# Boot-up-Grafik unter Linux: von der Firmware bis zum Login

System-Firmware bietet dem Kernel beim Start häufig einen einfachen Framebuffer für die Ausgabe an. Mit Hilfe von efidrm, simpledrm, u.a. kann der Linux-Kernel daher nun seit einiger Zeit einen modernen Grafik-Stack auf fast beliebiger Grafik-Hardware ausführen. Vortrag geht es unter anderem um die weiter unten folgenden Themen. Außer einem Interesse an Linux und Hardware-nahen Themen werden keine Voraussetzungen gemacht. Wer sich über den Grafikstack unter Linux informieren will, kann z.B. in meine Artikel [1][2] bei LWN.net reinschauen.

**Firmware und Bootloader** Auf PC-Systemen lässt sich die Display-Konfiguration nach Systemstart per UEFI oder VESA auslesen; auf eingebetteten Systemen werden die Informationen per DeviceTree bereit gestellt. Dazu gehören Auflösung, Farbformat, Länge einer einzelnen Bildzeile und die Speicheraddresse des Puffers. Je nach System lassen sich die Einstellungen im Bootloader noch ändern, aber spätestens mit Start des Kernels sind die Interfaces der Display-Firmware nicht mehr verfügbar. Weiterhin verfügbar ist allerdings der bereits konfigurierte Framebuffer.

**Kernel** Auf PC-Systemen werden die Einstellungen per Boot-Parameter vom Bootloader and den Kernel übergeben. Nach dem Start liest der Kernel die Werte aus und erstellt ein passendes Platform-Gerät für den Framebuffer. Auf Systemen mit DeviceTree wird das Platform-Gerät vom DeviceTree-Setup erstellt. Ab diesem Zeitpunkt ist der Firmware-Framebuffer für Gerätetreiber nutzbar.

**Kerneltreiber** Seit jeher bietet der Linux-Kernel passende Treiber im alten Fbdev-Framework. Diese Treiber bieten Kernel-Konsole oder Userspace Zugriff auf den vorkonfigurierten Framebuffers. Seit Kernel v5.14 bietet das DRM-Framework ähnliche Treiber. Im Gegensatz zu nativen Hardware-Treibern sind die generischen Treiber klein genug um im Kernel-Image oder der Boot-Partition gespeichert zu werden. Diese Treiber werden während des Starts des Linux-Systems verwenden bis native Gerätetreiber für die Hardware verfügbar sind. Zum Beispiel zeigt die Kernel-Konsole Start-Up-Logs oder Plymouth zeigt eine Animation an. Die Eingabe des Festplatten-Passwortes erfolgt ebenfalls mit Hilfe der generischen Ausgabe.

**Laden das nativen Treibers** Sobald der native Treiber der Display-Hardware verfügbar ist, muss der generische Treiber und das zugrunde liegende Gerät entfernt werden. Der native Treiber bindet meistens an ein PCI-Gerät. Liegt der generische Framebuffer im Grafikspeicher des PCI-Geräts, entfernt der native Treiber das Platform-Gerät per Hot-Unplug. Der generische Treiber wird dadurch entladen. Auf Systemen ohne dedizierten Grafikspeicher entfernt der native Treiber alle generischen Geräte. Die Hardware ist nun zur Benutzung verfügbar. Der Rest des Bootvorgangs bis hin zum laufenden System benutzt den nativen Treiber für die Grafikausgabe.

**Fehlerbehandlung** Der Kernel-Parameter *nomodeset* unterbindet das Laden der nativen Grafiktreiber. Probleme mit Hardware oder nativen Treiber lassen sich so abmildern. Die generischen DRM-Treiber bieten jedoch alle zentralen Funktionen des modernen Grafikstacks: Speicherverwaltung, Ioctls für Kernel-Modesetting, Buffer-Sharing per DMA-BUF-Framework. Selbst ohne funktionierenden Hardware-Treiber lassen sich so die meisten Funktionen des Linux-Desktops benutzen.

**Exotische Hardware** Das funktioniert auch ohne Hardware-Treiber. Falls Linux keinen Treiber für eine Grafikkarte anbietet, kann der generische Framebuffer benutzt werden. Die meiste Software unterstützt Pixel mit 32-Bit-XRGB-Format. Falls nötig emulieren die generischen DRM-Treiber dieses Format intern. Selbst wenn die Ausgabe nur ein 16-Bit- oder 8-Bit-Format bietet, muss die Software nicht angepasst werden. Bei Bedarf können fast beliebige Ausgabeformate unterstützt werden.

**Monitor** Monitore und ähnliche Geräte kodieren ihre Eigenschaften im EDID-Format. Zusammen mit dem Modesetting bieten Firmware-Schnittstellen häufig auch diese Daten an. Seit Kernel v6.16 exportieren die generischen DRM-Treiber auch EDID-Informationen an den Userspace. Kompositoren können damit die DPI-Werte korrekt einstellen. Unter VESA ist sogar Farbkorrektur möglich.

**Fbdev** Das Fbdev-Subsystem plus Xorg diente als Fallback für Systeme ohne nativen Grafiktreiber. Die Fbdev-Treiber bieten aber nur die Konsole und eingeschränkt Xorg. Mit den neuen generischen DRM-Treibern fällt dieser letzte Anwendungsfall weg. Die großen Desktops und Distributionen stellen daher zunehmend die Unterstützung der alten Umgebung ein, was zu kleineren und einfacheren Systemen führt.

## Literatur


[1] ZIMMERMANN, Thomas: *The Linux Graphics Stack in a Nutshell, Part 1.* `https://lwn.net/Articles/955376/`

[2] ZIMMERMANN, Thomas: *The Linux Graphics Stack in a Nutshell, Part 2.* `https://lwn.net/Articles/955708/`