**概述**  
     为了能让OS X的特性能够在普通PC上正常工作，就需要修改（可以利用补丁）DSDT和SSDT。本教程的目的，就是介绍如何利用现有补丁，修改你电脑的原始OEM DSDT/SSDT。  
     虽然你可以用其它型号的电脑的DSDT，但这样的后果是会造成更多问题。你不能保证用别的型号电脑的DSDT是没问题的。如果你用了别的电脑的ACPI文件，即便是硬件配置非常相似，也可能会造成 系统不稳定 和 一些奇葩的问题。  
  
使用补丁修改DSDT/SSDT包含了以下**步骤**：  
－提取原始ACPI文件  
－反编译原始文件  
－分析并筛选这些原始文件  
－使用补丁修改  
－编译并保存，再放到引导特定的目录

**提取原始ACPI文件**  
      所有BIOS都实现了把ACPI文件提交给操作系统的功能。所以，基于任何操作系统，你都可以通过操作系统提取出原始ACPI文件，之后再进行修改。Linux、OS X、Windows、Clover都可以用来提取原始ACPI文件。这些方式提取的文件是完全相同的，只是由于使用的提取工具不同，文件名会不同。  
      本教程将主要介绍3种提取方式：在OS X使用patchmatic工具（Rehabman编写的终端程序），在Clover启动界面按F4，还有使用Linux。

**使用patchmatic工具提取**  
      如果你已经有了OS X环境，并且还没有使用修改过的DSDT/SSDT，就可以使用patchmatic工具提取原始DSDT/SSDT。到这里下载编译好的patchmatic工具：<https://github.com/RehabMan/OS-X-MaciASL-patchmatic> （仔细阅读patchmatic工具的README）。为了方便使用，你可以把程序拷贝到 **/usr/bin** 文件夹。（放到这里，在任何地方都可以调用）

放好patchmatic之后，就可以在终端调用它了：  
cd ~/Desktop  
mkdir extract  
cd extract  
patchmatic -extract  
      patchmatic工具将会提取这次启动被加载的ACPI文件 到当前目录。如果你用了Clover的Fix功能（Clover会向DSDT/SSDT注入代码），那么得到的就不是原始文件了，所以，要确认你没有使用Clover的Fix功能。另外的情况是，如果你设置了(变色龙) DropSSDT=Yes,，或者 (Clover) DropOem=true，那么，引导程序就会自动屏蔽原始的SSDT，这时用patchmatic提取，就不会得到SSDT文件。还有就是，变色龙的GeneratePStates/GenerateCStates=Yes,参数，或Clover的 /ACPI/SSDT/Generate/CStates、/ACPI/SSDT/Generate/PStates参数，会生成并注入新的SSDT（非原始SSDT），这时，提取的SSDT在反编译的时候会出现问题。  
  
      以上就是提取原始文件的注意事项。一般，在Linux提取或在Clover启动界面按F4提取更容易一些。  
  
      注意：使用 'patchmatic -extract’ 命令可以作为 确认你修改的DSDT/SSDT是否被成功加载了的非常有用的诊断工具。

**在Clover启动界面按F4提取**  
     在Clover启动界面，你可以按 “F4”键，Clover会把原始ACPI文件提取到 EFI/Clover/ACPI/origin 目录。再进入系统，就可以在那个目录看到提取到的原始文件了。之后，就可以开始反编译、修改了。注意，有的BIOS可能按F4键没用，而需要Fn+F4，所以，如果不确定的话，F4 和 Fn+F4 都按一次。提取的时候，你不会看到任何是否完成的提示，只是会因为正在写文件而出现一点延迟现象。如果是用Clover的USB设备（U盘、硬盘）启动的，文件会被提取到USB设备，这样会感到比较明显的延迟。  
     有时候，Clover提取的SSDT可能会重复。重复的文件，会影响到反编译。如果你反编译的时候看到了这样的提示信息（duplicate definitions），就需要仔细检查重复的是哪个了（例如，看大小）。由于存在这样的问题，所以在Linux提取比较好。因为我还没有见过在Linux提取的出现重复的情况。

**在Linux提取**（推荐，还可以顺便提取声卡codec）  
     在Linux系统里，可以直接在系统分区里找到原始ACPI文件。你可以在 /sys/firmware/acpi/tables 和 /sys/firmware/acpi/tables/dynamic 看到。可以简单地用命令进行拷贝。  
  
      Linux系统**不需要安装**，做一个启动U盘就可以了：<http://www.ubuntu.com/download/desktop/create-a-usb-stick-on-windows>  
      （3楼有楼主翻译的制作步骤）  
  
打开Linux的终端输入：  
sudo cp -R /sys/firmware/acpi/tables DEST  
**注意**： 用FAT32格式的USB设备（U盘、硬盘）的挂载点的路径，替换命令里的 DEST  
  
      你的U盘或硬盘需要是FAT32格式的。FAT32格式可以避免出现文件权限问题，因为FAT32格式在Linux系统里没有文件权限的问题。用于替换DEST的 自动挂载的USB设备的名字，取决于你的Linux版本和它的启动方式。你可以在终端输入 **mount** 命令，来**查看当前的挂载点**。或者，在图形界面，把鼠标移到文件管理器上相应的USB设备名上。

**反编译ACPI文件**  
      虽然提取到的aml文件可以用MaciASL直接打开，但不建议这样做。用MaciASL直接打开aml文件，是软件在做对打开的这个文件的单独反编译。然而，ACPI文件之间有着紧密的联系，只对一个文件单独反编译，得到的文件会包含许多难于修复的错误。  
  
      所以，最好的方法是，使用iasl编译器，同时反编译所有文件。要做的准备工作有，新建一个文件夹用于存放所有DSDT和SSDT文件，并重命名所有文件，例如加上 .aml 扩展名。下载最新的iasl编译器来进行反编译。这里是Rehabman的修改版编译器：<https://bitbucket.org/RehabMan/acpica/downloads>。把iasl编译器拷贝到/usr/bin目录，会让调用iasl更方便。  
  
终端的反编译命令：  
cd “你存放DSDT和SSDT的文件夹名”  
iasl -da -dl \*.aml

     注意：不要用-da 命令去反编译 DSDT和SSDT以外的 ACPI文件，那将是无效的。  
  
      反编译成功后，你将得到 所有文件的dsl形式，修改就是使用dsl文件进行。修改好dsl文件后，使用MaciASL的 另存为，并选择 文件格式 "ACPI Machine Language Binary” ，再在文件名那里加上 .aml扩展名，保存。编译保存为aml之后，把它们放到引导程序规定的加载目录（变色龙 /Extra，Clover /EFI/Clover/ACPI/patched）。保存好dsl文件，以后修改还要用dsl文件。

**筛选ACPI文件**  
      生成的SSDT和原始的SSDT会产生冲突。大部分关于电源管理的贴子，会让删除所有原始的SSDT。但删除所有SSDT并不好，特别是对于一些新的电脑来说，很多代码都从DSDT移到了SSDT里，最好不要一棒子把所有SSDT都打掉。只需要 把和CPU相关的SSDT删除就可以了，因为这样的SSDT对黑苹果没有效果（也许它们还会造成其它问题），留下其它有用的SSDT，用于打补丁或者就保留着不做修改。  
  
      在反编译成功后，一个一个地打开检查dsl文件，以确认每一个具有的功能。如果是和CPU有关的，就把它拖到其它地方（也可以直接删除），最后放到加载目录的，不需要这些提取的CPU相关的SSDT。一般，和CPU相关的SSDT包含了 Scope \_PR.CPIx 这样的声明，可以通过这样的名字排除文件。  
  
      额外说明：对于Rehabman的联想U430，他试验了使用这些提取的CPU相关的SSDT（当然，是修复了错误的）。不确定这是不是有什么好处，但使用了提取的CPU相关SSDT，使得他的DSDT不再需要打"Fix PNOT/PPNT”这样的补丁了，好像电脑工作得很完美。  
  
以下是**常见的SSDT的特点和功能**：  
－**SATA**：可以留，也可以不留，自己决定就好。  
－**PTID**：一般对苹果系统来说，是没用的。也许这个文件还会有很多错误。一种比较少的应用情况是，在这个文件里，可以找到 读取风扇转速，温度，或者其它系统状态 的线索。  
－**IAOE**：如果存在这个文件，通常来讲，它会在DSDT里的 \_PTS 和 \_WAK 方法被调用。如果删除这个文件，就会影响到睡眠。  
－**GFX0**：通常包含了 Device GFX0 的SSDT，是和 集成显卡 相关的。亮度补丁就是针对这样的文件的。老的笔记本（Haswell以前的），GFX0一般是定义在DSDT里。Haswell的笔记本，它被定义在SSDT里（虽然也可以移到DSDT里）。  
－**PEGP**：包含PEGP的文件，一般是和 独显 和 双显卡切换 有关。有时候不只有一个这样的SSDT，为了实现 屏蔽独显 等功能，你需要保留所有包含PEGP的SSDT，并对它们做一些修改。  
  
      我们需要弄清楚，每个SSDT的大概是做什么的，哪些需要删除，哪些需要不做修改地保留，哪些需要修改。

**修复错误**  
      即使同时反编译所有文件（使用iasl的-da参数），还是难免存在错误。反编译后的文件之所以还会有错误，是因为一直以来iasl编译器的版本都在更新、iasl的实现细节变化了、不同的电脑与系统环境。存在这些错误的原因可能是（Rehabman认为），里面调用到的一些方法，实际上是Windows内部实现的（例如 MMTB方法和MDBG方法）。实际也有这样的情况，ACPI文件的代码原本就存在缺陷或者某些代码本来就是无效的（很难说得清楚）。  
  
      好了，在确认哪些文件是你需要的之后，就需要开始对它们进行修改了（修复错误、添加功能性代码），直到编译没有错误。现在有很多补丁源可以提供使用。  
  
MaciASL软件：<https://github.com/RehabMan/OS-X-MaciASL-patchmatic>  
Rehabman的综合补丁源：<https://github.com/RehabMan/Laptop-DSDT-Patch>  
  
（楼主补充：  
1、**MaciASL汉化版**：<http://bbs.pcbeta.com/viewthread-1495779-1-1.html>  
2、**Rehabman补丁源**（汉化版自带，添加到软件才能访问）：<http://raw.github.com/RehabMan/Laptop-DSDT-Patch/master>  
）  
  
      注意：以上补丁源里的**补丁**，**没有在DSDT Editor软件里测试过**。DSDT Editor软件存在着很多bug，内置的编译器也很老了。  
  
      在看别人的项目页面的时候，都请仔细阅读作者提供的README，到正确的地址下载软件，学习配置MaciASL。修复语法错误的补丁的名字，以[syn]开头。例如，对于一些比较老的电脑，提供了"Fix \_PLD Buffer/Package Error", "Fix TNOT Error", 和 "Fix FPED Parse Error”补丁。为了确定你需要用哪个补丁，需要看 编译结果 窗口给出的 提示信息，再定位到提示的行，查看错误行前后都有什么样的代码。你也可以尝试，只要Changes不为0或者预览框里有显示（说明可以打）的补丁，都可以打。如果你不熟悉编译结果的错误提示，可以一个一个试验看看能不能打。  
  
      对于一些错误，你可以简单地删除造成错误的那行代码来解决。当然，这要取决于这一行代码对于功能的实现，是否是必需的。例如，对于由External语句造成的错误，一般可以删除那行代码以修复错误。如果需要，你自己可以做一个删除这样的错误行代码的补丁。  
  
      学习一下ACPI规范 和 具有一些编程经验，有助于处理DSDT和SSDT。  
  
      我们的最终目的，是得到 编译没有错误的dsl文件（warnings/remarks/optimizations都不用管）。一旦你修复了所有错误，就可以开始为了你的黑苹果 添加功能、解决某些问题，而修改dsl文件了（使用现有补丁或手工修改）。

**常用补丁**  
      一般，你需要**先弄清自己的需求**，**再根据需求选择要用的补丁**。但也有一些补丁是通用的，并且这些补丁一般不会造成什么问题。例如，Rehabman的补丁源里的：  
  
"Fix \_WAK Arg0 v2"  
"HPET Fix"  
"SMBUS Fix"  
"IRQ Fix"  
"RTC Fix"  
"OS Check Fix"  
"Fix Mutex with non-zero SyncLevel"  
"Fix PNOT/PPNT"  
"Add IMEI”  
  
      **USB补丁**，用于解决 睡眠自动唤醒 的问题。  
  
根据自己电脑的硬件配置选择USB补丁：  
"6-series USB"  
"7-series/8-series USB”  
  
      **USB3 Mutliplex补丁**，有助于加载AppleUSBXHCI.kext（原生USB3.0驱动），于是就不需要安装第三方的GenericUSBXCHI.kext了。这个补丁是基于Mieze的发现而制作的。大部分DSDT需要对它做一些修改，才能使用（不然可能会提示有错误）。例如，对于惠普的Probook，Rehabman制作了这个补丁的修改版本。联想u310/u410可以直接使用这个补丁：  
"7-series USB3 Multiplex”  
  
      如果你的电脑是Haswell架构、8系列芯片组的，并且还没有加载AppleLPC.kext（电源管理的一种），你可以使用这个补丁，注入一个兼容ID，以加载AppleLPC：  
"Haswell LPC”  
  
      还需要考虑对一些设备名进行改名，并且，改名还要改得彻底。可以把一些设备名改成符合苹果规范的名字。（例如，使用"Rename GFX0 to IGPU”补丁，可以加载集成显卡的电源管理）。对于这一情况，DSDT/SSDT里，包含的所有GFX0，都需要改成IGPU（对所有含有GFX0的文件，都应用这个补丁）。  
  
      关于重复的标识符定义：你必须确保，你修改后的文件，不包含重复的标识符。典型的例子有，在需要添加一个\_DSM方法的时候，在同一个范围里，原本已经存在\_DSM方法了。编译器就会提示标识符重复。为了避免这样的问题，你可以一开始就对DSDT/SSDT用”Remove \_DSM methods”补丁，删除原本存在的所有\_DSM方法。

**关于一些特别的需求的修改**  
**电池状态显示**：  
1、原贴：[GUIDE] How to patch DSDT for working battery status](http://www.tonymacx86.com/mavericks-laptop-support/116102-how-patch-dsdt-working-battery-status.html)  
2、楼主翻译：<http://bbs.pcbeta.com/viewthread-1521462-1-1.html>  
**屏幕亮度调节**：  
1、原贴：<http://www.tonymacx86.com/yosemite-laptop-support/152659-guide-patching-dsdt-ssdt-laptop-backlight-control.html>  
2、楼主翻译：<http://bbs.pcbeta.com/viewthread-1571456-1-1.html>

屏蔽 英伟达/AMD Radeon独显：TBW，To Be Written，以后再写。

      当你在看某个特定型号的安装教程的时候，作者都会介绍用了哪些补丁，或者直接提供下载。你能容易的看出来，他们的补丁和Rehabman的补丁，在语法上是相似的。（例如这样的语法：'into\_all method label FOO code\_regex xxyy removeall\_matched;’）。这些补丁语句需要粘贴到MaciASL的补丁窗口，再应用。  
  
      如果你对制作补丁有兴趣，可以去看这里的MaciASL补丁语法说明：<http://sourceforge.net/p/maciasl/wiki/Patching%20Syntax%20Grammar/>  
  
      注意：在许多情况下，DSDT的补丁是和某个第三方kext、打过补丁的kext、Clover实时打补丁的kext相结合使用的（通过DSDT代码加载kext）。

**仿冒声卡驱动相关的补丁**  
      有两个和仿冒声卡驱动相关的补丁，需要和仿冒的声卡kext一起使用：  
"Audio Layout 12" (根据实际情况，自己修改 layout-id，例如把 12 改成 28 等)  
"IRQ Fix”（解决需要-f才能加载声卡驱动的问题）  
  
      注意，你需要有适合你的声卡的AppleHDA，还要知道这个仿冒声卡驱动用的是哪个layout-id。layout-id的值，是由修改驱动的人决定的。  
  
      要查看别人修改的AppleHDA用的layout-id是哪个值，首先，你需要知道自己声卡的codec id的十进制形式（例如，0x10ec0269 = 283902569）。然后，去查看 AppleHDA.kext/Contents/PlugIns/AppleHDAHardwareConfigDriver.kext/Contents/Info.plist，查看HDAConfigDefault节点下面的codec id（这个节点下，可能有一组或多组数据，所以，刚才的283902569就用上了）。和你的codec id匹配的那组里，LayoutID的值，就是你需要的layout-id值。有的驱动修改者，同一个codec id可能会对应多个layout-id，选择一个你需要的layout-id注入。

**让引导加载修改、编译好的文件**  
      为了使用你修改、编译好的DSDT/SSDT，你需要把它们放到引导能加载它们的目录。每种引导的加载目录是唯一的，且对文件的命名要求也不同。如果用MaciASL保存，另存为的时候，文件格式一定要选择"ACPI Machine Language Binary”（文件－另存为－文件格式），再自己加上 .aml 后缀。如果不选择文件格式，就相当于，硬是把dsl文本文件，存成了aml字节码文件（没有经过编译器转换），这会造成无法预知的问题。  
  
      **Clover**：aml文件应该放到Clover所在的分区（通常是EFI分区），如果存在 EFI/Clover/ACPI/patched. DSDT.aml，引导程序就会用这里的DSDT，替代掉原始的DSDT，提交给系统。Clover 3062版本以前，SSDT的命名必须是这样的：SSDT-x 或 SSDT-xx，x是一个整数（一直支持到SSDT-19）。Clover允许SSDT的名字不连续（例如，SSDT-1.aml, SSDT-5.aml, SSDT-6.aml是可以的）。Clover 3062以后的版本，会加载所有在ACPI/patched里的aml文件，对名字没有严格限制了。一定要注意，SSDT的加载顺序非常重要，你留下来的SSDT，要保持提取的时候的顺序。  
  
      关于Clover 3062版本以后的注意事项：ACPI/patched里SSDT的加载方式的改变，造成了加载顺序的不确定性。你应该要使用 config.plist/ACPI/SortedOrder 功能，指定SSDT的加载顺序。到这里可以看到更多细节：<http://www.projectosx.com/forum/index.php?showtopic=2656&view=findpost&p=43690>。Clover 3088版本以后，实现了SortedOrder参数。更多相关信息可以到这里查看：<http://www.projectosx.com/forum/index.php?showtopic=2562&view=findpost&p=43674>  
  
      **变色龙**（或Chimera）：修改、编译好的DSDT/SSDT，应该放到 系统盘的 /Extra（或者你的引导程序所在的其它地方的 /Extra）。如果/Extra里存在DSDT.aml，引导程序就会用这里的DSDT，替代掉原始的DSDT，提交给系统。变色龙要求，SSDT的名字是连续的。这样的情况，文件都会被加载：SSDT.aml, SSDT-1.aml, SSDT-2.aml, SSDT-3.aml，直到引导找不到接下来的名字的文件。另外，这样的情况：SSDT.aml, SSDT-1.aml, SSDT-4.aml, SSDT-5.aml，只会加载SSDT.aml 和 SSDT-1.aml。SSDT-4.aml 和 SSDT-5.aml不会被加载（变色龙会忽略这些文件）。  
  
      最后，如果不先屏蔽原始的OEM SSDT，就不能放自己修改的SSDT。最简单的方法，是设置DropSSDT=Yes（变色龙） 或者 ACPI/SSDT/DropOem=true（Clover），从而在系统启动的时候，先屏蔽原始的SSDT，之后再把目录里的SSDT（没有修改或修改过）提交给系统。

**Floating regions**  
      根据ACPI规范，OperationRegion可以被具体定义为 MMIO region, SystemMemory region, EmbeddedControl region 等。这些region通常包含了 修正过的 地址值，这些地址值 由 硬件构成、BIOS版本、BIOS设置 决定。这些region随时可能 随机地、不可预计地 发生改变。也就是所谓的"floating regions”（动态域）。  
  
      我们提取的DSDT/SSDT里的Regin，只是某一个时间点的状态，当BIOS发出命令，需要变动地址值时，我们提取的文件是做不到这样的变动的。在这样的情况下，你也许能注意到，某一个特定的功能变得断断续续，或者可能随机地出现一些稳定性问题。  
  
      综上而论，使用Clover的FixRegions功能，将是很好的解决办法。你可以在Clover的Wiki找到关于这个功能的说明。（国外的）论坛里的安装教程提供的config.plist都配置了这个参数。注意：只有在DSDT里的floating region，才能用这个参数修复。要修复SSDT里的floating region是非常困难的，目前，除了不处理SSDT里随机变动的floating region以外，还没有什么好的解决方案。研究floating region的处理，超出了本教程讨论的范围。

**本教程用到的资源**  
MaciASL (RehabMan fork): <https://github.com/RehabMan/OS-X-MaciASL-patchmatic>  
patchmatic: <https://github.com/RehabMan/OS-X-MaciASL-patchmatic>  
iasl (RehabMan fork): <https://bitbucket.org/RehabMan/acpica/downloads>  
ACPI 规范: <http://acpi.info/spec.htm>  
  
RehabMan github: <https://github.com/RehabMan?tab=repositories>  
  
Clover laptop guide: [[GUIDE] Booting the OS X installer on LAPTOPS with Clover UEFI](http://www.tonymacx86.com/yosemite-laptop-support/146864-guide-booting-os-x-installer-laptops-unibeast.html)  
Clover config.plist files for laptops: <https://github.com/RehabMan/OS-X-Clover-Laptop-Config>  
  
Clover thread: <http://www.projectosx.com/forum/index.php?showtopic=2562>  
Clover changes: <http://www.projectosx.com/forum/index.php?showtopic=2656>  
Clover installer discussion: <http://www.projectosx.com/forum/index.php?showtopic=3191>