Power Efficiency in OS X
Technology Overview

Introduction

The MacBook line of portable computers provides an amazing combination of power, portability, and battery life. The newest generation of Mac hardware enables breakthrough energy efficiency for even better battery life. Realizing the full potential of this new hardware requires a deeply integrated approach where system software and applications are optimized to be even smarter about using power.

Because Apple is responsible for all of the key components of the Mac—the hardware, the operating system, and the low-level firmware—we are uniquely able to address the challenges of maximizing battery life.

OS X Mavericks helps make the most of the energy-saving capabilities of your Mac hardware with optimizations at every level of the operating system, from the kernel and application frameworks to the built-in applications you use every day.

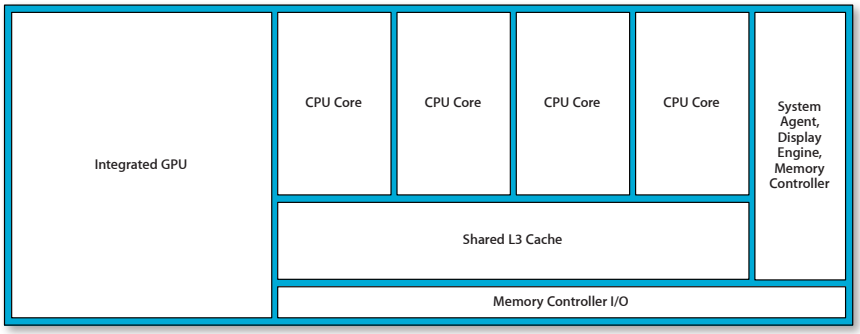
A Power Primer

CPUs

For most of the history of personal computers, microprocessors gained performance by running at faster and faster clock speeds. But with higher clock speeds came not only more computing performance, but also higher heat and power consumption. In fact, the amount of power that a CPU consumes as clock speed rises increases in a nonlinear way, so running a CPU at double the clock speed can require significantly more than double the power. Because of these issues, CPU vendors shifted their focus from cranking up clock speed to putting multiple processor cores into a single CPU in order to use less total energy per unit of computing power.

Nearly all modern computers use a feature called dynamic voltage scaling, which adjusts the clock speed and voltage of the CPU based on the current demands of the software running on the system. This voltage scaling is transparent to the apps on the system; it's managed by low-level firmware that selects the optimal clock frequency and voltage for a given workload.

With multiple processor cores, processors have the opportunity to run different processors at different power levels. Much like you can save power at home by shutting off lights in unoccupied rooms, modern CPUs can shut down individual processor cores or other components when they are not needed, for more energy savings. In addition, a processor that is still turned on can save power by entering an extremely low-power "idle" state for as little as a few microseconds at a time.



This block diagram shows the components of a modern CPU. OS X can manage the activity of different components of a processor independently, helping to balance power consumption and performance.

Processor idle

When you're sitting in front of your Mac not interacting with any apps, it may be using as little as 1 or 2 percent of the available CPU in order to perform background tasks. A task like launching a large application might take that utilization briefly up to 40 percent, but there are relatively few cases where the applications need to use all of the available CPU much of the time. One of the key energy-saving strategies for CPUs is known as idling: reducing the clock speed and the amount of power flowing to the processor core, sometimes to the point of turning off a processor core altogether. The effectiveness of idling as a power-saving strategy depends on the level of demand for resources that the operating system and applications place on the processor.

OS X Mavericks Power Technologies

The energy-efficiency technologies in OS X Mavericks are built with the capabilities of modern processors and the demands of modern apps in mind. These new technologies work together to achieve substantial power savings, while maintaining—and in some cases even improving—the responsiveness and performance of your Mac.

These technologies are rooted in a few key principles:

- They work without the need to modify existing apps, though small changes may lead to additional power savings.
- They keep as much of the hardware idle as possible given the demand for resources.
- When the Mac is on battery power, they do only work that the user is requesting or that is absolutely essential.

App Nap

While your Mac can run many different apps at the same time, the reality is that you're typically interacting with only one of them, and in some cases (such as using a full-screen app), you can't even see what the other apps are doing. App Nap takes advantage of this fact by putting applications that you can't see into a special low-power state that regulates their CPU usage as well as network and disk I/O. App Nap can be automatically triggered in the following situations:

- The app's windows are not visible.
- The app is not playing audio.
- The developer has not explicitly made the app exempt from App Nap by using the existing IOKit `IOPMAssertion` API (currently used in OS X to prevent the system from sleeping while an app is busy).
- An app is in background and hasn't drawn recently.

App Nap uses a number of power-saving measures, including:

- **Timer throttling**—Reduces the frequency with which an app's timers are executed. This can mean significant improvements in CPU idle time when running applications that frequently check for data.
- **I/O throttling**—Assigns the lowest priority to disk or network activity associated with a napping app. The rate at which an application can read or write data from a device is significantly reduced. In addition, I/O throttling reduces the chances that a background process will interfere with the I/O activity of an app that you are actively using.
- **Priority reduction**—Reduces the UNIX process priority of an app so that it receives a smaller share of available processor time.

In the unlikely event that an app needs to be manually set to not enter App Nap, you can bring up the Info panel for the app in the Finder, and select the Prevent App Nap checkbox. App Nap is never triggered for the foreground application.

Centralized Task Scheduling

Centralized Task Scheduling minimizes the amount of system maintenance and background work that is performed while your Mac is running on battery power. Some tasks are set to run on battery power only after a specified amount of time has passed (for example, Software Update checks every seven days, and can defer checking by up to one day if the user is on battery power), while other tasks may be configured to never run on battery power (such as background downloads of software updates).

Timer Coalescing

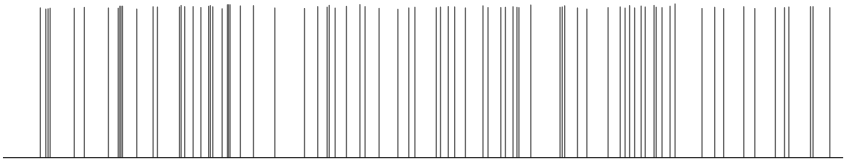
Because idling a processor is such an important power-saving strategy, frequently waking up the processor can hurt battery life. OS X Mavericks introduces a new technology—Timer Coalescing—that helps dramatically increase the amount of time that the CPU spends idling.

On a typical OS X Mountain Lion system, there are numerous applications and background processes that schedule routine work at some interval. For example, a program displaying an animation may need to draw a new frame 30 times each second, or a progress indicator in an application may check several times a second to get updates on the status of a job. In most cases, these tasks are initiated at a low level by something called a timer.

A timer is a request made by an app or background process to the operating system to perform an action after a specified amount of time. While timers are incredibly useful, they can also lead to an unintended expense: As the number of timers set by all of the apps on your machine increases, so does the amount of time the CPU potentially needs to spend working in a higher-power state.

The transition from a CPU doing work to being idle isn't instant, and the lower the power state the system is entering or leaving, the longer the transition. When trying to maximize battery life, those long transitions between CPU idle states are something that are best done infrequently.

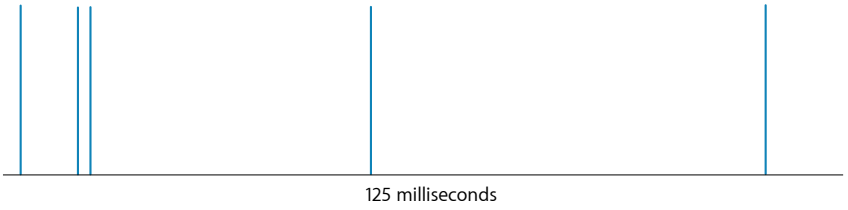
OS X Mountain Lion timers



Typical timer execution sample showing 125 milliseconds on an OS X Mountain Lion system with no user interaction. Frequent execution of timers from various processes can prevent the processor from idling.

The challenge becomes: How can the system do all of the required work while maximizing the amount of time the processor spends at idle? The answer is Timer Coalescing, which shifts the execution of timers by a small amount so that timers of multiple applications are executed at the same time. While the changes to when timers fire are quite small—anywhere from a few milliseconds to a few tens of milliseconds—Timer Coalescing can dramatically increase the amount of time that the processor spends idling.

OS X Mavericks timers



A sample of the same scenario on OS X Mavericks: 125 milliseconds of timer execution with no user activity. Executing timers at the same time reduces power use by maximizing processor idle time.

Conceptually this is done by applying a time window—based on the importance of each process—to every timer (see table below). A timer can be executed at any time during this window, meaning that it can be shifted forward or back a bit to line up with other timers that need to be executed at similar times. Timers associated with real-time or critical processes have a window of zero, meaning that they will always be executed at the specified time.

Process types and timer windows

Application (default)	1 ms
System daemon	70–90 ms
Background process	80–120 ms
Critical/real-time process	0 ms

Importance donation

Importance donation is a new mechanism in OS X Mavericks that enables a higher-priority process to "donate" its importance to a background helper process while the background process is performing an important task. This allows the background process to use less power, yet makes sure that it gets all the resources it needs for important tasks.

Power management firmware (XCPM)

The power management firmware on your Mac controls the power characteristics of the hardware. With its 2013 MacBook Air systems, Apple introduced a new generation of power management firmware called XCPM. It improves power efficiency by performing power management functions faster and providing more intelligent management of multiprocessor workloads.

The greater efficiency of the firmware means that the system can often enter a low-power state more quickly, leading to increased amounts of idle time and energy savings.

Efficient HD Video Playback

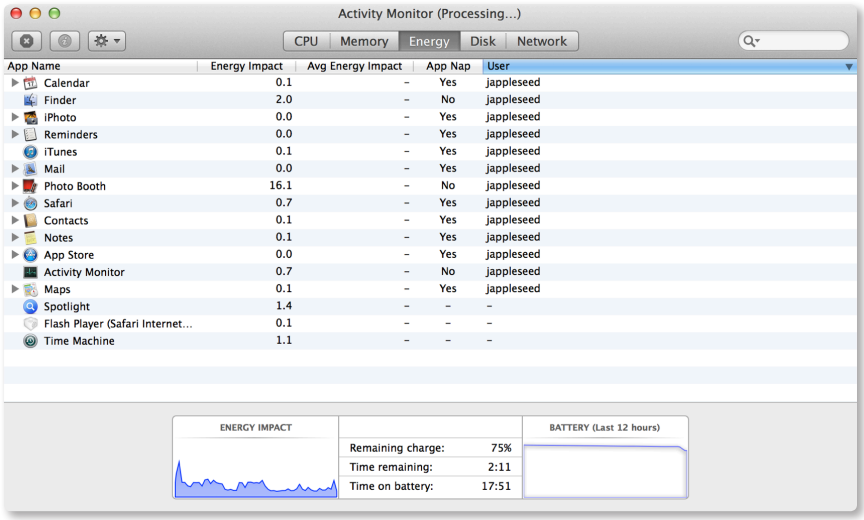
Playback of full-screen HD TV shows and movies from iTunes has been optimized, with CPU energy consumption reduced by up to 35 percent on the mid-2012 13-inch MacBook Air. These savings are achieved by a number of changes including:

- **GPU acceleration**—The video playback engine now leverages the GPU for scaling and color space conversion.
- **Larger I/O buffers**—Audio and video data are read in larger chunks, reducing the amount of disk access required.
- **Energy-efficient audio**—A new video playback mode uses larger buffers, and feeds the system audio driver data at its native sampling rate and bit depth.

User Tools

Activity Monitor

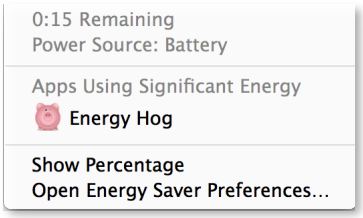
Activity Monitor now displays energy usage associated with each app on your system, so you can see exactly which apps are most efficient and where your battery power is being used. It also now reports battery charge state over time, making it easier to identify specific activities that use significant amounts of energy.



Activity Monitor in OS X Mavericks includes an app-by-app view of power usage.

Battery status menu

In addition to showing remaining battery percentage and estimated remaining battery life, the Battery status menu in OS X Mavericks shows any applications that are using a significant amount of energy. This gives you the option to quit an app that you are not using that may be contributing to shorter battery life.



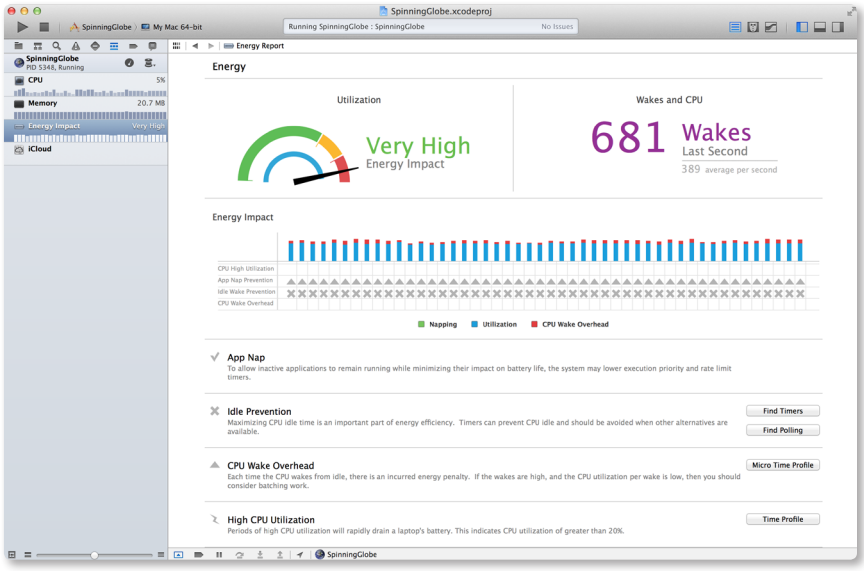
The Battery status menu in OS X Mavericks can identify apps that are using significant amounts of power.

Developer Tools

In addition to system technologies and user tools, OS X Mavericks introduces a number of tools that make it easy for developers to analyze energy usage in their apps and target changes that help their users see improved battery life.

Xcode Debug Gauges

As you begin running an app under the debugger in Xcode, the interface presents three Debug Gauges that instantly read the performance of your app to help you see how it's using system resources. The gauges—CPU, I/O, and Energy—summarize your app's key performance statistics in a single view, and can even be run side by side to present everything you need to know about how your app is using system resources, in a single window.



The Xcode Energy Debug Gauge gives you key energy statistics for your app at a glance.

Powermetrics
OS X Mavericks also introduces `powermetrics`, a new command-line tool for gathering information on CPU usage, including timer and interrupt wakeup frequency, CPU package C-state statistics, and average execution frequency for each CPU core.

APIs
Most of the energy-saving behavior of OS X Mavericks is automatic. For example, App Nap automatically throttles timers regardless of which API was used to create the timers. The operating system automatically fires any expired timers the moment a user begins interacting with an app, helping to ensure that any required work is done as the user sees the app.

The result of this automatic approach to power management is that developers don't typically need to do any work to take advantage of the energy-efficiency technologies in OS X Mavericks. To maximize power savings, developers can adopt new APIs in OS X Mavericks that help to provide better information to the operating system for the purpose of making energy management decisions.

- **NSProcessInfo**—New API in `NSProcessInfo` gives developers the ability to tell the operating system when they are performing long-running operations that may need to prevent App Nap or system sleep.
- **Grand Central Dispatch (GCD)**—In the GCD C API, the `dispatch_source_set_timer()` function's `leeway` parameter is now used when determining the window for Timer Coalescing. It allows developers to give the operating system more information about how timely their operations need to be, so the system can make better timer scheduling decisions. The new `dispatch_timer_strict()` function exempts a specific timer from App Nap and Timer Coalescing, so that developers can exempt a specific long-running operation of an app from App Nap and Timer Coalescing without requiring the entire app to be exempt.
- **NSTimer**—The `NSTimer` class introduces a new `tolerance` parameter that enables developers to specify how timely timer-driven events need to be, similar to the `leeway` parameter in GCD.

Conclusion

The Mac has long led the way in delivering power efficiency and performance. Fine-tuned at every level to help extend battery life, OS X Mavericks introduces powerful new technologies for energy efficiency as well as tools that help app developers tune their own apps for power efficiency. App Nap, Timer Coalescing, and Centralized Task Scheduling work in concert with the system apps and frameworks in OS X Mavericks to help you get more battery life out of your Mac, while delivering all of the performance you expect out of the apps you're using. All of this is only possible with the deep integration of hardware and software that Apple delivers.

For More Information

For more information about OS X Mavericks, visit www.apple.com/osx.

 October 2013 L519830A