Graphics on Linux

Colorado School of Mines Linux Users Group
October 10, 2019
What is Graphics?
A Computer
GUIs are cool. You can enter text in one window while browsing the internet in another.
CRT Monitors

Pixels are lit up for a fraction of a second by a beam scanning back and forth. The beam never stops. It is up to the computer to turn up the voltage when the beam is pointing at pixels it wants lit up and down otherwise. If the timing is off, things will look wrong.
This is a graphics adapter. It remembers what the screen should look like (the scanout buffer) and keeps the electron beam going so that the CPU can get on with other things.
Students at MIT wanted to use these new-fangled GUIs. Bob Scheifler ported the W Window System to Unix and split it up so that the GUI could run on a screen in a computer lab while the actual software was running on a mainframe somewhere. He called it the X Window System!
Display Server

User’s workstation

- Keyboard
- Mouse
- Screen

X Server

- X client (browser)
- X client (xterm)

Network

Remote machine

- X client (xterm)
X in the 1980s

- **X11 application**
- **X server**
  - **DIX (Device-Independent X)**
  - **XAA**
  - **DDX (2D driver)**

  **must be root for directly accessing the hardware (i.e. the (2D) GPU)**

- **Linux kernel**
- **CPU (registers & L1 & L2 & L3 & L4) & main memory**
- **GPU (registers & L1 & L2) & graphic memory**

**DDX (Device-Dependent X)**

**XFree86 Acceleration Architecture**
From MIT to X.Org

- X versions 1-9 (1984/5)
  - X version 10, MIT License (1986)
  - X11 (1987)
  - The MIT X Consortium (1988)
  - X386 (1991)
  - XFree86 (1992)
  - The Open Group (1997)
  - X.Org (1999)
  - The X.Org Foundation (2005)
  - Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
  - X11 (1987)
  - The MIT X Consortium (1988)
  - X386 (1991)
  - XFree86 (1992)
  - The Open Group (1997)
  - X.Org (1999)
  - The X.Org Foundation (2005)
  - Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
  - The MIT X Consortium (1988)
  - X386 (1991)
  - XFree86 (1992)
- The Open Group (1997)
- X.Org (1999)
- The X.Org Foundation (2005)
- Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
- The MIT X Consortium (1988)
  - X386 (1991)
  - XFree86 (1992)
- The Open Group (1997)
- X.Org (1999)
- The X.Org Foundation (2005)
- Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
- The MIT X Consortium (1988)
- X386 (1991)
  - XFree86 (1992)
- The Open Group (1997)
- X.Org (1999)
- The X.Org Foundation (2005)
- Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
- The MIT X Consortium (1988)
- X386 (1991)
- XFree86 (1992)
- The Open Group (1997)
- X.Org (1999)
- The X.Org Foundation (2005)
- Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
- The MIT X Consortium (1988)
- X386 (1991)
- XFree86 (1992)
  - The Open Group (1997)
  - X.Org (1999)
  - The X.Org Foundation (2005)
  - Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
- The MIT X Consortium (1988)
- X386 (1991)
- XFree86 (1992)
- The Open Group (1997)
- X.Org (1999)
- The X.Org Foundation (2005)
- Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
- The MIT X Consortium (1988)
- X386 (1991)
- XFree86 (1992)
- The Open Group (1997)
- X.Org (1999)
  - The X.Org Foundation (2005)
  - Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
- The MIT X Consortium (1988)
- X386 (1991)
- XFree86 (1992)
- The Open Group (1997)
- X.Org (1999)
- The X.Org Foundation (2005)
- Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
- The MIT X Consortium (1988)
- X386 (1991)
- XFree86 (1992)
- The Open Group (1997)
- X.Org (1999)
- The X.Org Foundation (2005)
- Wayland (2008)
From MIT to X.Org

- X versions 1-9 (1984/5)
- X version 10, MIT License (1986)
- X11 (1987)
- The MIT X Consortium (1988)
- X386 (1991)
- XFree86 (1992)
- The Open Group (1997)
- X.Org (1999)
- The X.Org Foundation (2005)
- Wayland (2008)
Games want to draw things in 3D so operating systems started shipping with APIs to draw 3D stuff. Silicon Graphics created a standard interface for 3D graphics called OpenGL. X added an extension called GLX so clients could tell X to draw 3D stuff using the OpenGL API. X is still doing all the math on the CPU.
X11 application  OpenGL application  Framebuffer application

GLX

X server  2D driver

Utah GLX driver

must be root for directly accessing the GPU

must be root

Linux kernel

FB driver

CPU (registers & L1 & L2 & L3 & L4) & main memory

GPU (registers & L1 & L2) & graphic memory

-rooted!

indirect rendering

(rooted!)
This is great and all. We have a device that can keep an image going without the CPU having to redraw the image every time, but what if I want to do nothing but have constant moving images.
A Slightly Smarter GPU

Around the time of the NES, the need for a device to handle graphics started to arise. The NES for example had a Picture Processing Unit (PPU). It had 2kB of Video RAM and 256 bytes of on-die ”memory/cache” to store 64 sprite indices on the screen.
The Use of these "processors"

- The CPU’s from here until the 1990’s were all custom solutions
  - They were used in high end workstations, game consoles, and arcade machines
  - These workstations had highend custom processors all with different names
  - Game consoles used cheaper solutions to having constant moving pixels along with arcade machines
The Use of these "processors"

- The CPU’s from here until the 1990’s were all custom solutions
- They were used in high end workstations, game consoles, and arcade machines
  - These workstations had highend custom processors all with different names
  - Game consoles used cheaper solutions to having constant moving pixels along with arcade machines
The Use of these "processors"

- The CPU’s from here until the 1990’s were all custom solutions
- They were used in high end workstations, game consoles, and arcade machines
- These workstations had highend custom processors all with different names
- Game consoles used cheaper solutions to having constant moving pixels along with arcade machines
The Use of these "processors"

- The CPU’s from here until the 1990’s were all custom solutions.
- They were used in high end workstations, game consoles, and arcade machines.
- These workstations had highend custom processors all with different names.
- Game consoles used cheaper solutions to having constant moving pixels along with arcade machines.
The idea of a Graphics Processing Unit

- By the beginning of the 90’s GUI were the new cool thing
- A lot of computers started having them and it allowed for more people to be able to use computers
- With the release of Windows 95 and ”modern” consoles companies started to do nothing but make these ”processors”
- IBM, Ricoh, and other hardware makers started making these processors.
- They all had a different name but with time everyone settled on GPU after one specific company’s release
The idea of a Graphics Processing Unit

- By the beginning of the 90’s GUI were the new cool thing
- A lot of computers started having them and it allowed for more people to be able to use computers
  - With the release of Windows 95 and ”modern” consoles companies started to do nothing but make these ”processors”
  - IBM, Ricoh, and other hardware makers started making these processors.
  - They all had a different name but with time everyone settled on GPU after one specific company’s release
The idea of a Graphics Processing Unit

- By the beginning of the 90’s GUI were the new cool thing
- A lot of computers started having them and it allowed for more people to be able to use computers
- With the release of Windows 95 and "modern" consoles companies started to do nothing but make these "processors"
  - IBM, Ricoh, and other hardware makers started making these processors.
- They all had a different name but with time everyone settled on GPU after one specific company’s release
The idea of a Graphics Processing Unit

- By the beginning of the 90’s GUI were the new cool thing
- A lot of computers started having them and it allowed for more people to be able to use computers
- With the release of Windows 95 and ”modern” consoles companies started to do nothing but make these ”processors”
- IBM, Ricoh, and other hardware makers started making these processors.
  - They all had a different name but with time everyone settled on GPU after one specific company’s release
The idea of a Graphics Processing Unit

- By the beginning of the 90’s GUI were the new cool thing
- A lot of computers started having them and it allowed for more people to be able to use computers
- With the release of Windows 95 and ”modern” consoles, companies started to do nothing but make these ”processors”
- IBM, Ricoh, and other hardware makers started making these processors.
- They all had a different name but with time everyone settled on GPU after one specific company’s release
Hardware Acceleration

The display adapter has gotten so much smarter. These things play chess now. There is no point calculating the color of buttons CPU-side. But how do you update your software to use this new fangled hardware acceleration?
You need someone to implement OpenGL so that it makes use of the GPU. At first Mesa was just an implementation of OpenGL on the CPU but quickly added support for hardware acceleration.

How can Mesa talk to the GPU if the X Server is hogging it? How can other applications use mesa without the overhead of GLX? By going through the Direct Rendering Manager (DRM).
Direct Rendering Infrastructure

- X11 application
- OpenGL application
- Framebuffer application

- X server
  - 2D driver

- OpenGL
  - DRI driver
  - FB driver
  - drm

- Linux kernel
- CPU (registers & L1 & L2 & L3 & L4) & main memory
- GPU (registers & L1 & L2) & graphic memory

must be root for directly accessing the GPU
Now that X has access to hardware acceleration, we can stop running GLX instructions on the CPU and use the full capabilities of our GPU. A program called the "compositor" can take in other programs’ windows as images (framebuffers) and use the GPU to copy them and move them around. This is necessary if you want transparent windows or if you want your computer to look like a cube.
AIGLX

X11 application

OpenGL application

Framebuffer application

X server

2D driver

OpenGL DRI driver

Linux kernel

CPU (registers & L1 & L2 & L3 & L4) & main memory

GPU (registers & L1 & L2) & graphic memory
xdpyinfo -queryExtensions | less
Wayland

Since everyone is using direct rendering to draw everything now anyway, we could get rid of all the complexity and make the display server, compositor, and window manager simple applications which manage input events and move other applications’ framebuffers around.

Enter Wayland! Wayland is a simpler display server that serves modern applications well. But Wayland and the X server can run side-by-side with X acting as any other Wayland client.
The Source of Your Nvidia Problems
Business Strategies

- AMD: Make our hardware the standard.
- Nvidia: Sell software features people depend on.
Business Strategies
- Nvidia Proprietary Driver
  - Closed-source, created by Nvidia
  - Better performance
  - Only works with X
- Nouveau
  - Open-source,\(^1\) reverse-engineered by community
  - Plays nicer with Linux graphics stack

\(^1\)MIT License
Graphics Switching

- Having a discrete GPU is nice... but so is having battery life.
- **Nvidia OPTIMUS**: switch between Intel integrated graphics and Nvidia discrete GPU depending on task
- Unsupported by proprietary driver
Having a discrete GPU is nice... but so is having battery life.

**Nvidia OPTIMUS**: switch between Intel integrated graphics and Nvidia discrete GPU depending on task

- Unsupported by proprietary driver
Graphics Switching

- Having a discrete GPU is nice... but so is having battery life.
- **Nvidia OPTIMUS**: switch between Intel integrated graphics and Nvidia discrete GPU depending on task
- Unsupported by proprietary driver
“Nvidia, fuck you!”
—Linus Torvalds, 2012
Graphics Switching Options For Linux

- Disable the GPU or integrated graphics in BIOS
- **VGA switcheroo**: hardware graphics mux, if present, is sys/kernel/debug/vgaswitcheroo/switch
- **PRIME**: GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee**: effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
  - Use VirtualGL or primus as bridge (primus is faster)
  - Performance isn't great
Graphics Switching Options For Linux

- Disable the GPU or integrated graphics in BIOS
- **VGA switcheroo**: hardware graphics mux, if present, is `sys/kernel/debug/vgaswitcheroo/switch`
- **PRIME**: GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee**: effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
  - Use VirtualGL or primus as bridge
  - *primus* is faster
  - Performance isn't great
Graphics Switching Options For Linux

- Disable the GPU or integrated graphics in BIOS
- **VGA switcheroo**: hardware graphics mux, if present, is `sys/kernel/debug/vgaswitcheroo/switch`
- **PRIME**: GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee**: effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
  - Use VirtualGL or primus as bridge (primus is faster)
- Performance isn't great
Graphics Switching Options For Linux

- Disable the GPU or integrated graphics in BIOS
- **VGA switcheroo**: hardware graphics mux, if present, is `sys/kernel/debug/vgaswitcheroo/switch`
- **PRIME**: GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee**: effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
  - Use VirtualGL or primus as bridge (primus is faster)
  - Performance isn't great
Graphics Switching Options For Linux

- **Disable the GPU or integrated graphics in BIOS**
- **VGA switcheroo**: hardware graphics mux, if present, is `sys/kernel/debug/vgaswitcheroo/switch`
- **PRIME**: GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee**: effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
  - Use VirtualGL or primus as bridge
  - Performance isn't great
Graphics Switching Options For Linux

- Disable the GPU or integrated graphics in BIOS
- **VGA switcheroo**: hardware graphics mux, if present, is `sys/kernel/debug/vgaswitcheroo/switch`
- **PRIME**: GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee**: effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
  - Use VirtualGL or primus as bridge
  - (primus is faster)
  - Performance isn’t great
- **Disable the GPU or integrated graphics in BIOS**
- **VGA switcheroo**: hardware graphics mux, if present, is `sys/kernel/debug/vgaswitcheroo/switch`
- **PRIME**: GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee**: effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
  - Use VirtualGL or primus as bridge
  - Performance isn’t great
Graphics Switching Options For Linux

- Disable the GPU or integrated graphics in BIOS
- **VGA switcheroo**: hardware graphics mux, if present, is sys/kernel/debug/vgaswitcheroo/switch
- **PRIME**: GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee**: effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
    - Use VirtualGL or primus as bridge
    - Performance isn’t great
Graphics Switching Options For Linux

- Disable the GPU or integrated graphics in BIOS
- **VGA switcheroo:** hardware graphics mux, if present, is `sys/kernel/debug/vgaswitcheroo/switch`
- **PRIME:** GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee:** effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
  - Use VirtualGL or primus as bridge
    (primus is faster)
  - Performance isn’t great
- **Disable the GPU or integrated graphics in BIOS**
- **VGA switcheroo**: hardware graphics mux, if present, is `sys/kernel/debug/vgaswitcheroo/switch`
- **PRIME**: GPU offloading in Linux kernel
  - Manually switch the active GPU
  - Requires logging out and logging in again
- **Bumblebee**: effort to replicate OPTIMUS
  - Disable Nvidia GPU when not necessary
  - Render on Nvidia GPU when necessary, then send output to integrated graphics to be displayed
  - Use VirtualGL or `primus` as bridge
  - Performance isn’t great
This presentation was from the Mines Linux Users Group. A mostly-complete archive of our presentations can be found online at https://lug.mines.edu.

Individual authors may have certain copyright or licensing restrictions on their presentations. Please be certain to contact the original author to obtain permission to reuse or distribute these slides.