

Introduction to Computer Engineering and Career Planning

Computer foundations and data representation

Prof. Dr. Mehmet Göktürk
Fall 2026 · Gebze Technical University

Tuesday 13:30–16:30 · Z23

CSE101 · Introduction to Computer Engineering and Career Planning · 8 ECTS

CSE107 · Introduction to Computer Science Laboratory · 2 ECTS

Principles of computing, assembly, C and Arduino.

Career-planning activities take place outside the scheduled class topics.

Aim of the course

Understand how hardware and software work together.

Read representations, trace operations and explain results.

Write and test small programs; connect them to physical input and output.

Build foundations that you can retain and reuse in later courses.

Terminology

Engineer
Engineering
Science
Computer Science
Computer Engineering
Data Scientist
Systems Engineering
Programmer
.....



Course outline · first half

Weeks	Topics
1–2	Foundations, data representation and computer architecture
3–4	Algorithms and assembly; operating systems and processes
5–6	Programming languages and C; physical computing and control
7–8	Software engineering; functions, arrays and algorithm efficiency

Topics build on each other; the weekly placement may adjust with progress.

Course outline · second half

Weeks	Topics
9–10	Networks, pointers and strings; the Web and file processing
11	Databases and records in C
12	Artificial intelligence and data structures
13–14	Security, privacy and integration; course synthesis and review

The project festival is after teaching ends; its date will be announced.

Lecture, recitation and lab

Activity	Schedule
CSE101 lecture	Tuesday · 13:30–16:30 · Z23
TA recitation	Wednesday late afternoon · about one hour
CSE107 lab	Friday · one assigned two-hour session

The three sessions reinforce the same foundations.

Week 1: introduction and orientation; no regular lab experiment.

CSE101 includes quizzes, assignments, a project and examinations.

CSE107 includes prelabs, lab-session performance and reports.

CSE107 has no midterm or final examination.

Grading weights, exact dates and detailed submission rules will be announced.

Weekly preparation and short quizzes

Read the announced topic before class.

Short questions use random student selection.

The latest quiz grade is the valid grade.

Students not yet assessed are included in the final teaching week.

CSE107: the weekly workflow

Friday evening: the prelab for the following week is released.

The day before the lab: submit your prelab.

During the lab: practise and demonstrate your work to the TAs.

The following week: submit the lab report. Check announced holiday adjustments.

Materials and support

PPTX decks are the core course material.

Website: weekly materials and course information.

Teams: communication, submissions and grades.

Lecture recordings support review.

Efe Ülgen Yılmaz · Ferda Abbasoğlu
Fatmanur Esirci · Arif Burak Dikmen

mehmetgokturk.com/cse101/

mehmetgokturk.com/cse107/

Science, engineering and programming

Perspective	Typical concern
Computer science	Representing problems and finding methods to solve them
Computer engineering	Designing systems where hardware and software work together
Programming	Expressing a method precisely and testing its behaviour

These activities overlap. Computing includes much more than writing code.

Computers were once people



People followed calculation procedures.

Machines automate operations on represented information.

Computers today



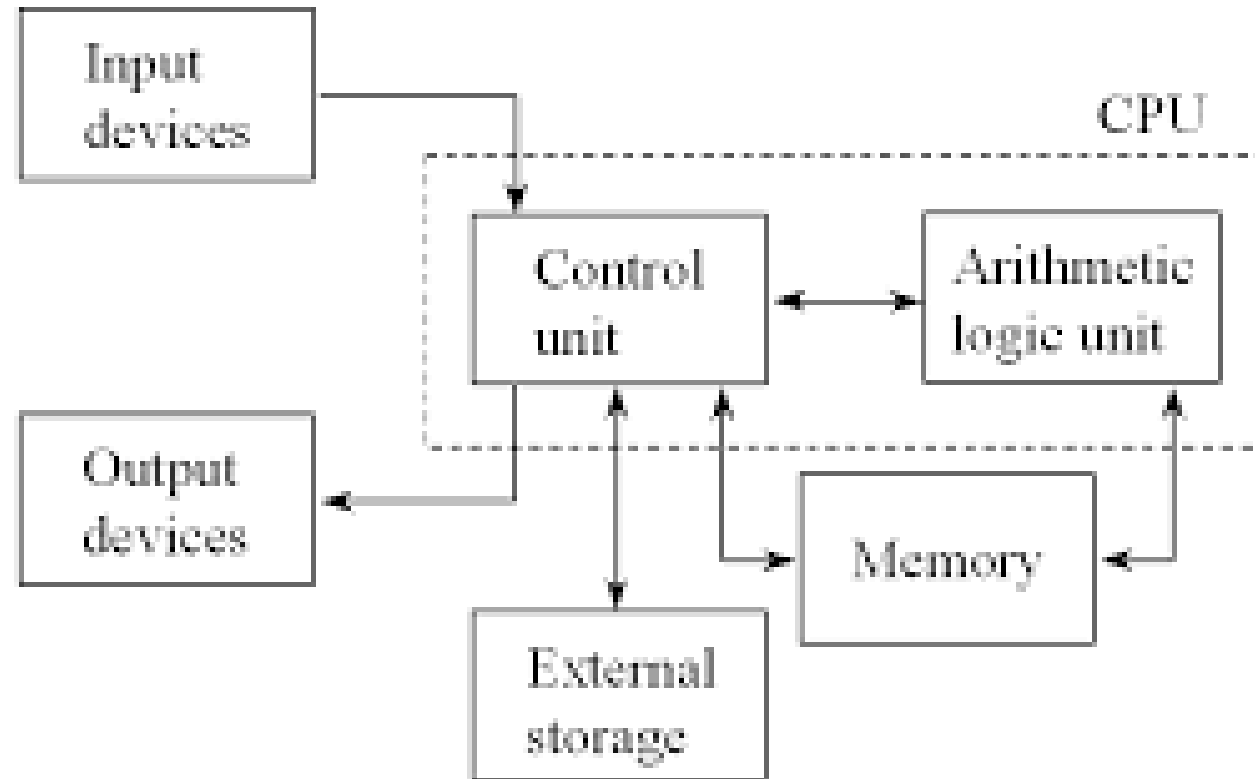
Input, processing, output—no magic



Something goes in.
The mechanism transforms it.
Something comes out.

Turning the handle advances the
mechanism; clock ticks
coordinate a computer's steps.

A computer: connected functional parts



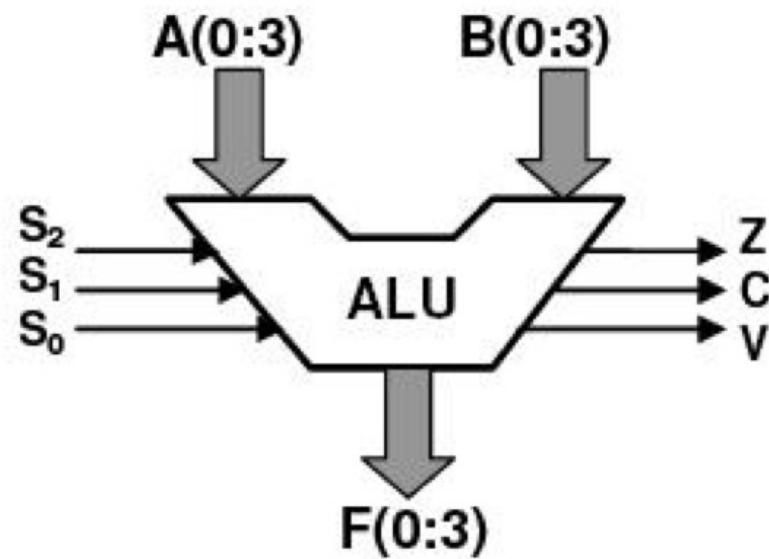
Simple Computer Architecture

Input, processing, memory and output work together under program control.

Computer (an early IBM film)



The arithmetic logic unit



Z, C and V are status flags

Z = 1 if F=0

C = Carry or Borrow

V = Overflow

S ₂	S ₁	S ₀	Function (F)
0	0	0	A+B
0	0	1	A-B
0	1	0	A-1
0	1	1	A+1
1	0	0	$A \wedge B$
1	0	1	$A \vee B$
1	1	0	NOT A
1	1	1	$A \oplus B$

The ALU operates on bit patterns. Control inputs select the operation.

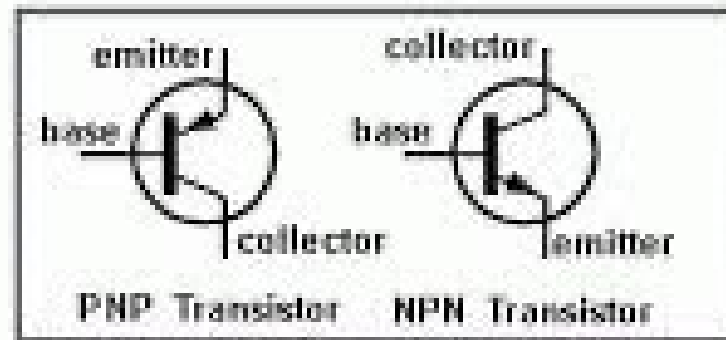
Transistors



A transistor can control conduction.

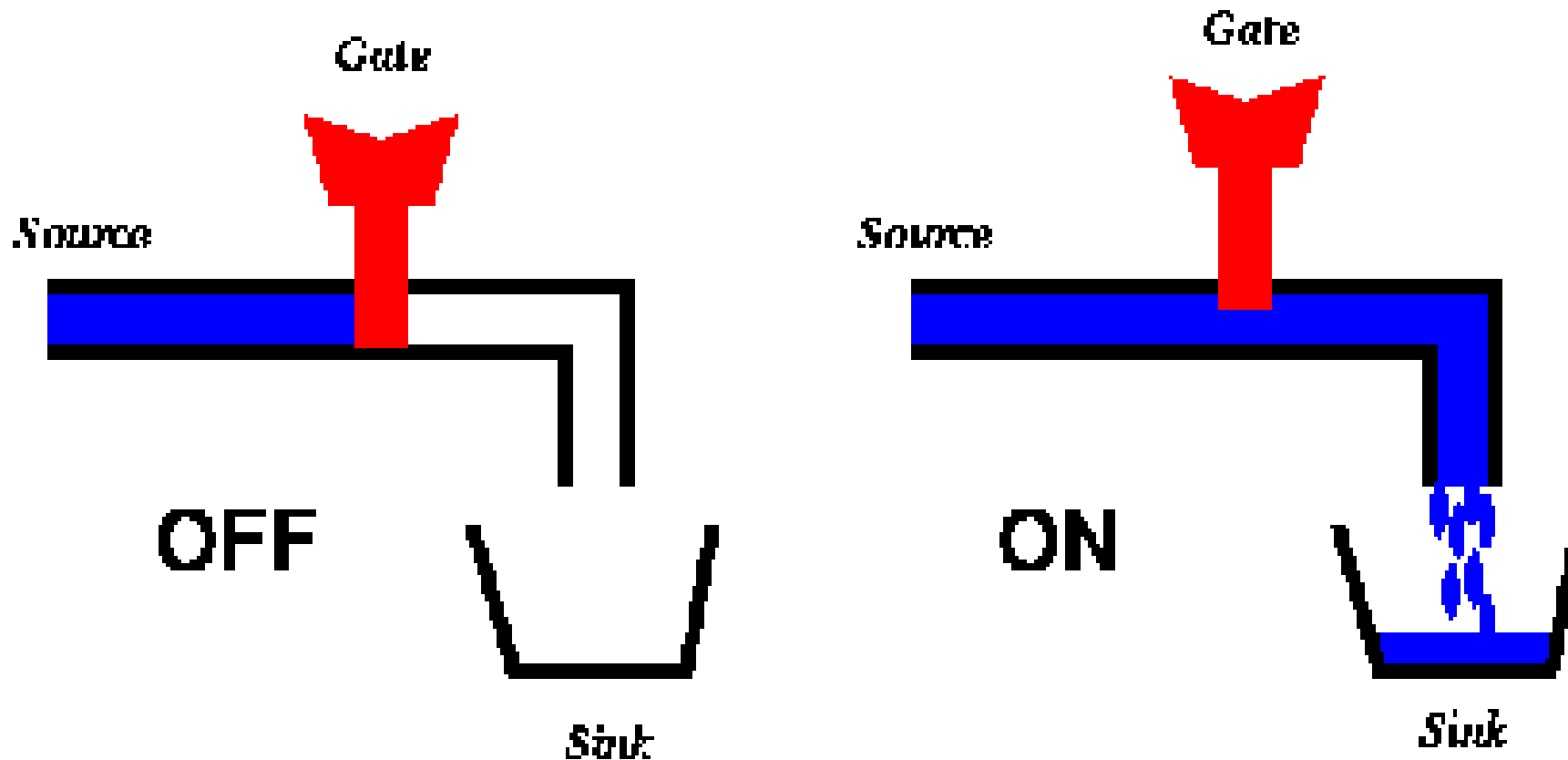
For digital logic, we use a simplified switching model: a path is enabled or disabled.

Transistor symbols and packages



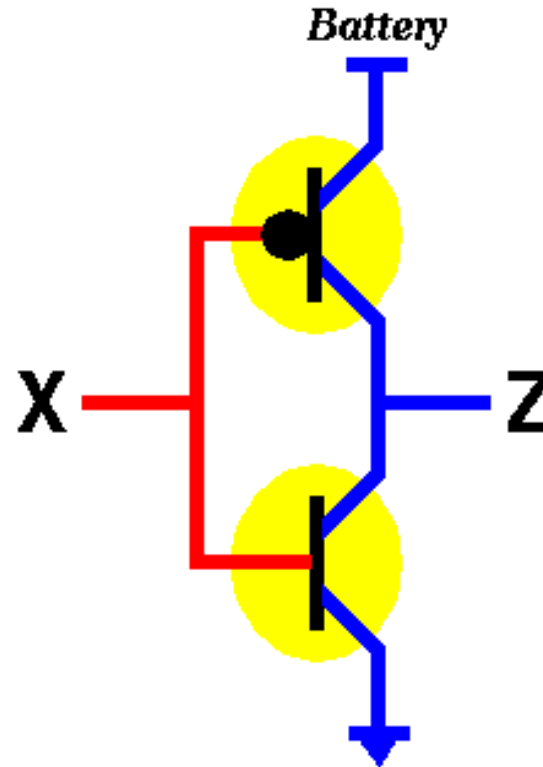
The drawing shows BJT terminal names. Other transistor types use different names.

Controlled flow: a physical analogy



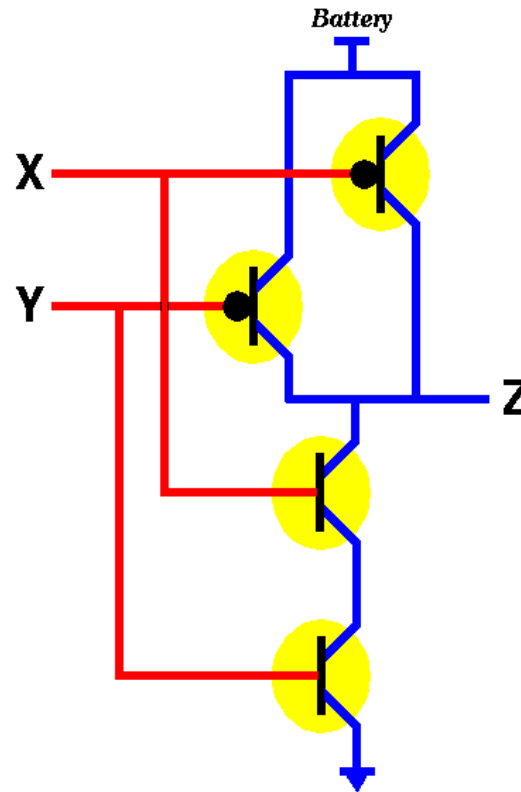
The control opens or closes a path. In a transistor, an electrical signal controls conduction.

An inverter: NOT



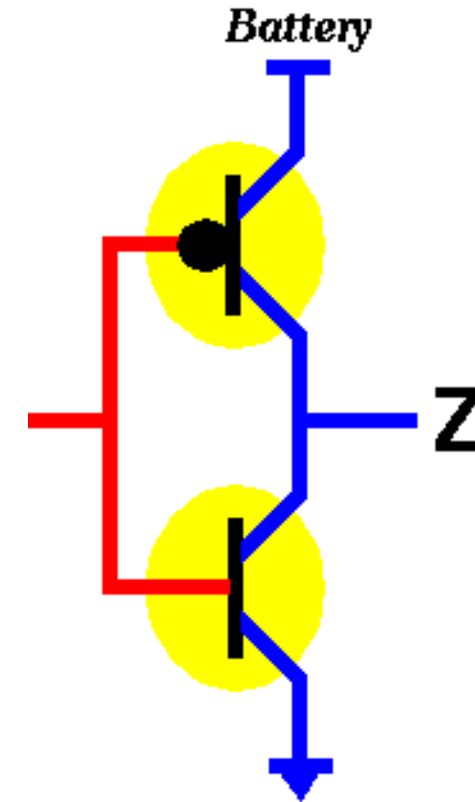
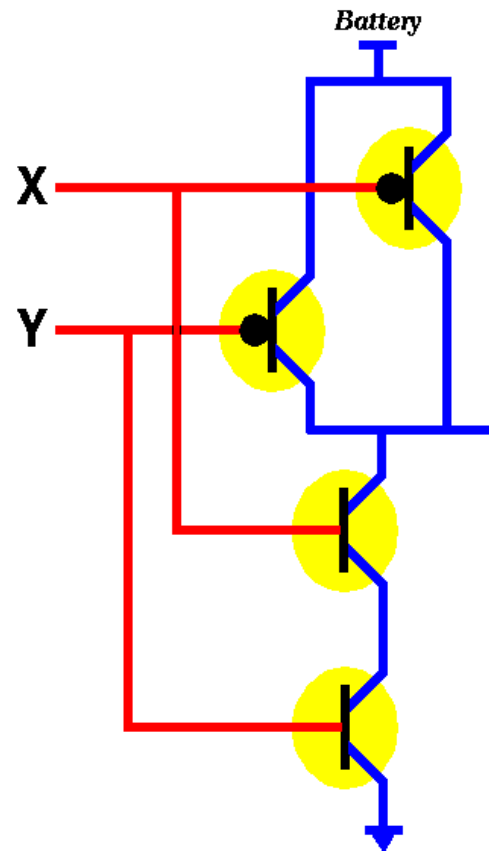
Input 0 gives output 1; input 1 gives output 0. The two paths act complementarily.

Two inputs: NAND



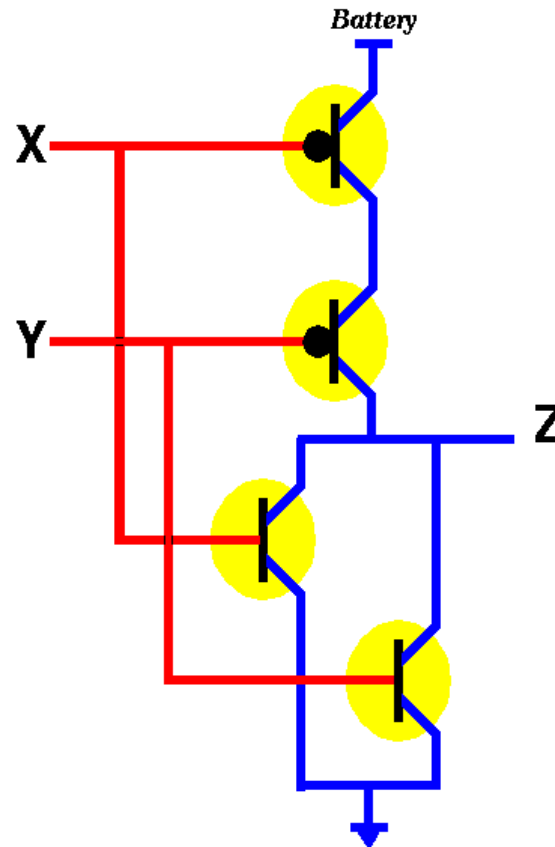
Only input 11 gives output 0. All other input combinations give output 1.

NAND followed by NOT gives AND



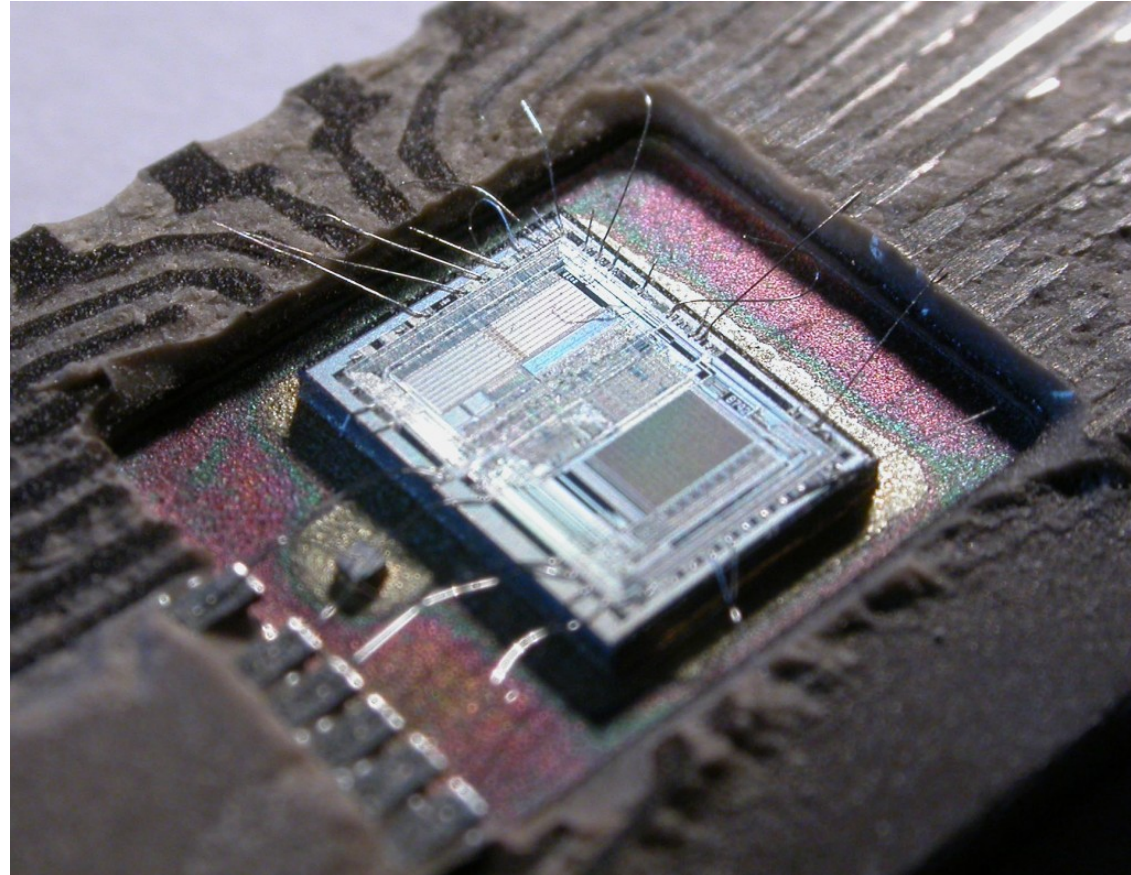
NAND output → inverter input. The final output is 1 only when both inputs are 1.

A different arrangement: NOR



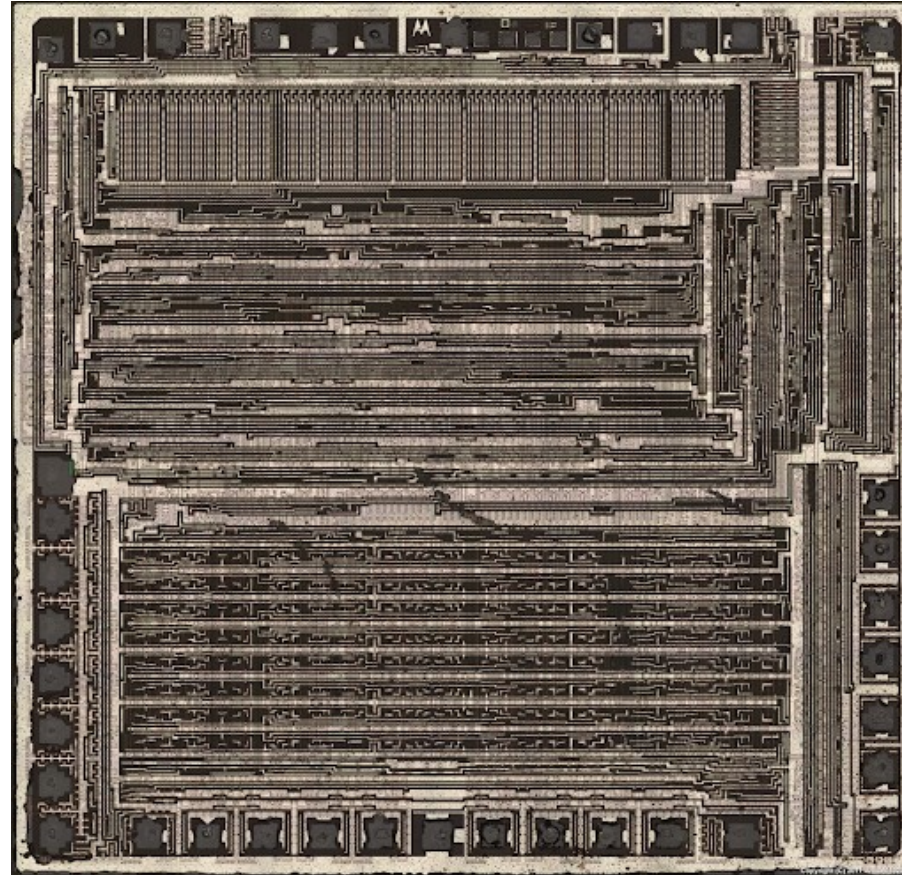
Either input at 1 makes the output 0. Only 00 gives output 1.

Integrated circuits



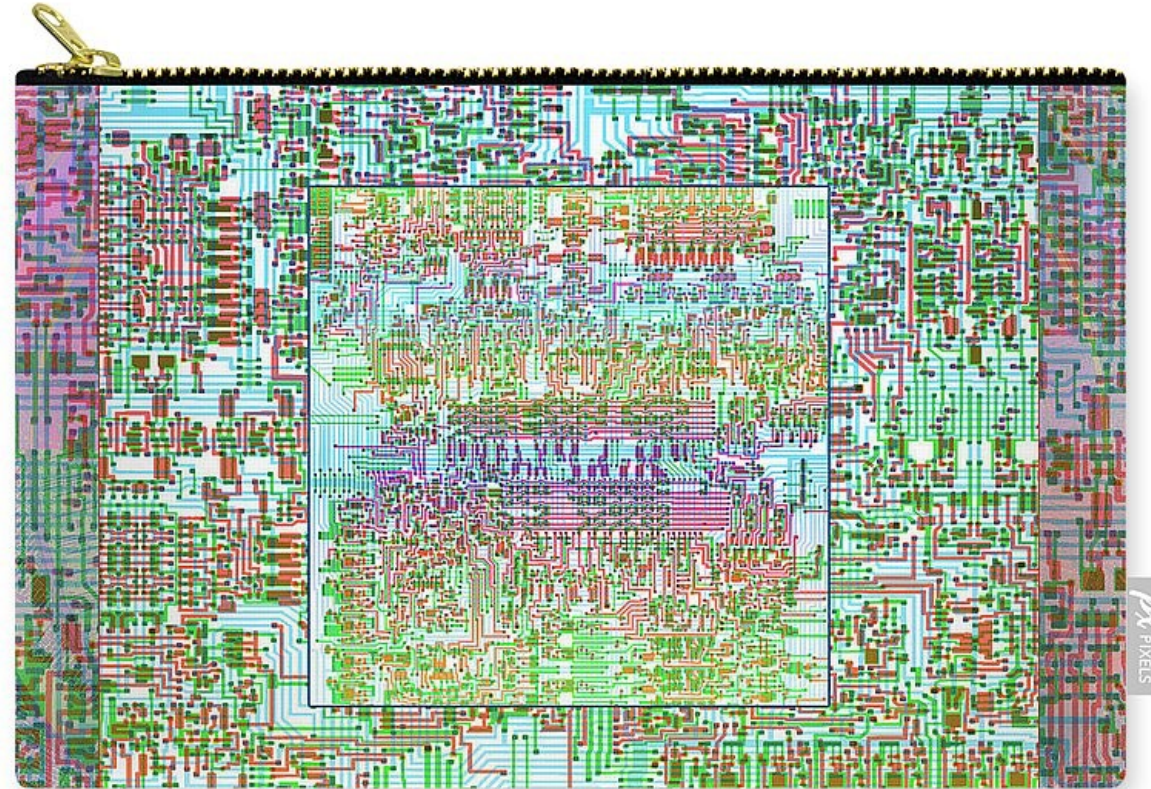
Many devices and connections are fabricated together, then connected to a package.

Inside the chip



Repeated structures and interconnections implement larger digital functions.

A processor is a physical circuit

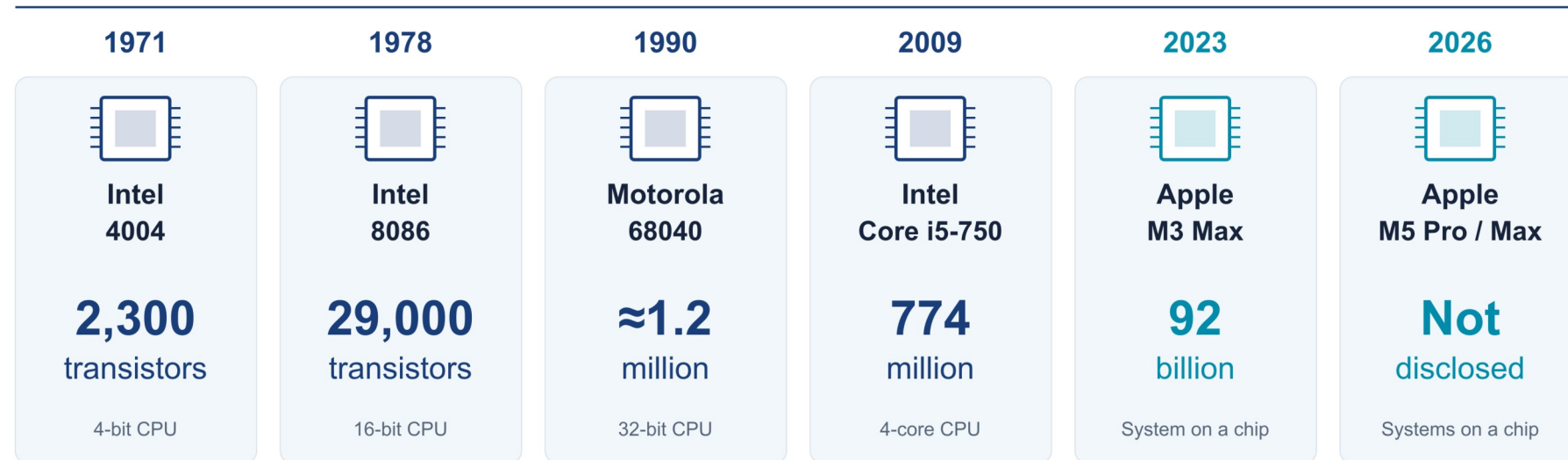


This processor layout illustrates the scale and organization of integrated circuitry.

How many transistors in a processor?

From thousands to billions of transistors

Six milestones in processor integration · 1971–2026



2,300 → 92,000,000,000

40 million times as many transistors

Counts cover the whole chip/SoC, including integrated functions and caches—not just CPU cores.

Transistor count is not a direct measure of speed. Spacing and chip symbols are schematic.

Sources: Intel; Motorola / IEEE Micro (1990); PC Perspective (2009); Apple Newsroom (2023, 2026).

M5 Pro/Max: no transistor count in Apple's launch announcement or technical specifications. Checked 17 Sep 2026.

Boolean operations

The AND operation

$$\begin{array}{r} 0 \\ \text{AND } 0 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 0 \\ \text{AND } 1 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 1 \\ \text{AND } 0 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 1 \\ \text{AND } 1 \\ \hline 1 \end{array}$$

The OR operation

$$\begin{array}{r} 0 \\ \text{OR } 0 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 0 \\ \text{OR } 1 \\ \hline 1 \end{array}$$

$$\begin{array}{r} 1 \\ \text{OR } 0 \\ \hline 1 \end{array}$$

$$\begin{array}{r} 1 \\ \text{OR } 1 \\ \hline 1 \end{array}$$

The XOR operation

$$\begin{array}{r} 0 \\ \text{XOR } 0 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 0 \\ \text{XOR } 1 \\ \hline 1 \end{array}$$

$$\begin{array}{r} 1 \\ \text{XOR } 0 \\ \hline 1 \end{array}$$

$$\begin{array}{r} 1 \\ \text{XOR } 1 \\ \hline 0 \end{array}$$

AND: both inputs. OR: at least one. XOR: exactly one of two inputs.

Gate symbols and truth tables

AND



Inputs	Output
0 0	0
0 1	0
1 0	0
1 1	1

OR



Inputs	Output
0 0	0
0 1	1
1 0	1
1 1	1

A truth table lists the output for every input combination.

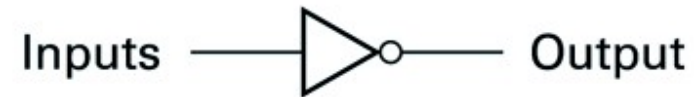
XOR and NOT

XOR



Inputs		Output
0	0	0
0	1	1
1	0	1
1	1	0

NOT



Inputs	Output
0	1
1	0

OR and XOR differ for input 11. NOT has one input.

Gate example

$$X = A \text{ AND } B \quad Y = X \text{ OR } C$$

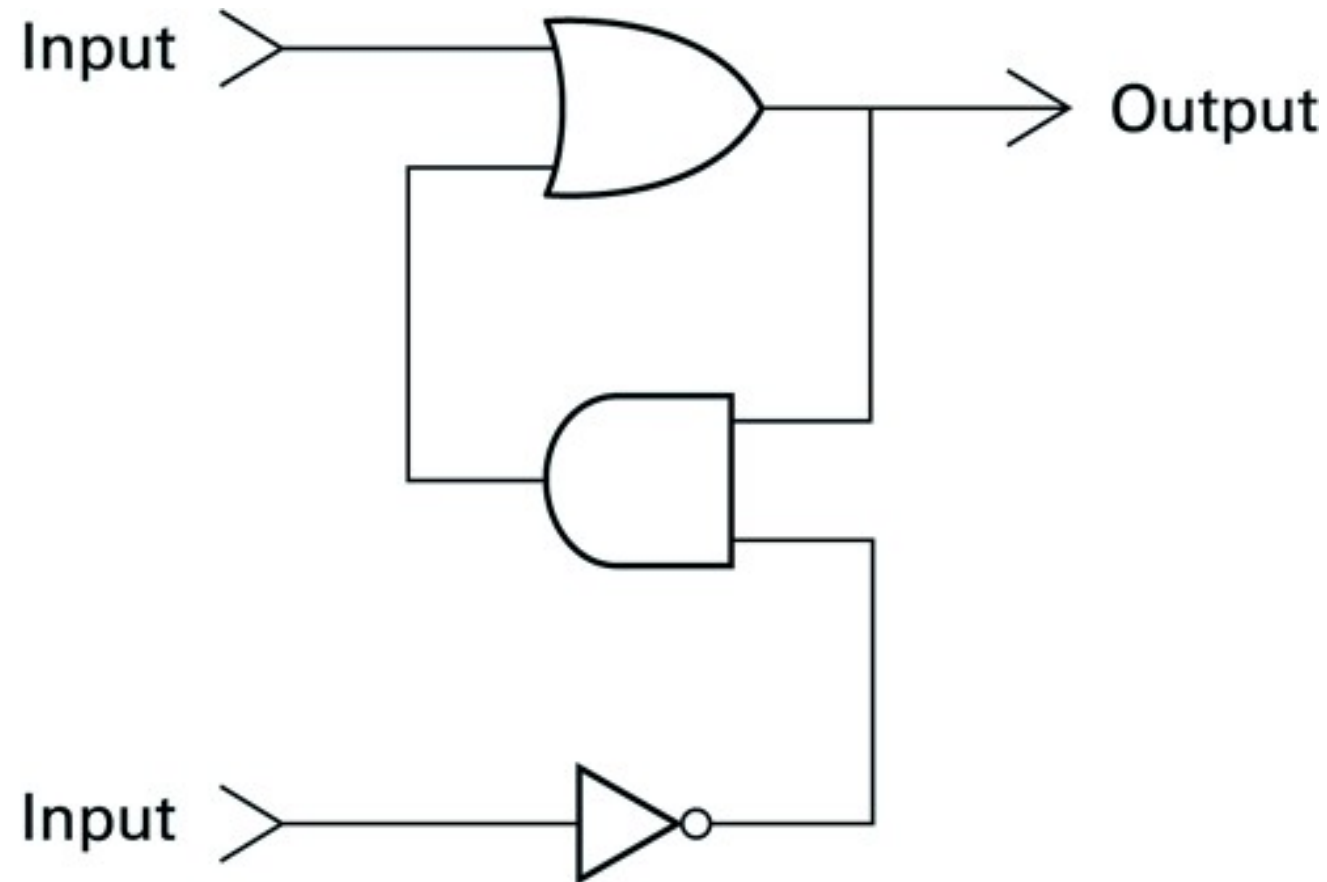
A	B	C	X	Y
1	0	0	?	?
1	0	1	?	?
1	1	0	?	?

Gate example: results

A	B	C	$X = A \text{ AND } B$	$Y = X \text{ OR } C$
1	0	0	0	0
1	0	1	0	1
1	1	0	1	1

Find the intermediate value X , then use it to find Y .

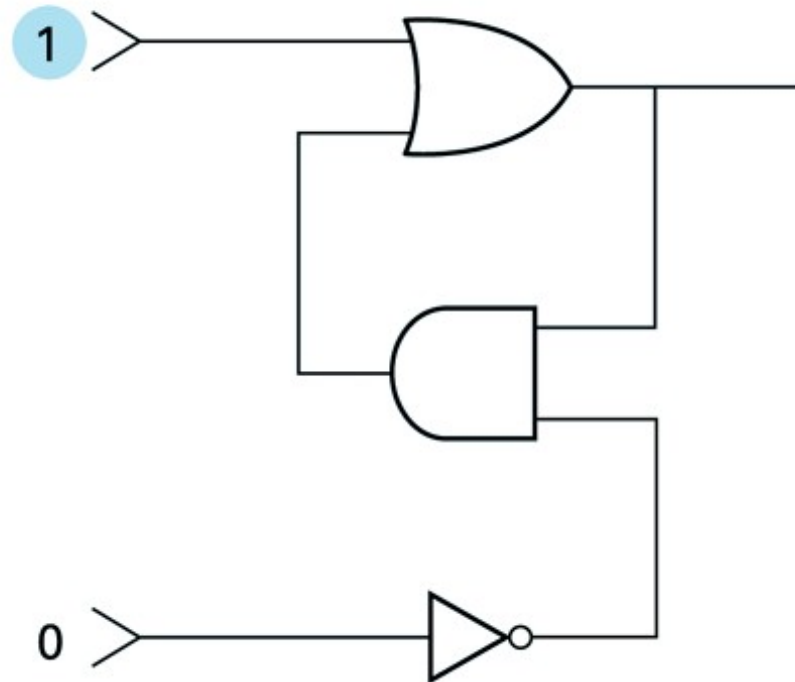
Feedback can retain a bit



The output feeds back into the circuit. Its state can persist after the input changes.

Setting the stored value: input

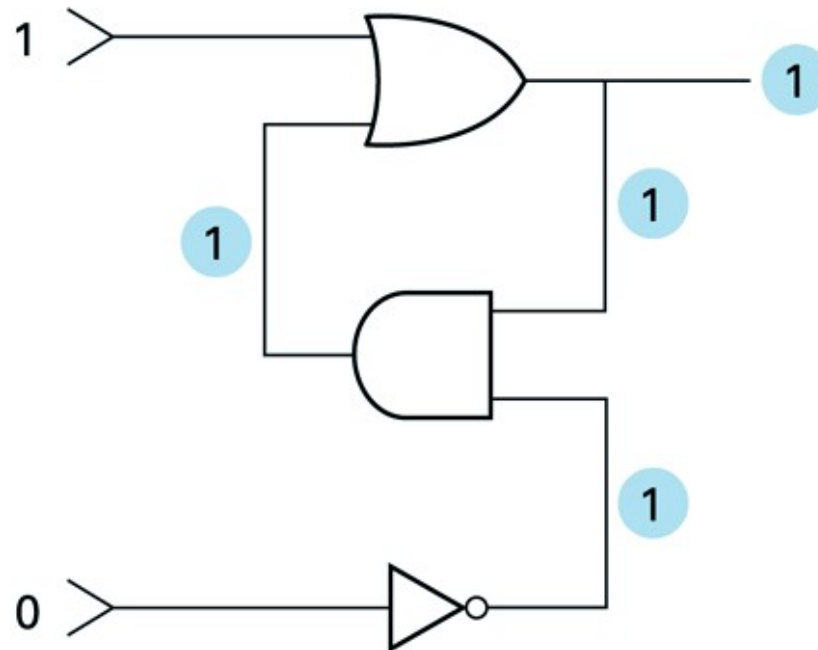
a. 1 is placed on the upper input.



The upper input is set to 1 while the lower input is 0.

Setting the stored value: propagation

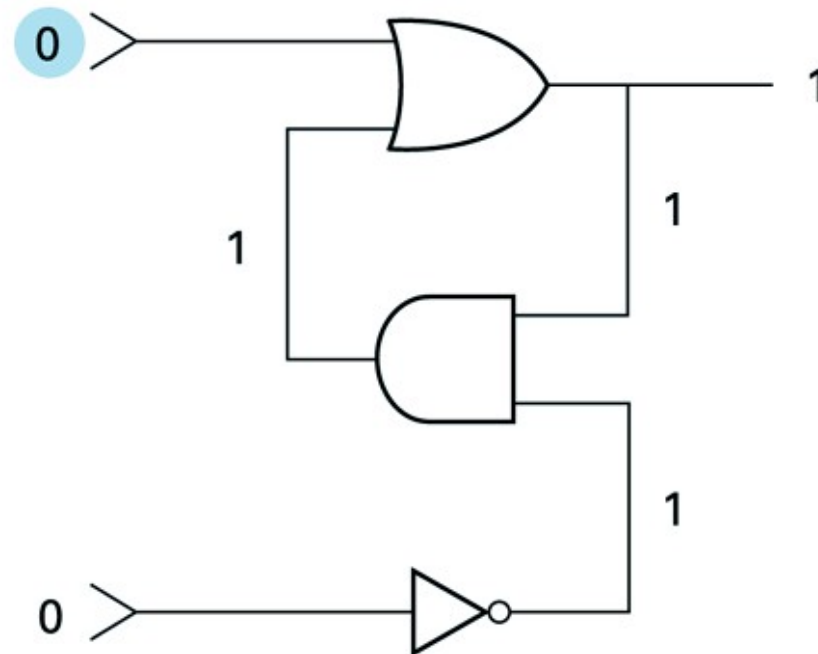
b. This causes the output of the OR gate to be 1 and, in turn, the output of the AND gate to be 1.



The 1 passes through the OR gate and returns through the AND gate.

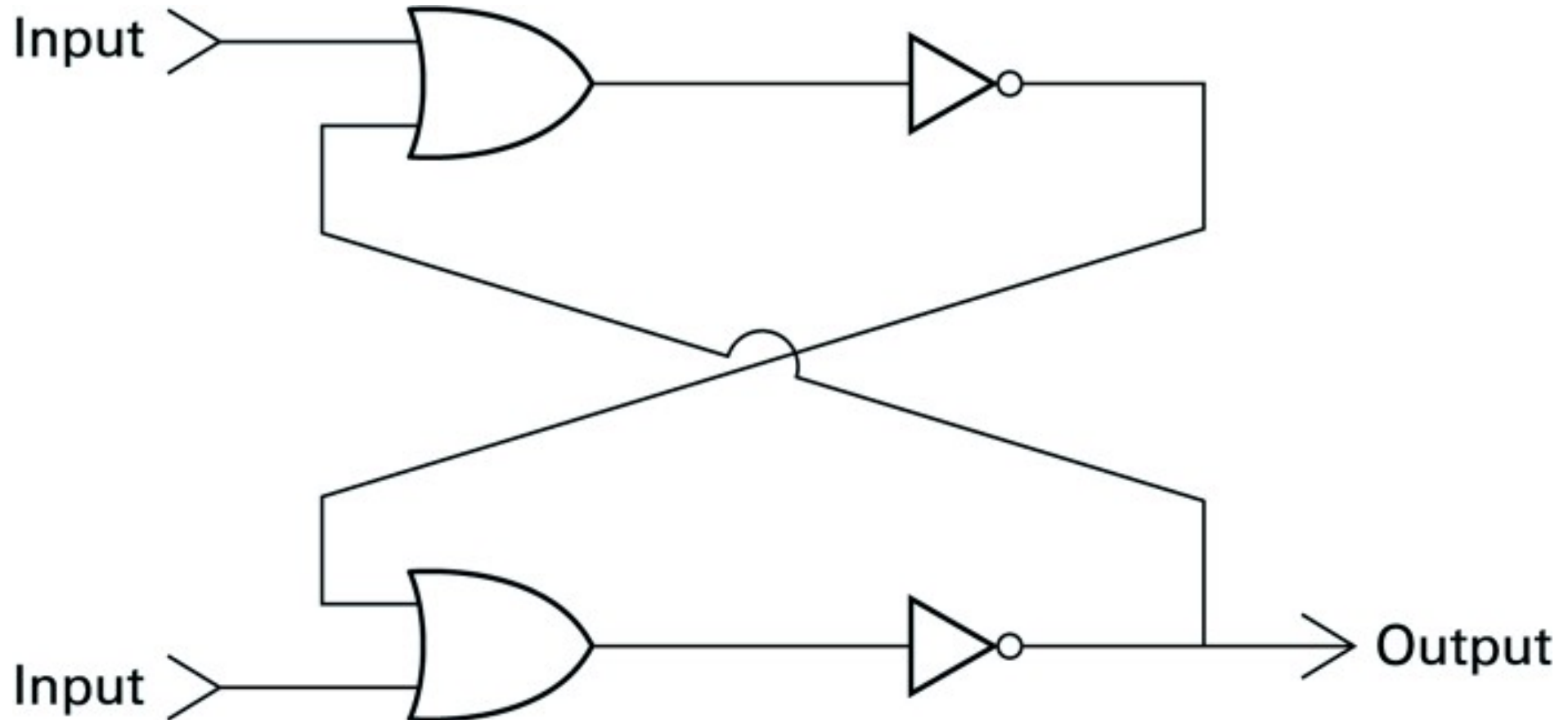
Holding the stored value

- c. The 1 from the AND gate keeps the OR gate from changing after the upper input returns to 0.



When the upper input returns to 0, feedback keeps the output at 1.

Another circuit with feedback



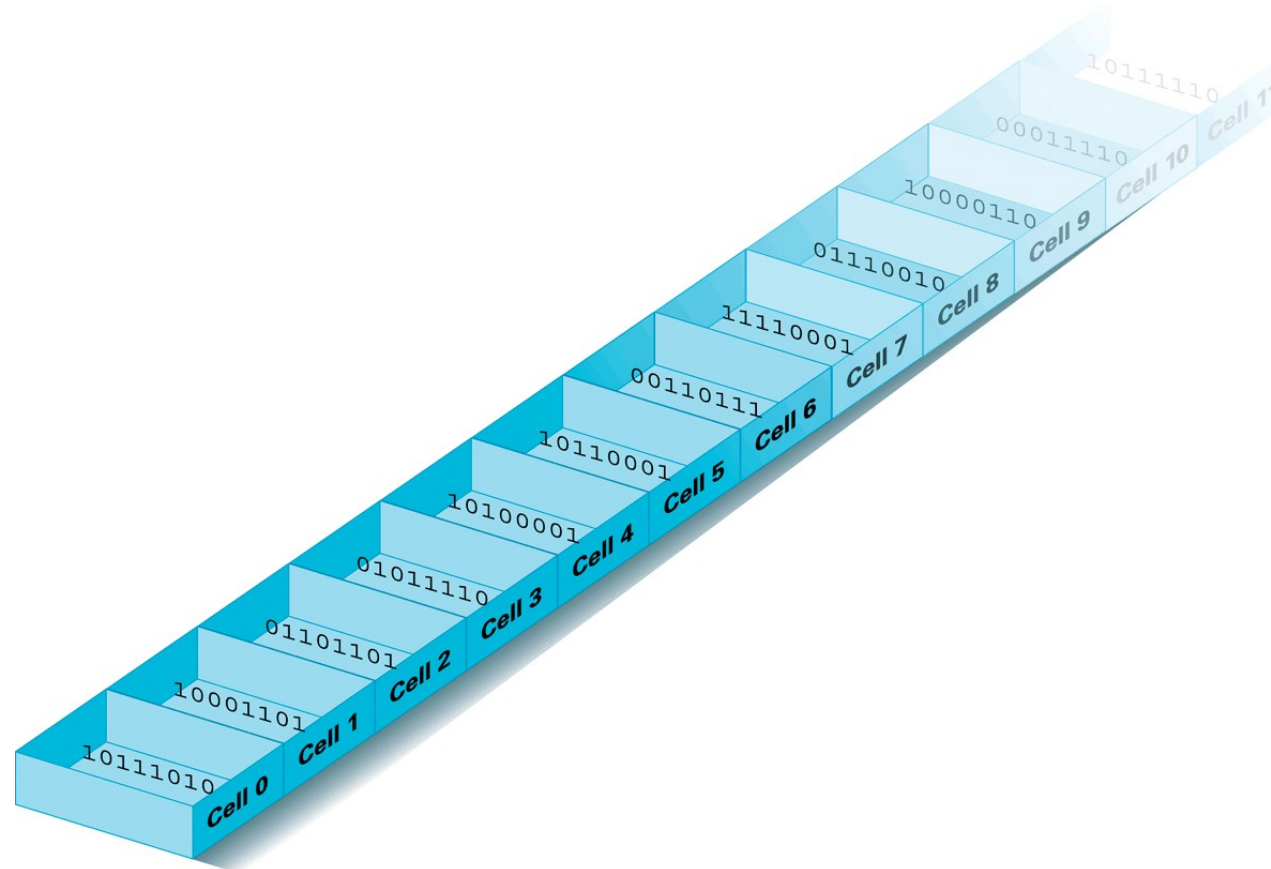
Cross-coupled gates provide another way to retain a logical state.

High-order end 0 1 0 1 1 0 1 0 Low-order end

Most significant bit Least significant bit

Bit positions have an order. The leftmost is the most significant in this representation.

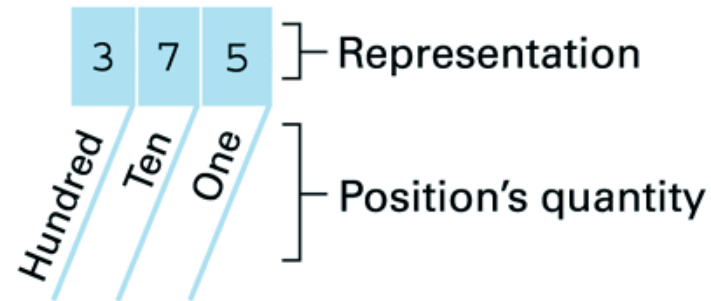
Memory cells have addresses



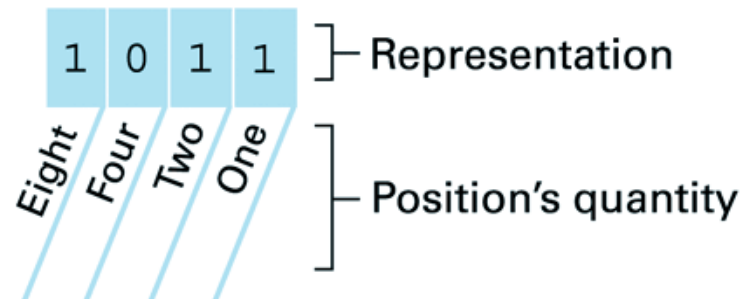
An address identifies a storage location; its contents are the value stored there.

Place value: decimal and binary

a. Base ten system

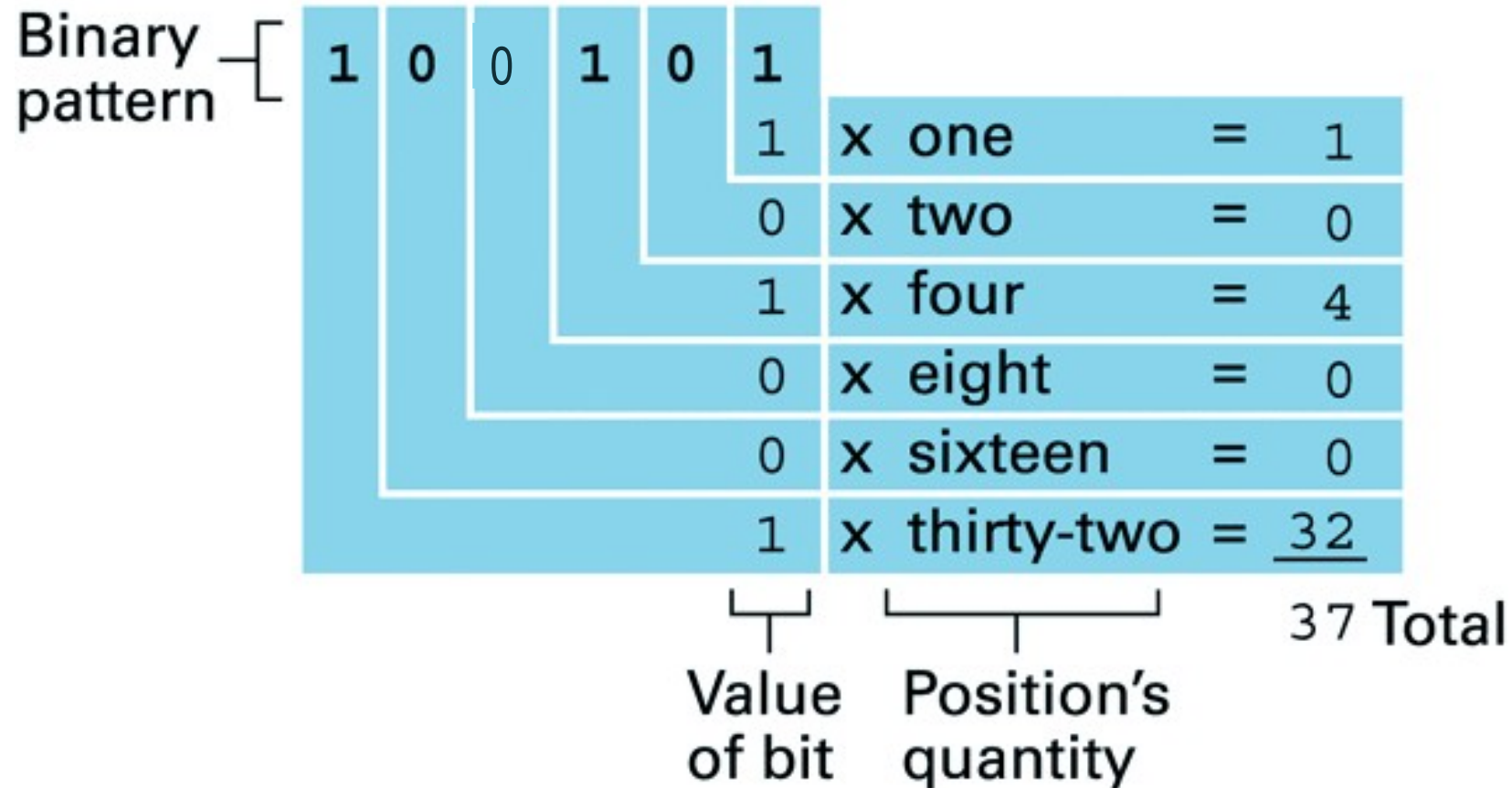


b. Base two system



Each step left multiplies the position weight by the base: 10 for decimal, 2 for binary.

Decode a binary representation



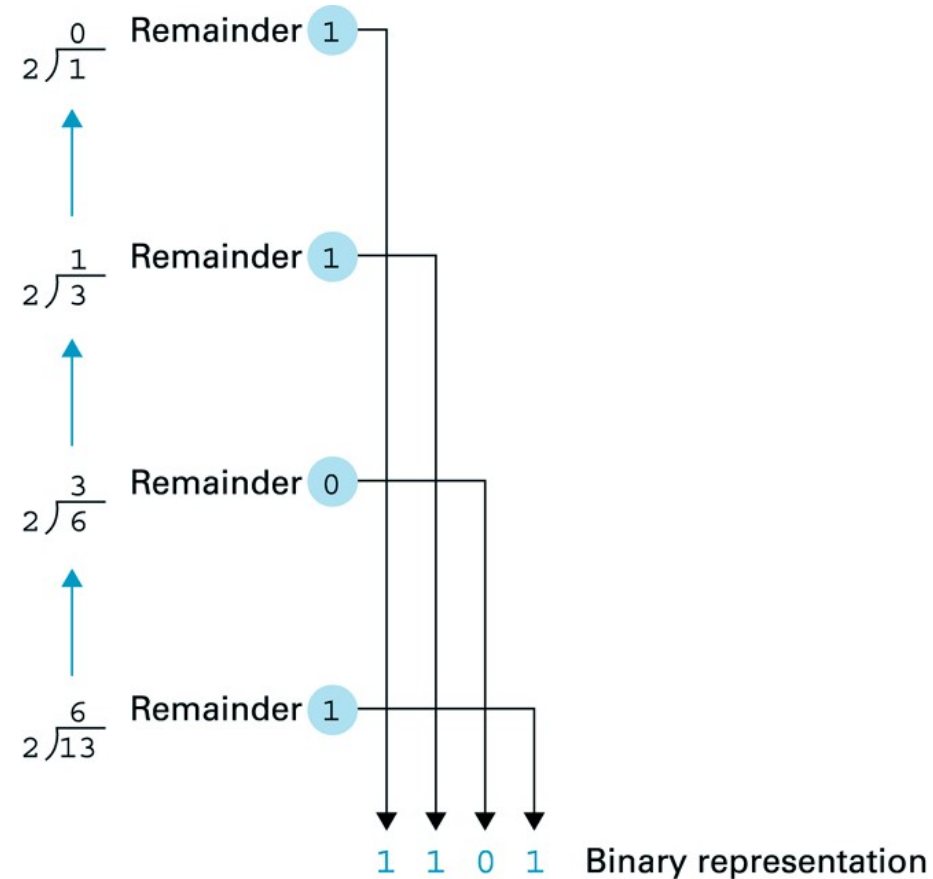
Include the weight of each position containing 1: $32 + 4 + 1 = 37$.

Decimal to binary: repeated division

- Step 1.** Divide the value by two and record the remainder.
- Step 2.** As long as the quotient obtained is not zero, continue to divide the newest quotient by two and record the remainder.
- Step 3.** Now that a quotient of zero has been obtained, the binary representation of the original value consists of the remainders listed from right to left in the order they were recorded.

Divide by 2, record each remainder, and read the remainders from last to first.

Worked example: thirteen



13 ÷ 2 leaves 1; 6 ÷ 2 leaves 0; 3 ÷ 2 leaves 1; 1 ÷ 2 leaves 1. Result: 1101.

Binary practice

Convert 11010 (base 2) to decimal.

Write 19 (base 10) in binary.

Explain why 00101 and 101 represent the same unsigned value.

Binary practice: answers

$$11010 = 16 + 8 + 2 = 26.$$

$$19 = 16 + 2 + 1 = 10011 \text{ (base 2).}$$

Leading zeros add no value: both 00101 and 101 represent 5.

Hexadecimal: four bits per digit

Bit pattern	Hexadecimal representation
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Hexadecimal uses 0–9 and A–F. One hexadecimal digit represents one group of four bits.

Hexadecimal example

0011 1010 (base 2)

0011 $\rightarrow$ 3 1010 $\rightarrow$ A

0011 1010 (base 2) = 3A (base 16)

Check in decimal: $3 \times 16 + 10 = 58$.

Characters are represented by codes

01001000

H

01100101

e

01101100

l

01101100

l

01101111

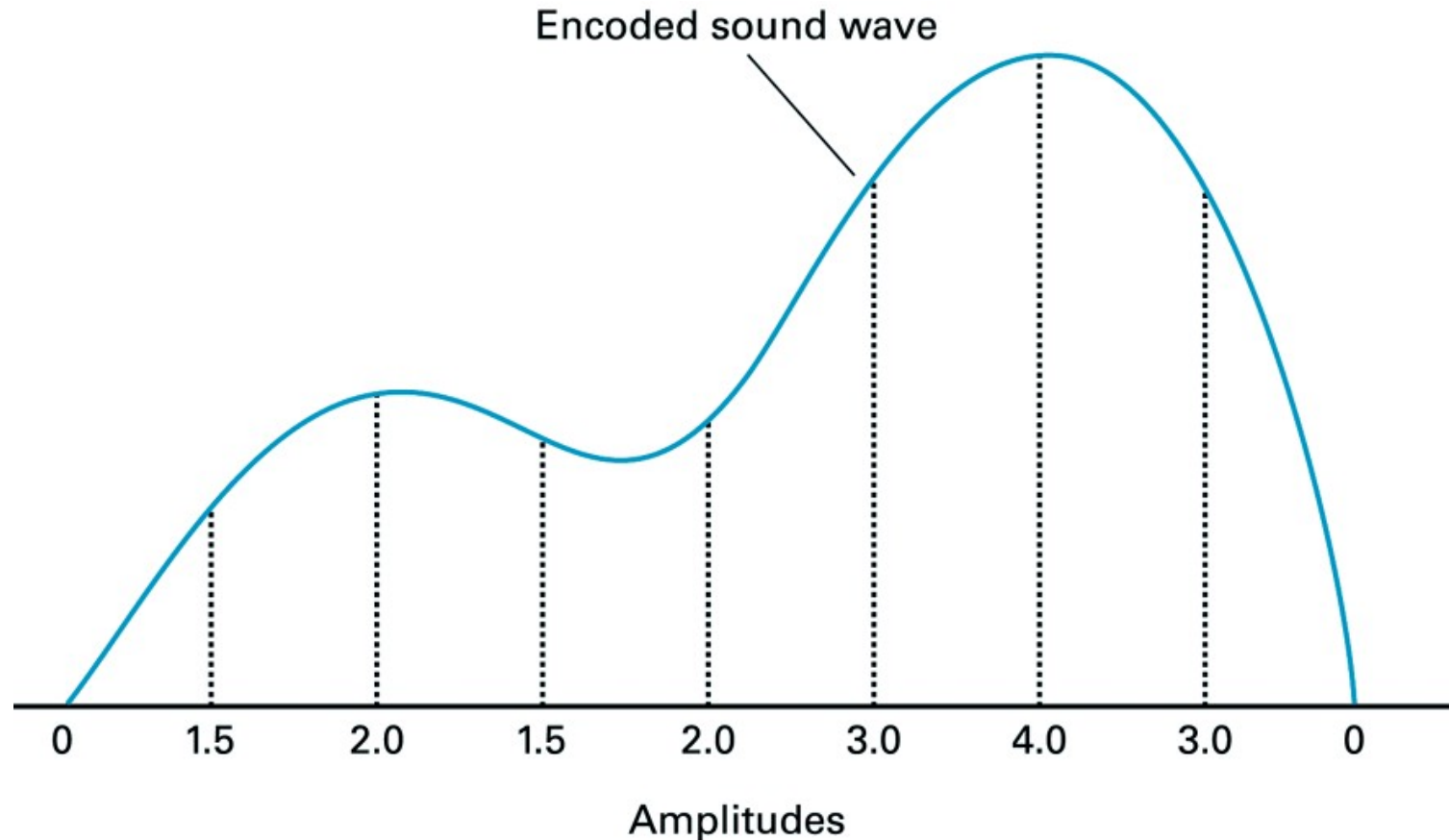
o

00101110

.

The bit patterns stand for characters under an agreed code—in this example, ASCII.

Digitization: a sound wave as numbers



Samples record the amplitude at selected times; the resulting sequence represents the signal.

From sampled values to stored bits

Stage	What changes
Sampling	Select measurement times
Quantization	Choose a representable value for each measurement
Encoding	Represent the selected values with bit patterns

Stored values approximate the original signal; the representation has finite precision.

A small quantization example

Measured value	Stored to nearest whole unit	Difference
1.2	1	-0.2
1.8	2	+0.2
2.4	2	-0.4

Finer representable levels can reduce rounding error, but require more information.

Binary addition

$$\begin{array}{r} 0 \\ + 0 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 1 \\ + 0 \\ \hline 1 \end{array}$$

$$\begin{array}{r} 0 \\ + 1 \\ \hline 1 \end{array}$$

$$\begin{array}{r} 1 \\ + 1 \\ \hline 10 \end{array}$$

1 + 1 produces 10: write 0 in the current position and carry 1 to the next.

Binary addition: 5 + 3

$$0101 + 0011 = 1000$$

Position	Calculation	Result bit / carry
1	$1 + 1$	0 / carry 1
2	$0 + 1 + \text{carry } 1$	0 / carry 1
4	$1 + 0 + \text{carry } 1$	0 / carry 1
8	$0 + 0 + \text{carry } 1$	1 / no carry

Unsigned and signed interpretations

4-bit pattern	Unsigned	Two's complement
0111	7	7
1000	8	-8
1111	15	-1

The same bits can have different meanings. The representation must be specified.

Two's complement representations

a. Using patterns of length three

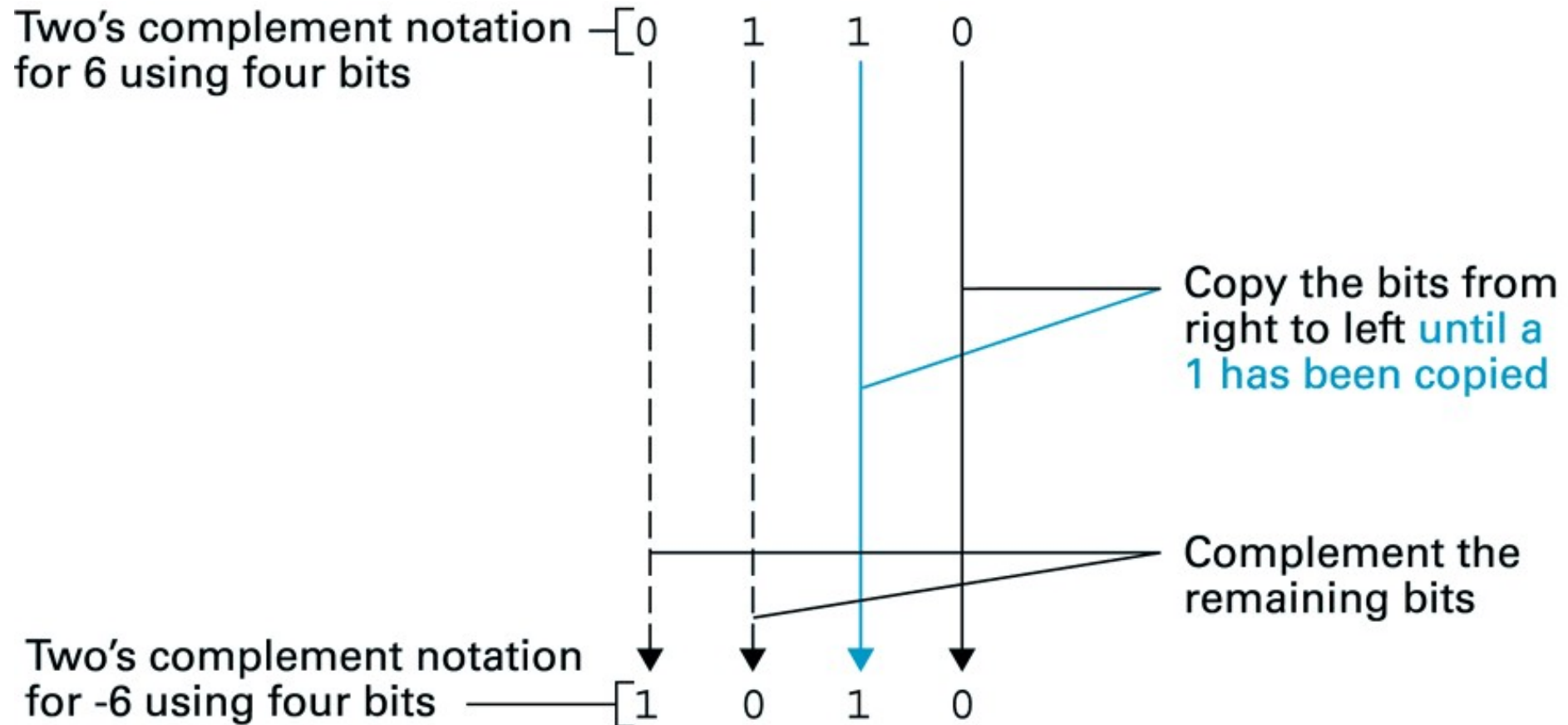
Bit pattern	Value represented
011	3
010	2
001	1
000	0
111	-1
110	-2
101	-3
100	-4

b. Using patterns of length four

Bit pattern	Value represented
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	-1
1110	-2
1101	-3
1100	-4
1011	-5
1010	-6
1001	-7
1000	-8

With 4 bits, the range is -8 through +7. With 3 bits, it is -4 through +3.

Representing -6 with four bits



Start with +6 = 0110. The resulting two's complement representation is 1010.

Another way to obtain -6

+6 in four bits: 0110

Invert the four bits: 1001

Add one: 1010

Check using the negative leading weight: $-8 + 2 = -6$.

Adding two's complement values

Problem in base ten		Problem in two's complement		Answer in base ten
$\begin{array}{r} 3 \\ + 2 \\ \hline \end{array}$	→	$\begin{array}{r} 0011 \\ + 0010 \\ \hline 0101 \end{array}$	→	5
$\begin{array}{r} -3 \\ + -2 \\ \hline \end{array}$	→	$\begin{array}{r} 1101 \\ + 1110 \\ \hline 1011 \end{array}$	→	-5
$\begin{array}{r} 7 \\ + -5 \\ \hline \end{array}$	→	$\begin{array}{r} 0111 \\ + 1011 \\ \hline 0010 \end{array}$	→	2

Use ordinary binary addition within the fixed width. Interpret the result as a signed value.

Fixed width and signed overflow

In 4-bit two's complement: $7 + 1$

$$0111 + 0001 = 1000$$

The true sum is +8, outside the range $-8 \dots +7$.

The pattern 1000 means -8 : signed overflow occurred.

A result bit pattern alone does not guarantee that the mathematical answer fits.

Signed-number practice

Interpret 1101 as a 4-bit two's complement number.

Represent -3 using four bits.

Does 4-bit two's complement addition $5 + 4$ fit in the available range?

Signed-number practice: answers

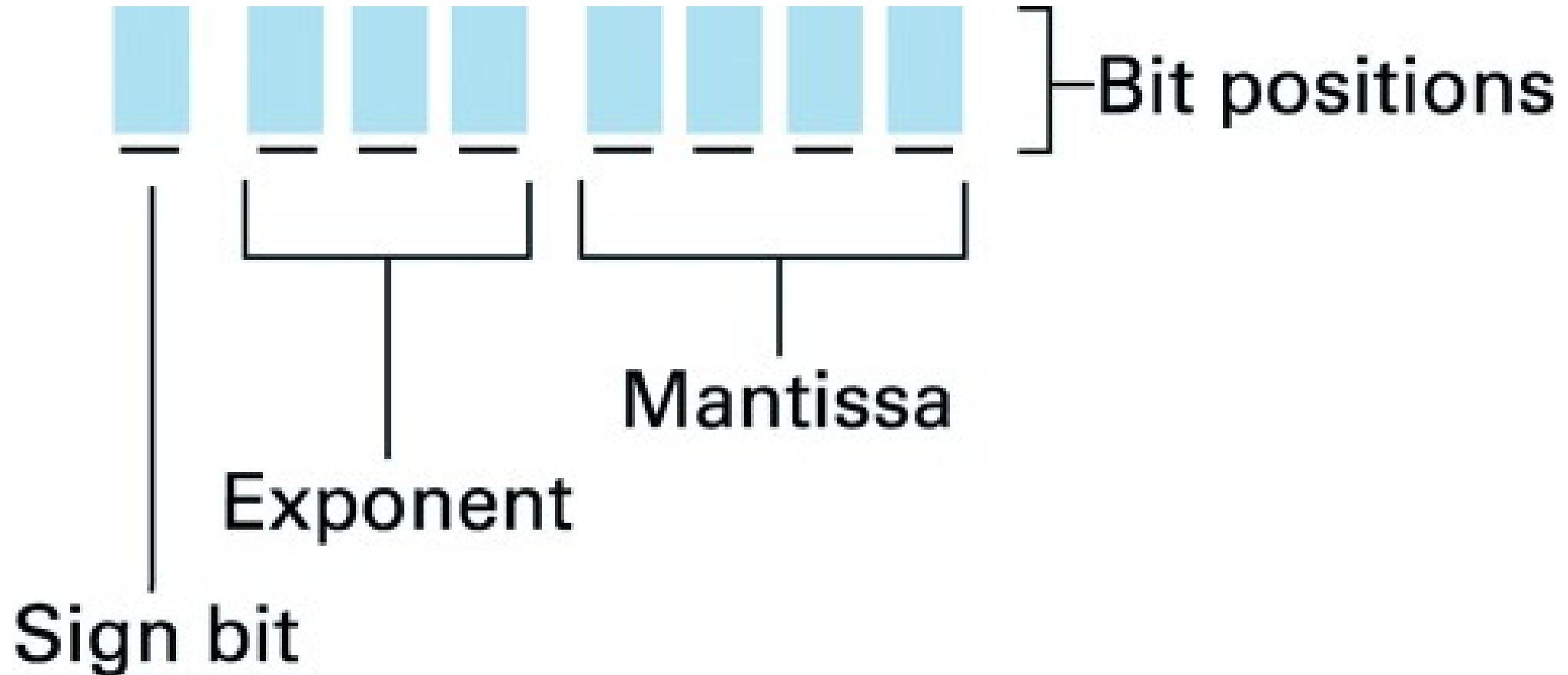
1101 represents $-8 + 4 + 1 = -3$.

$+3 = 0011$; invert to 1100; add 1 $\rightarrow 1101$.

$5 + 4 = 9$, which exceeds $+7$. The 4-bit pattern 1001 represents -7 , so overflow occurred.

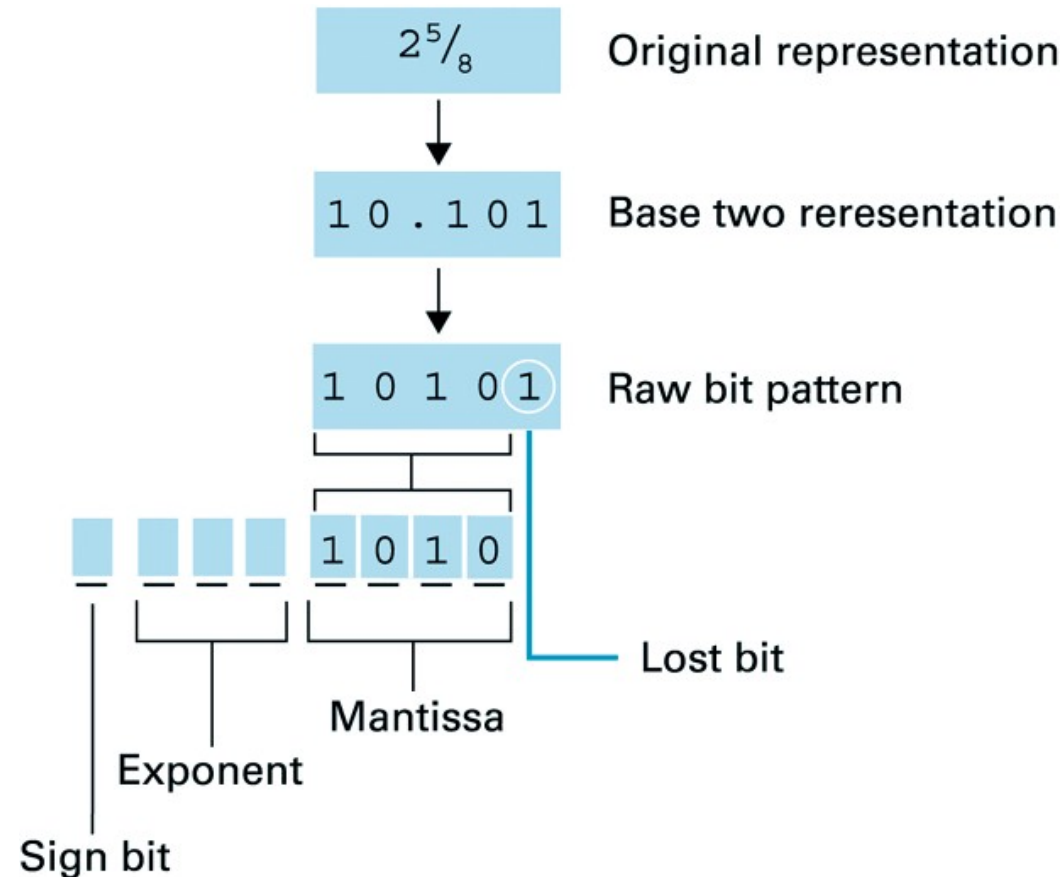


Floating point: fields with different roles



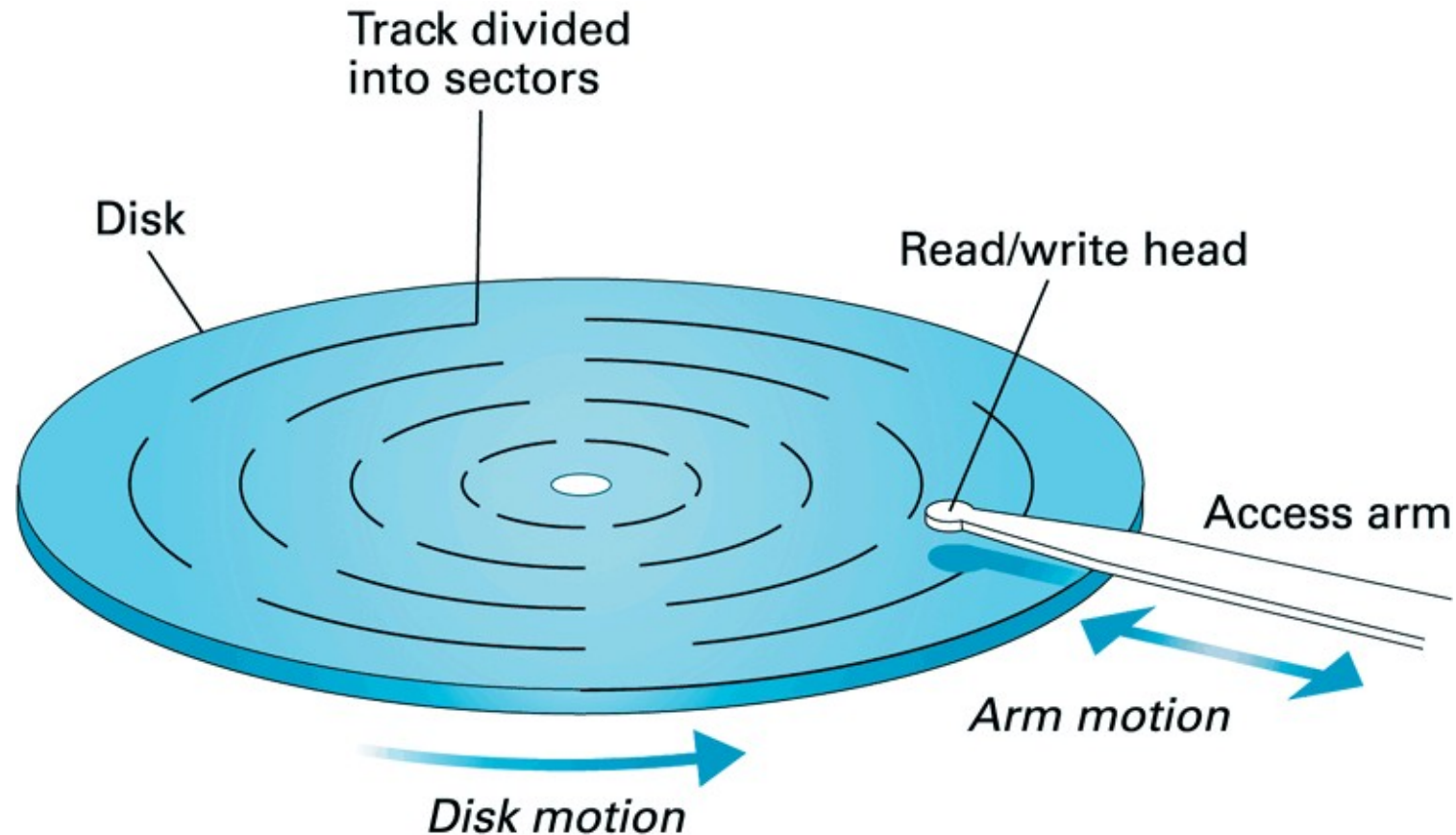
This textbook illustration separates sign, exponent and mantissa. It is a simplified format, not IEEE 754.

A simplified floating-point example



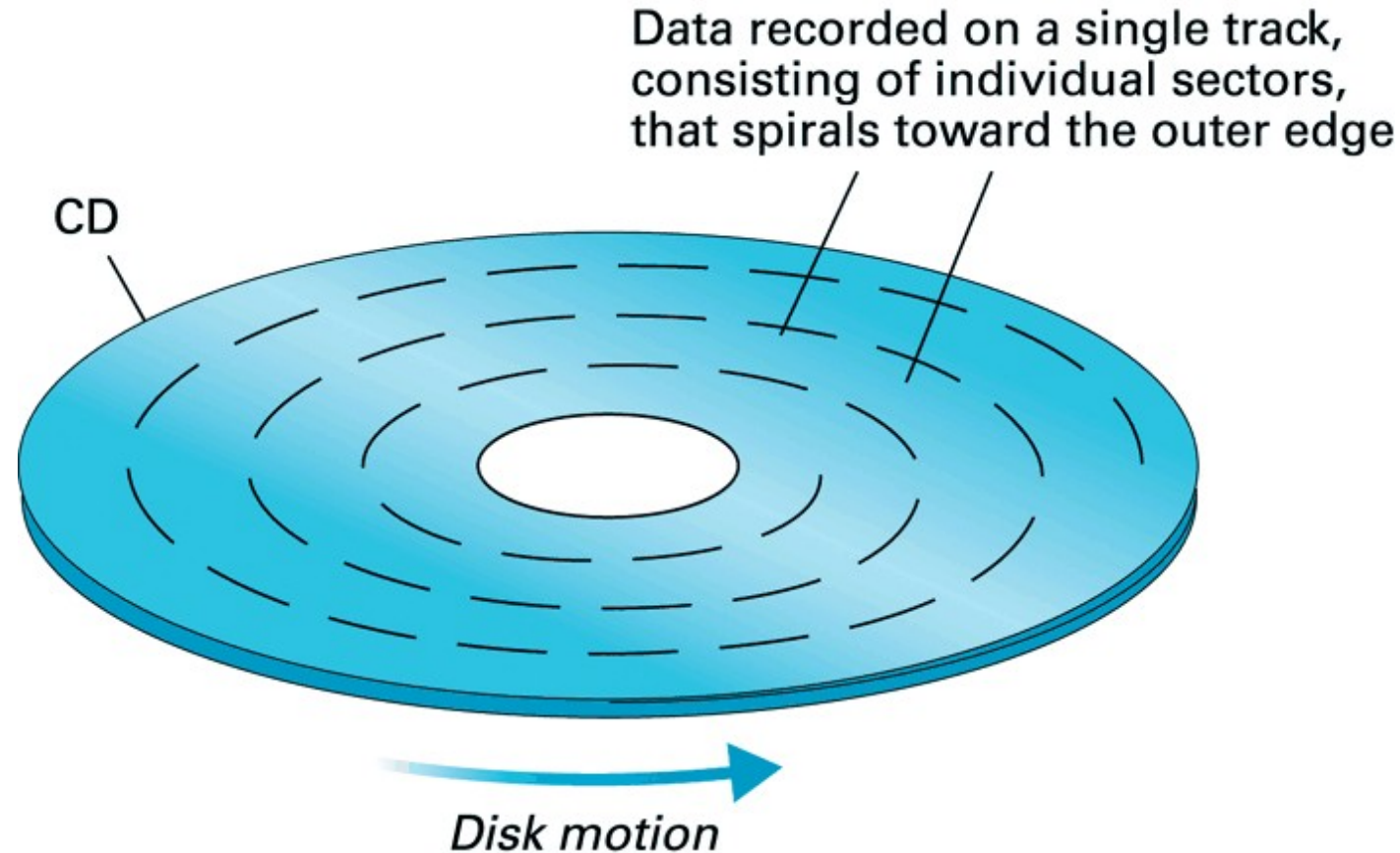
The value $2^5/8$ is 10.101 in binary. The small example format loses a low-order bit.

Storage example: magnetic disk



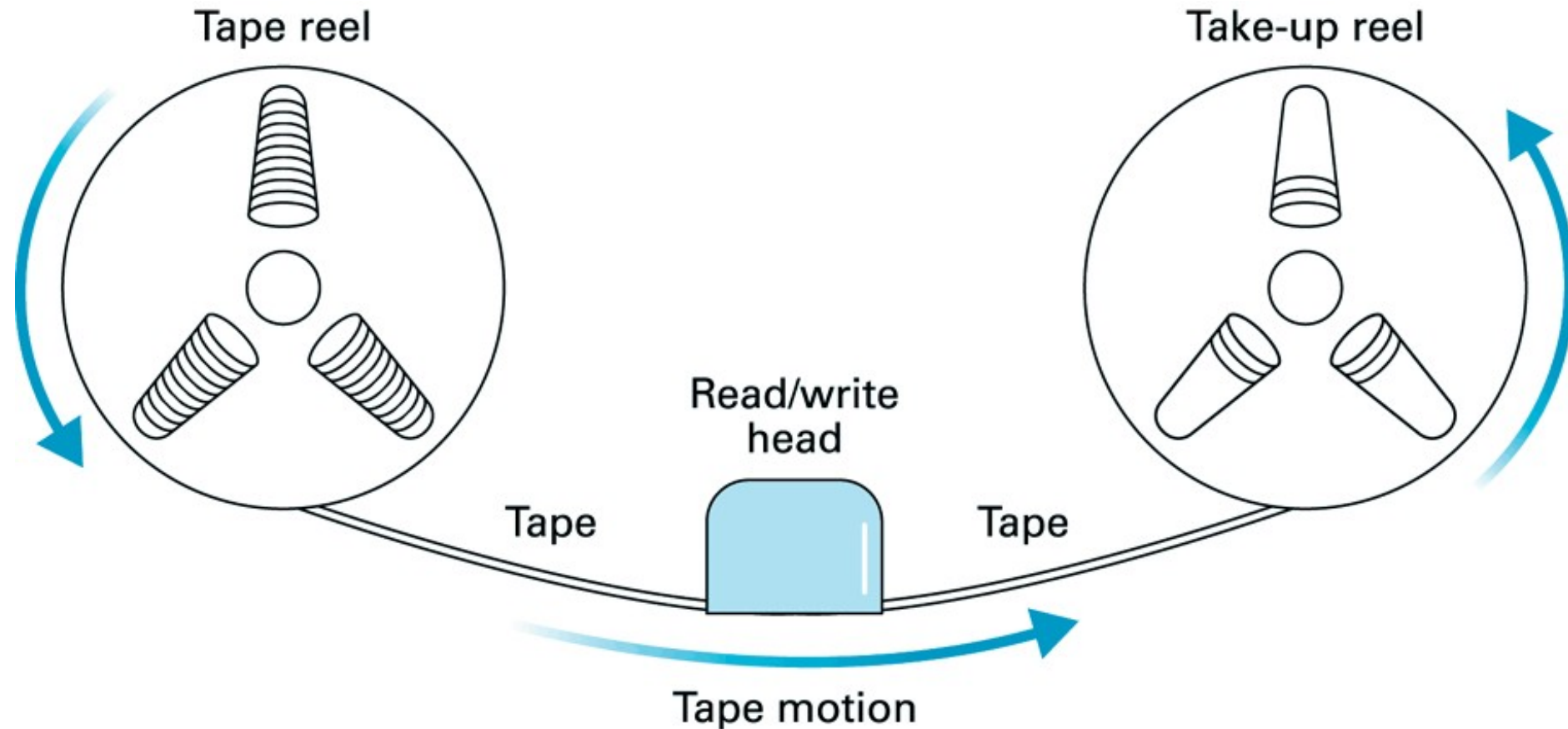
Tracks and sectors organize recorded data; a read/write head accesses the rotating surface.

Storage example: optical disc



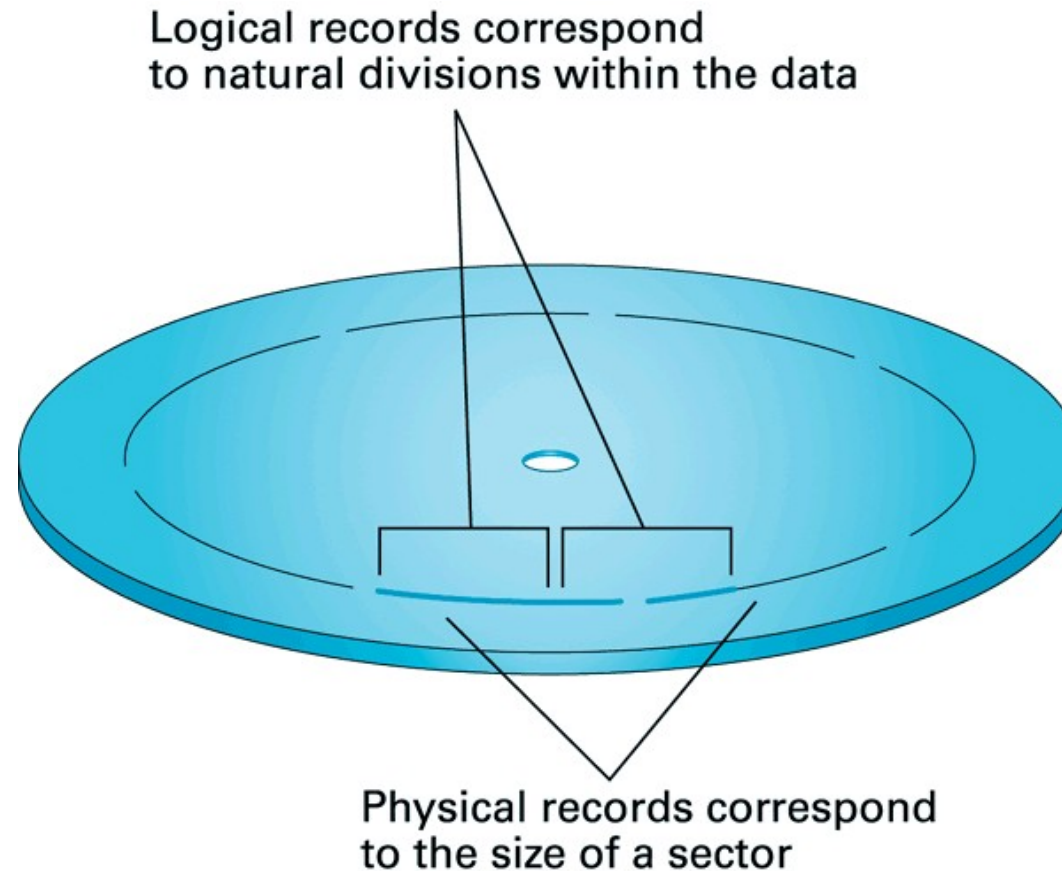
A spiral track carries the recorded information. This is an example of physical encoding.

Storage example: magnetic tape



Data is recorded along the tape. Reaching a later position requires moving through earlier tape.

Logical records and physical storage



A meaningful data record and a physical storage unit need not have the same boundaries.

The connections to retain

Physical devices implement logical operations and can retain state.

Bit patterns represent numbers, characters and sampled signals.

Width and interpretation determine range and precision.

A computer transforms represented information according to instructions.

Every student must buy an Arduino kit

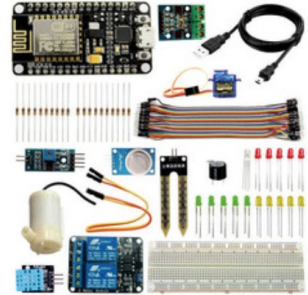


You must purchase your own kit.

Use an Arduino Uno or compatible clone.

A genuine board is also acceptable.

Starter kits are suitable

 YENİ R.E.X Evolution Serisi Super Star Transformers - 8 in 1 %27	 Arduino Proje Seti (Klon) (Kitaplı ve Videolu) %10	 Arduino Uno Başlangıç Seti (Klon) (Kitaplı ve Videolu)	 Arduino Süper Başlangıç Seti Uno Rev3 (Klon) (Kitaplı ve Videolu) %10
999,00 TL	431,85 TL	104,31 TL	258,17 TL
 YENİ %11	 %15		

Choose a modest kit. Historical examples shown here: prices and offers are obsolete.

An example kit configuration

Setleri > Arduino Uno Pro Başlangıç Seti (E-Kitap Hediye ve Videolu)

YENİ



Arduino Uno Pro Başlangıç Seti (E-Kitap Hediye ve Videolu)

Ürün Kodu : 21259

Marka / Menşei : Robotistan

★★★★★ : Yorumlar (1)

Stok Adedi : 10+

Fiyat : 479,82 TL + KDV

KDV Dahil Fiyat : 639,76 TL

İndirimli Fiyat : **575,78 TL** KDV Dahil **%10**

Adet:

- 1 +

SEPETE EKLE

Avnı Gün Karon Alışveriş İsteme Ekle

Board, USB data cable, breadboard, wires and basic components. Historical screenshot—not a current offer.

Computer and Arduino configurations



Computer + USB + Arduino
Develop and upload a program.
Exchange data with a running program.

Arduino running independently
Runs its uploaded program with suitable power.
Reads inputs and controls outputs.

Obtain your personal kit

Every student must buy their own Arduino kit.

A simple starter kit with a board and connection materials is enough.

Check the board model, included USB cable and kit contents before ordering.

Ask the instructor or TAs if you are unsure about a kit. Purchase timing will be announced.