Battery Powered Linux Mini−HOWTO

David Lechnyr

<david@lechnyr.com>

Revision History

Revision 2.31	2003−07−30	Revised by: drl
Minor updates		
Revision 2.0	2002−10−08	Revised by: drl
Major updates and conversion to SGML		
Revision 1.0	1997−12−21	Revised by: hm
Initial draft by Hanno Muller		

This document describes how to optimize and configure power management on a ready−configured Linux system for use on battery−powered laptops. This will be helpful for everyone who runs Linux on a portable computer system. APM/ACPI methods of power management are discussed along with power saving tips and techniques. There is also some discussion about the different types of batteries available.

Table of Contents

1. Power Management

If you have a relatively recent x86 laptop, odds are it supports either Advanced Power Management (APM) or Advanced Configuration and Power Interface (ACPI). ACPI is the newer of the two technologies and puts power management in the hands of the operating system, allowing for more intelligent power management than is possible with BIOS controlled APM. This is most useful for battery−powered laptops. You can only have one power management interface in control of your machine at a time, so it's important you decide which method best suits your situation.

1.1. Advanced Power Management (APM)

Advanced Power Management (APM) allows your computer's BIOS to control your system's power management without the knowledge of the operating system. The advantages to APM under Linux are that it's stable, well supported by Linux vendors and has a solid history behind it. However, not much development has been done with it over the past few years.

To use it, you'll need to enable APM in the kernel:

```
[*] Power Management support
<*>   Advanced Power Management BIOS support
[ ]     Ignore USER SUSPEND (NEW)
[ ]     Enable PM at boot time (NEW)
[ ]     Make CPU Idle calls when idle (NEW)
[ ]     Enable console blanking using APM (NEW)
[ ]     RTC stores time in GMT (NEW)
[ ]     Allow interrupts during APM BIOS calls (NEW)
[ ]     Use real mode APM BIOS call to power off (NEW)
```

Most of the other APM options exist as work−arounds for known problems with specific hardware devices, so you'll probably only want to enable the first one (CONFIG_APM).

- Advanced Power Management BIOS support (CONFIG_APM): You'll need to enable this in order to do anything useful with APM. User−space programs will receive notification of APM events (e.g., battery status change) and a /proc/apm device will provide you with battery status information.
- Ignore USER SUSPEND (CONFIG_APM_IGNORE_USER_SUSPEND): This is a workaround for NEC Versa M notebooks.
- Enable PM at boot time (CONFIG_APM_DO_ENABLE): Although it sounds nifty, most machines do not require this feature to be enabled and in fact can hang some systems at boot time.
- Make CPU Idle calls when idle (CONFIG_APM_CPU_IDLE): On some machines, this option provides increased power savings. On others, it will hang the system at boot time. Use with caution.
- Enable console blanking using APM (CONFIG_APM_DISPLAY_BLANK): Instead of blanking the virtual console actually turn off the screen. This won't work with X−Windows and actually can cause more problems that it solves.
- RTC stores time in GMT (CONFIG_APM_RTC_IS_GMT): If you want to store GMT (Greenwich Mean Time) in your RTC (Real Time Clock), say yes here.
- Allow interrupts during APM BIOS calls (CONFIG_APM_ALLOW_INTS): This is a workaround for some IBM Thinkpads that hang while suspending.
- Use real mode APM BIOS call to power off (CONFIG_APM_REAL_MODE_POWER_OFF): This is a workaround for a number of broken BIOSes. If your computer crashes instead of powering off properly, turn this on.

You'll want to install the APM daemon from http://www.worldvisions.ca/~apenwarr/apmd/ and configure your system startup scripts to activate it on boot:

```
# Start the APM daemon if it exists and if APM is enabled in the kernel
if [ -x /usr/sbin/apmd -a -d /proc/apm ]; then
        if cat /proc/apm 1> /dev/null 2> /dev/null ; then
                echo "Starting APM daemon:  /usr/sbin/apmd"
                /usr/sbin/apmd
        fi
fi
```

The APM daemon is actually made up of three primary programs:

- **apmd** − handles power management tasks
- **apm** − a command−line tool to print the current battery status or suspend the computer
- **xapm** − a simple battery meter for X

If you're looking for a simple, "works out of the box" approach to power management for your Laptop, APM is definitely the way to go.

A simple script to notify you how much battery time is remaining can be added to your `~/.profile` file:

```
if [ -f /proc/apm ]; then
        DUMMY=`cat /proc/apm | cut -d" " -f 7`
        # Don't display when fully charged
        if [ "$DUMMY" != "99%" ]; then
                LEVEL=`cat /proc/apm | sed -e "s/^.*% //"`
                echo "Battery at $DUMMY ($LEVEL)"
        fi
fi
```

1.2. Advanced Configuration and Power Interface (ACPI)

Advanced Configuration and Power Interface (ACPI) is the successor to APM, which places the responsibility of power management away from the BIOS and into the hands of the operating system. ACPI Linux is newer than APM Linux, more flexible in responding to power management events, has seen much more development as of late, and as a result of all this is prone to its own share of bugs from time to time.

If you're into cutting−edge development and are not intimidated with kernel builds and applying patches against source code, ACPI is worth consideration.

There are two parts to ACPI under Linux: The ACPI driver built into the kernel itself, and the ACPI daemon (ACPID). ACPID in its current incarnation is pretty simple: monitor /proc/acpi/event and do things in response. Even if you don't load the daemon, you'll still get the benefit of ACPI features built into the kernel such as processor thermal support.

You can determine which version of the ACPI driver you are using, along with supported suspend states, by using:

```
bash $ cat /proc/acpi/info
version: 20030619
states:  S0 S1 S3 S4 S4 S5
```

ACPI development is progressing at a steady rate, so you might want to consider patching your kernel against any recent updates to the kernel−level ACPI code. Once you have downloaded the patch for your specific kernel, you can patch it with something like:

```
bash$ gunzip acpi-[version-kernel].diff.gz
bash# cd /usr/src/linux-[version]
bash# patch -Np1 -i ../acpi-[version-kernel].diff
```

You'll want to recompile your kernel after this, of course:

```
      [*] ACPI Support
      [ ] CPU Enumeration Only
      <*>   AC Adapter
      <*>   Battery
      <*>   Button
      <*>   Fan
      <*>   Processor
      <*>   Thermal Zone
      < >   ASUS Laptop Extras
      < >   Toshiba Laptop Extras
      [*]   Debug Statements
```

You'll also want to install the ACPID daemon from http://sourceforge.net/project/showfiles.php?group_id=33140 and configure your system startup scripts to activate it on boot:

```
if [ -x /usr/sbin/acpid -a -d /proc/acpi ]; then
        echo "Starting ACPID Daemon:  /usr/sbin/acpid"
        /usr/sbin/acpid
fi
```

A bit of history... Microsoft was the first vendor to implement ACPI. This is both good and bad. It is good because when you buy a system, you can pretty much guarantee that it has passed Microsoft's hardware compliance tests, including the test of its ACPI implementation. However, these tests come up short in that they do not indicate compliance with the ACPI specification, but rather with Microsoft's implementation of ACPI. When that same machine is used with Linux, some classes of errors that did not manifest themselves under Windows may become apparent. To protect against this problem, the Linux ACPI driver maintains a "bad BIOS" blacklist of known BIOS's that are known to not be ACPI compliant, and as a result will refuse to enable ACPI if your system is listed.

Many manufacturers are now validating that their systems run on Linux. However, they use major Linux distributions with the default kernel. This means that it is somewhat difficult to get OEMs to ensure that their systems work with ACPI−enabled Linux until a major Linux distribution ships an ACPI kernel. This presents a slight dilemma in that Linux distributions want to ship kernels that run on as many systems as possible, but there have been some positive moves in this area lately.

To conserve energy while remaining quickly available, ACPI−compatible PCs may enter system sleep states. The ACPI specification defines five of these states, known as S−states. Unlike processor sleep states, no work is done by the system under S−states. Each state introduces greater power savings but requires commensurately more time to awaken and begin performing work. These are patterned on system states from the APM standard, a predecessor of ACPI.

Full details on ACPI sleep states are available at http://acpi.sourceforge.net/documentation/sleep.html. Processor states are described at http://acpi.sourceforge.net/documentation/processor.html.

For more specific background information on ACPI itself, you can visit the ACPI website at http://www.acpi.info

1.3. APM vs. ACPI: Which one?

There are currently two competing standards for providing power management: APM and ACPI. Both cannot be used at the same time, so which one is best for your situation? If you have a relatively recent (>2.4.20) kernel and are not intimidated by kernel builds and patching source code, you'll find many benefits with the flexibility of ACPI. If you just want to enable generic power management, or are using an older machine, choose APM. Neither method spins down idle hard drives; use hdparm for that instead. Either way, your system's BIOS must correctly support the power management scheme you'd like to use as well; if your system does not fully support either standard, some of the power management options might crash your system and/or cause data loss. You have been warned!

Even if you don't enable power management on your x86−laptop, Linux will always issue the HLT instruction to your processor whenever nothing needs to be done [1]. Many Microsoft Windows CPU cooling program use this technique. This results in lowering the power consumption of your CPU. Note that the system doesn't power down when it receives the HLT instruction; it just stops executing instructions until there is an interrupt.

There is generally no advantage to enabling either type of power management on servers or workstations that do not fall into these categories.

1.4. SMP, Hyper−Threading, IA64 & NUMA

Some SMP system manufacturers may have omitted the pre−ACPI tables used for SMP configurations. In this case, ACPI is required.

If you have a newer system that supports Hyper−Threading, you will need to enable ACPI (and, of course, SMP). Without it, your Linux system may be unable to discover and initialize all of the virtual processors.

IA64 machines require ACPI as well. Additionally, NUMA servers are starting to require it for proper initialization.

2. DPMS

DPMS (Display Power Management Signaling) is a standard to reduce power consumption in monitors. Typically, both the monitor and the video card must support the DPMS standard in order to receive any benefit from it. DPMS specifies four modes of operation (in order of increasing power savings): "Normal", "Standby", "Suspend" and "Off". Two signal lines, "Horizontal Sync" and "Vertical Sync" provide a method for signaling these four different states to a DPMS monitor.

A good technical resource on DPMS is available at http://webpages.charter.net/dperr/dpms.htm.

2.1. Normal

Normal means just that −− the monitor is fully powered and on. LCD laptop panels and LCD flat screens use considerably less power than traditional CRT monitors.

2.2. Standby

Standby is used to describe a very minor power savings level. This setting usually involves blanking the screen by turning off the electron (RGB) gun. However, the power supply is left on and the tube filaments energized. When you need to use the monitor again, the monitor will come back on very quickly. This option requires DPMS monitor and video card support along with enabling DPMS in XFree86. Standby is sometimes referred to as *hsync suspend mode* since the horizontal sync signal is turned off to signal this power management state to a DPMS monitor.

2.3. Suspend

Suspend is used to describe a very strong low power state. This setting usually involves the same power conservation as **Standby** however in addition the power supply to the monitor is turned off. This option requires DPMS monitor and video card support along with enabling DPMS in XFree86. Suspend is sometimes referred to as *vsync suspend mode* since the vertical sync signal is turned off to signal this power management state to a DPMS monitor.

2.4. Off

Off usually means just that −− the computer monitor is turned off. Usually, a small auxiliary circuit stays on to monitor the signals from the computer to turn the monitor back on when data needs to be displayed to the screen. Obviously, this keeps power consumption to a bare minimum (if not zero). While the power saving is substantial, to reactivate the monitor may take several seconds. This option requires DPMS monitor and video card support along with enabling DPMS in XFree86. Both the horizontal and vertical sync signals are turned off to signal this power management state to a DPMS monitor.

3. Power Management Methods

The basic goal of any power management technique is to reduce an entity's consumption. In the case of laptop power management, our focus is on decreasing CPU and hard drive usage. To make things a bit simpler, this is broken down into *obvious, semi−obvious,* and *non−obvious* techniques. Granted, your mileage may vary.

3.1. swsusp(8)

Suspend to Disk (S2D) is still an elusive task under Linux. The main project at the moment is swsusp, available at http://sourceforge.net/projects/swsusp. It's still fairly new and requires a bit of configuration to enable it.

3.2. hdparm(8)

hdparm is a Linux shell utility that can be used to spin down and improve the performance of various ATA/IDE drives. If it's not included with your system, you can fetch the source from http://freshmeat.net/redir/hdparm/4062/url_homepage/hardware. For example, the following provides 32−bit IO support with sync (−c3), DMA support (−d1), Advanced Power Management (−B128), write−caching (−W1), disk spin down after five minutes (−S60). gains me tremendous performance with added power savings. Note that your mileage may vary, and you'll want to adjust this for your specific system to prevent data loss (especially the −B and −m flags!).

In the following example, we run some read/write benchmarks of our hard drive before and after using **hdparm**. Note that while our cache reads remain about the same, our actual physical reads from the drive increase tremendously! If you like living on the edge, you can play with the −m, −c, −B, and −u switches with caution (see the man page).

```
bash# hdparm -tT /dev/hda
Timing buffer-cache reads:   588 MB in  2.01 seconds = 292.15 MB/sec
Timing buffered disk reads:   14 MB in  3.46 seconds =   4.05 MB/sec

bash# hdparm -k1 -K1 -c3 -d1 -W1 /dev/hda
bash# hdparm -tT /dev/hda
Timing buffer-cache reads:   596 MB in  2.01 seconds = 297.01 MB/sec
Timing buffered disk reads:   72 MB in  3.05 seconds =  23.58 MB/sec
```

3.3. sysklogd(8)

Examine your **/etc/syslog.conf** file for unnecessary logging activity and to optimize its performance. If you don't want to log any system activity, consider disabling syslogd and klogd entirely or, at the very least, minimize the amount of logging your system performs. You can also prefix each entry with the minus sign (−) to omit syncing the file after each log entry [2]. For example, this will log anything with a priority of *info* or higher, but lower than *warning*, to **/var/log/messages** or **/var/log/mail** without needing to sync to disk after each write. Since we want to keep all messages with a priority of *warning*, this will be logged to a different file without disabling disk syncing (to prevent data loss in the event of a system crash).

```
*.warning                       /var/log/syslog
*.info;*.!warning;mail.none     -/var/log/messages
mail.info;mail.!warning         -/var/log/mail
```

Another item to be aware of is the **−− MARK −−** messages that **syslogd(8)** writes. This will affect your hard drive inactivity settings. You can simply disable this by running **syslogd(8)** with:

```
if [ -x /usr/sbin/syslogd -a -x /usr/sbin/klogd ]; then
        # '-m 0' disabled 'MARK' messages
        /usr/sbin/syslogd -m 0
        sleep 1
        # '-c 3' displays errors on console
        # '-x' turns off broken EIP translation
        /usr/sbin/klogd -c 3 -x
fi
```

3.4. XFree86

There are essentially two different types of screen blanking that can be performed under XFree86: BlankTime and DPMS. The first is simply a fake "blanking" effect that doesn't actually save any power. The others are specific only to DPMS−compliant monitors, and must be specifically enabled to take effect. They are located in your **XF86Config** file, which normally resides in **/etc/X11/XF86Config**.

If you have a DPMS−compliant monitor, you might want to try enabling support for it under the *Monitor* section of your **XF86Config** file:

```
Section "Monitor"
        Option  "DPMS"
EndSection
```

To manipulate the DPMS functions, you can create/modify the following items in the *ServerLayout* section.

```
Section "ServerLayout"
        Option "BlankTime"      "10"    # Blank the screen in 10 minutes
        Option "StandbyTime"    "20"    # Turn off screen in 20 minutes
        Option "SuspendTime"    "30"    # Full hibernation in 30 minutes
        Option "OffTime"        "40"    # Turn off DPMS monitor
EndSection
```

It's worth noting that **BlankTime** is not actually a power saving level at all. The screen is sent a "fake" blanking effect and defaults to activate after 10 minutes. Alternately, it can indicate the number of minutes until the screensaver should activate. It has nothing to do with DPMS.

After activating your changes and restarting X−Windows, you might want to examine your logfile to see if your video card has any problems with your changes:

```
bash$ egrep "^\(WW|EE\)" /var/log/XFree86.0.log
```

There may be additional options that you can enable for your specific video card/chip driver; see the XFree86 Documentation website for specifics.

Of course, all of this can also be activated "on−the−fly" by using **xset(1)**. If you don't have access to your system's **XF86Config** file, a good place to put these commands would be in your **~/.Xsession** or **~/.xinitrc** file.

```
bash$ xset -dpms                        # Disable DPMS
bash$ xset +dpms                        # Enable DPMS
bash$ xset s off                        # Disable screen blanking
bash$ xset s 150                        # Blank the screen after 150 seconds
bash$ xset dpms 300 600 900             # Set standby, suspend, & off times (in seconds)
```

```
bash$ xset dpms force standby          # Immediately go into standby mode
bash$ xset dpms force suspend          # Immediately go into suspend mode
bash$ xset dpms force off              # Immediately turn off the monitor
bash$ xset -q                          # Query current settings
```

If instead you're using the Linux console (not X−Windows), you'll want to use **setterm(1)**:

```
bash$ setterm -blank 10                # Blank the screen in 10 minutes
bash$ setterm -powersave on            # Put the monitor into VESA power saving mode
bash$ setterm -powerdown 20            # Set the VESA powerdown to 20 minutes
```

3.5. KDE 3.1

3.5.1. Display Power Control

Assuming you've configured XFree86 to support DPMS, simply run **kcontrol** and choose **Power Control/Display Power Control**. From here, you can configure Standby, Suspend, and Power off settings for your monitor.

3.5.2. Laptop Battery

Assuming you've configured your kernel to support either APM or ACPI, simply run **kcontrol** and choose **Power Control/Laptop Battery**. From here, you can configure the various settings for your system based on the level of battery power remaining.

It's worth noting that some people running ACPI tend to see the following message:

```
Your computer seems to have a partial ACPI installation. ACPI was probably
enabled, but some of the sub-options were not - you need to enable at least
'AC Adaptor' and 'Control Method Battery' and then rebuild your kernel.
```

If you see this, either ACPI is not installed or, more likely, KDE does not recognize your particular Linux ACPI Subsystem. If patching the kernel with any ACPI updates does not resolve this, you must either not use this KDE function or, alternately, revert back to using APM.

3.6. Energy Star

Energy Star is a United States government−backed program to promote energy efficiency standards. Of interest:

- An ENERGY STAR qualified computer, in sleep mode, uses 70% less electricity than computers without power management features.
- An ENERGY STAR qualified monitor, in sleep mode, uses 90% less electricity than monitors without power management features.

Typically, Energy Star savings is accomplished by other power management settings and is not, in and of itself, a power management technique.

3.7. Swap File

Consider disabling your swap file in **/etc/fstab** to reduce hard drive access. If you've got lots of memory, this is definitely the way to go. One way to tell if you need your swap file is to enable it, use your system for a period of time, and examine **/proc/meminfo** and **/proc/swaps** to determine how much free memory you've got on average, and whether or not your swap file is even being utilized.

For example, today I've compiled several intensive programs and have been running my laptop for about eight hours straight. A simple examination of my system reveals:

```
bash$ cat /proc/swaps
Filename                                Type            Size    Used
Priority
/dev/hda3                                partition      136544  0       -1

bash$ cat /proc/meminfo
MemTotal:       513880 kB
MemFree:        254820 kB
Buffers:         42812 kB
Cached:         142880 kB
SwapCached:          0 kB
Active:         159644 kB
Inactive:        76888 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:       513880 kB
LowFree:        254820 kB
SwapTotal:      136544 kB
SwapFree:       136544 kB
Dirty:               0 kB
Writeback:           0 kB
Mapped:          86148 kB
Slab:            10748 kB
Committed_AS:   203944 kB
PageTables:       1140 kB
VmallocTotal:   516076 kB
VmallocUsed:      1468 kB
VmallocChunk:   514604 kB
HugePages_Total:     0
HugePages_Free:      0
Hugepagesize:     4096 kB
```

Given this, I'd opt to disable my swapfile if this is any indicator of my future usage.

3.8. tmpfs

Compile your kernel with tmpfs (temporary file system) enabled and mount your /tmp directory using it. The useful bit here is that nothing will be written to your hard drive on this mount point as it will act like a RAM disk (however nothing will be saved either). The advantage of tmpfs over the more traditional ramfs is that it lives in the kernel internal cache and grows and shrinks to accommodate the files placed there. See your kernel's Documentation/filesystems/tmpfs.txt for full information. If you don't specify a maximum size, it will default to a ceiling limit of half your available memory. An example /etc/fstab with 100MB temporary ram file mounted on /tmp would look like:

```
tmpfs   /tmp    tmpfs   size=100m,mode=1777     0 0
```

3.9. Miscellaneous Tuning

Modifying /proc/sys/vm/bdflush allows a user to specify under what circumstances dirty buffers are flushed to disk, how many such buffers exist, etc. Details are in linux_src_tree/Documentation/sysctl/vm.txt (thanks to Marc Liberatore for pointing this out).

Boot your system and list the currently loaded modules with lsmod. Anything listed here most likely needs to be loaded on a regular basis; compiling these in as part of your kernel rather than as loadable modules may help to decrease the amount of time they must be loaded from disk, and to a very minor degree, decrease the amount of disk access required to start your system.

Examine your crontab settings to see if anything is being run on a regular basis. Comment out any unnecessary items. Don't forget to examine every user's crontab, including the user 'nobody'. If you don't need to schedule any background activity, consider disabling crond alltogether. The same advice goes for atd.

If you run httpd to test and/or develop web pages, try altering the values of MinSpareServers and StartServers to 1. Don't define any CustomLogging or at least increase the value of LogLevel to warn. If you're really sure of yourself, you can change the ErrorLog directive to point to /dev/null.

Consider creating a power−saving script that will immediately take your laptop into low−power mode:

```
#!/bin/sh
if [ -x /usr/sbin/hdparm ]; then
        hdparm -y /dev/hda
fi

if [ -x /usr/X11R6/bin/xset ]; then
        xset dpms force off
fi
```

Additionally, it's worth considering anything in the following areas:

- Adjust your system's BIOS settings to decrease or turn off your display's backlight.
- Adjust your system's BIOS settings to reduce the CPU clock speed while on battery.
- Avoid using PCMCIA devices while on battery. Better yet, eject your PCMCIA cards when not in use.
- Avoid using external devices with your computer while on battery. This includes printers, external monitors, zip drives, and portable cameras.
- Avoid using built−in devices while on battery. This includes cdroms and floppy drives.
- Use simple software. A full blown multimedia application will create a lot more system load and disk activity than a small simple word processor
- Use a simple window manager. While Gnome and KDE are nice, the extra time it takes to load and run is not worth it while on battery power. One nifty idea is to use a different **xinitrc** script to launch a different, more simple window manager based on whether or not your system is on battery power.

3.10. Power Saving Myths

It used to be beneficial to recompile the Linux PCMCIA drivers to allow the slots to have APM power support. However, most of the functionality of these drivers are now built into the kernel itself. If you're interested in specifics, the PCMCIA project page is available at http://sourceforge.net/projects/pcmcia−cs/.

Some people believe that APM offers better power savings over ACPI, and vice−versa. While their power management techniques differ, in actual battery−usage tests, both reportedly perform about the same.

Contrary to popular belief, Lithium Ion (see below) batteries *do* suffer from a memory effect. Luckily, the effect is not large over the lifespan of a typical battery (3−4 years). Anyone who tells you different is selling something.

---

4. Types of Batteries

There are currently three types of batteries commonly used for laptops: Nickel Cadmium, Nickel Metal Hydride, and Lithium Ion.

4.1. Nickel Cadmium (Ni−Cd)

Nickel Cadmium (Ni−Cd) batteries were the standard technology for years, but today they are out of date and new laptops don't use them anymore. They are heavy and very prone to the "memory effect". When recharging a NiCd battery that has not been fully discharged, it "remembers" the old charge and continues there the next time you use it. The memory effect is caused by crystallization of the battery's substances and can permanently reduce your battery's lifetime, even make it useless. To avoid it, you should completely discharge the battery and then fully recharge it again at least once every few weeks. As this battery contains cadmium, a toxic material, it should always be recycled or disposed of properly.

NiCad batteries, and to a some degree NiMH batteries, suffer from what's called the *memory effect*. Memory Effect means that if a battery is repeatedly only partially discharged before recharging, the battery will forget that it can further discharge. The best way to prevent this situation is to fully charge and discharge your battery on a regular basis.

4.2. Nickel Metal Hydride (Ni−MH)

Nickel Metal Hydride (Ni−MH) batteries are the cadmium−free replacement for NiCad. They are less affected by the memory effect than NiCd and thus require less maintenance and conditioning. However, they have problems at very high or low room temperatures. And even though they use less hazardous materials (i.e., they do not contain heavy metals), they cannot be fully recycled yet. Another main difference between NiCad and NiMH is that NiMH battery offers higher energy density than NiCads. In other words, the capacity of a NiMH is approximately twice the capacity of its NiCad counterpart. What this means for you is increased run−time from the battery with no additional bulk or weight.

4.3. Lithium Ion (Li−ion)

Lithium Ion (Li−ion) are the new standard for portable power. Li−ion batteries produce the same energy as NiMH but weighs approximately 20%−35% less. They do not suffer significantly from the memory effect unlike their NiMH and Ni−Cd counterparts. Their substances are non−hazardous to the 0. Because lithium ignites very easily, they require special handling. Unfortunately, few consumer recycling programs have been established for Li−ion batteries at this point in time.

4.4. Smart Batteries

Smart batteries are not really a different type of battery, but they do deserve special mention. Smart batteries have internal circuit boards with chips which allow them to communicate with the laptop and monitor battery performance, output voltage and temperature. Smart batteries will generally run 15% longer due to their increased efficiency and also give the computer much more accurate "fuel gauge" capabilities to determine how much battery run time is left before the next recharge is required.

4.5. General Battery Care

Even if the battery case looks the same, you cannot just upgrade to another battery technology unless your laptop has been pre−configured from the manufacturer to accept more than one type of battery type, since the recharging process is different for each of the three types of batteries.

A battery that is not used for a long time will slowly discharge itself. Even with the best of care, a battery needs to be replaced after 500 to 1000 recharges. But still it is not recommended to run a laptop without the battery while on ac power −− the battery often serves as a big capacitor to protect against voltage peaks from your ac outlet.

As the manufacturers change the shapes of their batteries every few months, you might have problems to find a new battery for your laptop in a few years from now. This is somewhat of a concern only if you anticipate using the same laptop several years from now. If in doubt, buy a spare battery now − before it's out of stock.

New batteries come in a discharged condition and must be fully charged before use. It is recommended that you fully charge and discharge the new battery two to four times to allow it to reach its maximum rated capacity. It is generally recommend that you perform an overnight charge (approximately twelve hours) for this. Note: It is normal for a battery to become warm to the touch during charging and discharging. When charging the battery for the first time, the device may indicate that charging is complete after just 10 or 15 minutes. This is a normal with rechargeable batteries. New batteries are hard for the device to charge; they have never been fully charged and are not broken in. Sometimes the device's charger will stop charging a new battery before it is fully charged. If this happens, remove the battery from the device and then reinsert it. The charge cycle should begin again. This may happen several times during the first battery charge. Don't worry; it's perfectly normal. Keep the battery healthy by fully charging and then fully discharging it at least once every two to three weeks. Exceptions to the rule are Li−Ion batteries which do not suffer from the memory effect.

Batteries should be stored in a discharged state since they can self−discharge and may become inactive after a long storage period. They should not be stored for any length of time while connected to the laptop. High humidity and temperatures can cause the battery to deteriorate, so these should be avoided during storage.

Do not remove and carry a battery pack in your pocket, purse, or other container where metal objects (such as car keys or paper clips) could short−circuit the battery terminals. The resulting excessive current flow can cause extremely high temperatures and may result in damage to the battery pack or cause fire or burns.

5. Appendix

This document was lovingly handcrafted on a Dell Latitude C400 laptop running Slackware Linux 9.0, in case anyone asks.

This document would not have been possible without the excellent material initially developed by Hanno Muller `<kontakt@hanno.de>`.

Notes

[1] source/arch/i386/kernel/process.c

[2] syslogd.c