I/O Kit Device Drivers for Mac OS X

Designing and debugging your driver

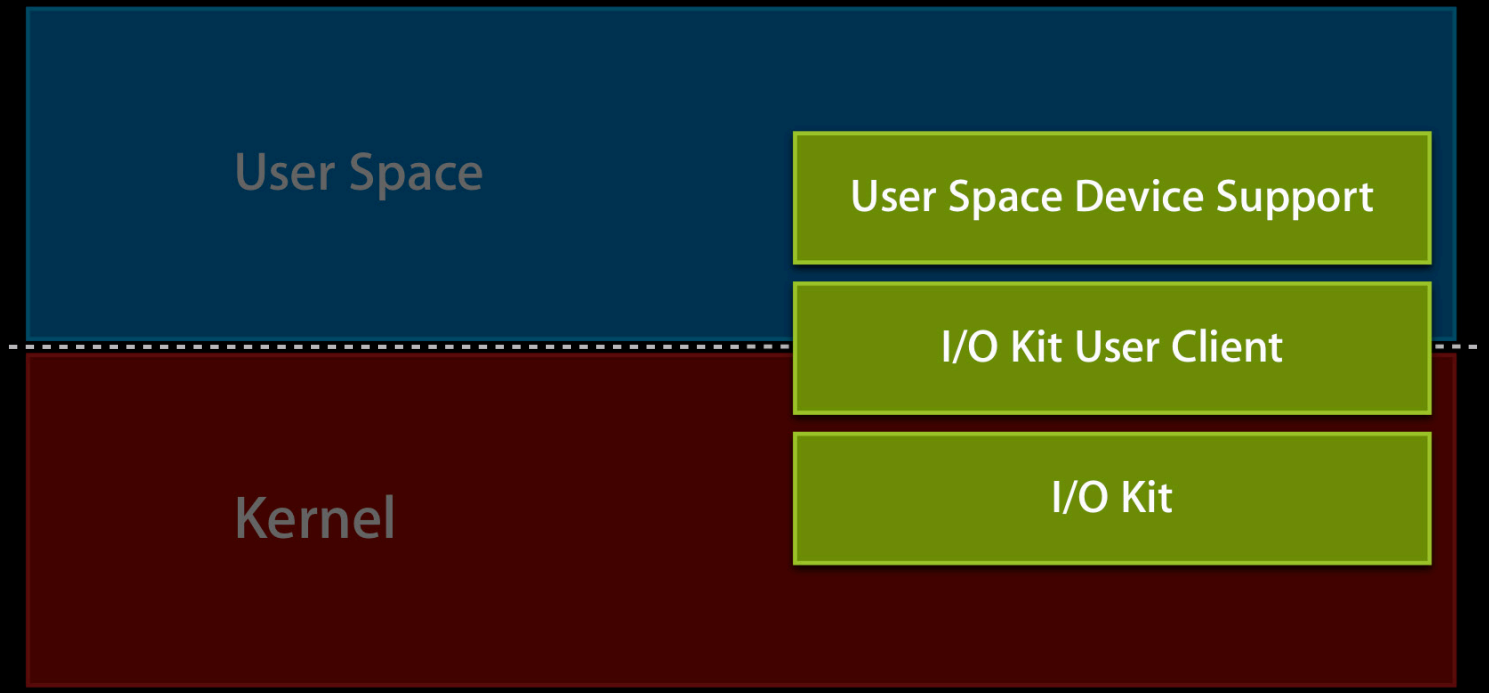
Thane Norton
I/O Kit Team Member

What You Will Learn

- Driving your hardware from user space or the kernel
- The kinds of drivers that have to be in the kernel
- Ways to debug kernel level code
- Special challenges involved in creating your own IOUserClient

What Is I/O Kit?

A set of frameworks for driving hardware



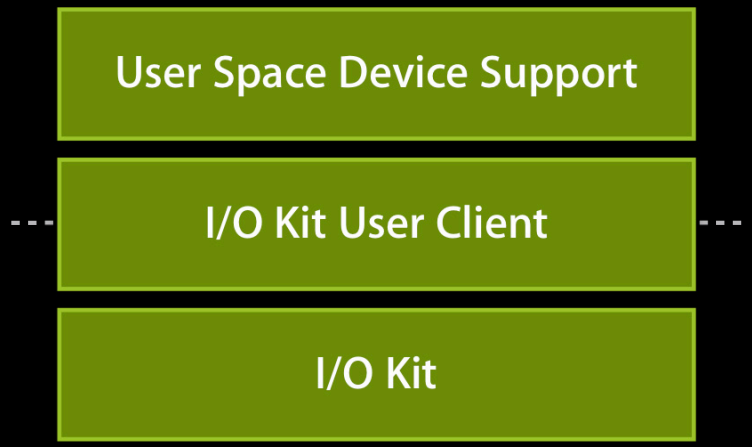
What Is I/O Kit?

Only on Mac OS

- Device driver model for Mac OS X
- Framework for applications to access devices

- I/O Kit is not available in the iPhone SDK
  - See session:

    Developing Apps that Work with iPhone Accessories



User Space I/O Kit Device Support

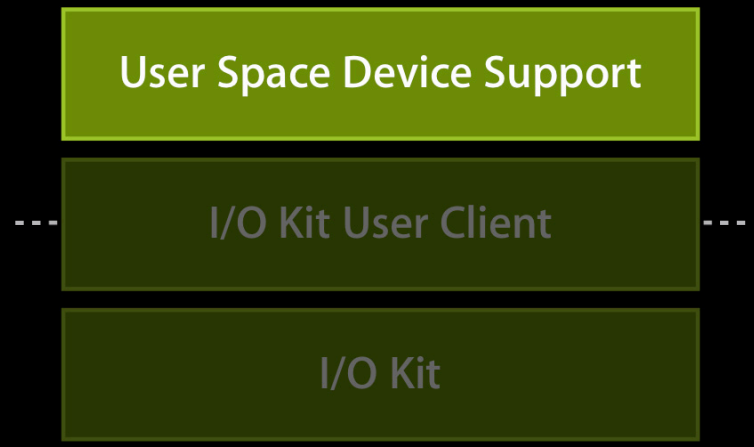
The place you want to be

- If you only need to support your own application, build support into your app
- Requires only IOKit.framework
- Drag and drop install of app
- Multiple apps can share access using I/O Kit

User Space I/O Kit Device Support

The place you want to be

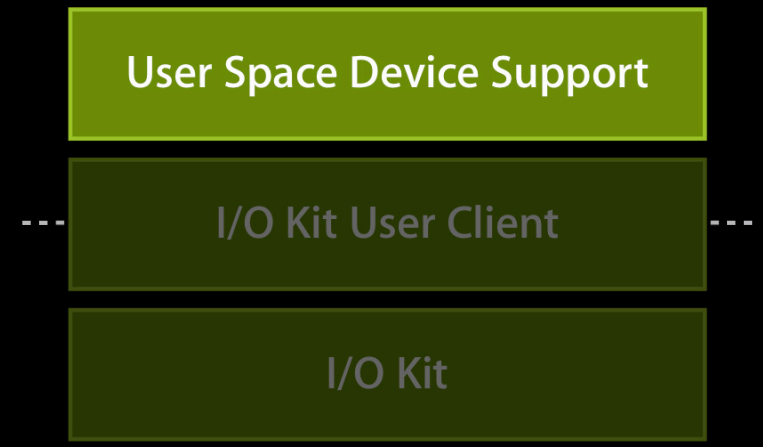
- If multiple applications need to share device support code, create a framework
- Can still be built into an application
- Will need an installer otherwise



User Space I/O Kit Device Support

The place you want to be

- If you need to supply services, create a daemon or background application
- Can use launchd to be launch on demand
- See also:
  - `man launchd.plist`
  - See also session: Launch-on-Demand



User Space I/O Kit Device Support

The place you want to be

- Easier to debug
- More robust
- More access to system services
- Better logging
- Should not cause panics

User Space I/O Kit Device Support

- Nearly full access to hardware through I/O Kit user clients
  - Almost any USB device can be supported
- Better memory management
- Identical control over thread priority
  - Kernel tasks are not treated specially

User Space I/O Kit Device Support

Testing matrix

	i386 or x86_64	PowerPC or PowerPC64
10.6	✓	
10.5	✓	✓
10.4		✓

User Space I/O Kit Device Support

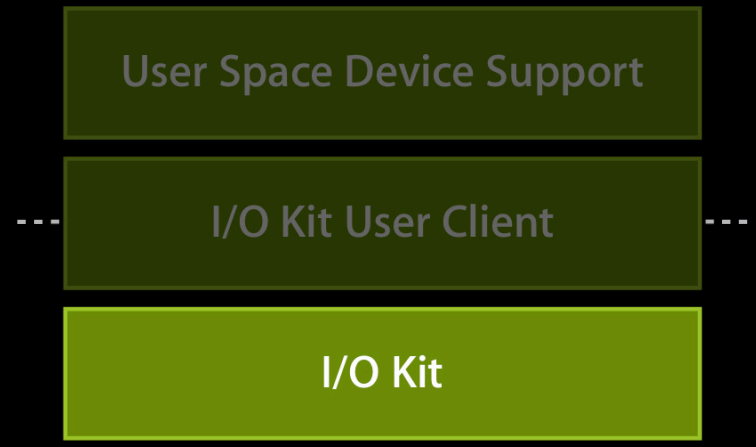
Not possible when…

- Your client is in the kernel
- You require access to other kernel resources
- You need to respond directly to primary interrupts
  - PCI drivers must be kexts

I/O Kit Kernel Driver

a.k.a. kext (kernel extension)

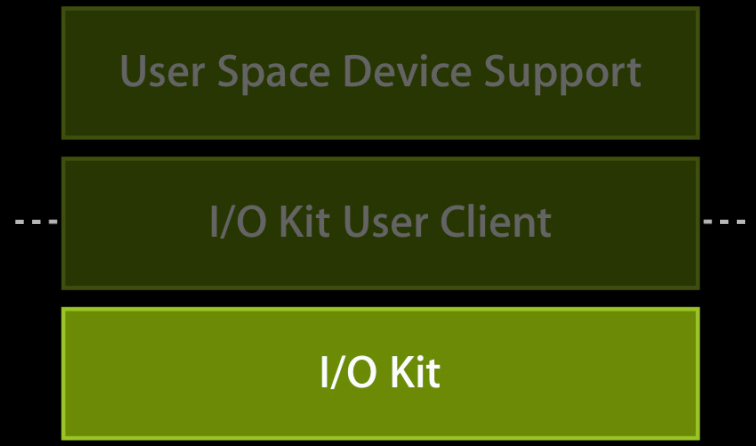
- Read the "Kernel Extension Programming Topics"
  - One day of work can get you a pretty good workflow
- Memory allocated by kexts is pre-wired
- Memory from user space is not
  - You must prepare memory from user space before performing physical I/O



I/O Kit Kernel Driver

a.k.a. kext (kernel extension)

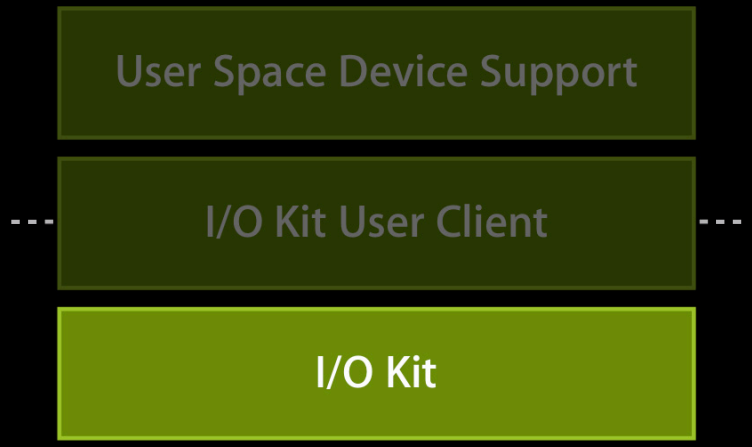
- Logging is limited
  - Uses ring buffers
  - Flooding the log will cause messages to drop and be garbled
  - Can't use Apple System Log
  - Can use kprintf() for FireWire logging
    - See `man fwkpfv`



I/O Kit Kernel Driver

a.k.a. kext (kernel extension)

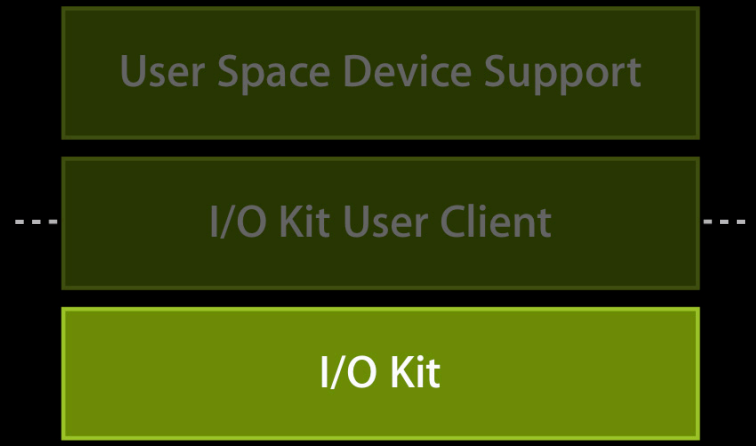
- Test cycle often requires reboot
- Debugging
  - Requires advance preparation
  - Requires two machines
  - Connected via Ethernet or FireWire
  - Use the Kernel Debug Kit for your kernel
- Read the "Kernel Extension Programming Topics"
  - Also see "man fwkdp"



I/O Kit Kernel Driver

a.k.a. kext (kernel extension)

- Must test kext on all supported kernels every release
- Must match kernel architecture
  - i386 compilation for 32-bit Intel kernel
  - x86_64 compilation for 64-bit kernel
  - 32-bit PowerPC compilation for a Leopard PowerPC kernel



I/O Kit Kernel Driver

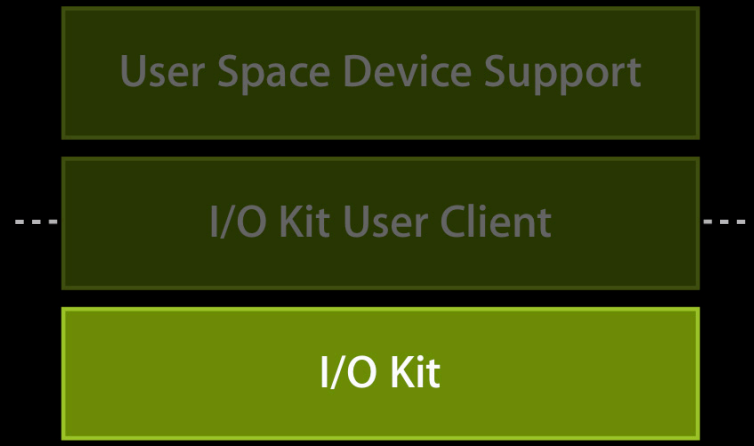
Testing matrix

			
	x86_64	i386	PowerPC
10.6	✓	✓	
10.5		✓	✓
10.4			✓

I/O Kit Kernel Driver

The art and skill of multi-architecture compilation

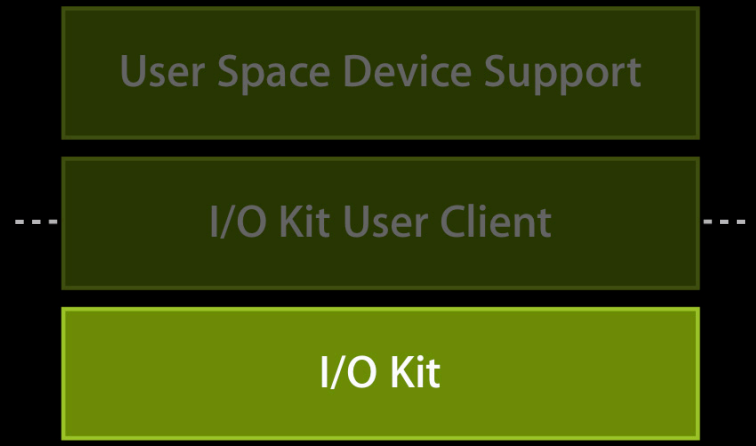
- Memory descriptors changed on 10.5 to support x86_64 user processes
  - Kernel was 32-bit only
- Use conditional compilation and availability macros
  - Weak linking not (yet) supported



I/O Kit Kernel Driver

The art and skill of multi-architecture compilation

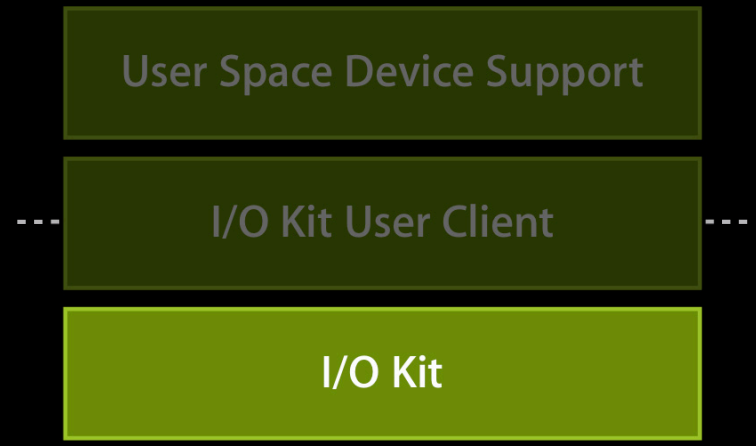
- kexts can be nested to ease packaging
- Can easily support all Leopard or better kernels with one kext
  - Support for earlier kernels is possible
- From WWDC09 see session: Creating I/O Kit Drivers for Multiple Architectures and OS Versions



I/O Kit Kernel Driver

Panics

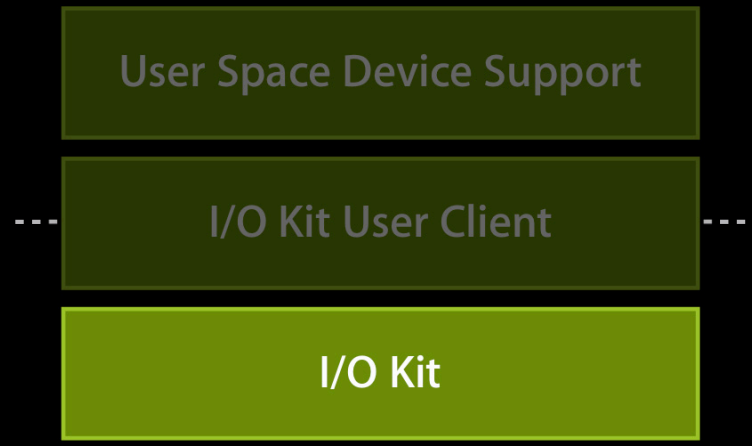
- Harder to debug
- Two-machine debugging works well
  - If you have it set up
- Setting machines up to "dump core" can be a big boon
  - See TN2118 and `man fwkdp`
- Covered in depth by TN2063



I/O Kit Kernel Driver

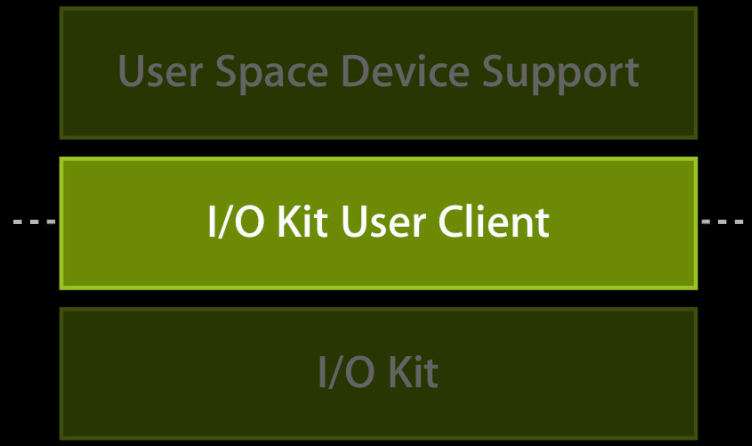
Summary

- Closer access to hardware
- More challenging than user space
- New class of defects (panics)
- Limited system resources



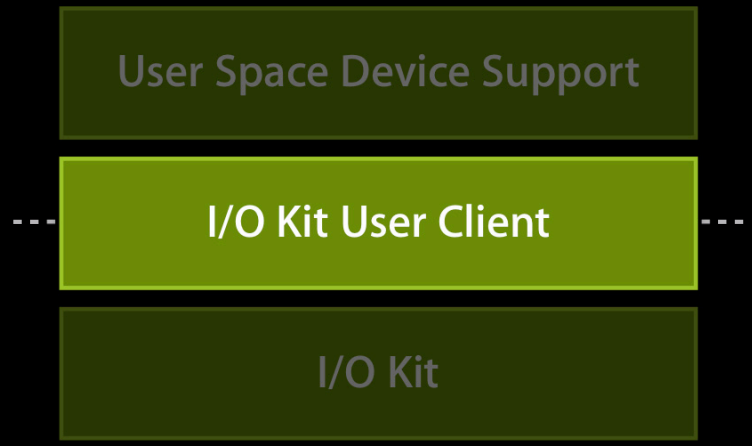
I/O Kit Custom User Client

- Custom interface across the user/kernel boundary
- Even harder to debug than a kext
  - Must debug two processes in different instruction/memory spaces
- There is almost always a better choice



I/O Kit Custom User Client

- Code running in the kernel is trusted
- IOUserClient allows you to penetrate the trust boundary
- If you make a custom user client, you become the gatekeeper



I/O Kit Custom User Client

- Must validate…
  - All data coming in
  - All connections from applications
    - See initWithTask and clientHasPrivilege for more info
- See also:
  - SimpleUserClient
  - AppleSamplePCI

I/O Kit Custom User Client

		x86_64	i386	PowerPC
10.6	x86_64 client	✓	✓	
	i386 client	✓	✓	
	PowerPC 32 client	✓	✓	
10.5	x86_64 client		✓	
	i386 client		✓	
	PowerPC 64 client			✓
	PowerPC 32 client		✓	✓
10.4	x86_64 client		✓	
	i386 client		✓	
	PowerPC 64 client			✓
	PowerPC 32 client		✓	✓

I/O Kit Custom User Client

Without Rosetta

		x86_64	i386	PowerPC
10.6	x86_64 client	✓	✓	
	i386 client	✓	✓	
	PowerPC 32 client	✗	✗	
10.5	x86_64 client		✓	
	i386 client		✓	
	PowerPC 64 client			✓
	PowerPC 32 client		✗	✓
10.4	x86_64 client		✓	
	i386 client		✓	
	PowerPC 64 client			✓
	PowerPC 32 client		✗	✓

PCI and Power Management

Thane Norton
I/O Kit Team Member

PCI Message-Signaled Interrupts

- Preferred interrupt technique (less interrupt sharing)
- MSIs allow additional interrupt sources available via IOService
  - Index 0 is reserved for the legacy pin based shared interrupt

```
Pin = IOFilterInterruptEventSource::filterInterruptEventSource
                (this, interruptHandler, interruptFilter, provider, 0);
```

  - MSI interrupts do not need a filter, and have indices > 0

```
MSI = IOInterruptEventSource::interruptEventSource
                (this, interruptHandler, provider, 1);
```

- Enabing an MSI source disables legacy pin interrupt source
- Supported on all Intel hardware running 10.4.7 or later

Power Management

User space

- Better performance = better battery life
- Improving performance is the number one thing you can do
- Periodic activity is worse than batching
- Reduce sporadic disk access
- Use System Load Advisory API to guide background behavior
- Use System Power messages from IOKit.framework to discover system power state changes
- Use IOPMAssertion APIs to prevent idle sleep when necessary
- Should not do anything different for SafeSleep

Power Management

Kernel and kext

- Any sleep can become a SafeSleep if power is lost
  - Should not do anything different for SafeSleep
- Use IOService APIs to join power tree
  - Let you request and be notified of PM state changes
  - kexts only implement mechanism; policy belongs in user space
- Maintenance Wake
  - New for Snow Leopard added for Bonjour network presence
  - Brief, partial wake with screen and audio off
    - Triggered in Bonjour Sleep Proxy server is active
    - Limited to 30 seconds

Productizing Your Driver

Dean Reece
I/O Kit Team Manager

What You'll Learn

- How to take a driver from prototype to product
- How to deliver your driver to your customers
- What resources are available to help you

Qualification

- A simple checklist can catch many common errors before your driver gets into the field
  - Basic correctness checks
  - Common transition cycling
  - Memory footprint analysis
- Keep records of results for comparing between releases

Qualification

Examine your Info.plist

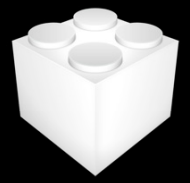
- Is “IOKitDebug” absent or set to “0”?
- Did you advance your version number?
- Is your “CFBundleIdentifier” correctly formed?

```
com.yourcompany…
```

- Remember “OSBundleCompatibleVersion” is only used for libraries!
- Check your use of “OSBundleRequired”

Qualification

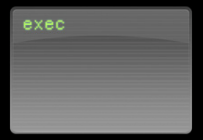
Kext bundle structure



- Run “find” on the kext and make sure every file is expected
- Doublecheck owner and permissions
  - Files should be root:wheel 644
  - Folders should be root:wheel 755
- Run “kextutil -tn” on kext and investigate any warnings or errors
  - Prior to SnowLeopard, use “kextload -tn”
  - Use “kextlibs” to help fix issues with OSBundleLibraries

Qualification

Kext bundle binary



- Make sure Xcode build configuration is set to “release”
- Verify correct architectures are present (use “file” command)
  - Expect i386, x86_64, and possibly ppc
- Run “nm” on the kext binary to see what symbols are present
  - Use C++filt and grep to make results more readable:

```
nm driver.kext/Contents/MacOS/driver | c++filt | grep -v " U "
```

  - Are all global symbols properly prefixed?

```
com_yourcompany_…
```

Qualification

Loading and running the kext

- Watch system.log and kernel.log while loading and starting your kext
  - Are debugging log messages present?
  - Investigate any warnings reported by IOKit or kext management
- Unload your kext and verify it unloads cleanly
  - Failure to unload can indicate reference leaks
- Things to verify
  - System sleeps/wakes correctly while driver is in use
  - Driver unloads after device is removed (if applicable)
  - Driver behaves correctly on SafeBoot (hold shift key down)

Qualification: Cycling

Cycling

- Use available tools to monitor resource usage while cycling:
  - ioclasscount, ioalloccount, zprint, and top
  - Observe values at start and watch for steady ramp in consumption
- Things to exercise:
  - kext load/unload
  - Device attach/detach
  - Driver open/close
  - System sleep/wake (use SleepX)
  - Common product-specific transitions
- Set goals for each cycler (1,000 cycles) as a quality metric

Packaging Your Kext

- You can use PackageMaker.app to create a software installation bundle for Installer.app
- Automatically follows the correct steps to install kexts correctly
  - Permissions correctly managed
  - No need to touch Extensions folder
- This is fully described on the Apple Developer website:
  - Packaging a Kernel Extension for Distribution and Installation

Installing Your Kext

- Install drivers in /System/Library/Extensions
  - Do not install them in /Library/Extensions
  - Can be located in app bundle if loaded explicitly by the app
- If delivering multiple kexts as a single product, you can nest them one level deep inside the PlugIns folder of a single kext
- Always touch the extensions folder after adding, updating, or removing kexts: `touch /System/Library/Extensions`
  - You should not directly manage the kext caches
- To install a kext without requiring a restart, see **Technical Q&A QA1319**

Installing Your Kext

If you use an alternate distribution format, make sure that…

- The installed kext matches your built kext (files, contents, permissions)
- The Extensions folder is touched even after an upgrade install
- Old kext bundle contents don’t survive an upgrade install

Getting Help

Apple's developer website is your hub

- Hardware & Drivers page: http://developer.apple.com/hardwaredrivers
- Developer Forums: http://developer.apple.com/devforums
  - 64-Bit migration forum
- Darwin and other mail lists
  - darwin-kernel, darwin-drivers, darwin-development
  - usb, firewire, ata-scsi-dev
- Bug Reporter: http://developer.apple.com/bugreporter
- DTS Incident

More Information

Craig Keithley
I/O Technology Evangelist
keithley@apple.com

Documentation
http://developer.apple.com/hardwaredrivers

Apple Developer Forums
http://devforums.apple.com

Labs

Mac OS X Kernel Lab	Core OS Lab B Wednesday 4:30PM
USB and FireWire Lab	Core OS Lab A Wednesday 9:00AM
USB and FireWire Lab	Core OS Lab B Thursday 2:00PM
USB and FireWire Lab	Core OS Lab A Friday 9:00AM
Bluetooth Lab	Core OS Lab B Wednesday 9:00AM
iPhone OS Accessories Lab	Core OS Lab B Tuesday 2:00PM

Q&A


WWDC10

The last slide after the logo is intentionally left blank for all