AMD Computing Platform Presentation 1

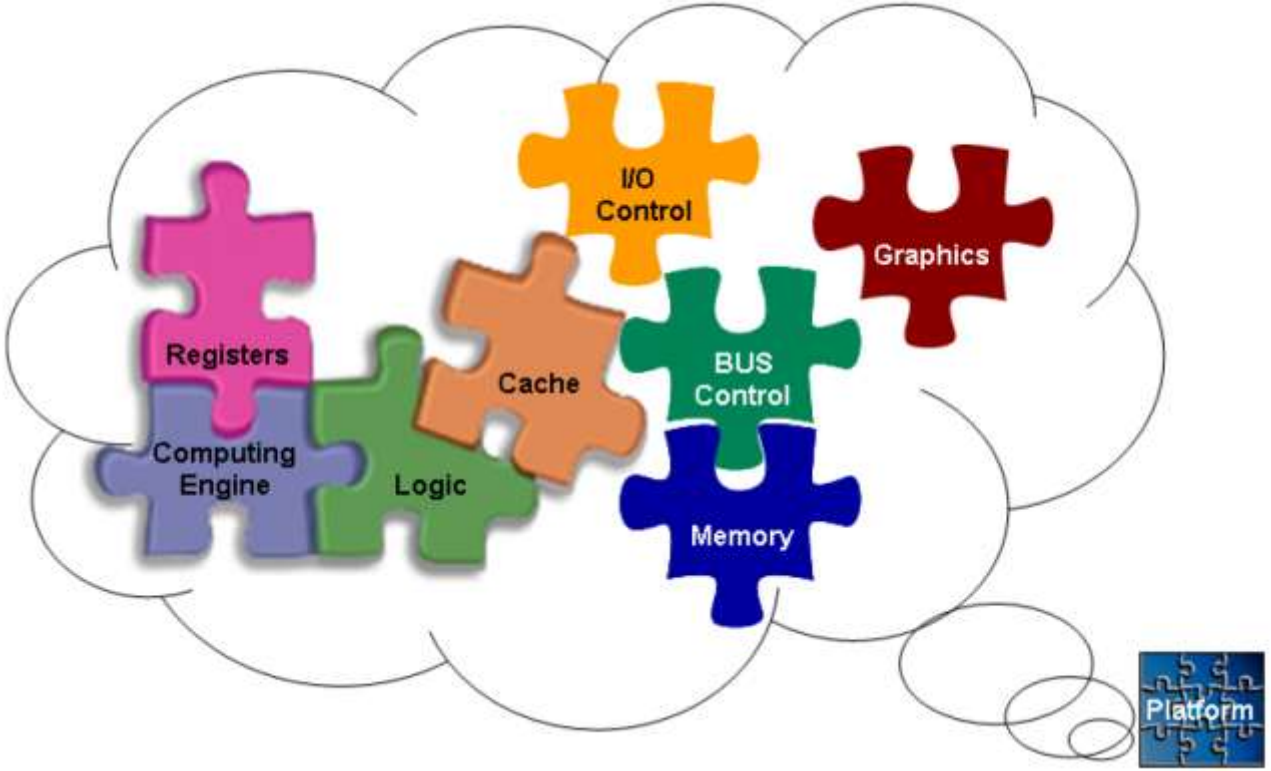
Guangming Zhu

1. AMD Computing Platform 101
   - Introduction to computing platforms
   - AMD CPU
   - AMD GPU
   - AMD chipset
   - AMD APU
2. AMD Computing Platform 101 Volume II
   - Introduction to platform software---System Boot
   - BIOS
   - Operating systems
   - APIS
   - Drivers

Platform

Platform

- A platform describes some sort of hardware architecture or software framework that allows software to run.
  - A computing engine------process instructions and data.
  - A storage place------host the instructions and data.
  - A form of control logic-------bridge the computing engine to peripheral devices.

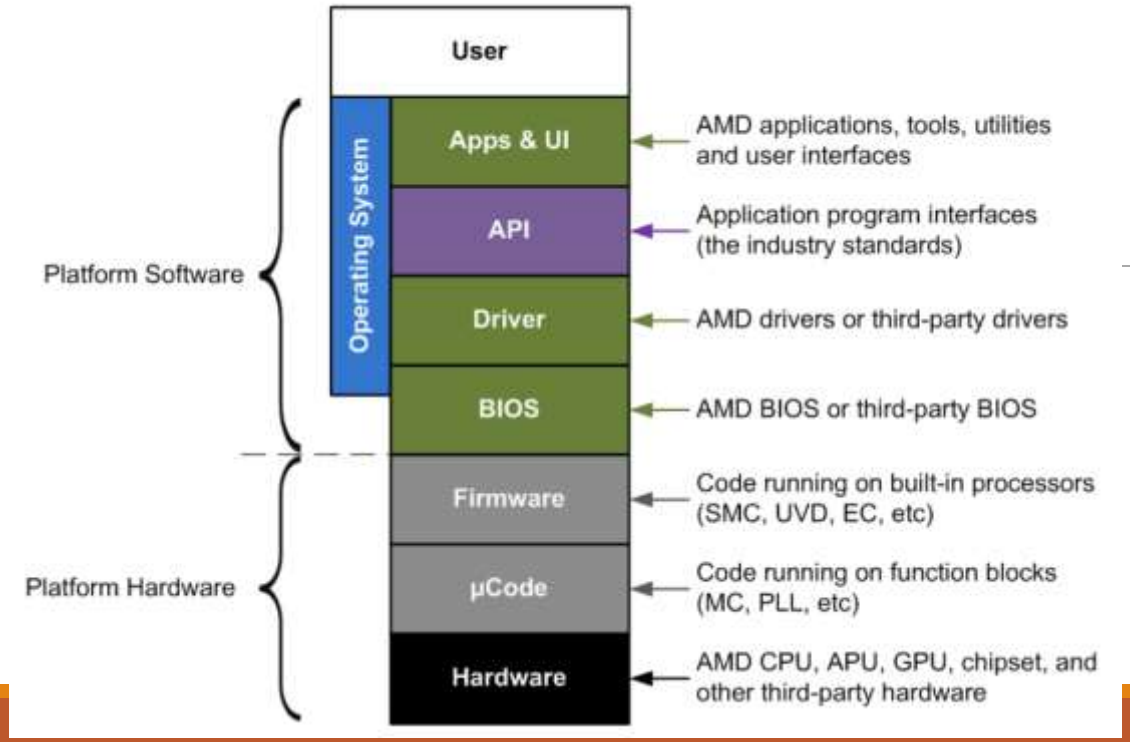


Example Building Blocks of a Computing Platform

Platform software

- Software is the medium that enables users to control and communicate with the hardware components of a computing platform.

- In some special cases, software is also part of the hardware. For example, the memory controller of certain microprocessors cannot exist without first being loaded with its microcode.

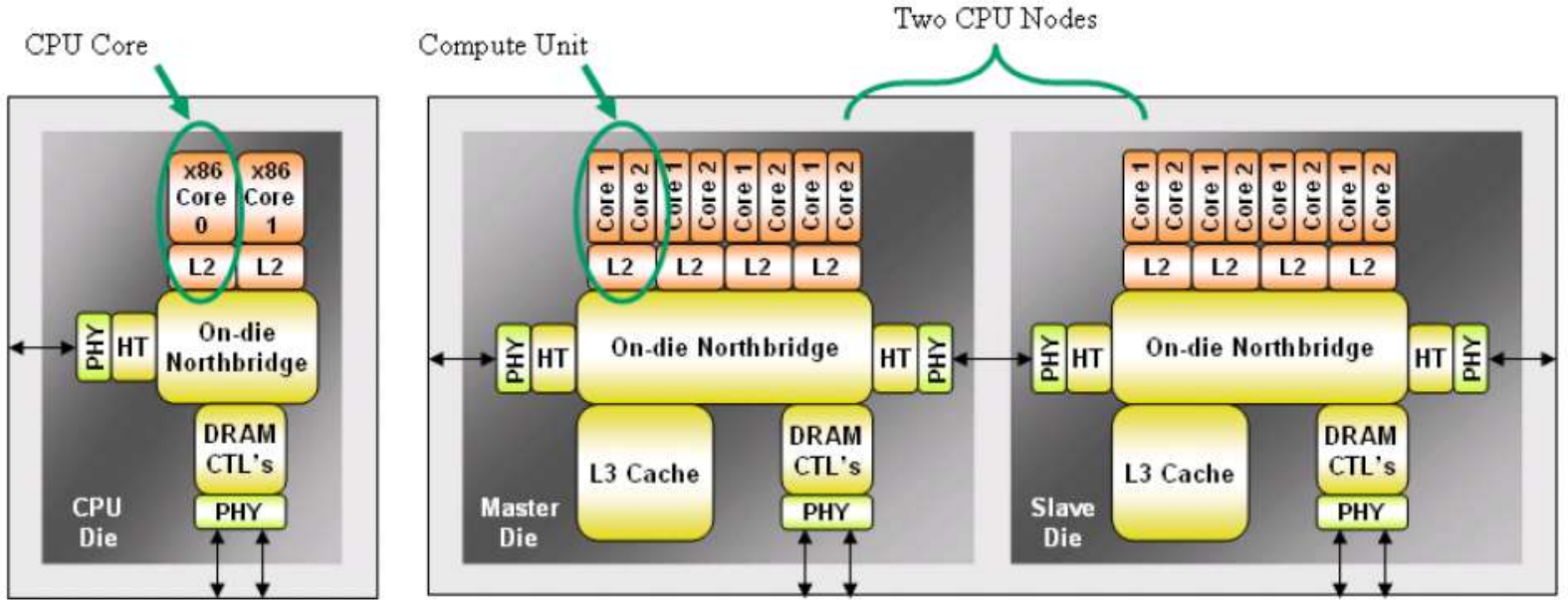


platform software architecture

CPU

CPU

- CPU is “a programmable logic device that performs instructions, logic, and mathematical processing in a computer platform.”
  - CPU core---Hardware that supports one execution thread.
  - Compute unit---A pair of CPU cores that shares some logic components and operates on a single frequency and voltage domain.
  - CPU node---A group of CPU cores integrated with an on-die Northbridge and a memory interface on a single silicon die.
  - CPU processor---A complete package with one or more CPU nodes.



AMD CPU Examples

CPU Identification

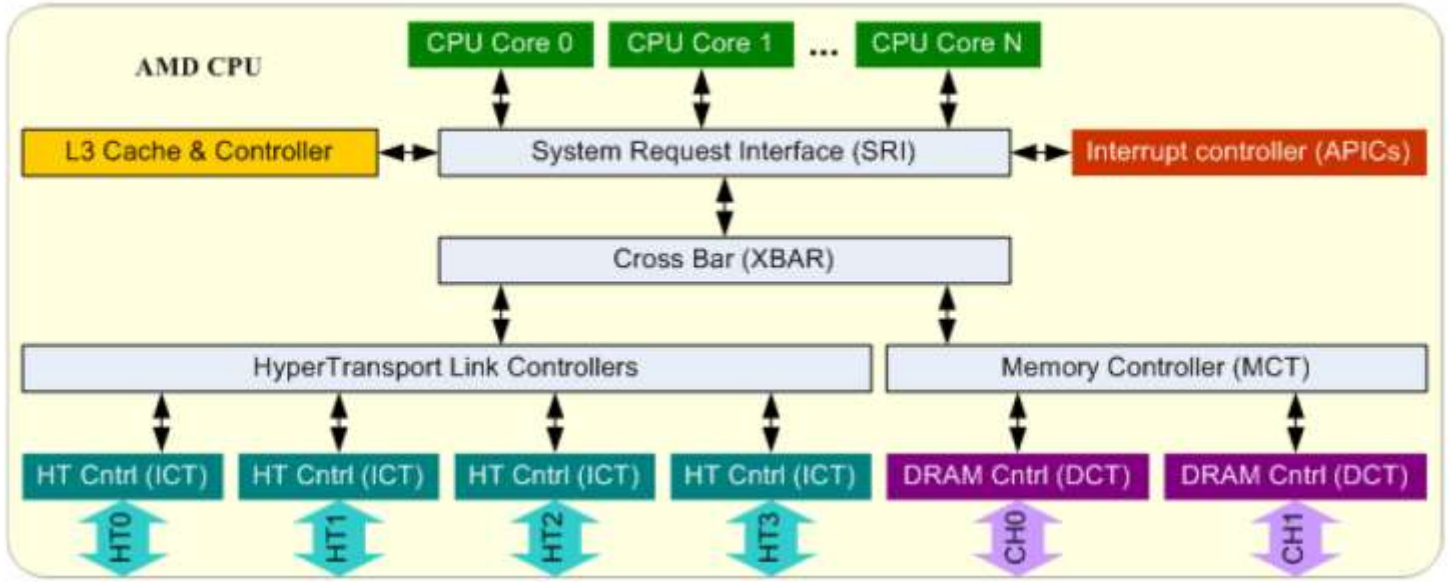
- Each AMD CPU processor is assigned a set of identification information that defines the uniqueness of its hardware.

  - AMD CPU Vendor ID and Signature--- “CPUID” is an instruction that software uses to read CPU registers and to determine the manufacturer, type, and feature set of a CPU processor.

  - AMD CPU Package---which defines the physical makeup of a CPU. The CPU package is used to differentiate a processor's physical and electrical variants.

CPU

- Key function blocks of the AMD CPU are:
  - CPU core
  - Level 3 (L3) cache and controller
  - System request interface (SRI)
  - Interrupt controllers (APICs)
  - Crossbar (XBAR)
  - HyperTransport (HT) link controller
  - HyperTransport (HT) interface controller (ICT)
  - Memory controller (MCT)
  - DRAM controller (DCT)

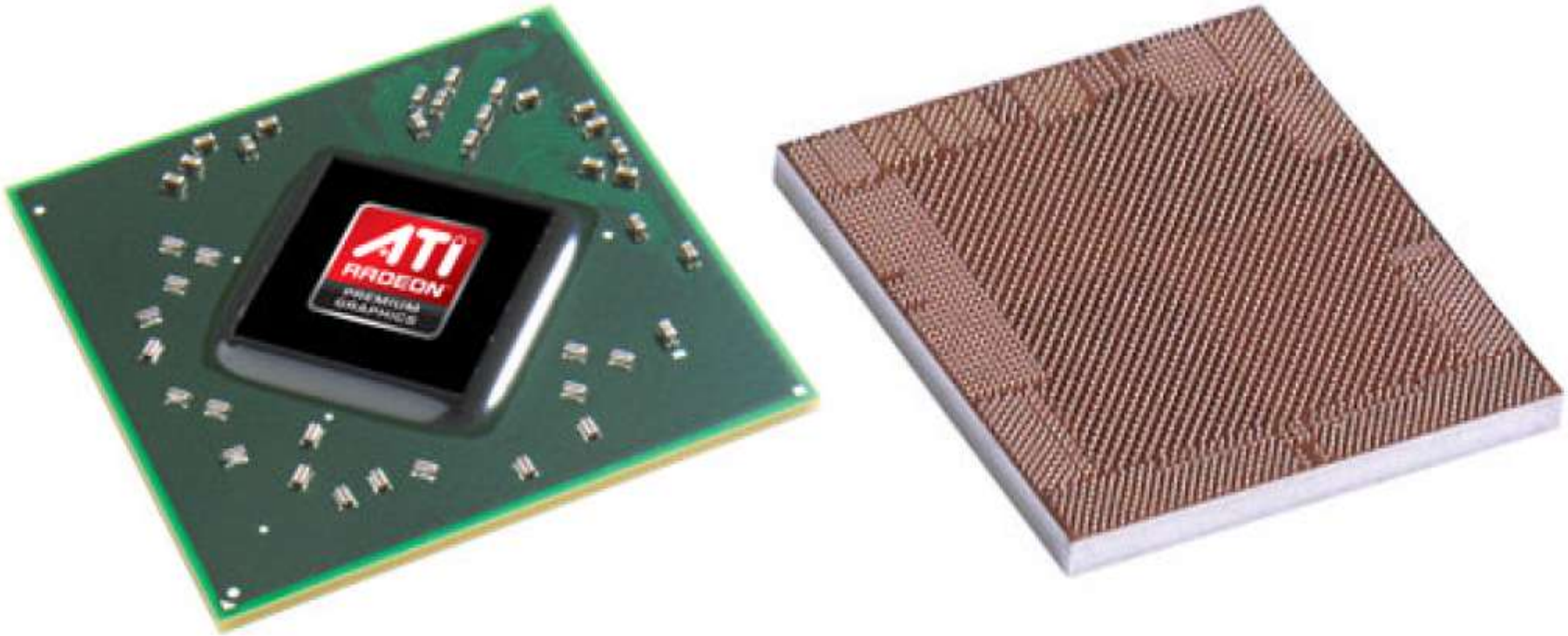


Single-node AMD CPU Architecture
Example Key function

GPU

GPU

- “a single-chip processor used primarily for computing 3D functions such as lighting effects, object transformations, and 3D motion. The GPU allows video displays to be updated more quickly than with the CPU, thus allowing the CPU to be available for other tasks.”

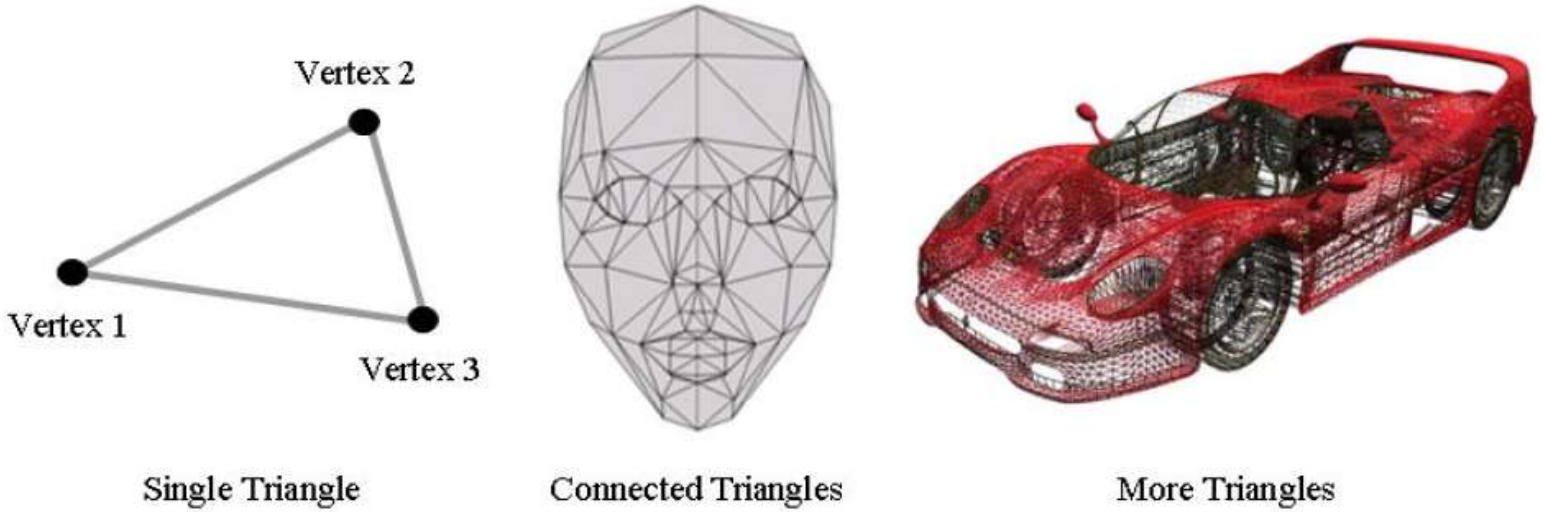


Discrete GPU in a BGA Package

The Basics of 3D Graphics

- 3D graphics is simply the means by which a 2D image on a 2D display is given the illusion of depth.

- Polygons are the most commonly used 3D primitive.
- Polygons are geometric shapes defined by three or more vertexes (points).
- The most commonly used polygon primitive is the triangle.

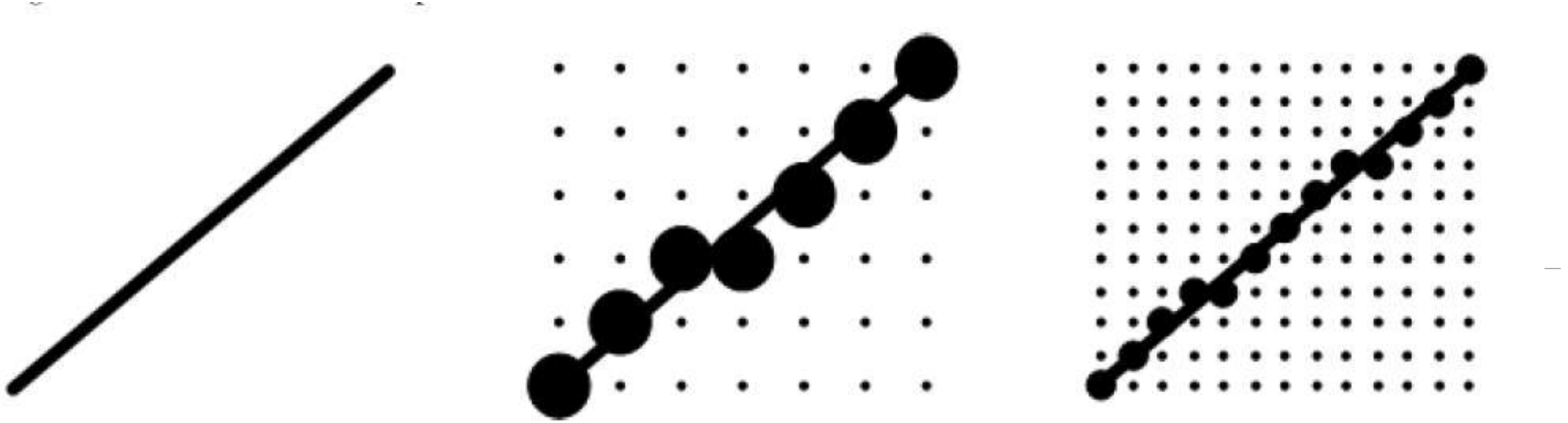


3D Objects Modeled Using Connected Triangles

The Basics of 3D Graphics

- GPUs deal with images in the form of bitmaps, which are two-dimensional arrays of pixels.

- Display resolution is equal to the number of pixels that can be shown.
- Insufficient resolution for an image of a given size leads to aliasing artifacts, also known as "pixelation."



Pixelation Example

Pixels

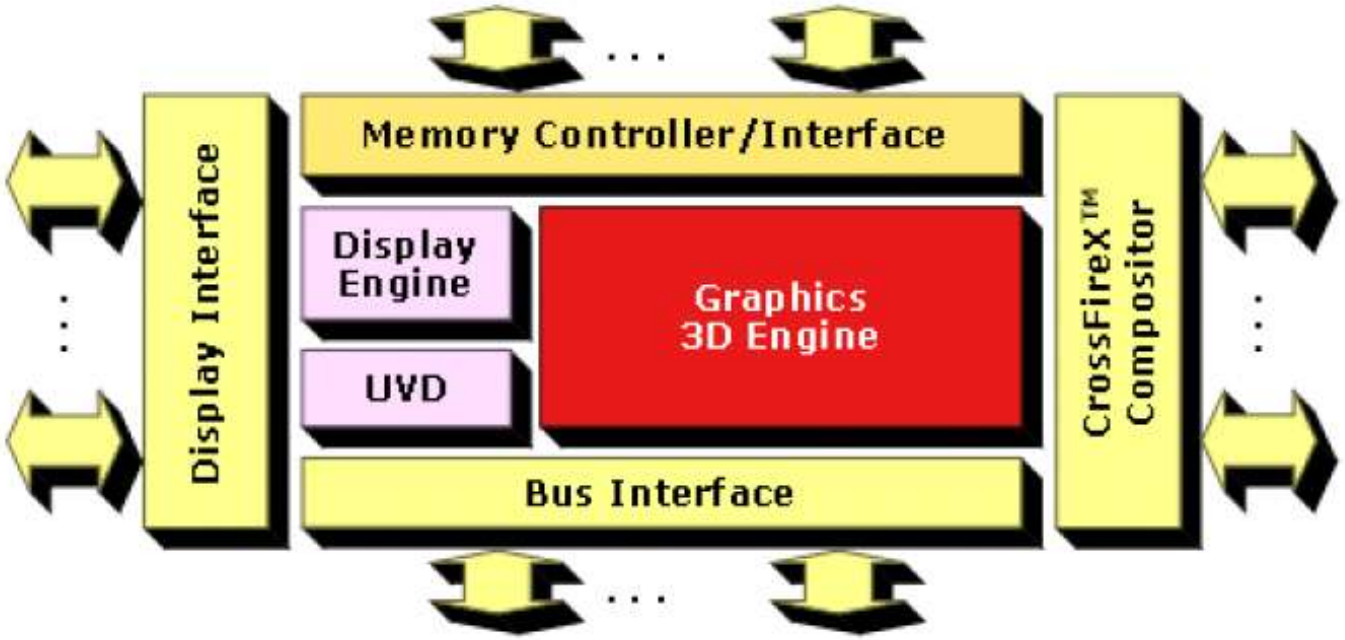
- Each pixel consists of a location (coordinate) and a color value.

- Color depth is represented by bits per pixel (bpp), which determines the maximum number of colors a pixel can be.

- Common bpp values include:
  - 8 bpp = 256 colors.
  - 16 bpp = 65,536 colors, also known as highcolor or thousands of colors.
  - 24 bpp = 16,777,216 colors, also known as truecolor or millions of colors.

Key Components of the AMD GPU

- Bus interface
- Memory controller and memory interface---A discrete GPU directly access its dedicated local graphics memory, known as the frame buffer.
- Graphics 3D engine
- ATI CrossFireX compositor
- Display engine and display interface
- Unified Video Decoder(UVD)



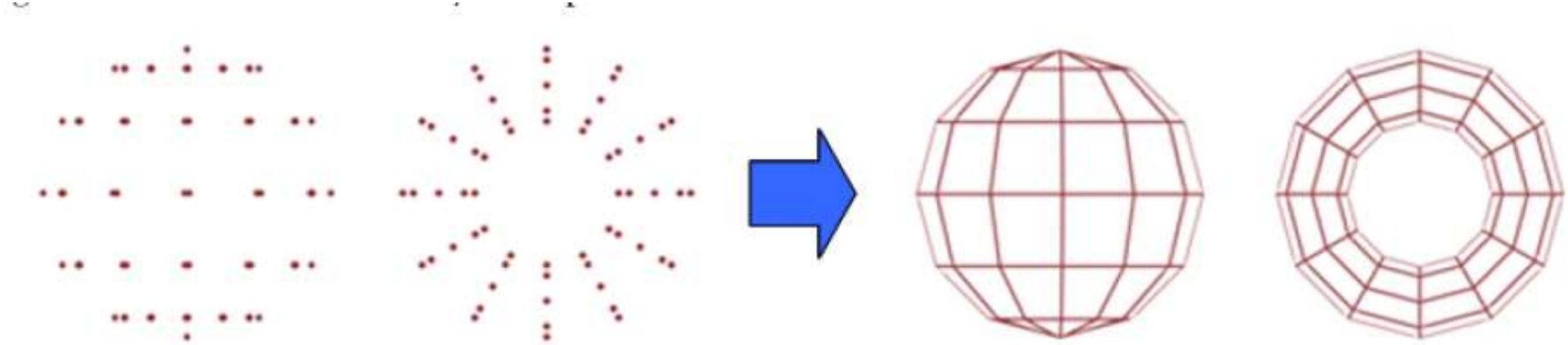
Creating a 3D World

- When a computer creates a 3D world, its CPU sends commands and data to the GPU so that each frame can be rendered.
- The GPU generates a list of all visible objects for each frame to be rendered, discards elements that are obscured or fall entirely outside the field of view.
- The GPU then loads all necessary polygons, textures, and animation data into the graphics memory.
- Once the data is set up, the actual drawing of scenes can begin.

  - Vertex Assembly
  - Tessellation
  - Transformation
  - Lighting
  - Triangle Setup
  - Texture Mapping
  - Texture Filtering
  - Z (Depth) Test
  - Transparency
  - Anti-aliasing

Creating a 3D World---Vertex Assembly

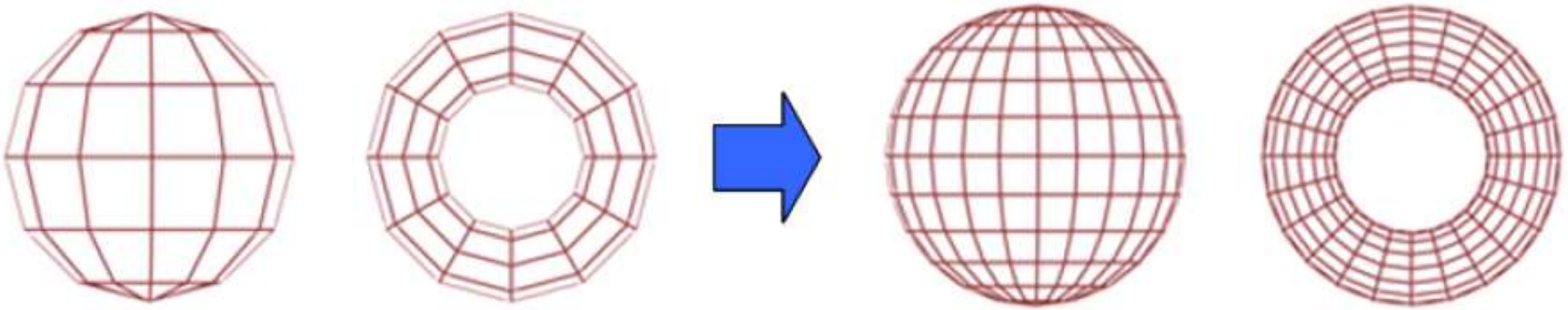
- Data is inputted into the GPU in the form of vertexes.
- During vertex processing, the points are placed in their corresponding locations are then used to form triangles.



Vertex Assembly Example

Creating a 3D World---Tessellation

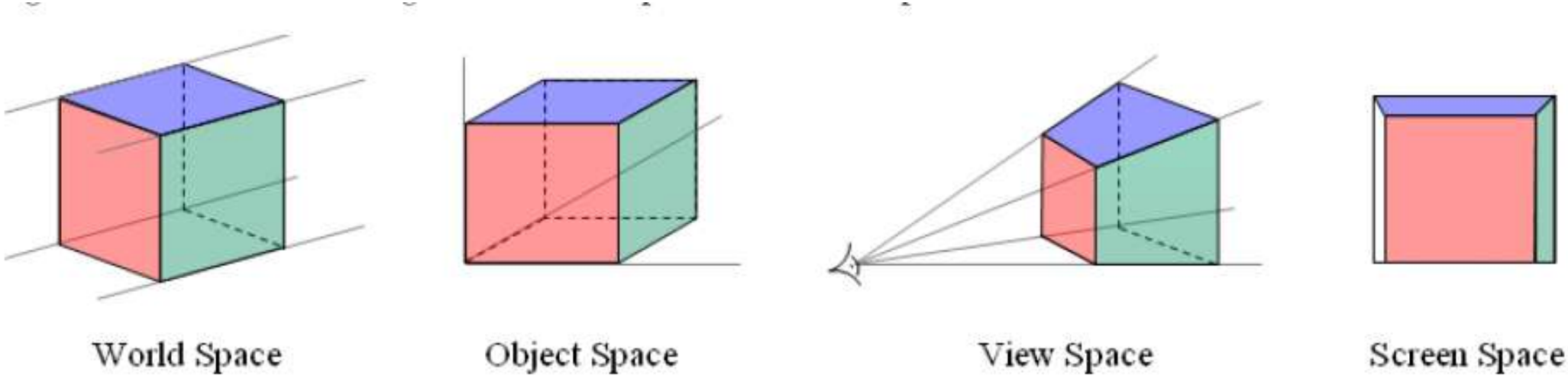
- Tessellation is the process of converting a curved surface into a mesh of flat polygons.
- It also involves adding new vertexes to form additional triangles to give details to 3D objects.



Tessellation Example

Creating a 3D World---Transformation

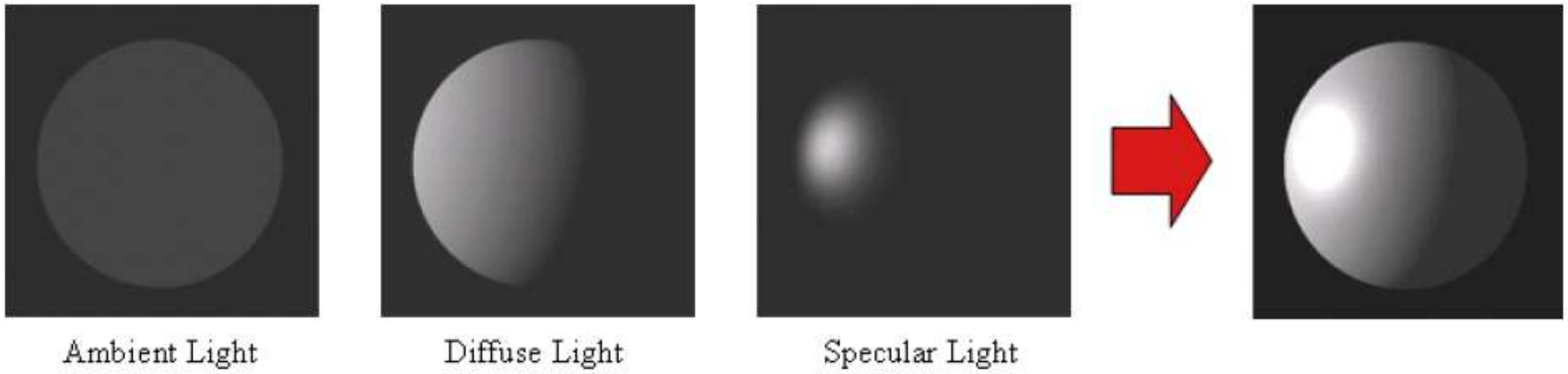
- Transformation is the process of calculating and modifying the positions of vertexes.
- Coordinates are used to assign locations to objects relative to some point of origin (for example, X = 0, Y = 0, Z = 0).



Transforming from World Space to Screen Space

Creating a 3D World---Lighting

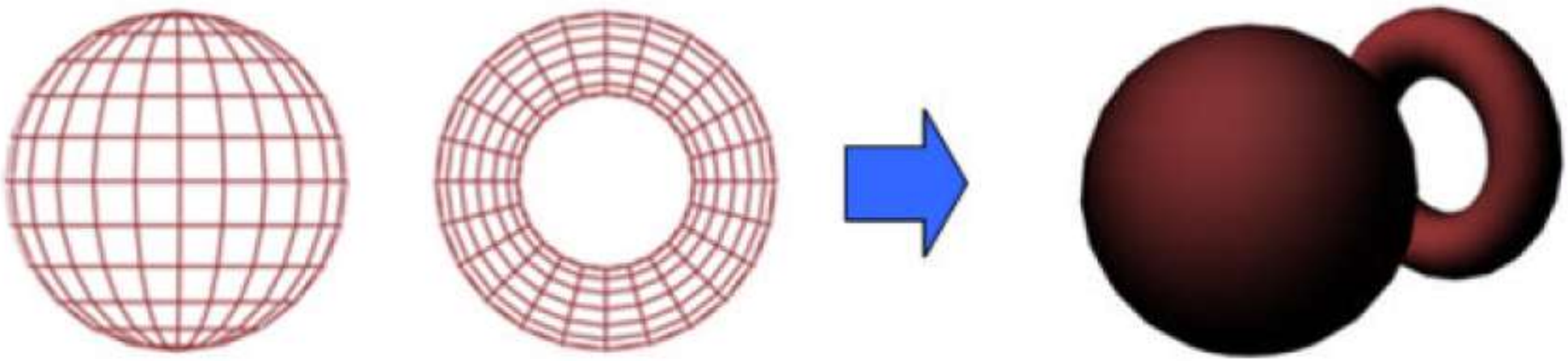
- At least one light source is required in the 3D digital world, otherwise all images would be completely dark.
- Each light source contains at least three components:
  - Ambient
  - Diffuse
  - Specular



Lighting Components

Creating a 3D World---Triangle Setup

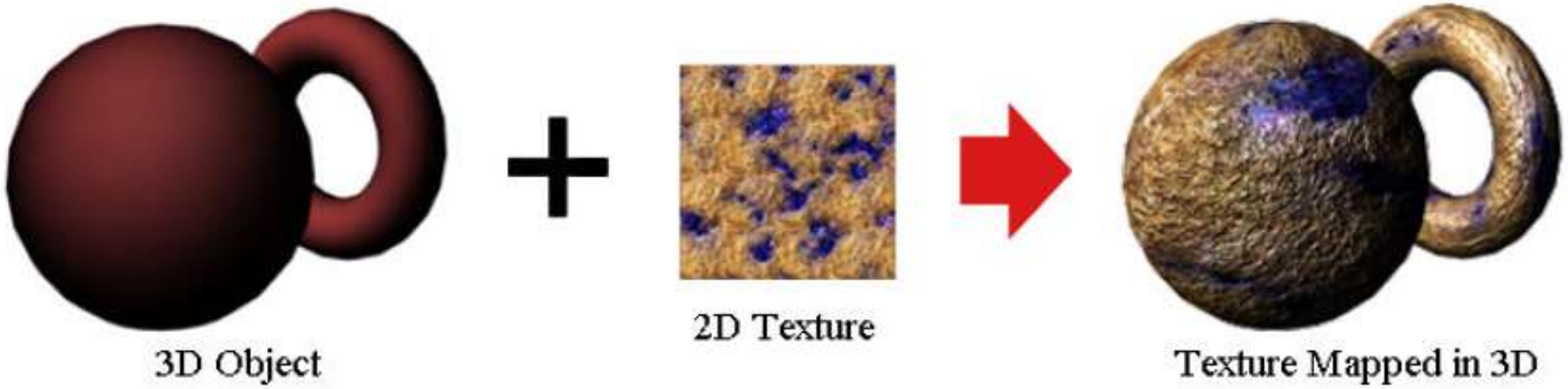
- Triangle setup is the step in the rendering process that converts polygons into pixels.
- It determines which screen pixels lie inside the polygon being processed.
- Color values and texture coordinates are assigned to each pixel, known as Gouraud shading.



Triangle Setup and Gouraud Shading Example

Creating a 3D World---Texture Mapping

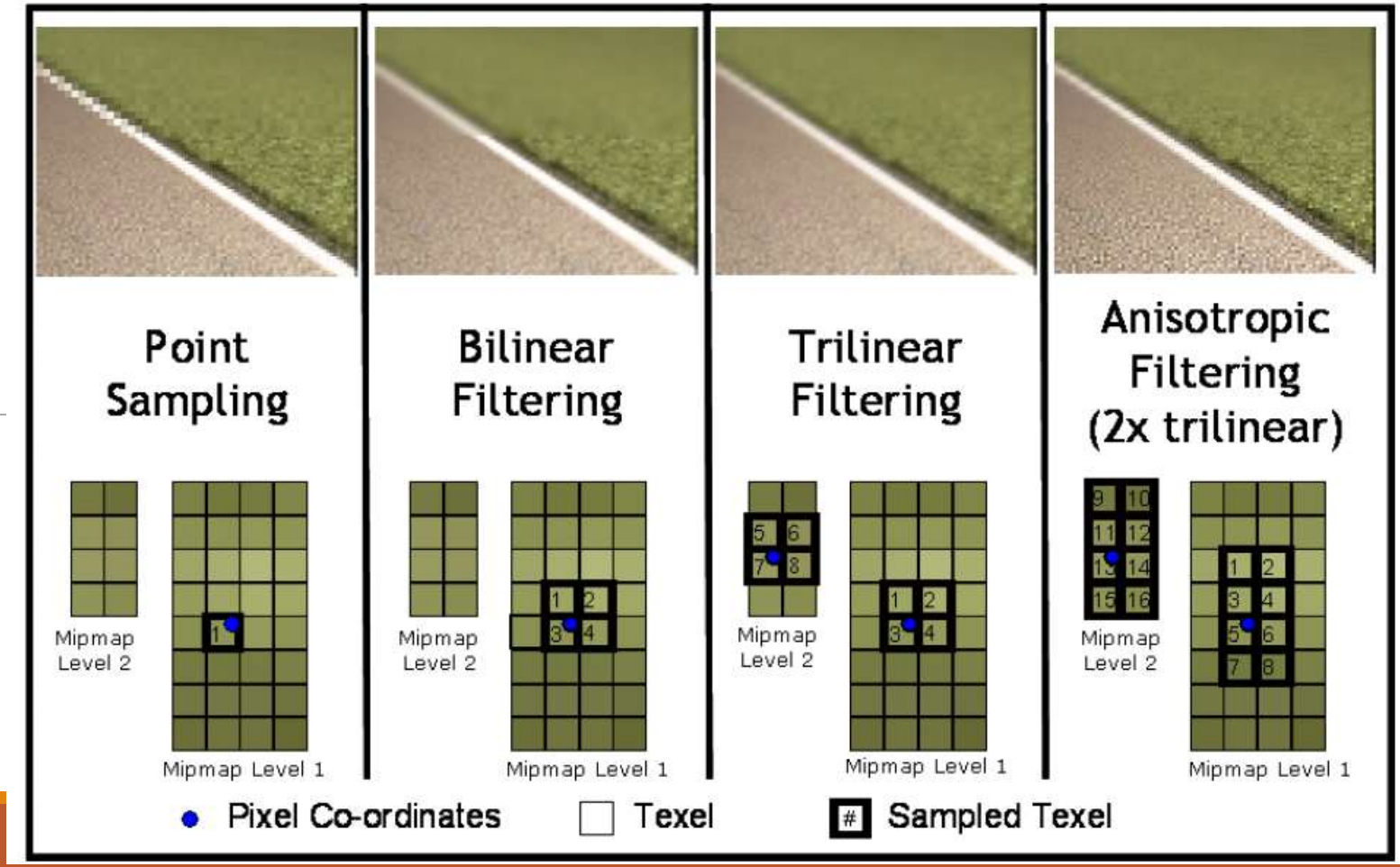
Every surface in a 3D scene must be assigned a set of material parameters such as diffuse color, reflectivity, and transparency.
Textures are images applied to the surface of an object to give it a more realistic appearance.



Texture Mapping Example

Creating a 3D World--- Texture Filtering

- Texture filtering improves the appearance of surfaces.
- The following are common filtering techniques (in order of increasing quality).
  - Point sampling (pick nearest)
  - Bilinear filtering
  - Trilinear filtering
  - Anisotropic filtering

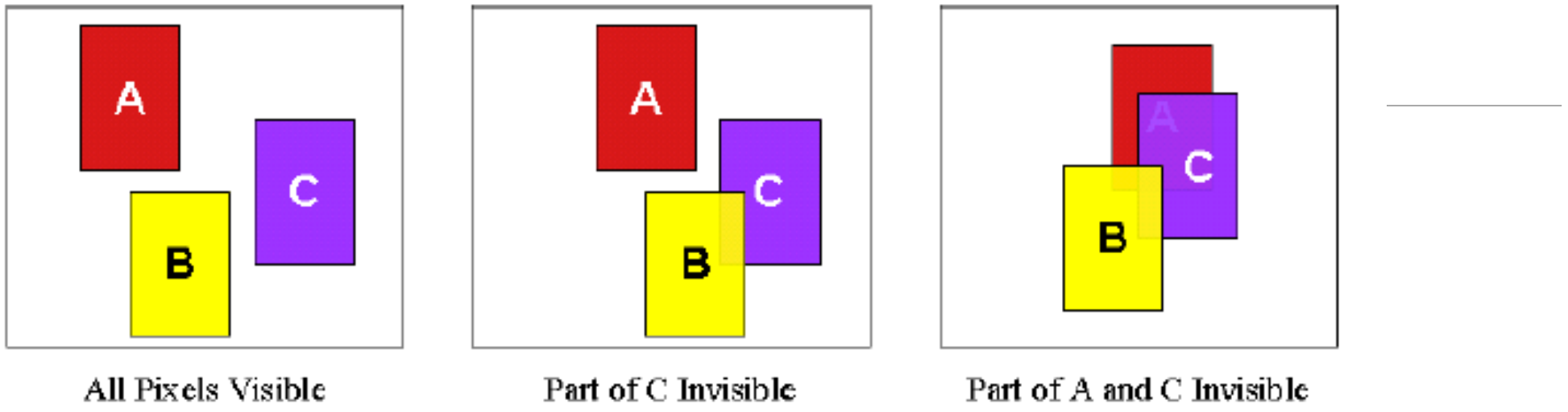


Texture Mapping Example

Creating a 3D World---Z (Depth) Test

Depth is the third dimension (Z-axis) that makes a 2D image look like a 3D object. The Z (depth) test determines whether each pixel being rendered will be visible in the final image.
Before the current pixel is written to the frame buffer, its z-value, or depth, is compared with that of the pixel already in the frame buffer.

- If the depth value of the current pixel is greater, the current pixel is discarded because it appears “behind” the existing pixel.
- If the depth value of the current pixel is smaller, the color values of the existing pixel in the frame buffer are overwritten by those of the current pixel.



Z Test Example

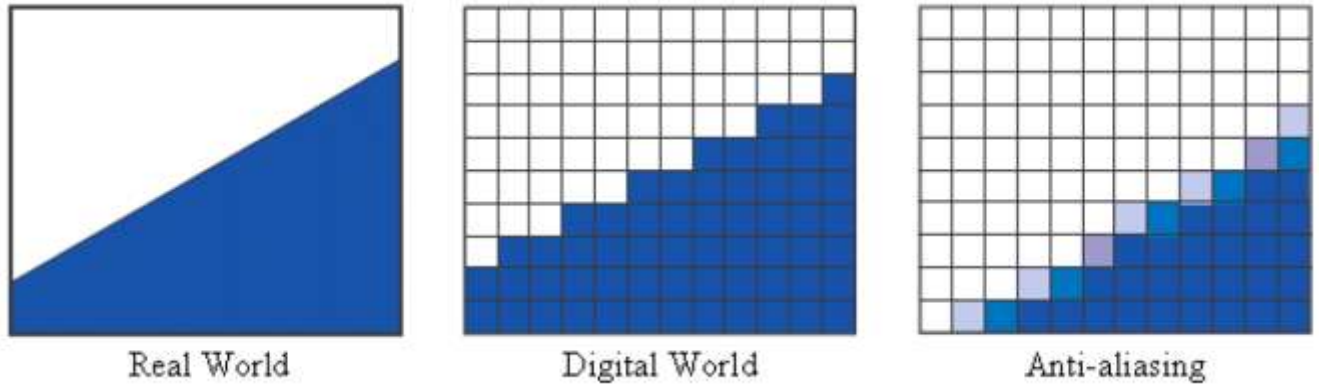
Creating a 3D World---Transparency

- Transparency allows an object appearing behind other objects to be seen.
- Objects appearing in the back are blended into the front object rather than removed.
- Each pixel has an optional transparency parameter known as alpha.
- The following equation is a linear blending example used to compute transparency:

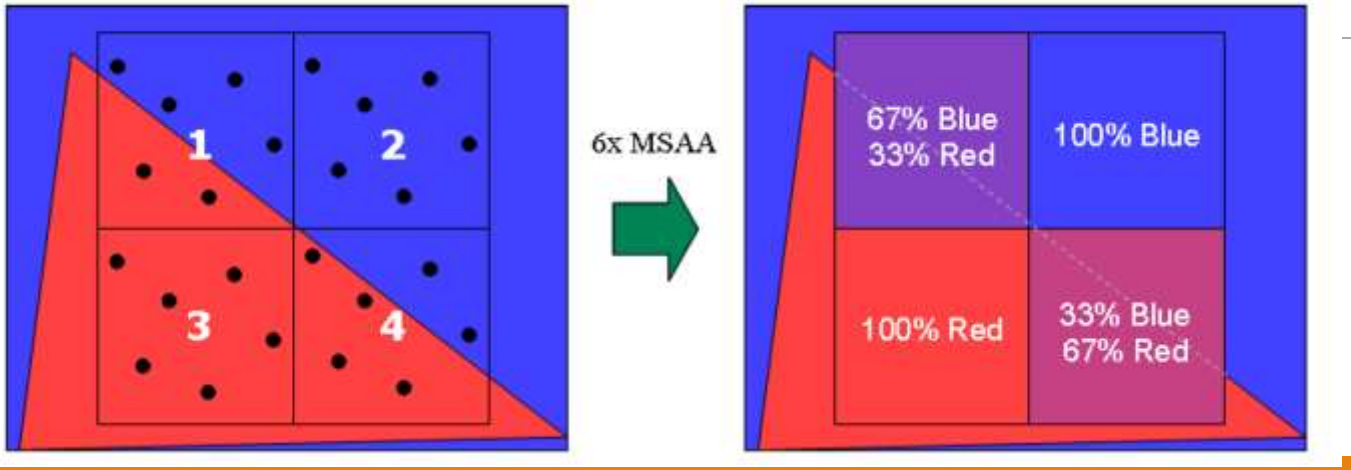
$$\text{New\_Color} = \text{Old\_Color} * (1 - \text{Alpha}) + \text{New\_Color} * \text{Alpha}$$

Creating a 3D World---Anti-aliasing

- Anti-aliasing (AA) is a sampling technique used to create more realistic looking images.
- It smoothes jagged edges.



Anti-aliasing Examples



6× Multi-sample Anti-aliasing Example

Drivers

Drivers

- To initialize, control, and manage each unique hardware device, a small hardware-specific program is required to translate generic operating system instructions to instructions that the target hardware device can understand.

- Keeping device drivers independent from the operating system benefits independent hardware vendors(IHVs) and independent software vendors(ISVs) because it allows feature-added drivers and new hardware devices to be released by the vendors without requiring operating system updates from operating system vendors.

User-mode and Kernel-mode Drivers

- Device drivers can pose potential risks to system security and stability whenever certain low-level function calls are required to “touch” platform-critical hardware components. For example, a poorly-designed device driver can cause a system to crash.
- To minimize these risks, device drivers are divided into two types: user-mode and kernel-mode drivers. The following is a description of each type:
  - User-mode drivers---These drivers run in the user space of an operating system. They have access to only the user address space and always run in a different thread from the requesting process. User-mode drivers are isolated from the system address space and internal platform structure. As a result, user-mode drivers pose a lower risk to system security and stability than kernel-mode drivers.
  - Kernel-mode drivers---These drivers run in the kernel space of an operating system. They have complete access to the system address space and the internal platform structure. As a result, kernel-mode driver errors can cause a system to crash.

Driver Architecture

- Device drivers are not only hardware-dependent, and they are also dependent on the operating system. Device drivers are typically designed with different architectures to support different operating systems.

  - Windows Driver Architecture
  - Linux Driver Architecture

Windows Driver Foundation

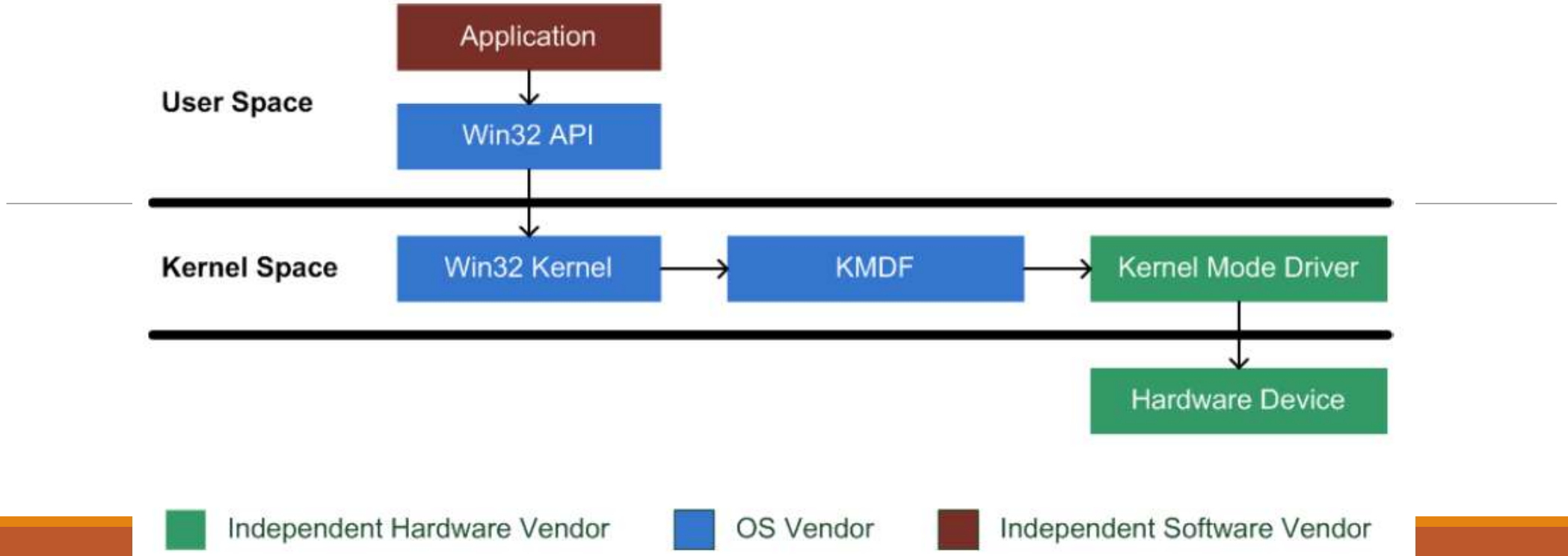
- Windows Driver Foundation(WDF) is a driver model that provides basic infrastructures for both kernel-mode and user-mode device drivers. These infrastructure are generic software code that implement common features and manage most interactions with the operating system.

- The infrastructures consist of separate sets of code for kernel-mode and user-mode drivers.

  - Kernel Mode Driver Framework, or KMDF.
  - User Mode Driver Framework, or UMDF.

Kernel Mode Driver Framework(KMDF)

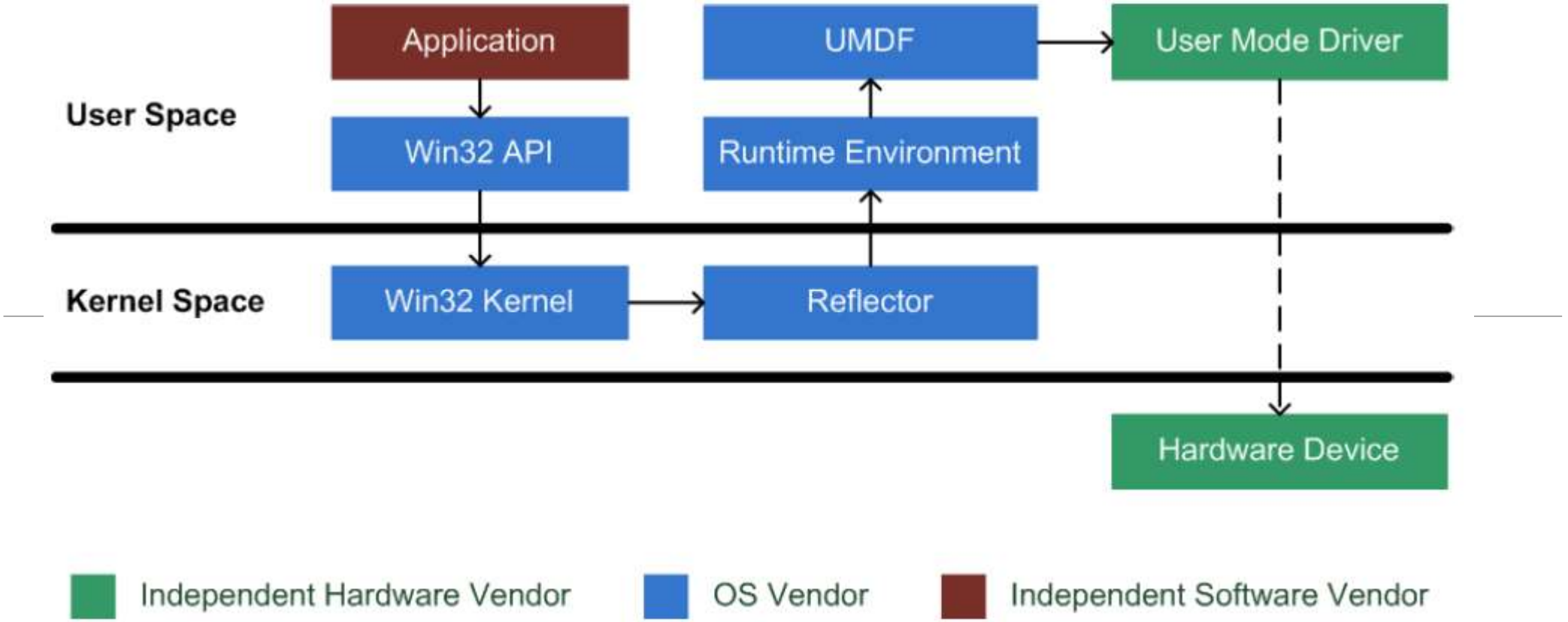
- KMDF provides the fundamental features that are needed by a kernel-mode driver to support plug and play events, power management, input and output queues, direct memory access(DMA), and synchronization.
- KMDF provides many low-level functions, it is not part of the operating system kernel. This allows KMDF to isolate kernel-mode drivers from the operating system.
- When a kernel-mode driver crashes, the operating system recovers the affected resources that are used by the driver and prevents the system from halting.



Example Request Flow to Kernel-mode Driver

User Mode Driver Framework(UMDF)

- UMDF manages requests using the same method as KMDF, except that UMDF cannot be directly called by the operating system kernel. This is because the kernel code runs in the kernel space and the UMDF code runs in the user space.



Example Request Flow to User-mode Driver

Windows Driver Foundation

- Other than the DMA function, the AMD kernel-mode driver also contains the following key components:

  - Common ASIC initialization library(CAIL)
  - Power and performance management library (PPLib)
  - Display abstraction layer(DAL)

System Boot

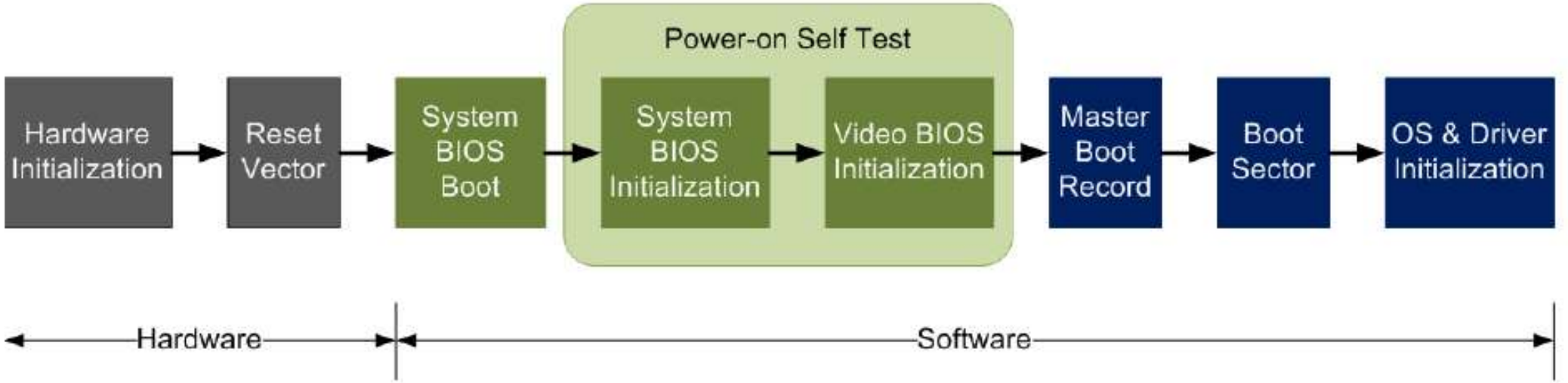
System Boot

- As soon as the first instruction code is fetched by the CPU or APU x86 core, Software initialization starts.
- The following sections describe the events that occur during system boot:

  - System Boot Process
  - POST
  - Display
  - OS Location

System Boot Process

- After power up reset, the system BIOS is the first platform software that is run.
- The system BIOS must find and load the operating system once the initial setup is complete.
- The following diagram shows the basic steps that make up the system boot process:



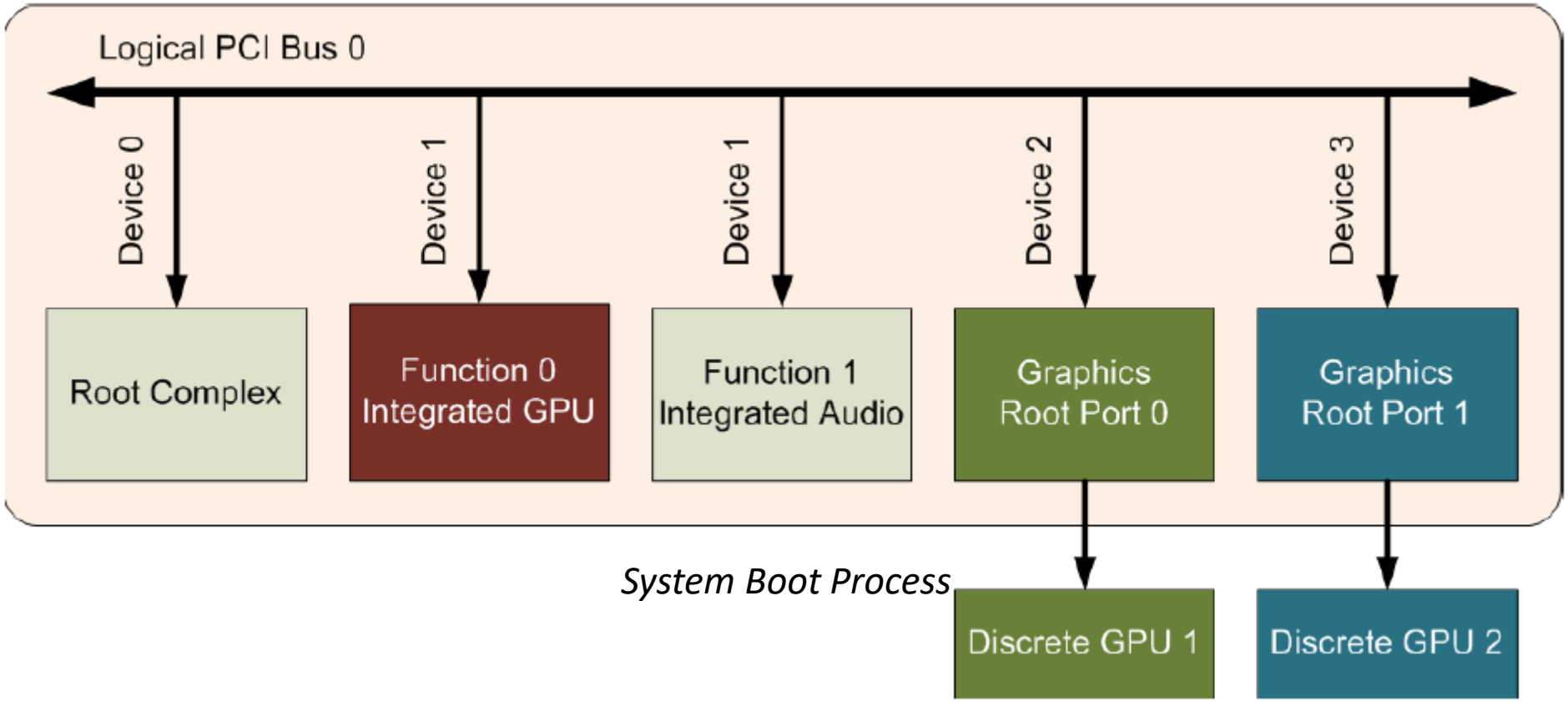
System Boot Process

System Boot Process

- While it initializes the hardware, the system BIOS code also carries out certain levels of verification to detect missing or malfunctioning components. This combination of initialization and verification is also known as “Power-on Self Test”(POST).
- During POST, the BIOS initializes and tests the chipset, system memory, and resources for I/O devices.

Display

- The display of a modern computing platform is physically controlled by one or more graphics processing units(GPUs).
- The diagram shows an integrated GPU and two discrete GPUs.



System Boot Process

Software View of GPU Connections

OS Location

- Among the responsibilities of the system BIOS, loading the operating system(OS) is the main task. All other tasks are done in preparation for this main task.
- The system BIOS must also find the medium that carries the operating system. This medium is known as the initial program load(IPL) device or “boot device”. The hard disk drive is usually set as the first IPL device.
- Using the logical block addressing(LBA) method, a hard disk is divided into multiple sectors, with each sector carrying 512 bytes(offset by 000-1FF) of data.
- When the system BIOS searches for a bootable hard disk, it examines the data from the very first sector(sector 0) of the target hard disk drive. If the hard disk is bootable, the hard disk drive sector 0 contains a special data structure known as the Master Boot Record(MBR).
- The MBR consists of a short program(known as the “boot loader”), a partition table, and a few signature bytes. The boot loader can launch its own or other operating system.

Q & A

THANK YOU