# Red Hat Enterprise Linux Kernel Device Drivers

**Duration: 5 Days** **Course Code: RHD362**

## Overview:

Red Hat® Enterprise Linux® Kernel Device Drivers (RHD362) teaches experienced C programmers already familiar with the Linux kernel architecture the skills and strategies they need to develop device drivers.
The course covers device driver models (including character, block, and network device drivers); device interaction (including port I/O, memory mapped I/O, interrupt handling, and DMA transfers); managing PCI and USB devices; strategies for deferring activity using tasklets and work queues; device registration using the Unified Device model and the sysfs filesystem; and process interaction, including basic file operations, polling, and wait queues.

## Target Audience:

Experienced C programmers with a good understanding of the Linux kernel who want to learn how to develop device drivers for Linux systems.

## Objectives:

- Introduction and Review of Kernel Programming
- Device Drivers
- Unified Device Model
- Interrupt Handling
- Advanced File Operations
- Interacting With Devices
- Direct Memory Access
- PCI Drivers
- USB Drivers
- Introduction to Network Device Drivers
- Introduction to Block Device Drivers

## Prerequisites:

- Experience in C programming
- Red Hat Enterprise Linux Kernel Internals (RHD361) or equivalent experience

## Content:

- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures

- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets

- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures

- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel

- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation

- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types

- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions

- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs

- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues

- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules

- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

Advanced File Operations

- IOCTLs

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree

- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree

- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information

- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks

- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks

- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O

- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues

- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues

- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem

- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers

- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers

- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission

- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization

- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization

- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations

- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access

- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access

- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

<u>Device Drivers</u>

- Device Drivers

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions

- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations

- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations

- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions

- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission

- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission

- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers

- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations

- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations

- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts

- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together

- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together

- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration

- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types

- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices

- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

Introduction to Network Device Drivers

- Networking

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space

- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse

- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures

- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents

- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules

- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure

- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types

- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources

- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls

- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

Interacting With Devices

- Device Interaction

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context

- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context

- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration

- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers

- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA

- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions

- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions

- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together

- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules

- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules

- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations

- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests

- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests

- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device

- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()

- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()

- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

Unified Device Model

- Device Model Objects

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices

- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

Direct Memory Access

- Managing DMA

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures

- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards

- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices

- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors

- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O

- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers

- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues

- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration

- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

Introduction to Block Device Drivers

- Block Devices

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration

- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules

- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)

- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)

- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks

- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

<u>PCI Drivers</u>

- Peripheral Component Interconnect

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access

- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers

- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()

- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations

- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem

- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes

- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission

- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions

- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)

- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations

- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers

- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers

- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together

- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices

- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

Interrupt Handling

- Interrupt Context

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations

- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration

- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example

- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations

- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices

- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices

- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space

- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures

- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets

- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration

- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel

- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types

- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces

- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices

- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues

- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs

- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues

- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Linux Kernel Variants
- Interacting with the Kernel
- Kernel Implementation
- The Linux Source Tree

- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks
- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests
- Writing Kernel Modules
- Compiling Kernel Modules
- More about Kernel Modules
- Review of Kernel Structures
- Device Driver Design
- Interacting with Character Device Drivers
- Interacting with Block Device Drivers
- Interacting with Network Interface Cards
- Device Nodes
- Character Device Drivers: Registration
- The Device Node Problem
- File Operations
- Defining File Operations
- Registering File Operations
- File I/O Conventions
- The sysfs filesystem
- Device Model Structures
- Tying it all Together
- Netlink Sockets and User Events
- Userspace event agents
- Device Registration
- Miscellaneous Devices
- Handling Interrupts
- Interrupt Structures
- ISR Registration
- System Context
- Softirqs
- Deferred Activity: Tasklets
- Deferred Activity: Work Queues
- Custom Work Queues
- Example: Serial Mouse
- IOCTL Conventions
- Implementing IOCTLs
- Lockless IOCTLs
- Sleeping
- Wait Queues
- Wait Queues
- Variations on wait_event()
- Sleeping in System Calls
- Multiplexed I/O
- Polling
- Managing Exclusive Access
- I/O Ports
- I/O Memory Buffers
- Accessing I/O Memory Buffers
- Userspace Drivers
- Initializing DMA
- Coherent Allocations
- Streaming DMA Mappings
- PCI Configuration Space
- PCI Driver Registration
- Registering PCI Drivers
- Driver Probe Function
- Initializing PCI devices
- Releasing a PCI Device
- Managing PCI Resources
- Accessing Configuration Space
- Listing PCI Devices
- USB Descriptors
- USB Transfer Types
- Linux USB Implementation
- USB Drivers
- USB Registration
- Initializing USB Interfaces
- USB Class Drivers
- USB Request Blocks

- Using USB request blocks
- Synchronous USB Requests
- More information
- Network Devices
- Network Device Callback Functions
- Network Driver Initialization
- Network Driver Transmission Queue Control
- Network Statistics
- Socket Buffers (Packets)
- Socket Buffer API
- Transmission
- Reception
- Example Implementations
- I/O Scheduling
- Linux 2.6 Elevator Functions
- Block Device Driver Implementation
- Block Device Registration
- Block Device File Operations
- The gendisk structure
- Request Queues
- Initialization Example
- Handling Requests

## Further Information:

For More information, or to book your course, please call us on 00 966 92000 9278

training@globalknowledge.com.sa

www.globalknowledge.com/en-sa/

Global Knowledge - KSA, 393 Al-Uroubah Road, Al Worood, Riyadh 3140, Saudi Arabia