The Art of Graphics Programming

Technical Context

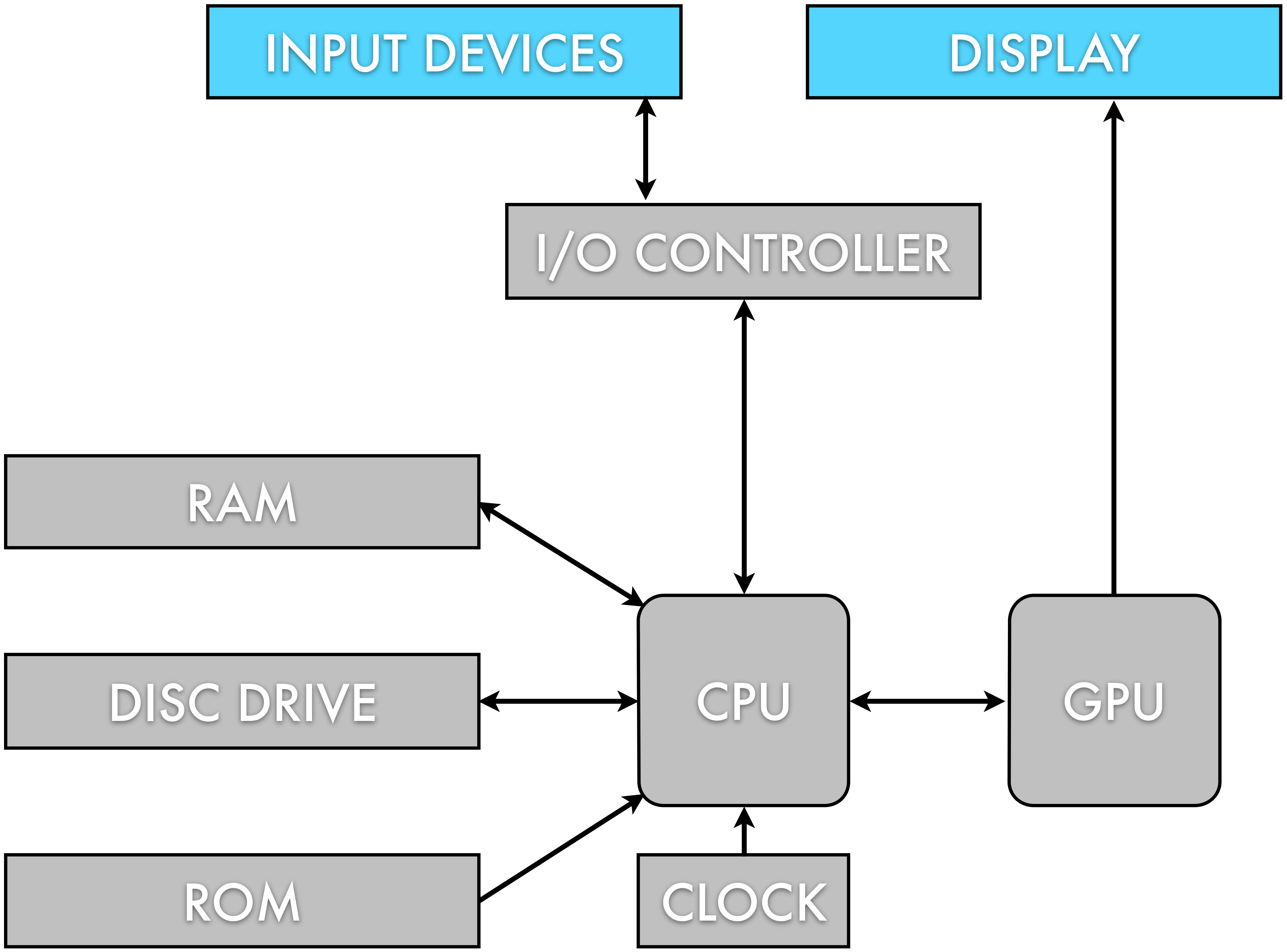
Basic Anatomy

CPUs, GPUs, Programming Languages & APIs

Patrick Hebron
NYU ITP

Hardware Component Classifications

Hardware Anatomy (Simplified)



A Central Processing Unit (CPU)

is a large-scale integrated circuit, capable of performing the basic arithmetical, logical and input/output calculations required by a computer system.

The CPU serves as the primary clearing house for instructions assigned by the OS and other applications.

As Eric Rosenthal says,
“The CPU is the CPU because it’s the CPU.”

Examples: Intel i7 Series and AMD Opteron Series

A Graphics Processing Unit (GPU)

is a specialized integrated circuit, designed for the accelerated handling of floating point (decimal) calculations as well as the construction and manipulation of geometric and textual data related to graphics display.

A GPU is also accelerated through its use of numerous parallel computational nodes.

Examples: NVidia GeForce Series and ATI Radeon Series

Random-Access Memory (RAM)

is a type of computer memory storage hardware
that uses a solid-state medium to facilitate
the rapid retrieval and writing of data.

RAM is designed in such a way that any memory unit can
be accessed as quickly as any other. However, RAM is *volatile*,
meaning that its memory state is lost when power is removed.

RAM is generally used for the temporary storage of data
required by the runtime processes of the OS and other software.

A Disc Drive

is a type of computer memory storage hardware used for
the persistent storage of large quantities of data.

Until recently, this term would generally imply a
magnetic spinning-platter medium. In the past few years,
high-speed solid-state drives have become increasingly prevalent.

Disc drives are slower than RAM. However, they are
less expensive per memory unit and are also non-volatile.

Software Component Classifications

An Operating System

is a collection of software that manages a computer's hardware components and provides services such as event handing and memory allocation to programs running within the OS.

Modern operating systems often provide additional APIs such as windowing environments, user interface widgets and network protocols that assist third-party development for the OS.

Examples: Mac OS, Windows, Linux, iOS and Android

A Virtual Machine

is an operating system that runs within another operating system.

This allows several user terminals to be hosted on a single hardware system or for a single terminal to draw upon the resources of multiple hardware systems.

Virtual machines are also used to simplify the process of creating cross-platform software. For example, software written in Java and run within the JVM will behave almost identically on Mac, Windows, Linux and Solaris computers.

Examples: Java Virtual Machine (JVM), VMware and Parallels

A Programming Language

is a human-readable vocabulary and syntax, which is used to define logical instructions that can be interpreted and executed by a computer.

This term generally implies a language that is *Turing complete*, meaning that it is capable of representing any deterministic calculation whatsoever.

Examples: C, C++, Objective-C, Java and Python

A Graphics API

provides a series of tools related to the creation, manipulation and realtime rendering of 2D or 3D geometries.

Due to the computational volume of this task, specialized hardware (the GPU) is commonly used to accelerate the process.

A graphics API provides an interface for and manages the relationship between the software and hardware components of a realtime rendering pipeline.

Examples: OpenGL and DirectX

An Application Framework

mediates the low-level operations of a
programming language and additional APIs in order
to provide developers with easier access to
functions that would be commonly
used in the development of software of a particular kind.

An application framework is not a programming language, per se.
Instead, it can be seen as a shorthand vocabulary
that encapsulates or reduces a larger, more generalized
and often more complex vocabulary.

Examples: Processing, OpenFrameworks, Cinder, Pocode and Polycode

A Shader Language

exists within a Graphics API and provides a syntax and vocabulary for the expression of logical instructions specifically related to the capabilities of a graphics card's specialized design.

In general, shaders are used to define and augment the material properties of computer-generated surfaces. These properties include a surface's perspectival distortion, illumination and texturing.

Examples: GLSL, Cg, HLSL and RenderMan

A GPGPU API

or, *General Purpose* GPU API, provides an interface for performing non-graphics calculations through the high-performance, specialized hardware of a graphics card.

Examples: OpenCL and CUDA