

WHAT IS INSIDE YOUR PHONE OR COMPUTER?

A public, self-guided, all-level lesson about the parts that make everyday technology work.

From a photo you take → the chips, memory, sensors, power, and software behind it

EXPLORER

BUILDER

ENGINEER / RESEARCHER

WHAT YOU NEED

Paper + a pencil. A phone is optional. No device teardown required.

WHAT YOU WILL MAKE

A labeled device blueprint and a system story explaining one experience.

Your goal: understand one more component, explain an experience, and discover what to ask next.

How to use this public deck

Start anywhere. Complete the paper activity even if you are watching alone.

EXPLORER

Name a component, explain its purpose in everyday words, and trace a simple experience.

BUILDER

Draw a system, identify a constraint, and propose one safe improvement or test.

ENGINEER / RESEARCHER

Reason about interfaces, memory hierarchy, throughput, power, thermal limits, and evidence.

STOP / PAUSE / DRAW whenever a prompt appears. You do not need a computer or previous technical knowledge. A question mark is a useful result.

A successful lesson produces: an answer, an artifact, a next step, or a better question.

Four weeks, one connected story

We increase the resolution of the SAME everyday technology experience.

WEEK 1

Technology is a system, not a screen.

WEEK 2

A tap travels from input to feedback.

WEEK 3

A good experience depends on performance.

WEEK 4

Open the device: what does each part do?

TODAY: locate the physical parts and show how they cooperate with software.

Start with a human experience: take a photo

You tap the shutter. A moment appears on your screen and remains after restart.

WHAT YOU NOTICE

1. A preview appears.
2. You tap the shutter.
3. The image is processed.
4. You find it later in your gallery.

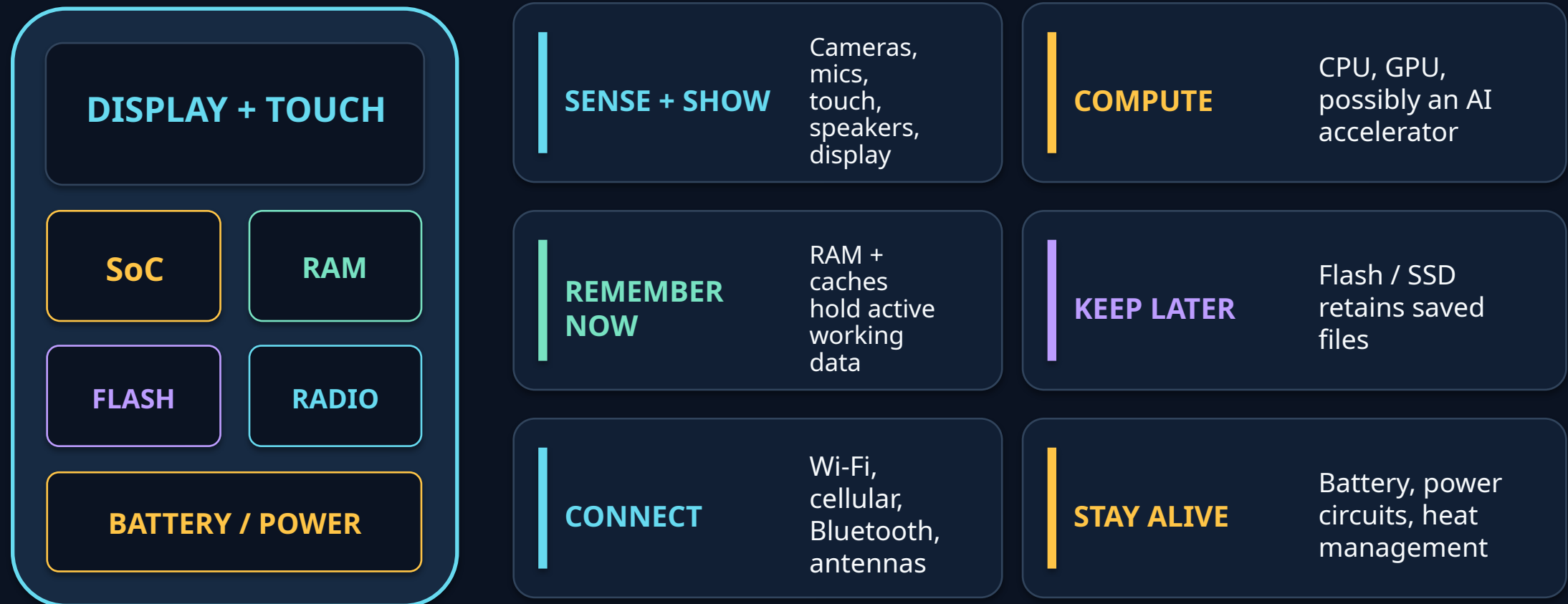
WHAT MUST EXIST

Camera sensor • processor • working memory • persistent storage • screen • power • operating system and camera app

Guiding question: which part detected light, which processed it, and which kept the result?

Conceptual exploded view: inside a phone

This is a teaching map. Real layouts differ by model; some functions share chips.



CPU: the general-purpose instruction worker

The CPU executes program instructions and coordinates many kinds of work.

EVERYDAY LANGUAGE

The CPU follows instructions, makes decisions, runs parts of the OS, and supports apps.

MORE PRECISELY

Programs become machine instructions. CPU cores execute operations using registers, caches, and main memory.

COMMON TRAP: “More GHz always means faster.” Actual results depend on workload, architecture, memory, cooling, and software.

GPU, AI accelerators, and different kinds of work

Not every computation belongs on the CPU.

CPU

Flexible general-purpose compute: logic, OS work, scheduling, and many application tasks.

GPU

Highly parallel workloads such as drawing graphics, rendering games, and some computations.

NPU / AI ACCELERATOR

Specialized math for certain machine-learning workloads, when supported.

Real phones often integrate several of these functions into one system-on-chip (SoC). Not every device has a separate NPU.

RAM versus storage: the distinction that matters

Working now is different from keeping a file after power is off.

RAM = WORKING SPACE

Keeps active data easy to access while apps are running. Typical RAM is volatile: its contents do not persist when power is removed.

STORAGE = KEEP IT

Flash storage in phones, SSDs in many computers, or hard drives in some PCs hold files and installed software beyond a shutdown.

Examples: photo being processed = RAM | photo saved in gallery = storage

Useful analogies have limits

A mental model can help you start without pretending the machine works exactly like a person.

DESK / SHELF ANALOGY

RAM is like working space on a desk. Storage is like a shelf where you keep materials afterward.

IMPORTANT CORRECTION

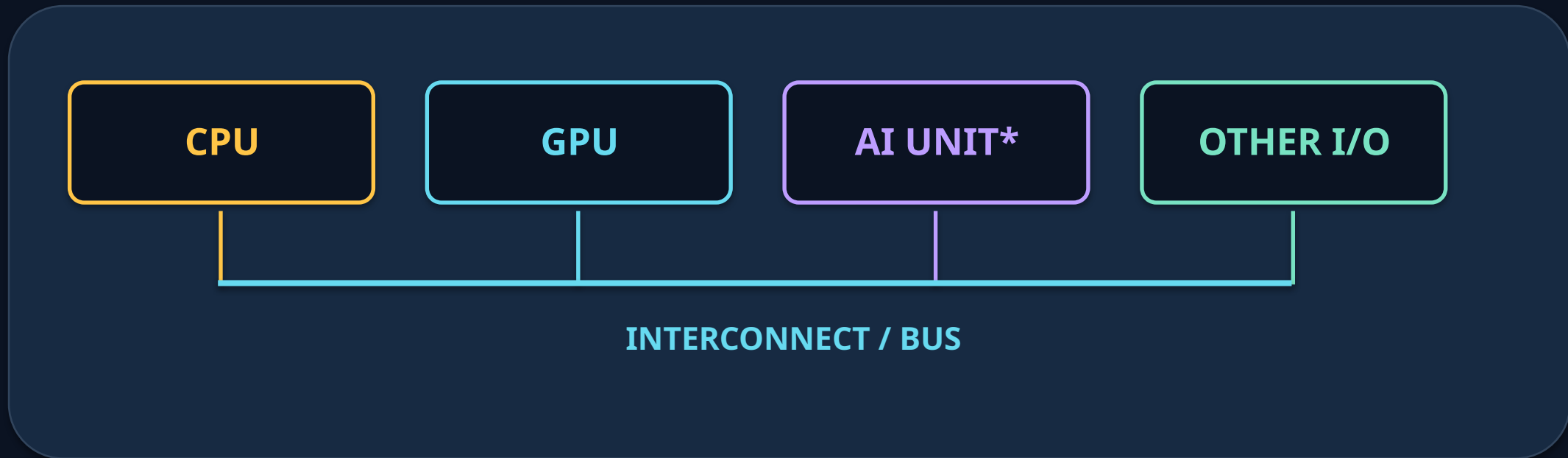
RAM is electronic circuitry, not a literal desk. Files may be cached, and computers manage several memory levels.

ONE QUESTION TO CHECK

If you save a photo and restart the phone, why is the photo still there? Explain without assuming RAM kept it.

SoC, motherboard, and interconnects

A “computer” is a set of cooperating components, not one magic chip.



PHONE: often integrated SoC with many functions on one chip. PC: components may be integrated, separate, or upgradeable. *If present.

Phone versus desktop/laptop

The major jobs are similar. Packaging, cooling, power budgets, and repairability differ.

SMARTPHONE

High integration; limited space and battery; radios, cameras, sensors, display, and SoC packed tightly. Most parts not designed for casual upgrades.

PC / LAPTOP

Often larger thermal and power budgets; display and input may be separate; RAM/storage/graphics can be more or less replaceable depending on model.

Don't assume every phone or computer has the same internal layout.

The operating system is software, not a chip

The OS helps programs use hardware safely and consistently.

APPLICATION

Camera, browser, music player

OS + DRIVERS

Services, permissions, memory and device access

HARDWARE

CPU, sensors, storage, radios, display

Android, Windows, macOS, iOS, and Linux differ in implementation but all manage software/hardware interaction.

Inputs and outputs: how the world meets the device

“Inside the computer” also includes components that sense and present information.

INPUTS: SENSE

Touch • camera • microphone •
accelerometer • keyboard • mouse •
buttons • motion sensors

OUTPUTS: PRESENT

Display • speakers • vibration motor •
indicator lights • external display • other
actuators

Try it: name two inputs and two outputs of the phone or computer you use most.

Camera: light becomes data

A camera is a system, not just a lens.



SOFTWARE MATTERS TOO

The camera app and OS coordinate permissions, preview, capture, processing, and saving.

Note: processing and buffering order differs by device; this is a conceptual path, not a precise timing trace.

How your device connects beyond itself

Hardware contains radios and antennas; remote services usually live elsewhere.

WI-FI

Connects through a compatible local wireless network and its infrastructure.

CELLULAR

Uses mobile radio infrastructure and service, where available.

BLUETOOTH / GNSS

Bluetooth connects nearby devices; GNSS receives satellite signals for positioning.

A strong radio signal is not proof that every app, server, or network dependency is healthy.

The hidden constraint: power and heat

Higher performance can draw more power and create more heat.

POWER SYSTEM

Battery / adapter supplies energy. Power-management circuitry distributes and regulates it.

THERMAL SYSTEM

Heat moves through the enclosure, cooling structures, fans (on some PCs), and the environment.

Safety: do not open, puncture, bend, heat, or remove device batteries for this lesson.

Follow one photo through the whole system

Now connect all the pieces: it is not “just the camera.”

1 YOU

choose the moment

2 CAMERA

senses light

3 SoC / SOFTWARE

processes pixels

4 RAM

holds working data

5 STORAGE

keeps the file

6 DISPLAY

shows the result

The exact pipeline differs; the useful question is what each component contributes to the outcome.

Hardware problems can look like software problems

A symptom points to possibilities, not a guaranteed diagnosis.

"My app freezes."

RAM pressure, storage, CPU load, OS/app bug, or network wait

"My video call breaks up."

Radio/network conditions, app, microphone, server, or device load

"My phone gets hot."

Demanding workload, environment, charging, or fault

"My photos will not save."

Storage capacity, permission, app, or file system issue

Avoid confident diagnoses from one observation. Test safe, distinguishable hypotheses.

Great design involves tradeoffs

There is no single “best” device for every human need.

BATTERY vs SPEED

Fast chips may require more power; efficient designs try to deliver useful work per watt.

SIZE vs COOLING

Thin devices have less room to move heat; sustained workloads can behave differently.

COST vs ACCESS

More hardware costs money; the right design fits real needs, budget, repair, and longevity.

Builder question: for a low-cost school device, what experience should work reliably FIRST?

YOUR ACTIVITY: draw an imaginary device

Paper + pencil is enough. Do not disassemble real electronics.

DRAW A LARGE RECTANGLE AND LABEL IT

- A. Human goal (top)
- B. CPU / processing
- C. RAM / working memory
- D. Flash or SSD / saved files
- E. Screen / input
- F. Connection / radio
- G. Battery or wall power
- H. ONE arrow explaining how data moves

SUCCESS CHECK

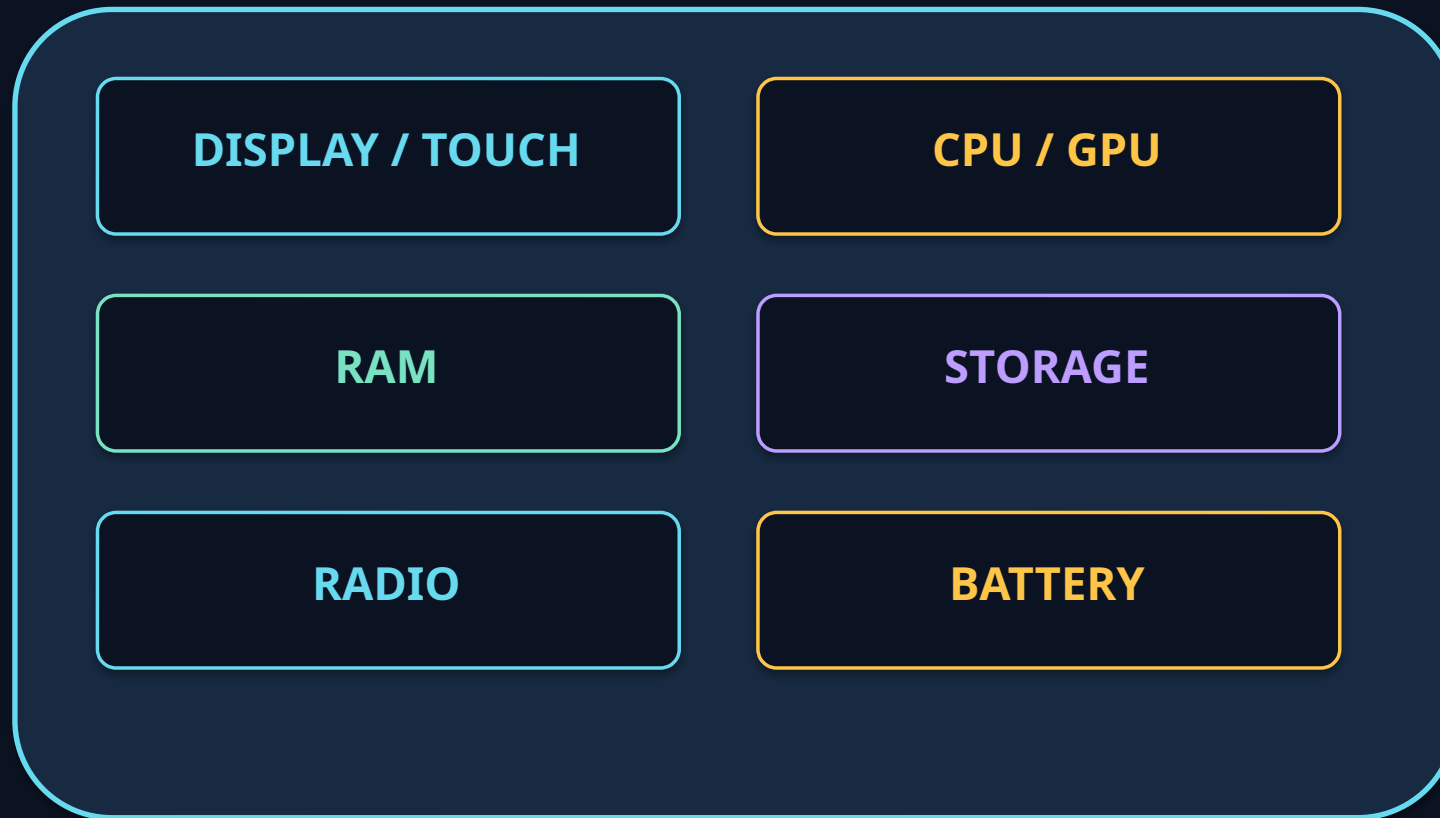
Can someone else look at your drawing and say what each part contributes?

Unknown answer? Write "?" and continue.

PAUSE HERE and draw for 5 minutes before swiping.

Worked example: an imaginary study device

Compare this to your own diagram. It does not need to match.



EXPLAIN A LESSON

A learner taps an assignment. The app and OS run instructions on the CPU, use RAM for active data, load saved materials, draw the screen, and may use Wi-Fi to reach the class portal.

Optional phone path: inspect safely

No paid apps or developer settings required; do not share screenshots of private data.

1. DEVICE INFORMATION

Find the operating system/version and model using Settings/About (wording varies). Record only the component concepts.

2. STORAGE

Find the storage usage overview. Why does saving a video use storage, not just RAM?

3. BATTERY

Look for battery usage/health summaries if your device provides them. What costs energy?

Do not post serial numbers, device IDs, accounts, photos, passwords, or personal usage logs.

Builder challenge: design for an actual person

You have a tight budget. Choose what to prioritize and explain why.

Choose ONE: student who studies offline | gamer | remote worker | older adult learning basics

DRAW YOUR PRIORITIES

Which two subsystems matter most? Is battery life more important than maximum speed? How much storage is needed?

DEFEND ONE TRADEOFF

What gets less budget? What failure would harm the user most? How would you test your choice?

Successful artifact: labeled hardware diagram + 3 sentences explaining your design decision.

Engineer / Researcher extension

Turn the diagram into testable hypotheses, not unsupported performance claims.

MEMORY HIERARCHY What changes when data is served from cache, RAM, or persistent storage?

INTERFACES Where do signals, interrupts, drivers, and software boundaries appear?

POWER / THERMALS How might sustained load differ from a 2-second burst?

EVIDENCE What would you measure before blaming the CPU or adding more RAM?

Check yourself: four questions

Try answering from memory. The next slide provides explanations.

1. A saved photo survives restart. RAM or persistent storage?

2. Is the operating system a physical chip?

3. Does fast Wi-Fi alone guarantee smooth video calls?

4. Which parts turn a camera scene into a saved image?

PAUSE: write your answers before moving on.

Check yourself: explanations

Knowing WHY matters more than remembering a single word.

1. PERSISTENT STORAGE

RAM supports active work; saved photos reside in nonvolatile storage.

2. NO

The OS is software that coordinates applications and hardware.

3. NO

Server health, delay, jitter, software, and device resources also matter.

4. A CHAIN

Sensor + processing + working memory + software + storage + display.

Your next step: choose a deeper path

You can keep learning without buying hardware or opening your device.

EXPLORER

Teach someone RAM vs storage with a real photo example. Ask them to explain it back.

BUILDER

Pick a device need; refine your blueprint into inputs, compute, memory, connection, and power.

ENGINEER

List the interfaces and experiments needed to justify your device tradeoffs.

NEXT THURSDAY / WEEK 5: How Does the Internet Get From Here to There?

Technology is for everyone: a successful question today can become a project tomorrow.

Continue learning and share your blueprint

Public companion resources; no software purchase or physical teardown is required.

gunnchOS3k / project ecosystem

<https://gunnchos.com>

WAIKE learning pathways

<https://github.com/gunnchOS3k/waike-research-ops>

The Technology Landscape

<https://github.com/gunnchOS3k/gunnchos-technology-landscape>

Intel: CPU versus RAM

<https://www.intel.com/content/www/us/en/learn/what-is-ram-vs-processor.html>

Android architecture (technical extension)

<https://developer.android.com/guide/platform/index.html>

Apple battery and performance

<https://support.apple.com/en-us/101575>

Public learning resource. Diagrams illustrate concepts; they do not show a specific manufacturer's internal design.