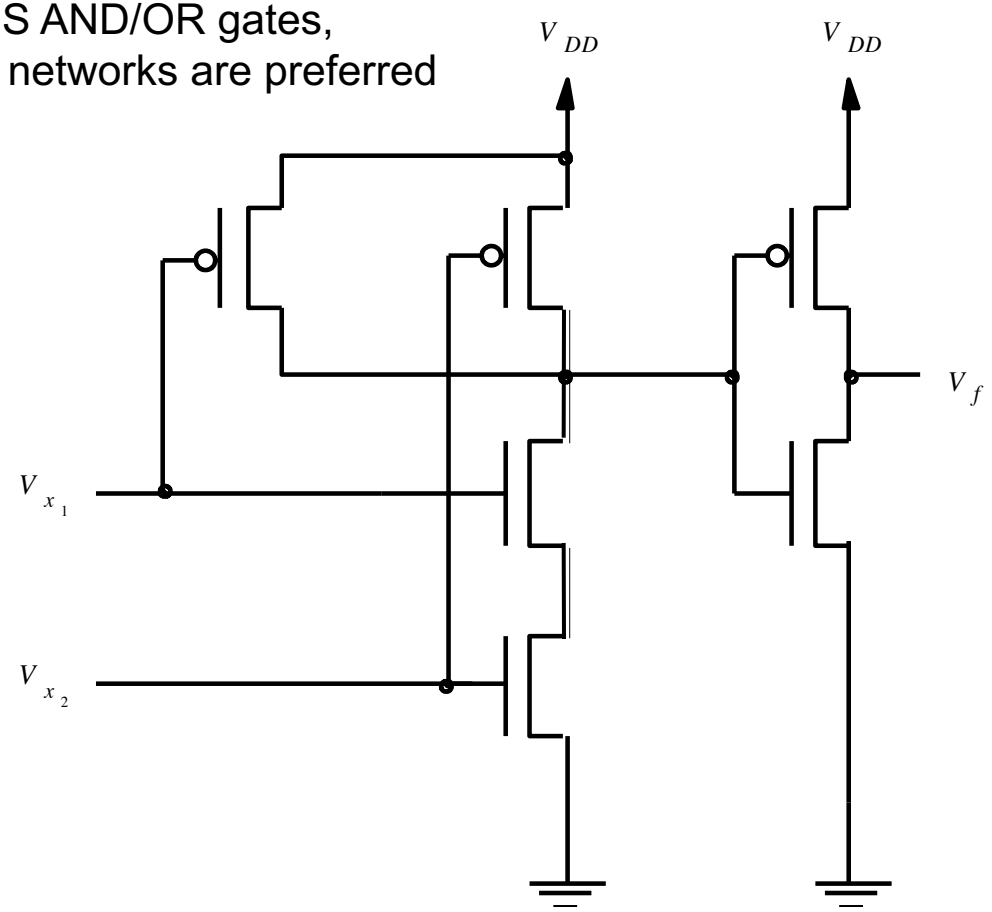
x_1	x_2	T_1	T_2	T_3	T_4	f
0	0	on	on	off	off	1
0	1	on	off	off	on	0
1	0	off	on	on	off	0
1	1	off	off	on	on	0

(b) Truth table and transistor states

B&V3, Figure 3.14

CMOS realization of an AND gate

Since two more transistors are needed
to construct CMOS AND/OR gates,
NAND/NOR logic networks are preferred

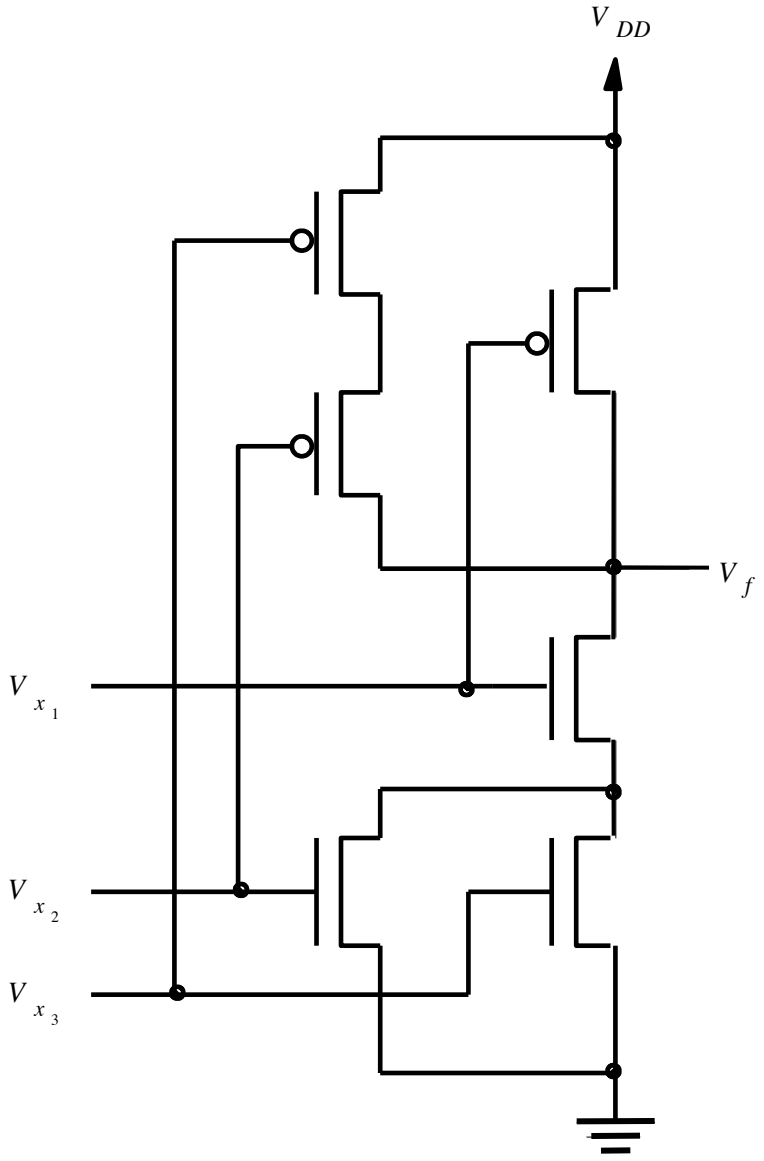


B&V3, Figure 3.15

The circuit for $f = \overline{x}_1 + \overline{x}_2\overline{x}_3$

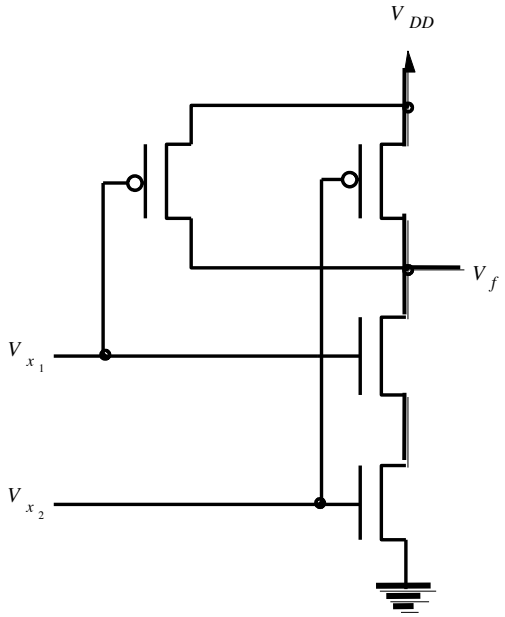
- The PUN is readily derived since all variables appear in complemented form
- Use DeMorgan to derive the PDN:

$$\overline{f} = \overline{\overline{x}_1 + \overline{x}_2\overline{x}_3}$$
$$= x_1(x_2 + x_3)$$



B&V3, Figure 3.16

Note voltage levels for the NAND circuit…



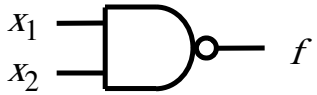
(a) Circuit

V_{x_1}	V_{x_2}	V_f
L	L	H
L	H	H
H	L	H
H	H	L

(b) Voltage levels

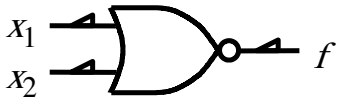
Obtained by assigning logic=1 to voltage=L & logic=0 to voltage=H

x_1	x_2	f
0	0	1
0	1	1
1	0	1
1	1	0



(a) Positive logic truth table and gate symbol

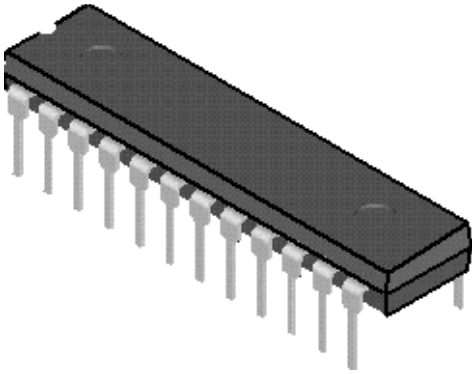
x_1	x_2	f
1	1	0
1	0	0
0	1	0
0	0	1



(b) Negative logic truth table and gate symbol

B&V3, Figure 3.18

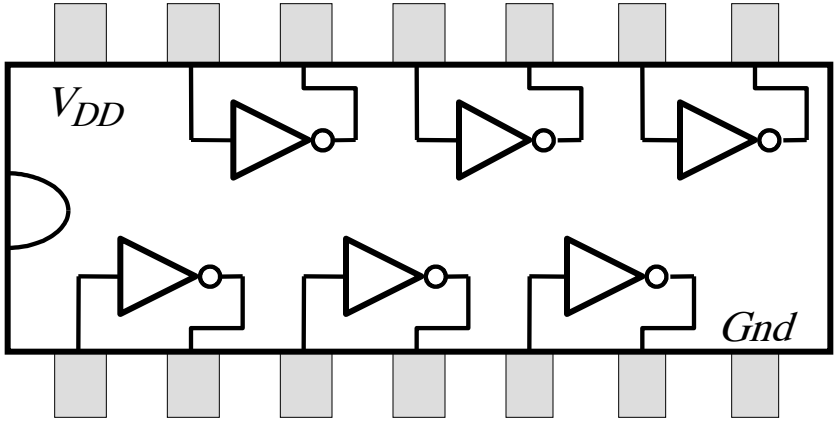
A 7400-series chip



(a) Dual-inline package

7400-series packages, a highly successful example of integrated circuit technology introduced in the 1960s, were widely used until the mid-1980s.

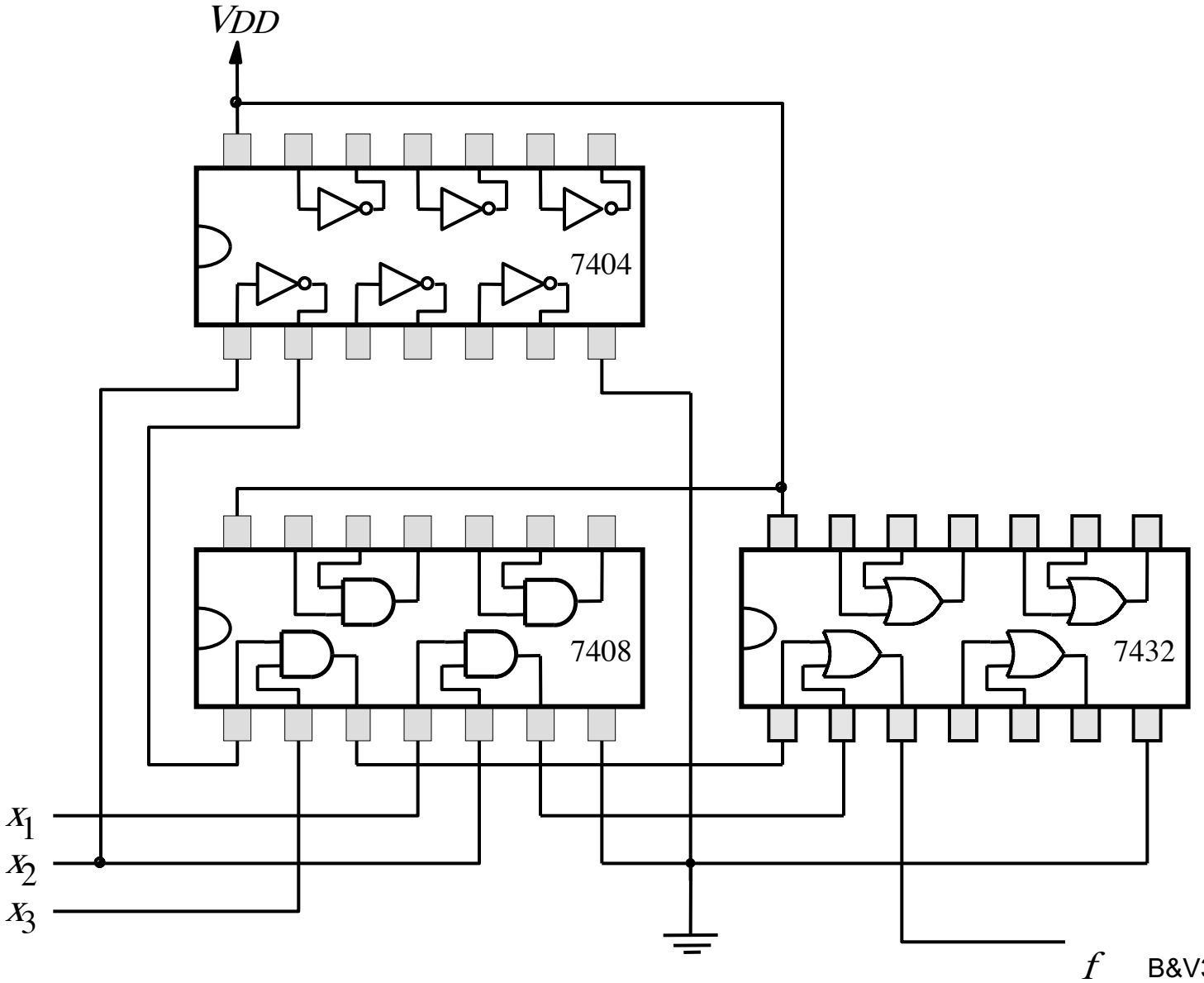
Because of their low logic capacity, these standard chips are seldom used today



(b) Structure of 7404 chip

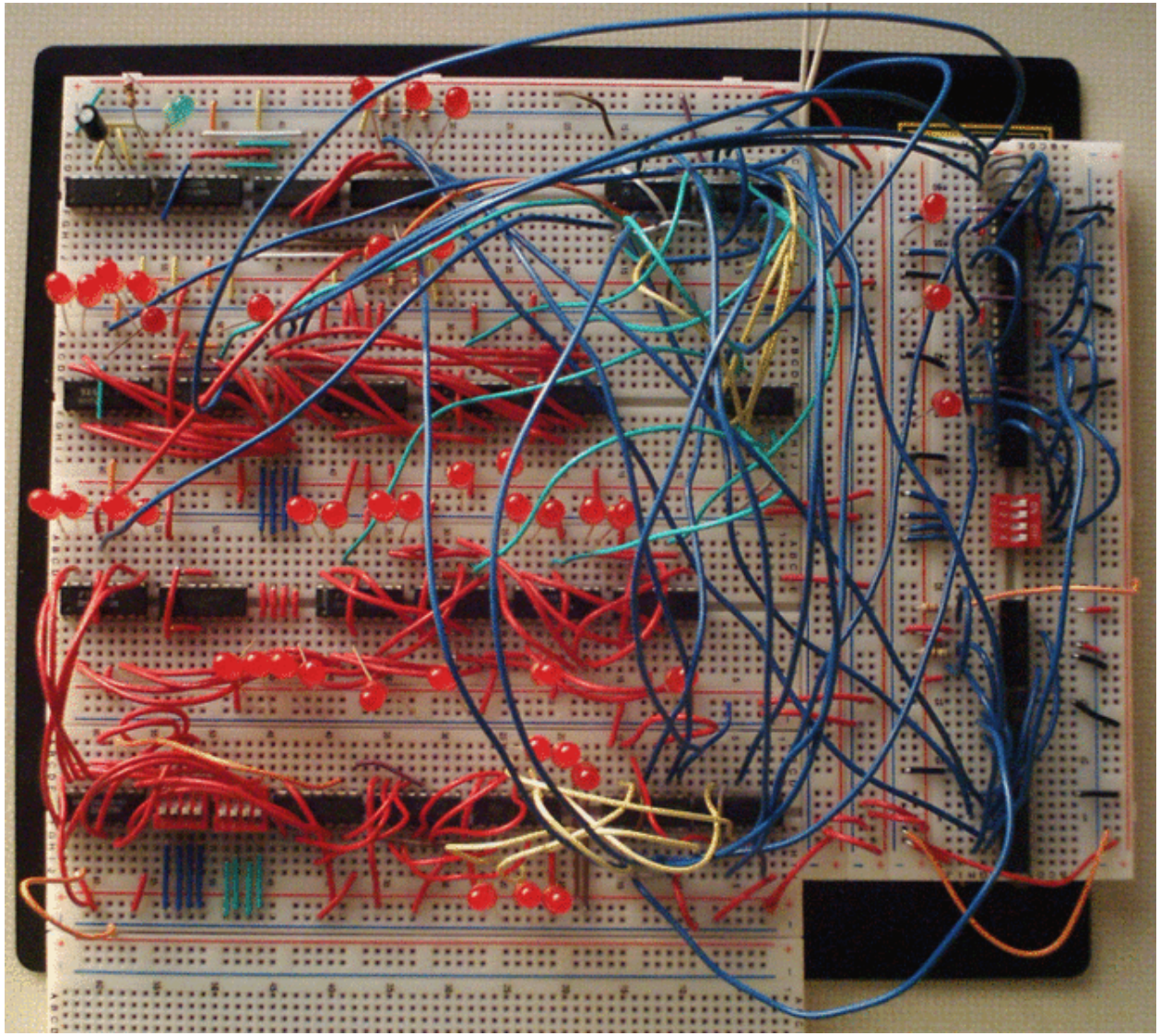
B&V3, Figure 3.21

An implementation of $f = x_1x_2 + \overline{x}_2x_3$



B&V3, Figure 3.22

A 4-bit, 2-register, 6-instruction 7400 series computer



Wikipedia

Evolution of integrated circuit technology

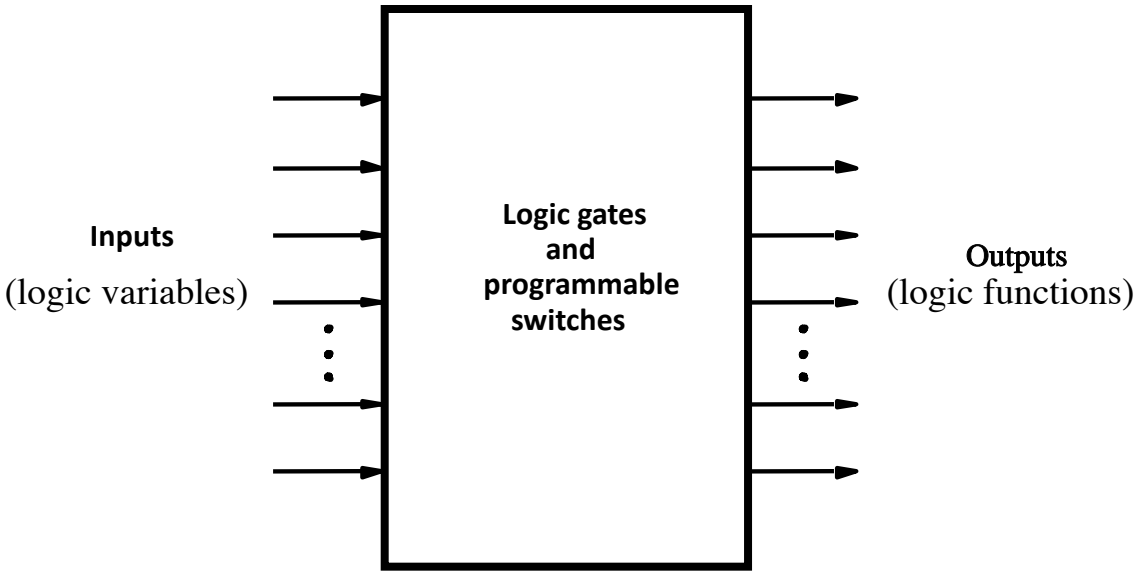
- Over time, chips have been classified according to their size
  - The earliest chips, such as 7400-series chips, comprise only a few logic gates – the technology used to produce these chips is referred to as *small-scale integration (SSI)*
  - Chips that included 10 to 100 gates represented *medium-scale integration (MSI)*
  - Until the mid-1980s, chips that were too large to qualify as MSI were classified as *large-scale integration (LSI)*
- More recently, this classification scheme has lost practical value, as most integrated circuits contain many 1,000,000s, if not billions, of transistors – these are all made with what is referred to as *very large scale integration (VLSI)* technology
  - These devices support the trend in digital hardware to integrate as much circuitry as possible onto a single chip
- This trend to ever greater levels of integration is driven by application needs, cost, performance, reliability and profit factors

The problem with fixed devices…

- The functions provided by 7400-series parts are fixed and cannot be tailored to suit particular designs

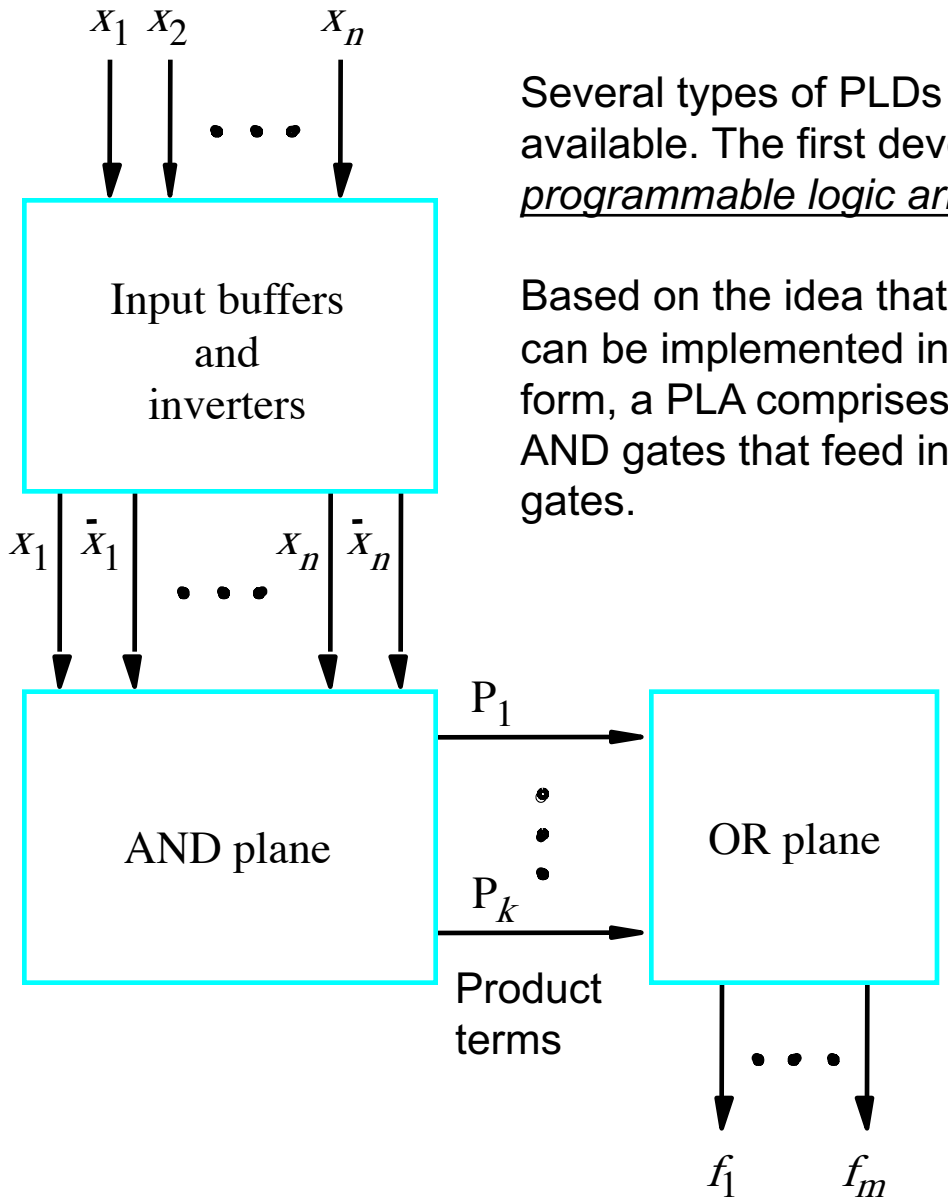
Programmable logic device as a black box

- It is possible to manufacture chips that contain relatively large amounts of logic circuitry with a structure that is **not fixed**
  - Such devices were introduced in the 1970s and are called *programmable logic devices (PLDs)*
  - PLDs are general-purpose chips for implementing logic circuitry
  - They can be viewed as a black box containing logic gates and programmable switches to allow the gates to be interconnected as desired



B&V3, Figure 3.24

Programmable Logic Array (PLA)

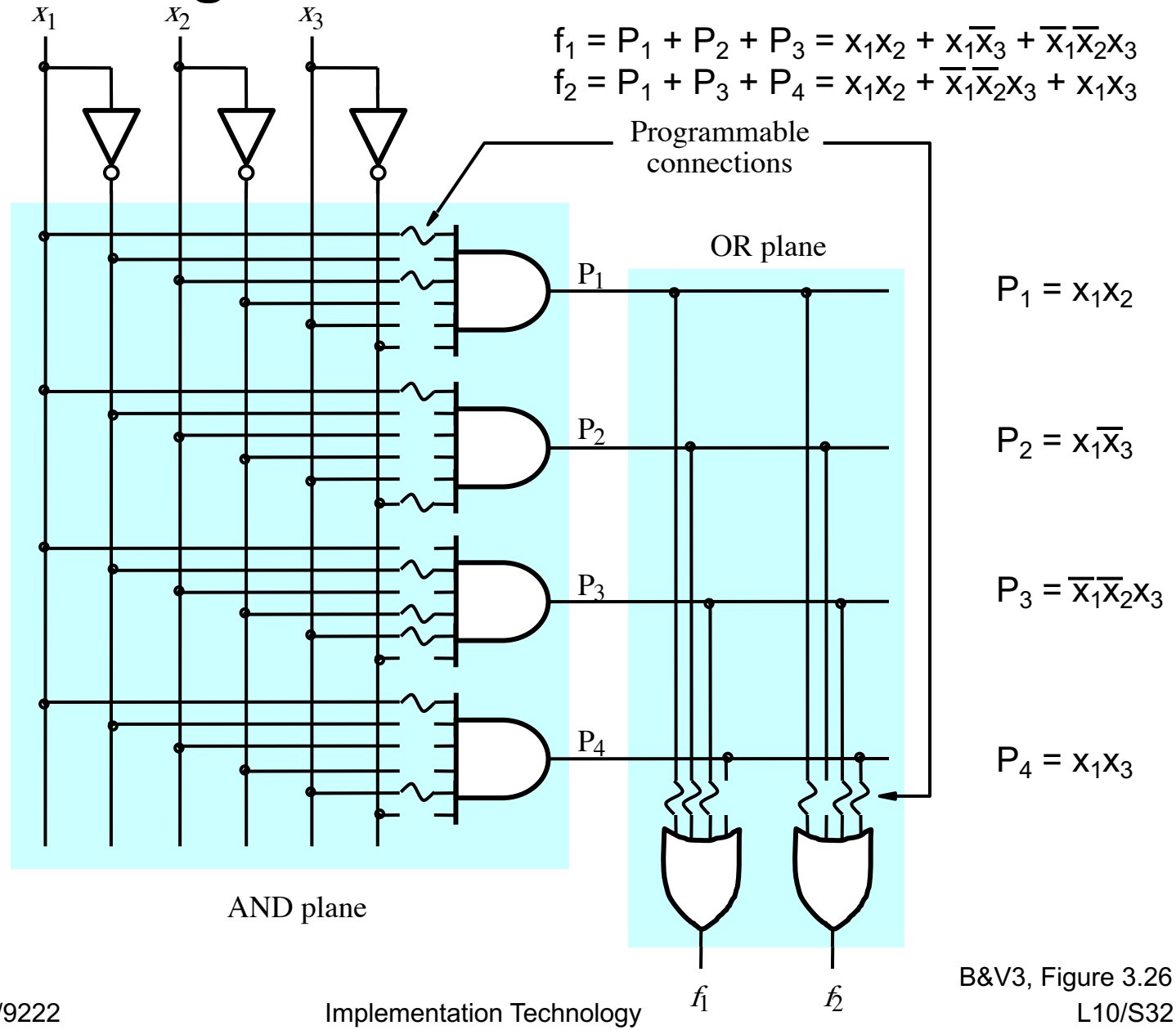


Several types of PLDs are commercially available. The first developed was the *programmable logic array (PLA)*.

Based on the idea that logic functions can be implemented in sum-of-products form, a PLA comprises a collection of AND gates that feed into a set of OR gates.

B&V3, Figure 3.25

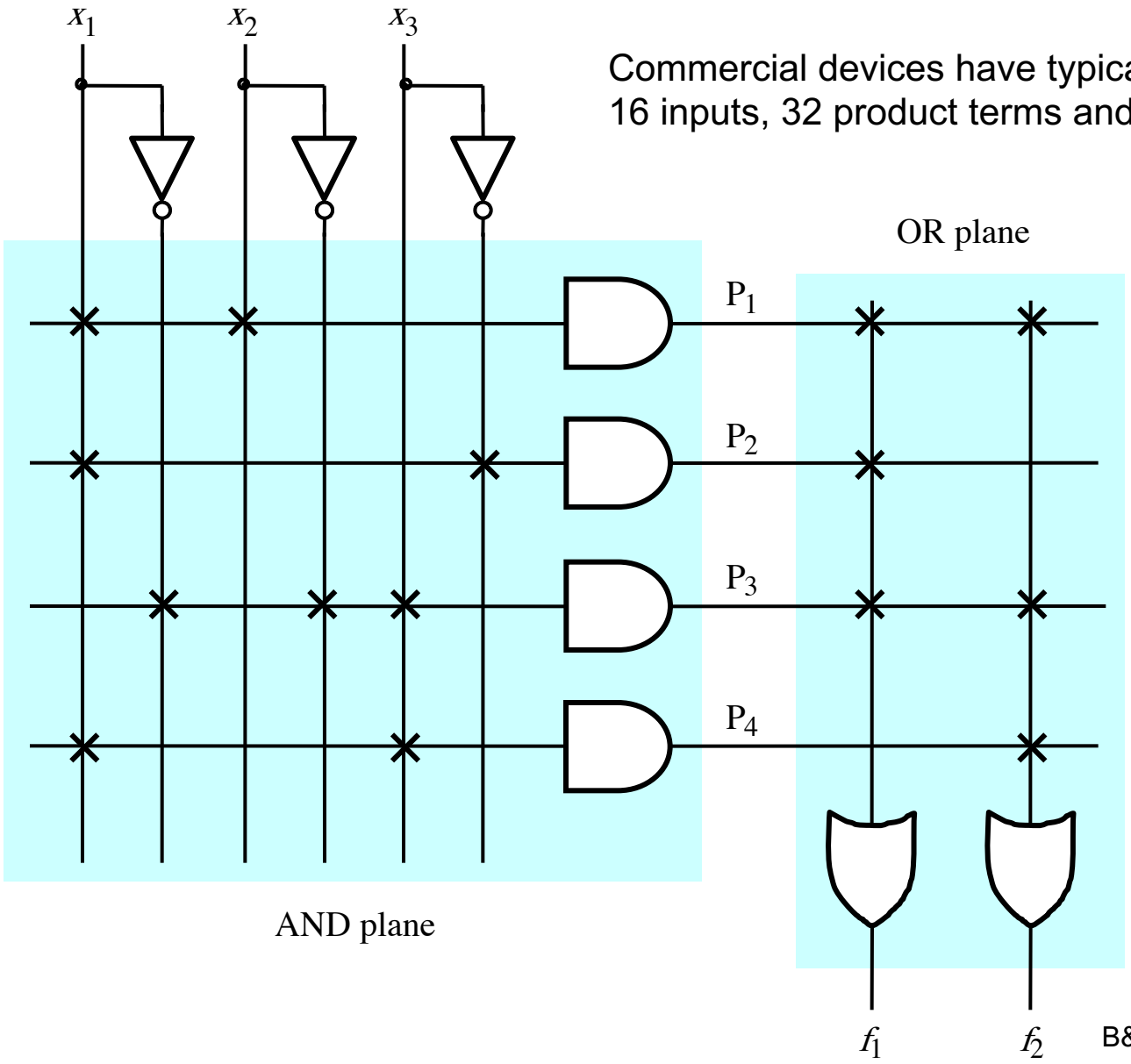
Gate-level diagram of a PLA



B&V3, Figure 3.26

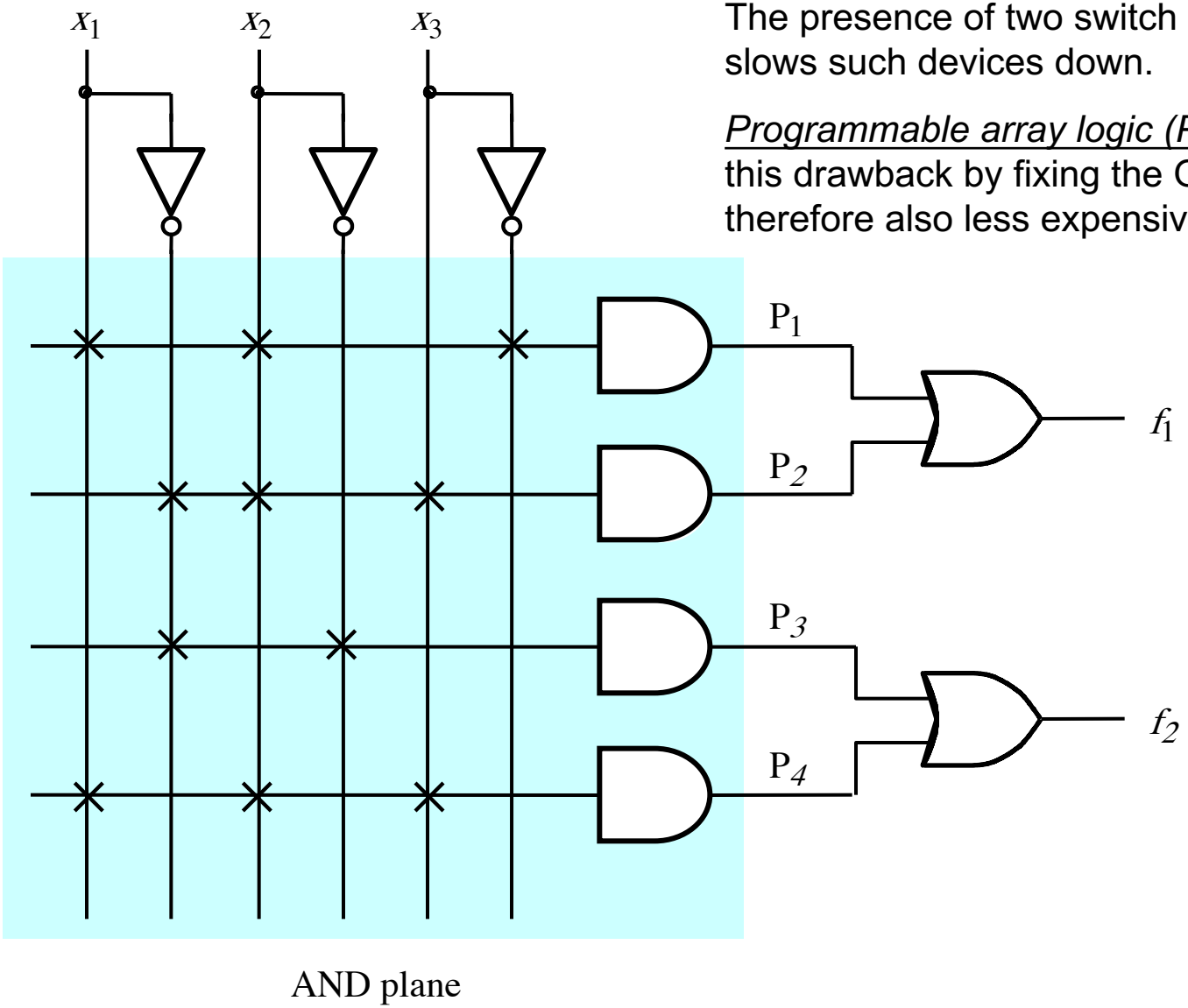
More conventional schematic of the previous PLA

Commercial devices have typical sizes of 16 inputs, 32 product terms and 8 outputs.



B&V3, Figure 3.27

Programmable Array Logic (PAL)

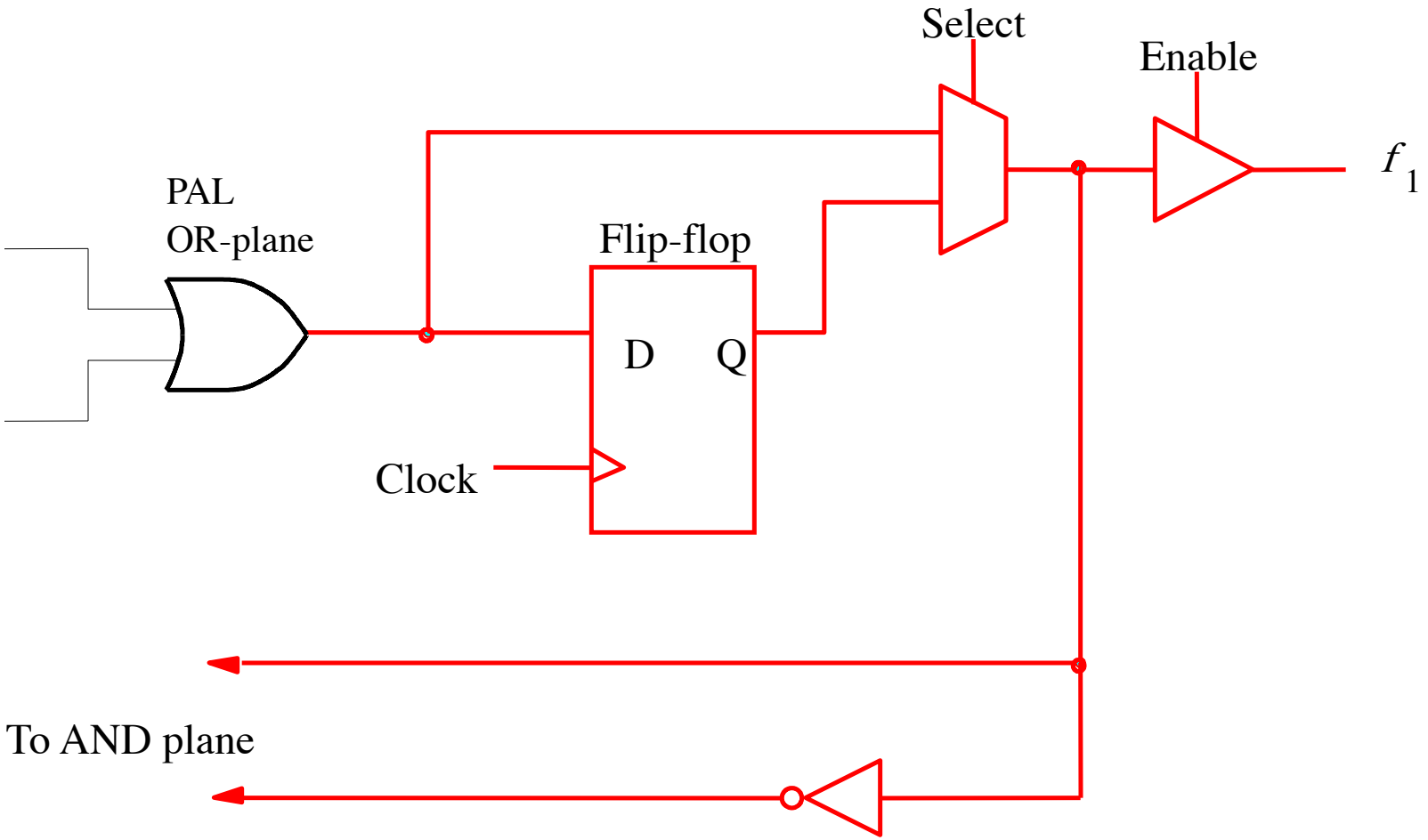


The presence of two switch planes in PLAs slows such devices down.

Programmable array logic (PAL) overcomes this drawback by fixing the OR plane. They are therefore also less expensive to manufacture

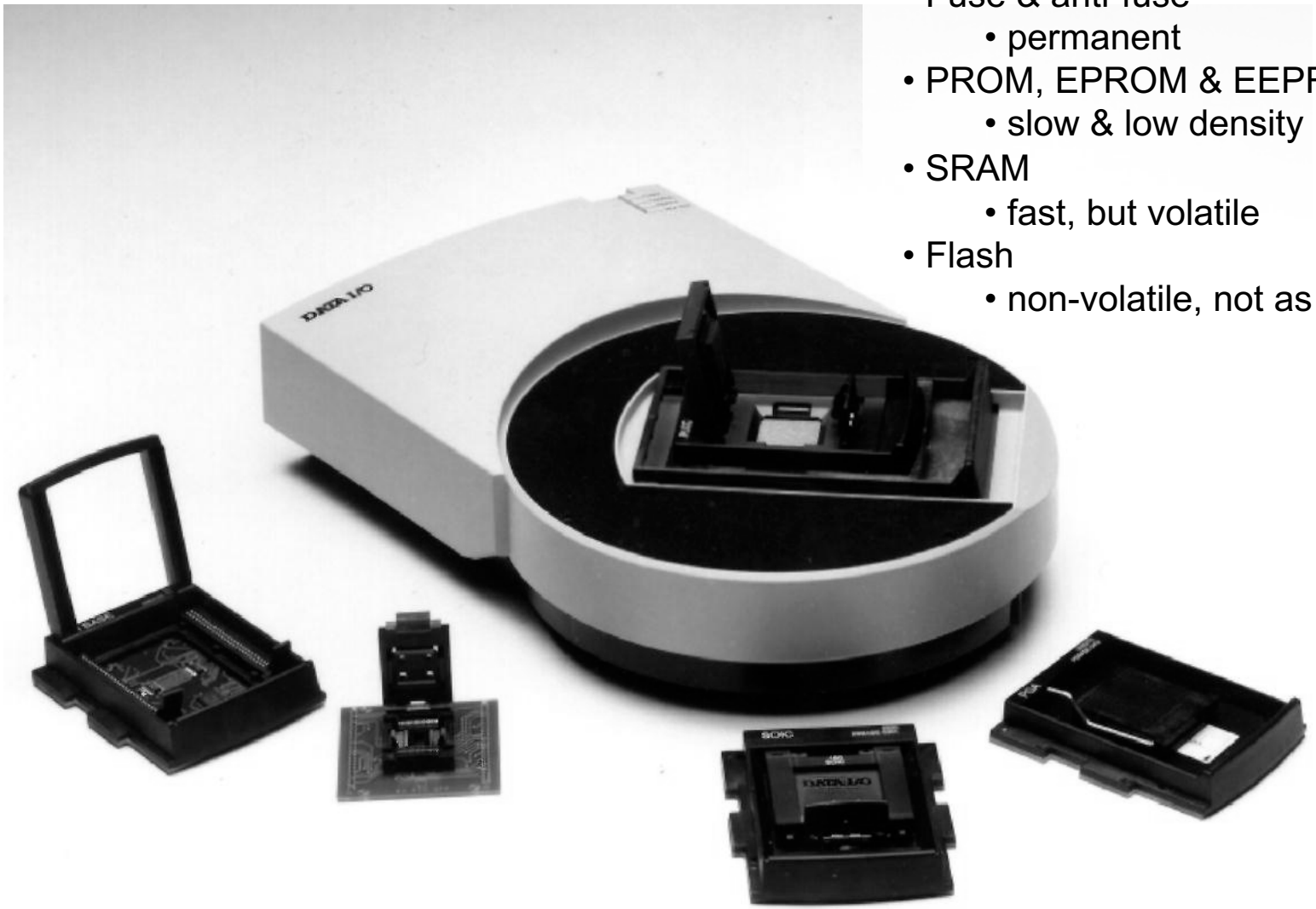
B&V3, Figure 3.28

Enhancing the flexibility of PAL outputs



B&V3, Figure 3.29

A PLD programming unit

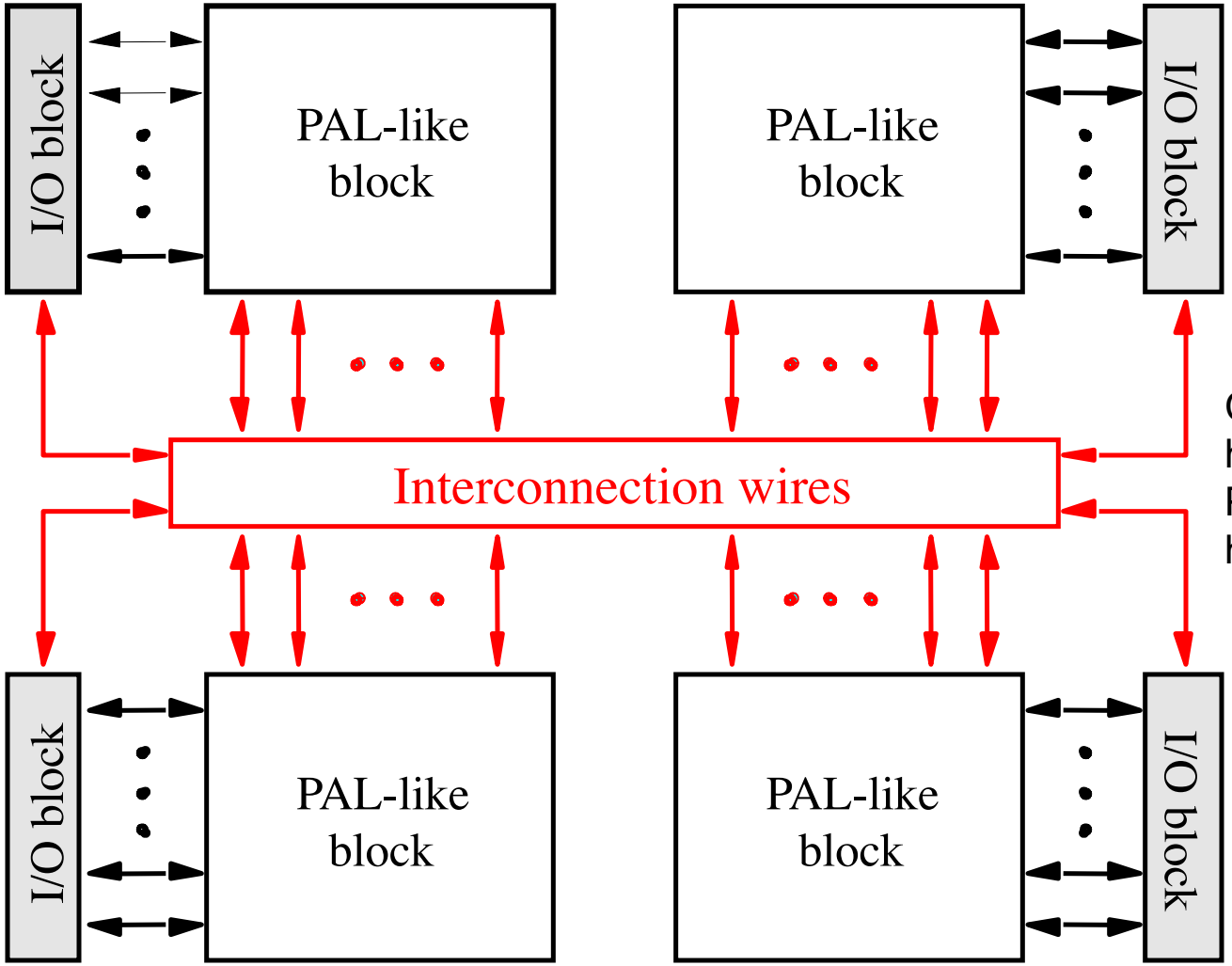


Programming methods:

- Fuse & anti-fuse
  - permanent
- PROM, EPROM & EEPROM
  - slow & low density
- SRAM
  - fast, but volatile
- Flash
  - non-volatile, not as fast

B&V3, Figure 3.30 (courtesy of Data IO Corp)

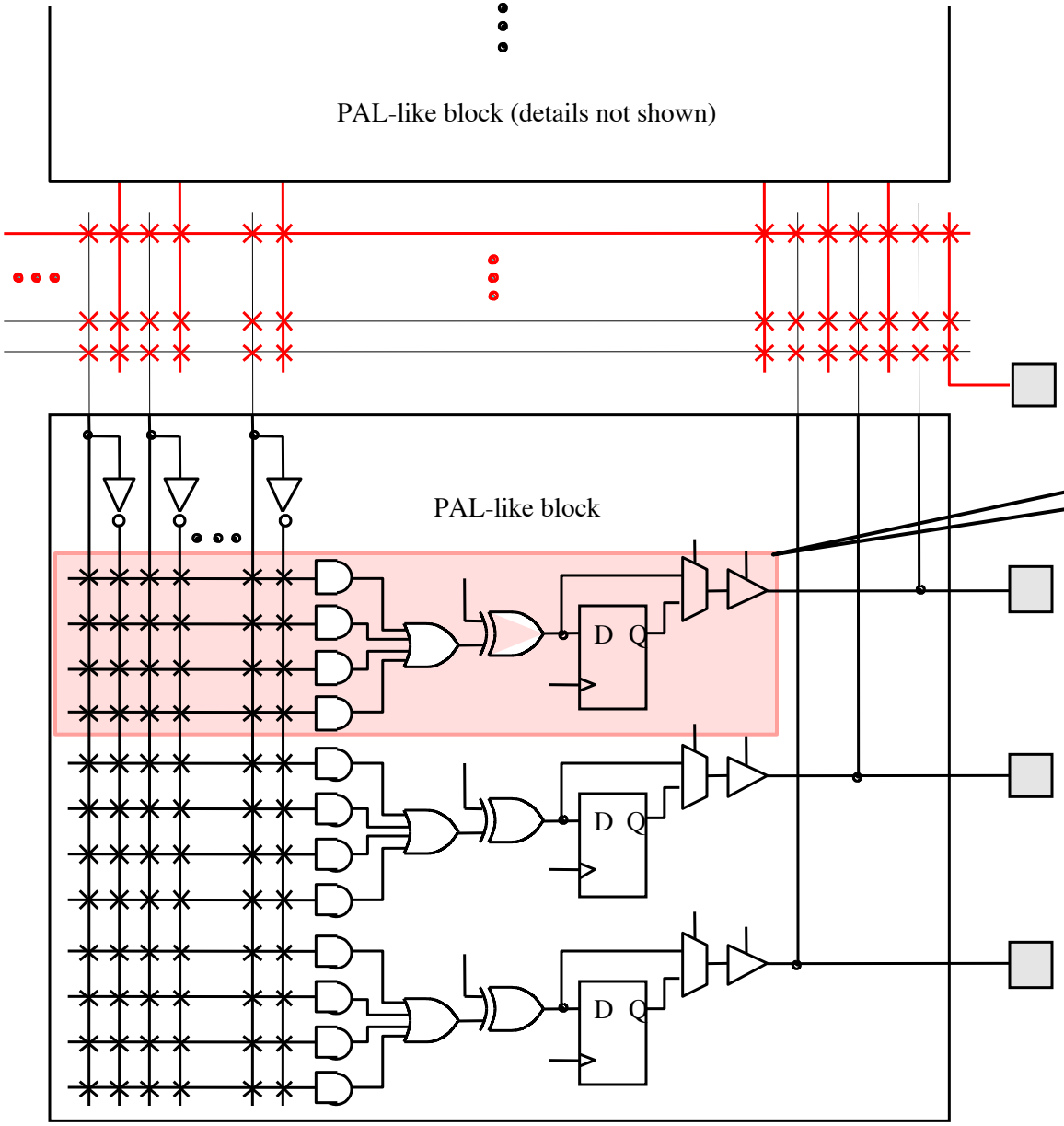
Structure of a complex programmable logic device (CPLD)



Commercial CPLDs have between 2 and 100 PAL-like blocks as seen here.

B&V3, Figure 3.32

A section of a CPLD



Macrocell

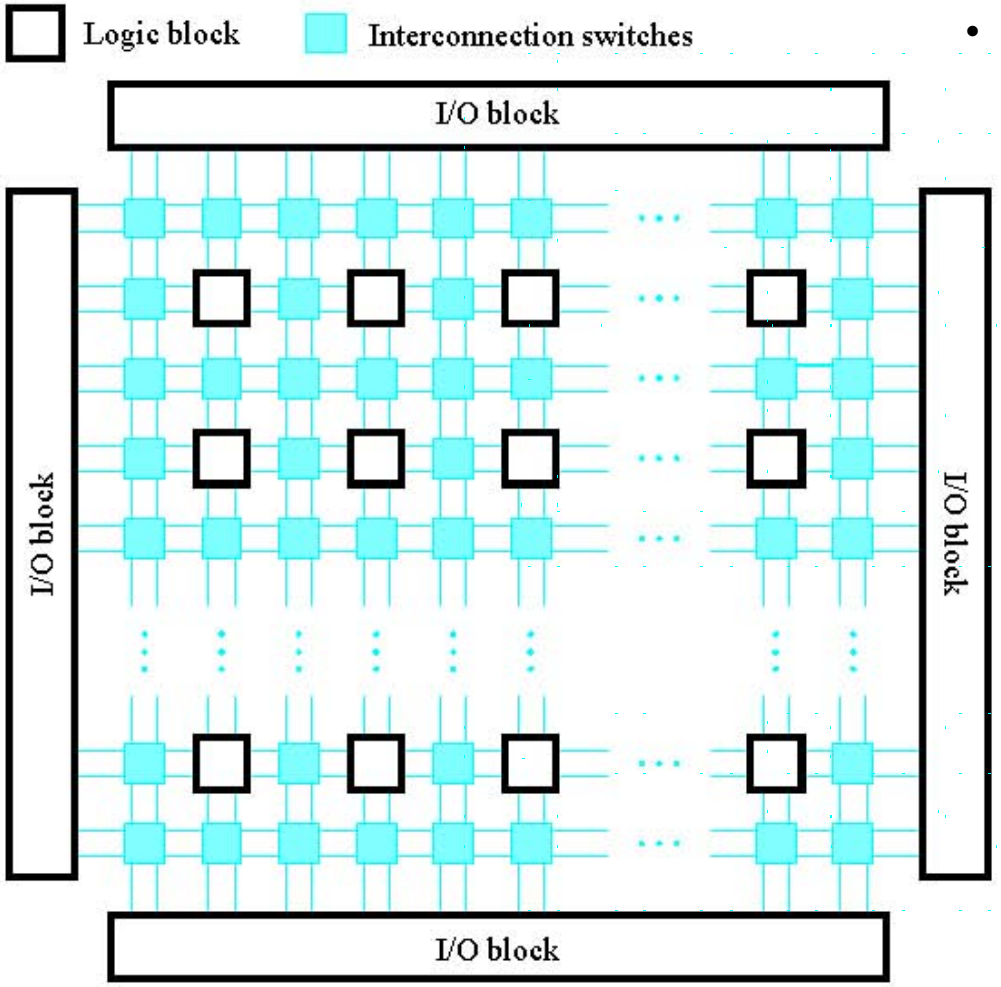
PAL-like blocks for commercially available CPLDs typically have 16 macrocells with 5–20 inputs per OR gate.

XOR used to optionally complement output.

Three-states allow pins to be used as outputs or as inputs.

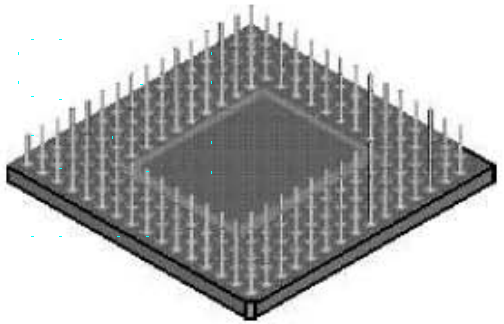
B&V3, Figure 3.33

A field-programmable gate array (FPGA)



(a) General structure of an FPGA

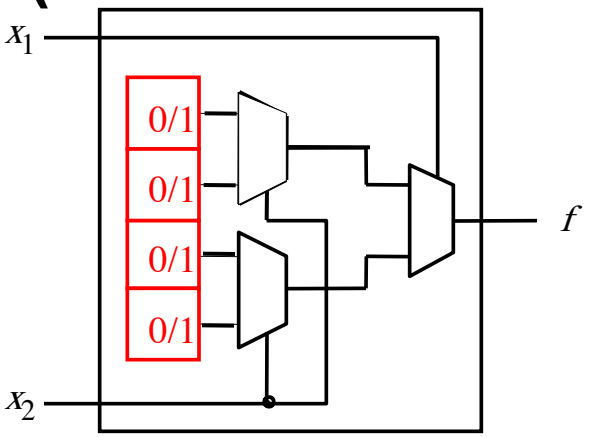
- For cost and performance reasons, as few chips as possible are used to implement a design
  - 7400-series chips implement the equivalent of just a few two-input NAND gates (the prevalent metric for chip size)
  - An SPLD or CPLD macrocell represents about 20 equivalent gates; thus a PAL with 8 macrocells can accommodate a circuit that needs up to about 160 gates and a large CPLD with 500 macrocells can implement circuits of up to 10,000 equivalent gates
  - Modern FPGAs can be used to implement circuits of millions of equivalent gates in size



(b) Pin grid array (PGA) package (bottom view)

B&V3, Figure 3.35

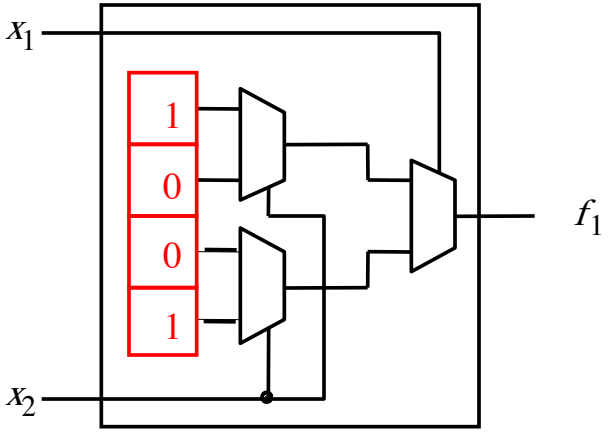
A two-input lookup table (LUT) as an FPGA logic cell (recall from Week 1)



(a) Circuit for a two-input LUT

x_1	x_2	f_1
0	0	1
0	1	0
1	0	0
1	1	1

(b) $f_1 = \bar{x}_1 \bar{x}_2 + x_1 x_2$

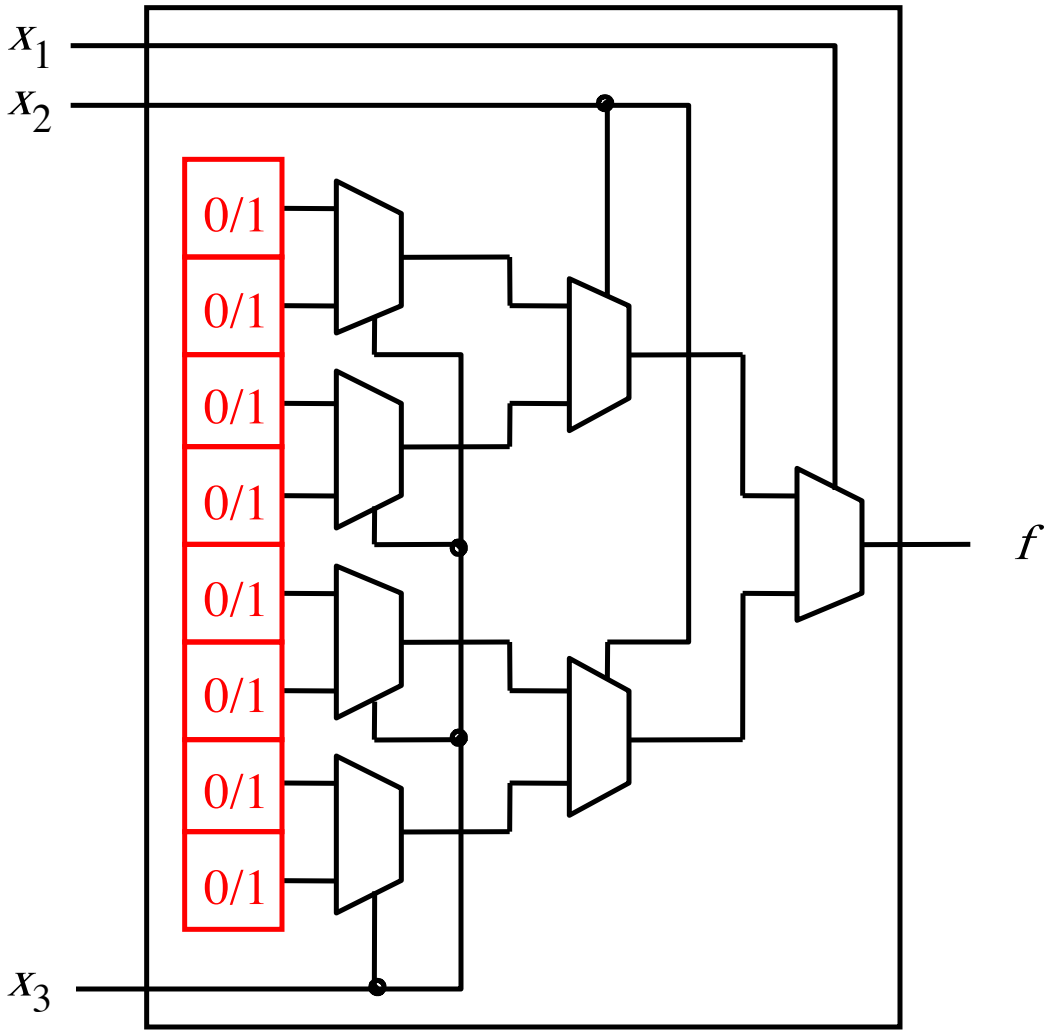


(c) Storage cell contents in the LUT

An *n*-input LUT serves as a small 2^n address memory to implement an arbitrary Boolean function of *n* variables

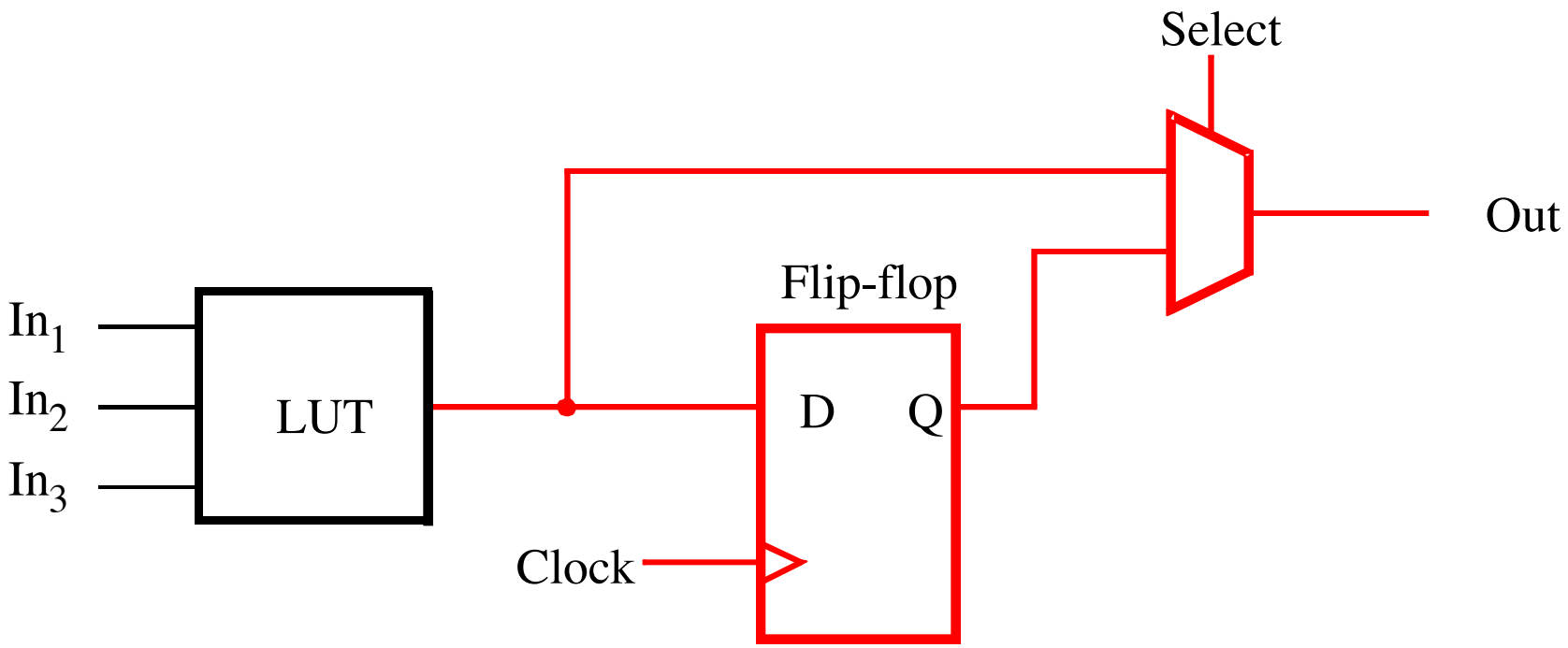
B&V3, Figure 3.36

A three-input LUT



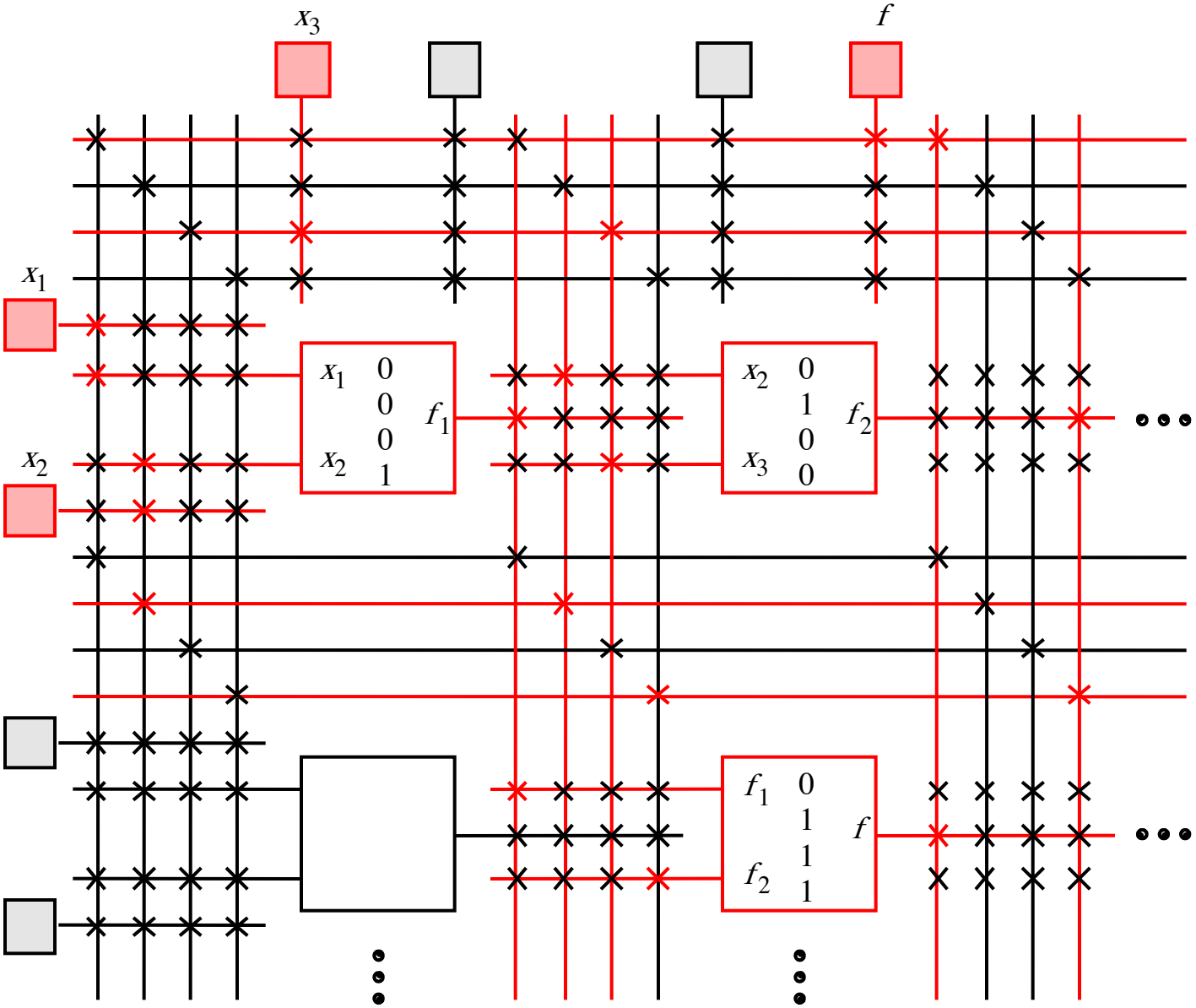
B&V3, Figure 3.37

Inclusion of a flip-flop in an FPGA logic block



B&V3, Figure 3.38

A section of a programmed FPGA



$f_1 = x_1x_2$
$f_2 = \overline{x}_2x_3$
$f = f_1 + f_2$
$= x_1x_2 + \overline{x}_2x_3$

In an FPGA, the LUT contents, routing switches and connection boxes are most commonly provided by SRAM cells

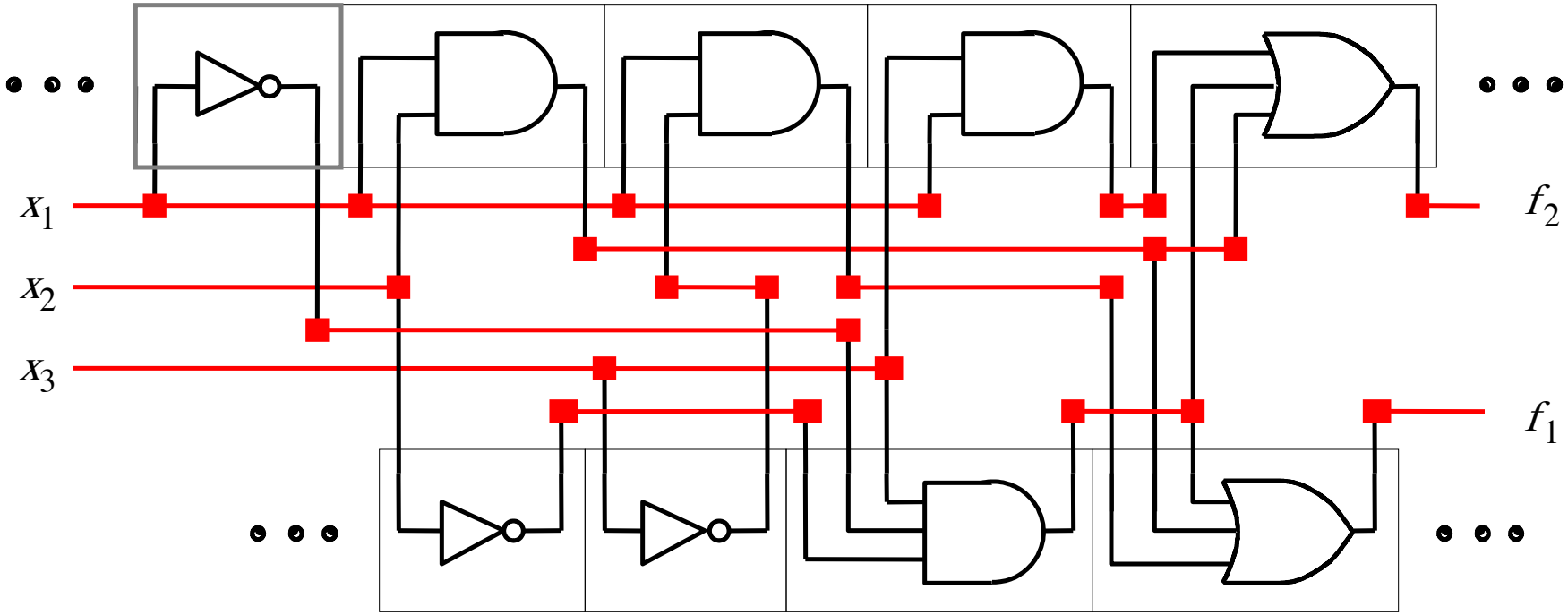
B&V3, Figure 3.39

Custom chips

- The key factor that limits the size of a circuit that can be accommodated in a PLD is the presence of the programmable switches
- Although the *switches afford PLDs their flexibility*, they *consume significant amounts of space* and thus *lead to increased cost, increased power consumption and reduced speed*
- To provide the largest number of logic gates, highest circuit speed or lowest power, a so-called *custom chip* can be manufactured
- Whereas PLDs are prefabricated and available off-the-shelf, *custom chips are created from scratch – the designer has complete flexibility to determine the number of transistors, their placement, and how they are to be interconnected*
- This enormous design effort leads to *significant expense*, and thus they are *only produced when standard parts do not meet requirements or the quantity that is to be sold justifies the cost*
  - Examples of products usually realized as custom chips include: *microprocessors, memory chips, GPUs, FPGAs, WiFi transceivers, digital broadcast receivers, engine management units,…*

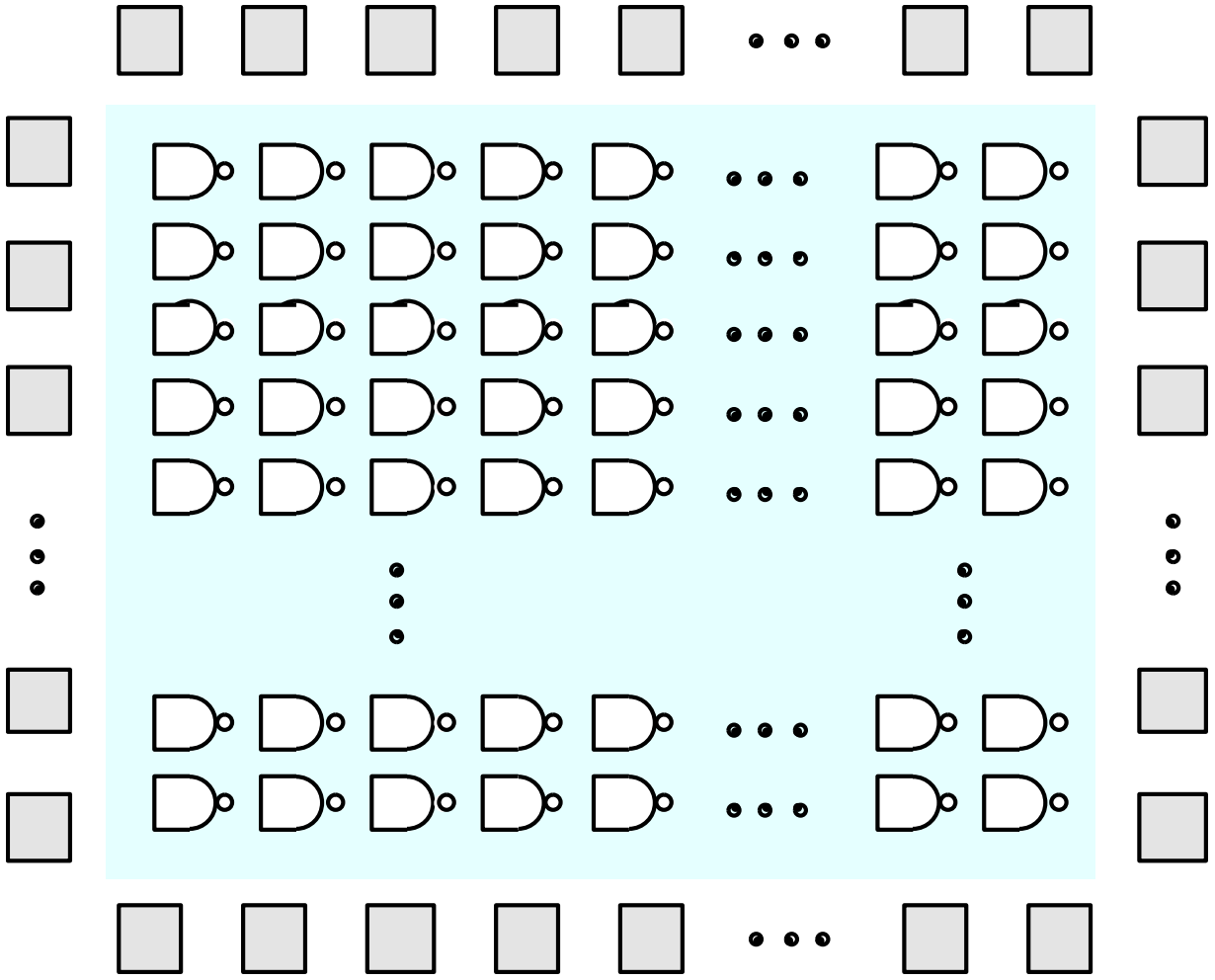
A section of two rows in a standard-cell chip

- When the designer doesn't need complete flexibility over the layout of individual transistors, a technology known as *standard cells* can be used
- Chips made using this technology are also called *application-specific integrated circuits (ASICs)*
- A standard cell design uses standard gates available from a library and is thus mostly concerned with their interconnection rather than the layout of transistors to create the cell



B&V3, Figure 3.40

A sea-of-gates gate array



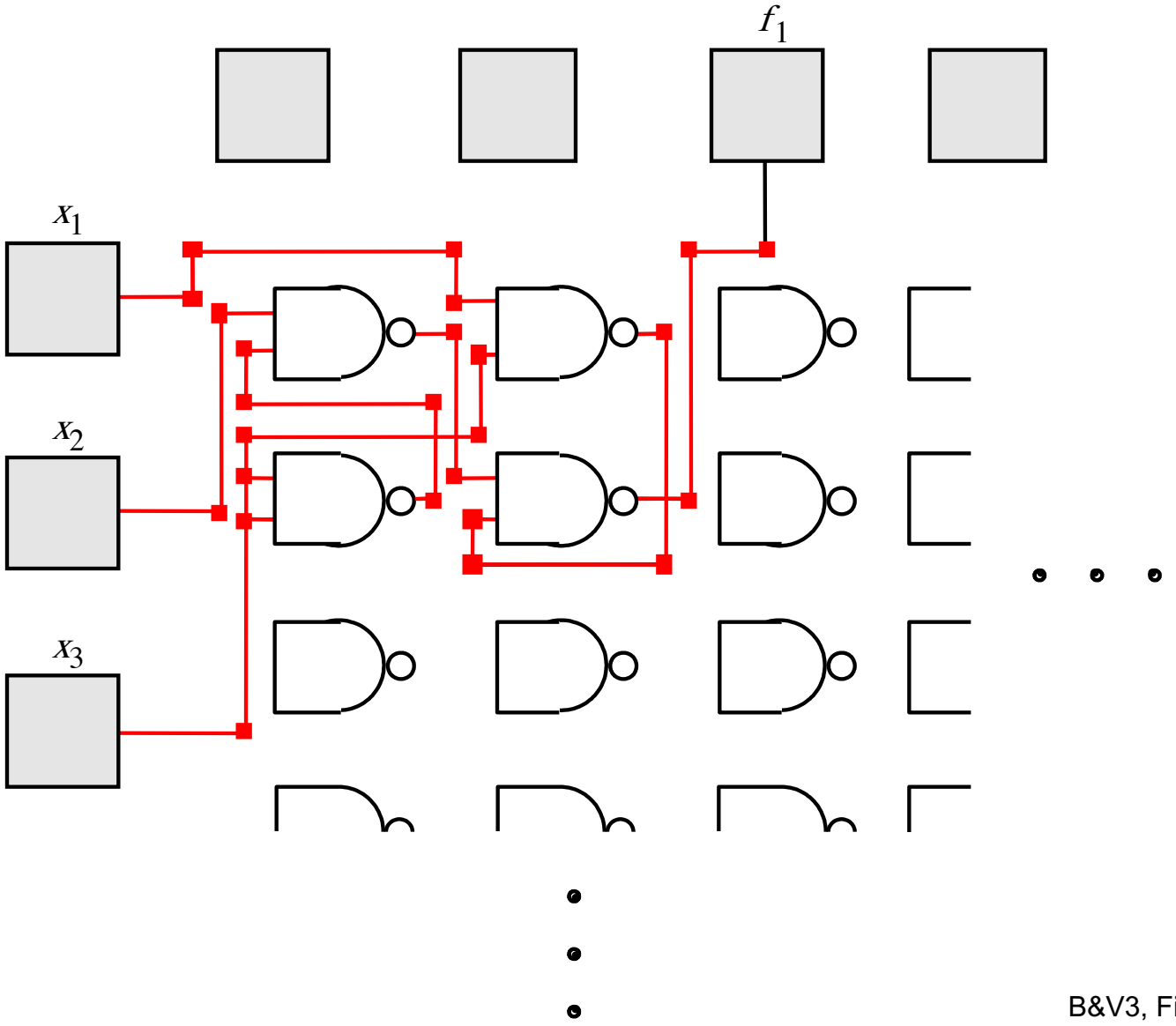
In *gate array* technology, a predefined array of gates is prefabricated onto the silicon wafer.

The designer is then only concerned with the interconnections that are required to implement the design.

These are added in a late fabrication step. Such devices are said to be *mask programmable*.

B&V3, Figure 3.41

The logic function $f_1 = x_2\overline{x}_3 + x_1x_3$ in a gate array



B&V3, Figure 3.42