SNS COLLEGE OF TECHNOLOGY

An Autonomous Institution

Coimbatore-35

Department of Computer Science and Engineering

23CST206–OPERATING SYSTEMS AND VIRTUALIZATION

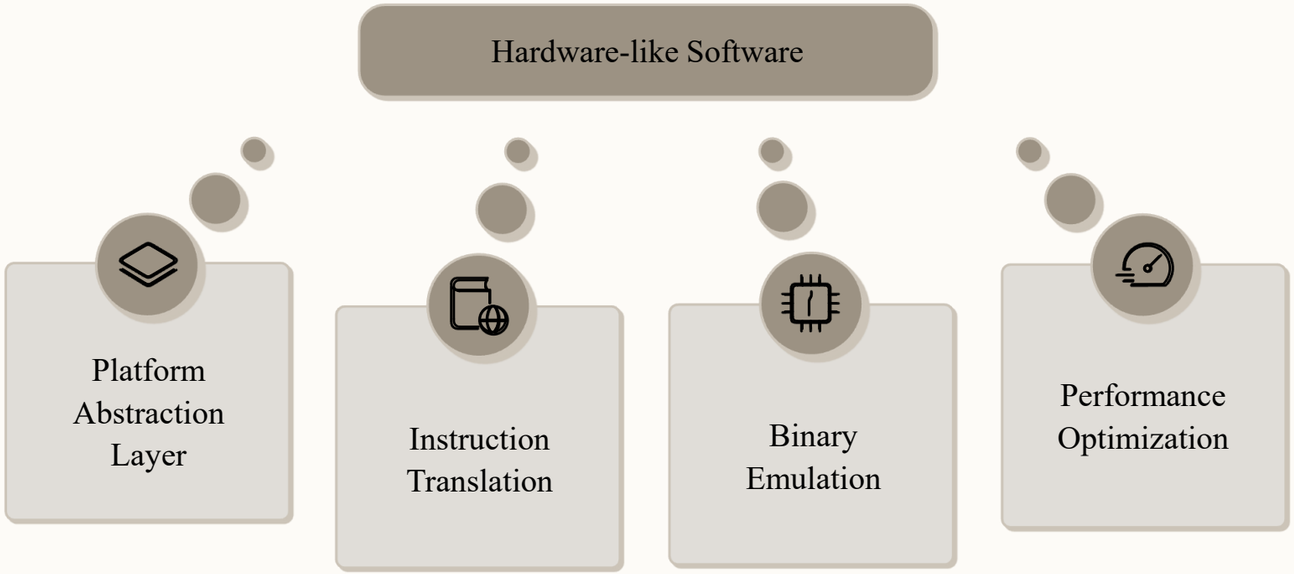
B.E- CSE /IV SEMESTER

UNIT – IV VIRTUALIZATION

Topic 2:Hardware Emulation

Hardware Emulation

Software that behaves like hardware, enabling programs designed for one platform to run on different hardware systems.



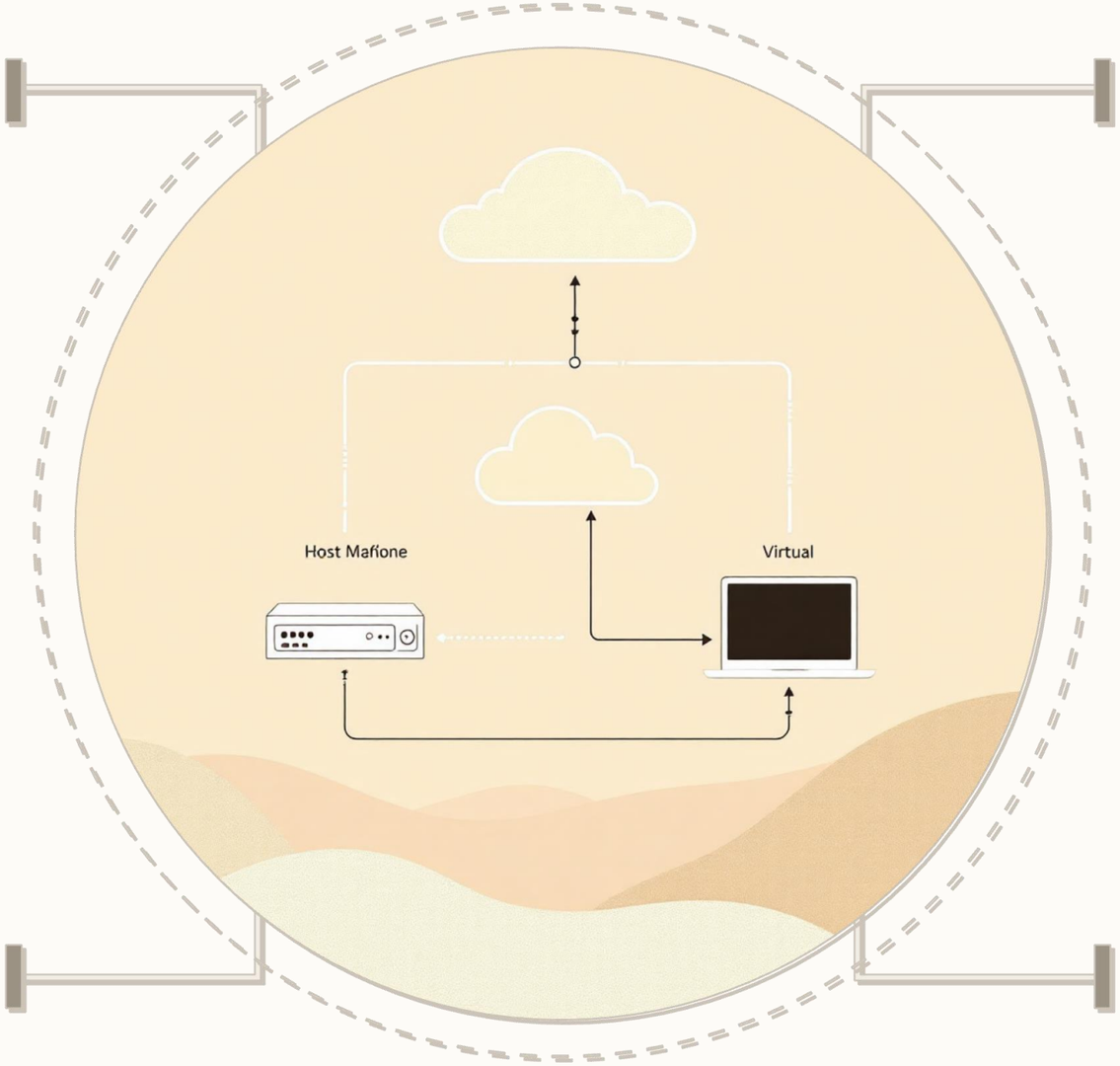
What is Hardware Emulation?

Host System

Runs emulation software

Emulation Layer

Translates hardware behavior



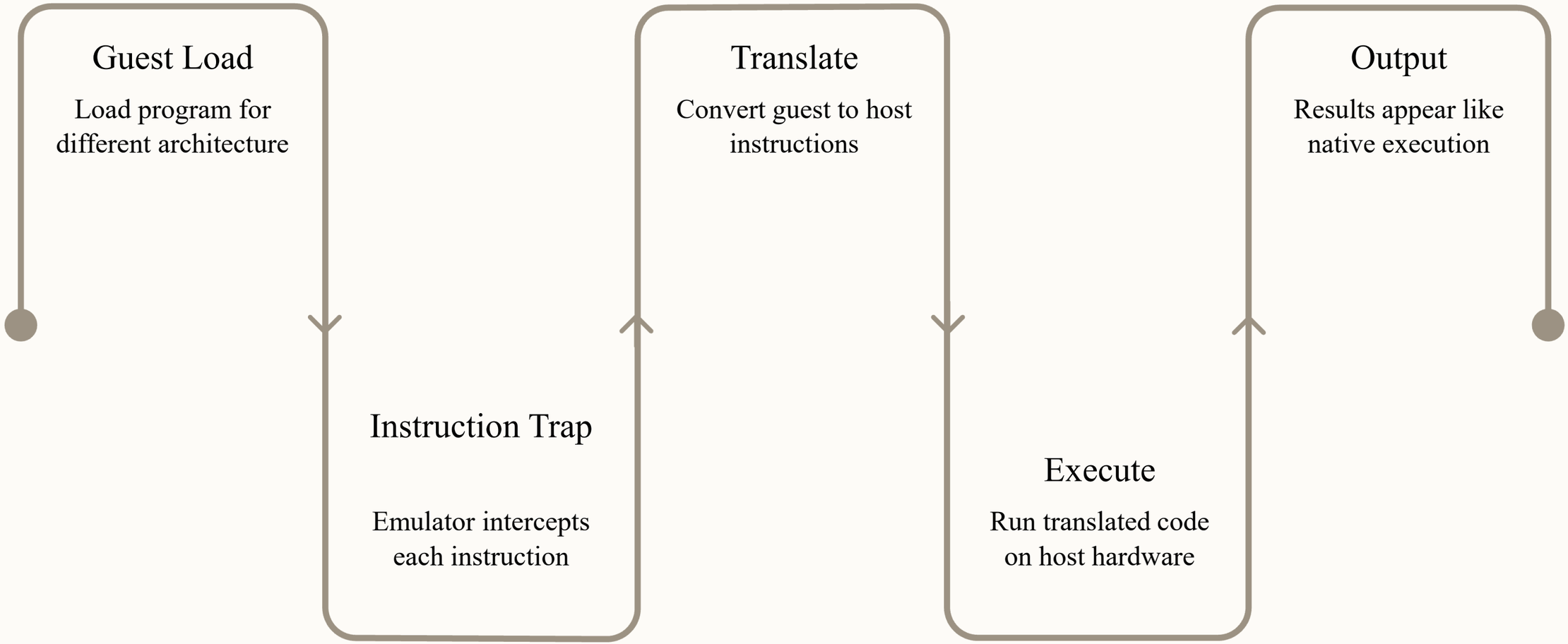
Guest System

Sees virtual hardware

Applications

Run unchanged on guest

How Hardware Emulation Works



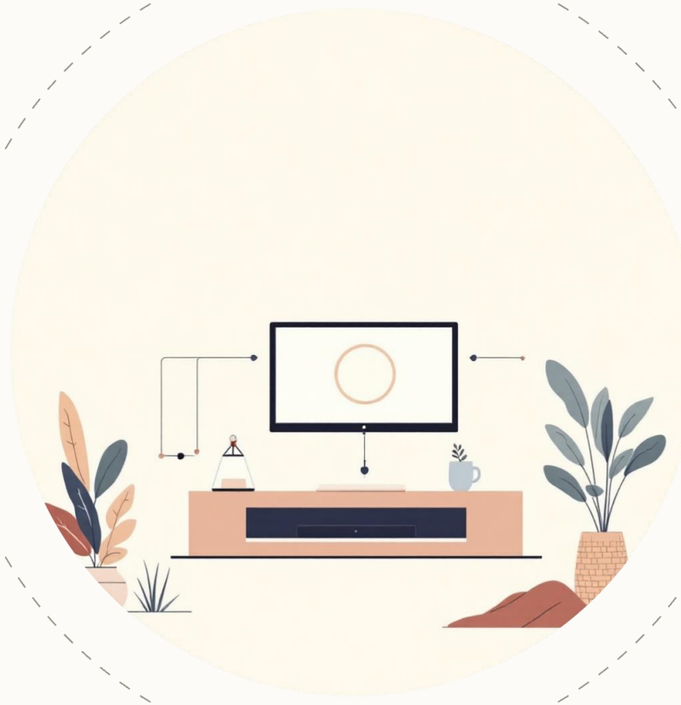
Key Components

Host System

Physical machine with CPU, memory, I/O

Emulator Software

Intercepts and translates guest instructions



CPU Emulation

Simulates guest CPU instruction set

Memory Management

Virtual memory keeps guest isolated

Types of Emulators

CPU Emulators

Game Console Emulators

OS Emulators

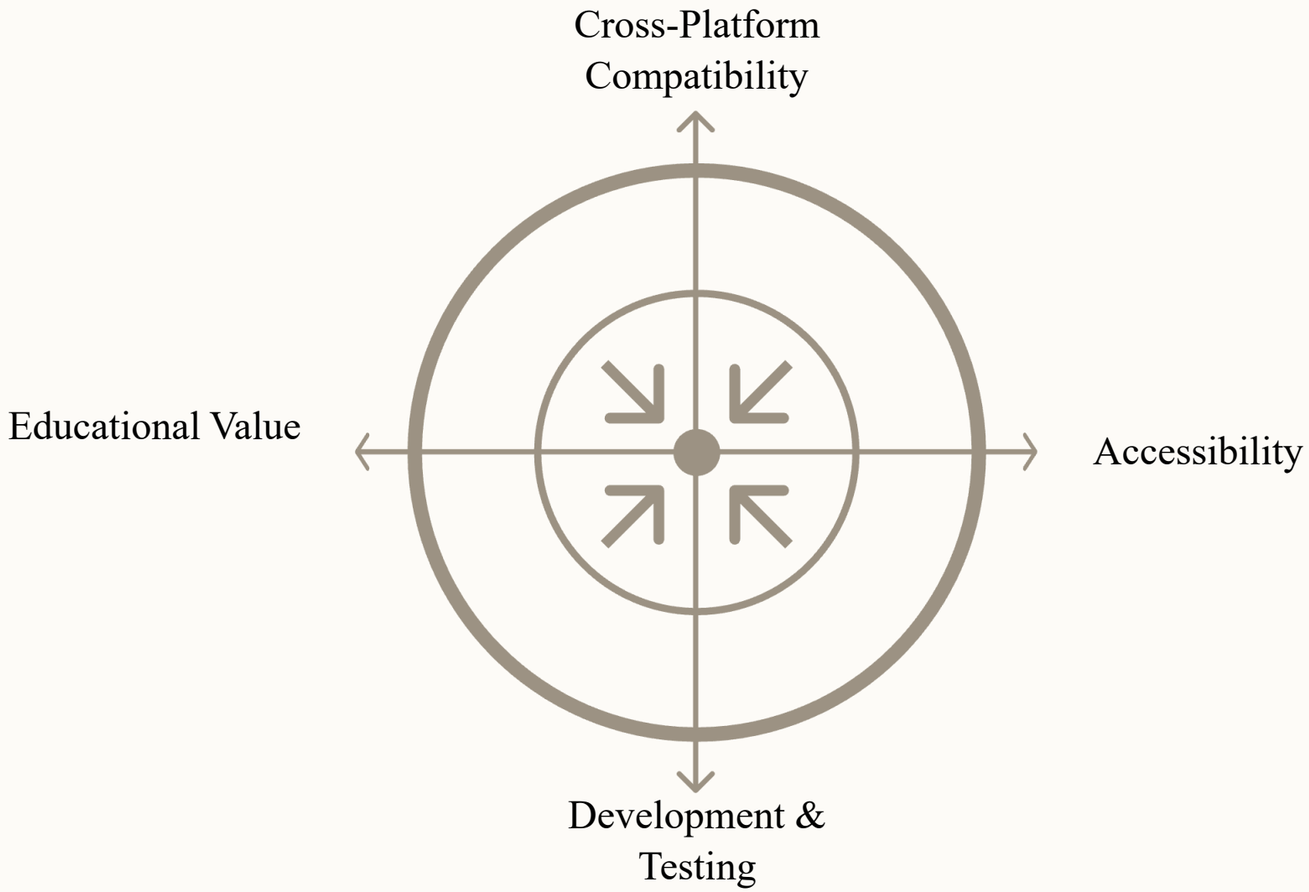
Network Emulators

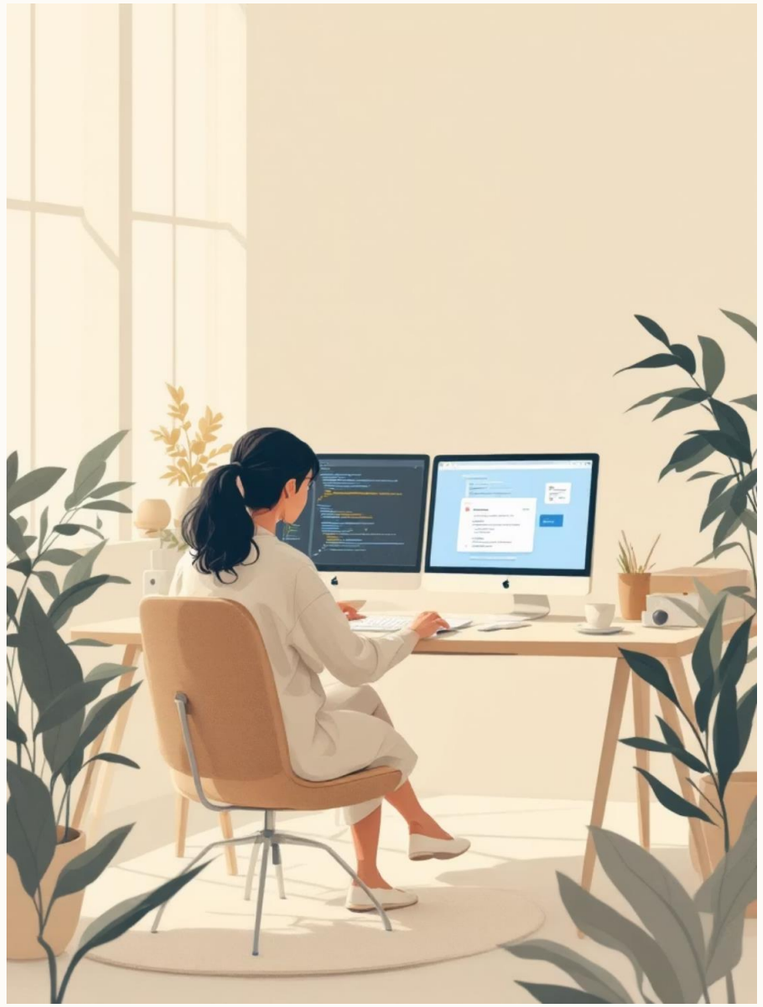
Application-Specific

Applications of Emulation

- Gaming
- Software Development
- Legacy Support
- Education & Training

Advantages


Cross-Platform Compatibility
Educational Value
Accessibility
Development & Testing



Disadvantages

Performance Overhead

User Expertise

Accuracy Challenges

Configuration Complexity

Popular Emulator Examples

QEMU

Emulates different CPU architectures for cross-platform development and testing

Bochs

x86 hardware emulator for running legacy operating systems and software

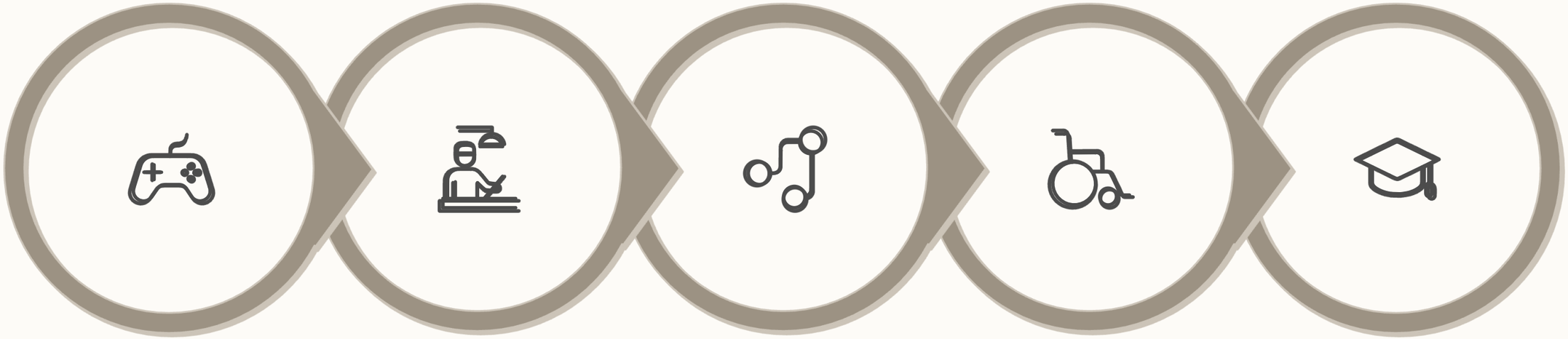
DOSBox

Emulates old DOS hardware for running classic games and legacy software

Android Emulator

Emulates mobile hardware for Android app development and testing

Real-World Use Cases



Game Preservation

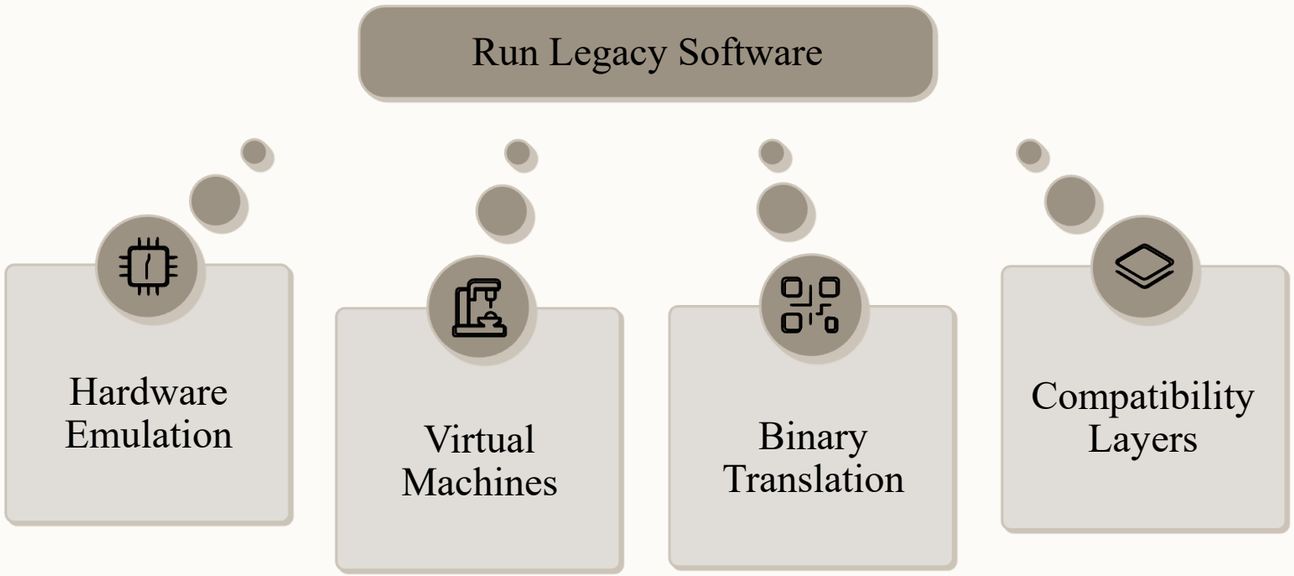
OS Development

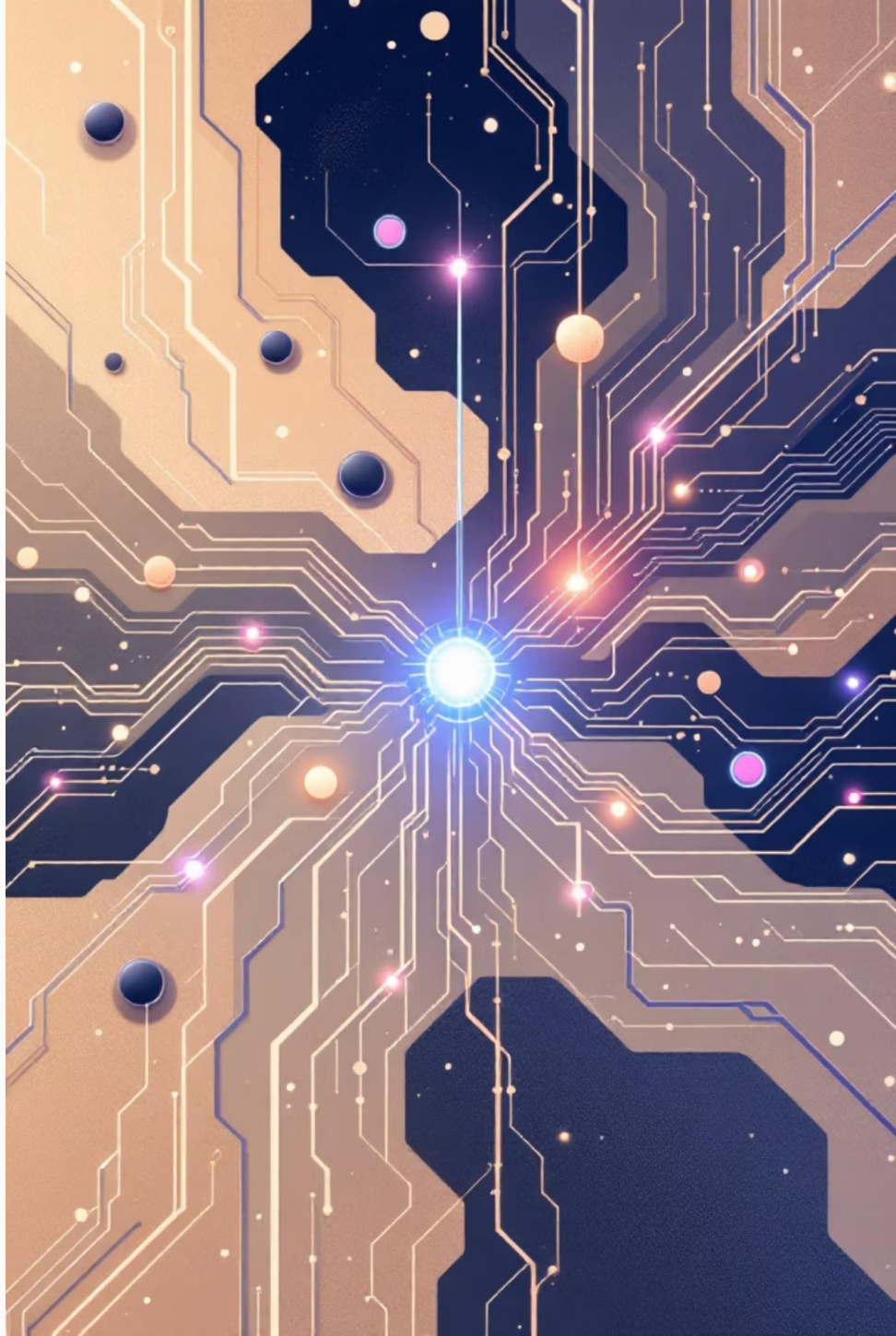
Embedded Testing

Mobile App Testing

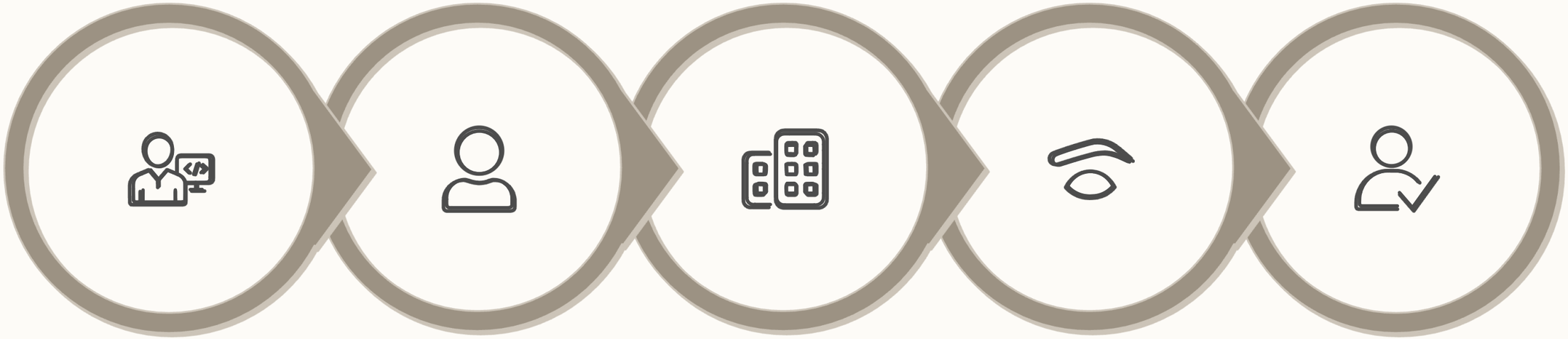
Academic Labs

Design Thinking Approach





Empathize



Developers

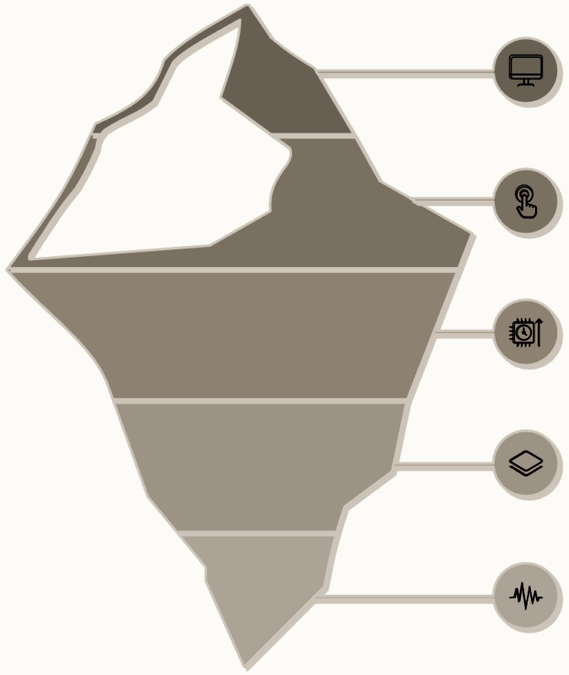
Users

Institutions

Cross-Platform Tests

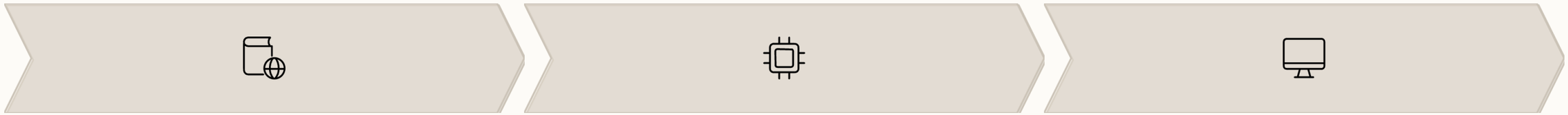
Legacy Compatibility

Define



- Visible Output
- User Interfaces
- Emulation Engine
- Hardware Abstraction
- Timing & Electrical Models

Ideate



Instruction Translation

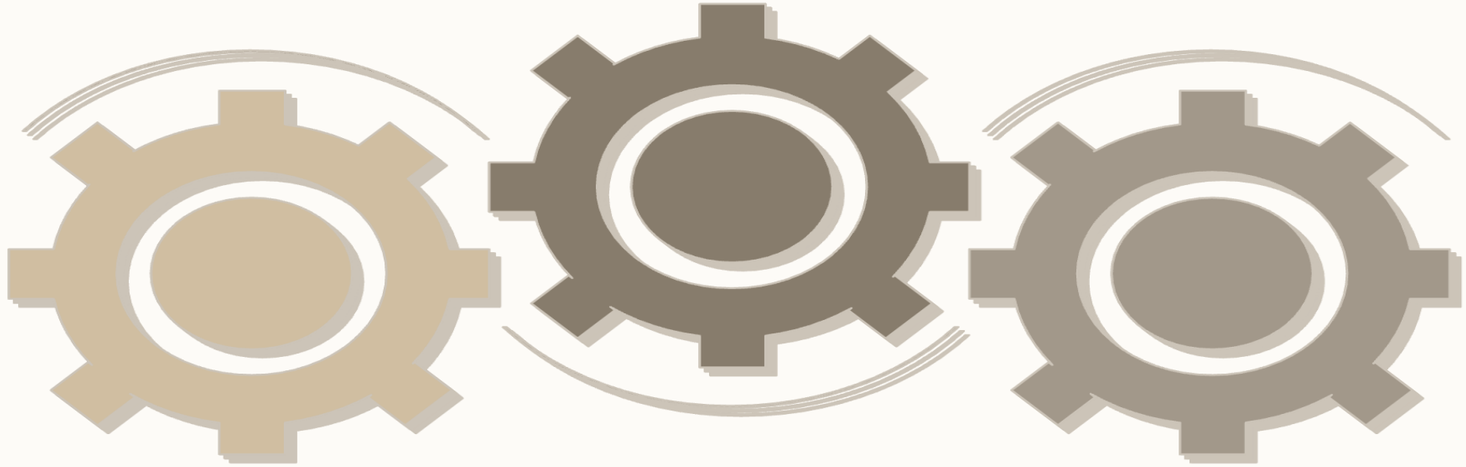
Emulate CPU & I/O

Full System Emulators

Prototype

Load Guest OS

Boot the operating system within the emulated hardware.



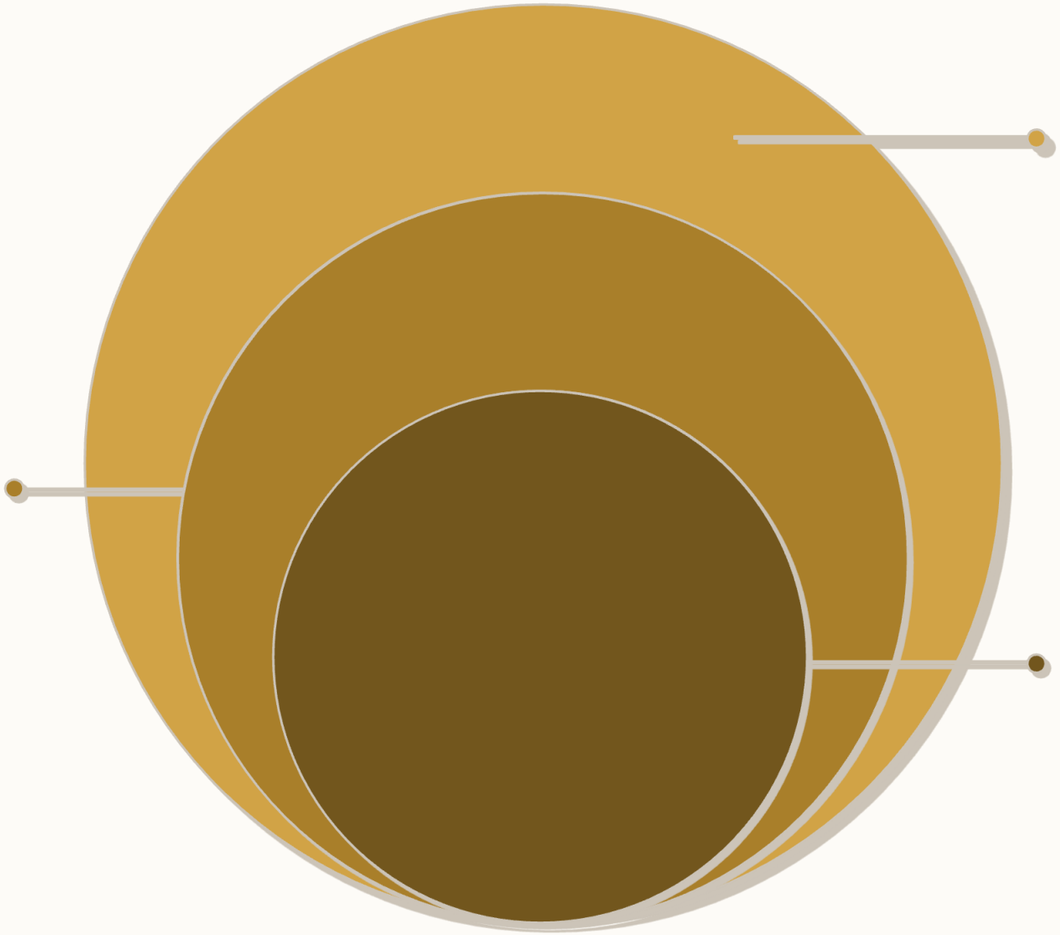
Install Emulator

Set up the virtual environment for the guest OS.

Translate Instructions

Convert guest commands for the host machine to execute.

Test



Check Compatibility

Confirm seamless integration with existing systems and workflows.

Measure Performance

Track key metrics to evaluate efficiency and effectiveness.

Verify Correctness

Ensure all data and processes meet established standards.

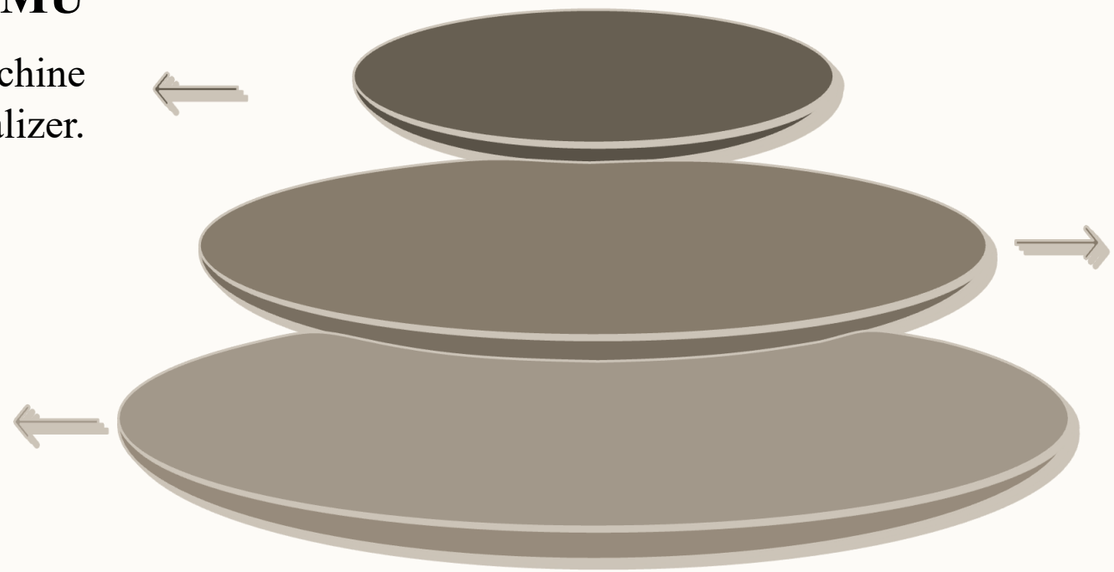
Implementation

QEMU

An open-source machine emulator and virtualizer.

VMware

A leader in virtualization and cloud computing.



VirtualBox

A powerful x86 and AMD64/Intel64 virtualization product.

Puzzle:

Hardware Emulation in OS

A student is using a modern x86-based laptop to run an old ARM-based operating system for testing purposes. The OS runs correctly even though the laptop does NOT have ARM hardware.

1. Each CPU instruction issued by the ARM-based OS is first captured by a software layer.
2. The captured instructions are converted into instriuctions understandable by the x86 processor.
3. All memory access and I/O device operations are handled in software.
4. The program produces output exactly as if it were running on native ARM hardware.

a Identify the technology being used.

b Name the software layer responsible for instruction capture and translation.

c Correct sequence of steps:
- → Guest OS or Program Execution
- → Instruction Interception
- → Execution by Host Hardware
- → Memory and I/O Management
- → Output Generation

d Give one real-world example of such a system.

Answer:

a **Technology Used:** Hardware Emulation

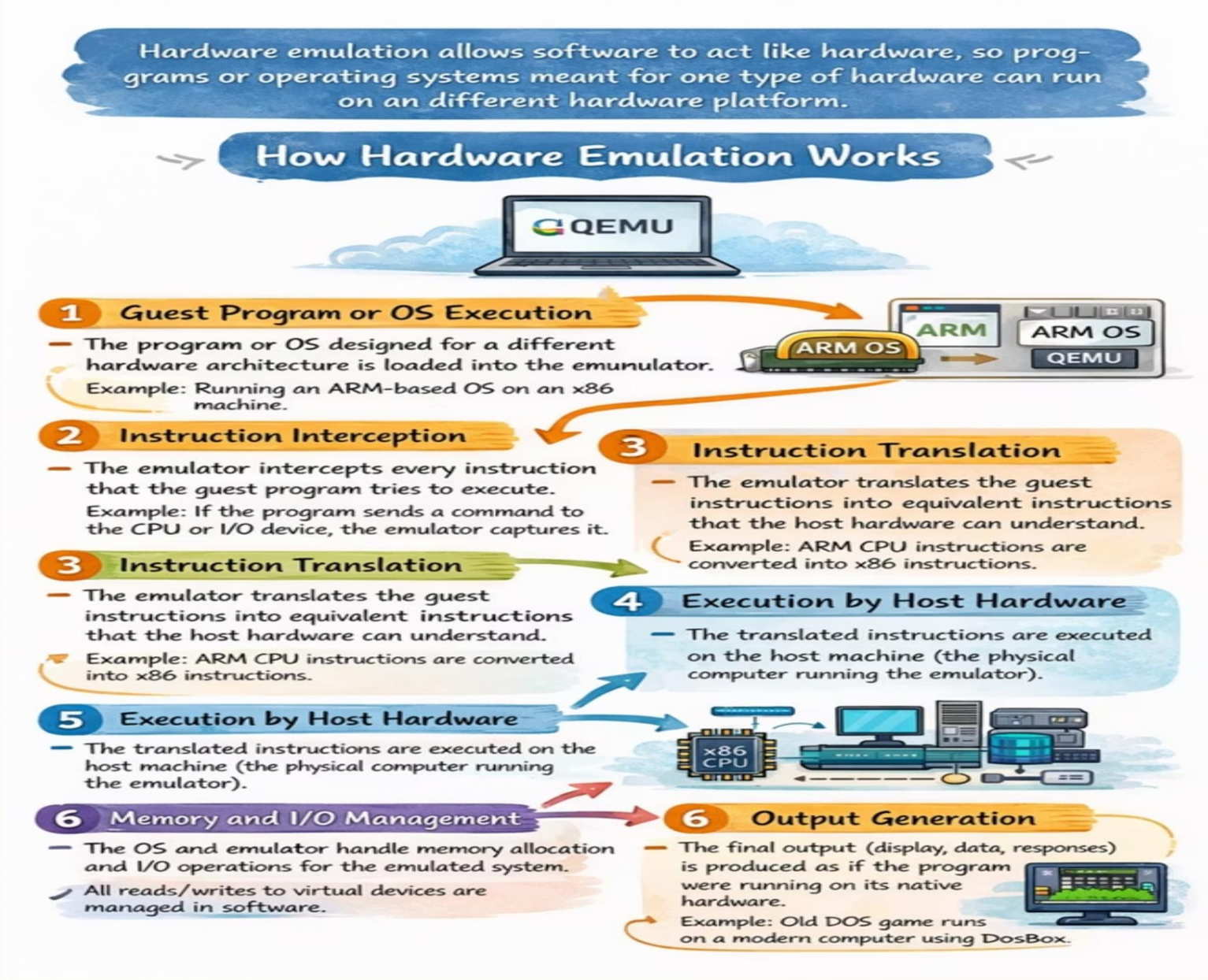
b **Software Layer Responsible:** **Emulator** (e.g. QEMU, DOSBox)

c **Correct Sequence of Steps:**
- → Guest OS or Program Execution
- → Instruction Interception
- → Execution by Host Hardware
- → Memory and I/O Management
- → Output Generation

d **Real-World Example:**
- ✔ Running an old DOS game using DOSBox on a modern PC

Key Learning Point

Hardware emulation works at the instruction level and is slower than virtualization, but it enables cross-architecture **compatibility**.


Hardware emulation allows software to act like hardware, so programs or operating systems meant for one type of hardware can run on an different hardware platform.
How Hardware Emulation Works
QEMU
1 Guest Program or OS Execution
The program or OS designed for a different hardware architecture is loaded into the emunulator.
Example: Running an ARM-based OS on an x86 machine.
ARM OS
ARM
ARM OS
QEMU
2 Instruction Interception
The emulator intercepts every instruction that the guest program tries to execute.
Example: If the program sends a command to the CPU or I/O device, the emulator captures it.
3 Instruction Translation
The emulator translates the guest instructions into equivalent instructions that the host hardware can understand.
Example: ARM CPU instructions are converted into x86 instructions.
3 Instruction Translation
The emulator translates the guest instructions into equivalent instructions that the host hardware can understand.
Example: ARM CPU instructions are converted into x86 instructions.
4 Execution by Host Hardware
The translated instructions are executed on the host machine (the physical computer running the emulator).
5 Execution by Host Hardware
The translated instructions are executed on the host machine (the physical computer running the emulator).
x86 CPU
6 Memory and I/O Management
The OS and emulator handle memory allocation and I/O operations for the emulated system.
All reads/writes to virtual devices are managed in software.
6 Output Generation
The final output (display, data, responses) is produced as if the program were running on its native hardware.
Example: Old DOS game runs on a modern computer using DosBox.