Training - Displaying and rendering graphics with Linux

2-day session

Warning: this agenda can still undergo some changes. The main topics will remain the same, but their order and details may still change.

Title	**Training - Graphics display and rendering with Linux**
Overview	Image and color representation Basic drawing Basic and advanced operations Hardware aspects overview Hardware for display Hardware for rendering Memory aspects Performance aspects Software aspects overview Kernel components in Linux Userspace components with Linux
Materials	Materials for this course are still under development. They will be released under a free documentation license after the first session is delivered.
Duration	**Two** days - 16 hours (8 hours per day). 75% of lectures, 25% of demos.
Trainer	One of the engineers listed on: `https://bootlin.com/training/trainers/`
Language	Oral lectures: English or French. Materials: English.
Audience	People developing multimedia devices using the Linux kernel
Prerequisites	**Basic knowledge of concepts related to low-level hardware interaction (e.g. registers, interrupts), kernel-level system management (e.g. virtual memory mappings) and userspace interfaces (syscalls). Basic knowledge of concepts related to hardware interfaces (e.g. clocks, busses).**

Required equipment	**For on-site sessions only** Everything is supplied by Bootlin in public sessions. • Video projector • Large monitor
Materials	Print and electronic copies of presentations slides

Day 1 - Morning

Lecture - Image and color representation

- Pixels and quantization
- Frames, framebuffers and dimensions
- Color encoding and depth
- Colorspaces
- Sub-sampling
- Alpha component
- Pixel formats formalization

Introducing the basic notions used for representing color images in graphics.

Lecture - Basic drawing and operations

- Lines
- Circles, ellipses and arcs
- Gradients (linear and circular)
- Format and colorspace conversion
- Bit blitting
- Alpha blending
- Colorkeying
- Clipping
- Integer scaling

Presenting how to draw basic shapes and perform basic operations on a framebuffer.

Lecture - Advanced operations

- Filtering and convolution
- Blur
- Fractional scaling
- Anti-aliasing
- Dithering

Providing basic notions about filtering, with very common examples of how it's used.

Demo - Drawing and operations

- Drawing outlines for various shapes
- Filling shapes with solid colors
- Filling shapes with gradients
- Compositing shapes without alpha
- Compositing shapes with alpha
- Compositing shapes with a color-key
- RGB to YUV conversion
- Blur

Illustrating the concepts presented so far.

Day 1 - Afternoon

Lecture - Hardware components overview

- Display hardware components
- 2D processing hardware components
- 2D rendering hardware components
- 3D rendering hardware components
- Video input hardware components

Presenting the hardware involved in graphics pipelines.

Lecture - Display hardware

- Frame streaming and timings
- Side channels and additional data
- Display interfaces
- Transcoders

Presenting the inner workings of display hardware.

Lecture - Processing and rendering hardware

- ISPs and DSPs overview
- Display engine processing
- Vector processors overview
- GPU architectures overview

Describing the architecture of processing and rendering hardware.

Lecture - Memory management and constraints

- Tearing, artifacts and double-buffering
- Alignment and tiling
- Contiguity and IOMMUs
- Cache coherency

Presenting all things related to managing framebuffer memory.

Lecture - Performance aspects

- Common bottlenecks
- DMA operations
- Zero-copy frame sharing
- Costs and benefits of offloading

Presenting possible performance issues and how they are solved in hardware.

Demo - Pipelines debugging and performance

- Misconfiguration patterns
- Tearing example
- CPU rendering performance
- Accelerated rendering performance
- Zero-copy performance

Illustrating what can go wrong and what can be improved in a graphics pipeline.

Day 2 - Morning

Lecture - Software aspects overview

- Complete display pipelines
- Roles handled by the kernel
- Roles handled by userspace

Showing what software components are required for modern computer graphics and how they are divided between kernel and userspace.

Lecture - Linux kernel components

- DRM/KMS subsystem for display
- DRM/GEM subsystem for rendering
- 2D hardware support
- Legacy framebuffer interface
- Framebuffer console, VT switching

How display and rendering is organized in the Linux kernel subsystems.

Demo - Setting up a display with DRM/KMS

- Legacy framebuffer interface
- DRM initialization
- Legacy DRM interface
- Atomic DRM interface
- Using planes

Interacting with the kernel subsystems directly and showing related code paths.

Day 2 - Afternoon

Lecture - Userspace components

- libdrm shim
- Display servers: functionalities, Xorg and Wayland architectures
- 3D acceleration with OpenGL and mesa
- 2D acceleration though OpenGL
- 2D rendering libraries: pixman, cairo
- UI toolkits overview: Qt, GTK/GDK, SDL

Presenting each element involved in the userspace graphics stack.

Demo - Graphics libraries usage examples

- Direct Xorg usage
- Direct Wayland usage
- Basic cairo rendering
- Basic OpenGL rendering
- Basic SDL usage

Showing how to get started with userspace libraries.