1.1 Glossar - Begriffe rund um das Thema Hackintosh

Also fangen wir mal an und zwar am Besten erstmal mit einem kleinen Überblick bevor es dann ins Detail geht. Wer sich mit dem Thema "MAC OS X", UNIX und Hackintosh beschäftigt, der wird unweigerlich früher oder später über bestimmte Begriffe stolpern, die fest in den Kontext dieser Systeme gehören. Hier eine kleine Übersicht:

Kernel - Ohne Ihn geht gar nix!

Der Kernel ist der Kern eines Betriebssystems und somit freilich etwas, dass es nicht nur beim Hackintosh gibt sondern in jedem Betriebssystem. Der Kernel bildet die Unterste Ebene des Betriebssystems und regelt die Kommunikation der Software mit der Hardware, er regelt und verwaltet Prozessor und Speicherzugriffe und koordiniert die Abarbeitung von verschiedenen Aufgaben (tasks,threads). Wer sich ernsthaft dafür interessiert findet in diesem WIKI Beitrag eine Menge lesenswertes zum Thema Betriebssystemkerne und deren Aufbau und Aufgaben.

Der Kernel bei MAC OS heißt **mach_kernel** und gehört zur Familie der Mikrokernel was bedeutet, dass in ihm nur die grundlegendsten Systemfunktionen "Hart verdrahtet" vorliegen und alles weitere zur Laufzeit durch Treiber abhängig von der eingesetzten Hardware zugeladen wird. Diese Treiber bezeichnet man in der Unix Welt als Kernel Erweiterungen oder in Englisch halt KernelEXTensions (KEXT).

KernelExtensions oder auch .kext - Der Motor für viele Aufgaben

Wie wir schon im Abschnitt über den Kernel gelernt haben sind in ihm nur grundliegende Systemfunktionen direkt enthalten, alles weitere wird bei Bedarf über eine Extension (kext) zur Verfügung gestellt und zwar in den meisten Fällen genau dann, wenn es benötigt wird und nur genau so lange wie es benötigt wird. Der Vorteil dieses Vorgehens ist, dass der Kern des Systems klein bleibt und damit schnell und effizient arbeiten kann.

KernelExtensions können dabei ganz unterschiedliche Aufgaben übernehmen je nachdem in welcher Ebene des Kernels sie zum Einsatz kommen stellen sie systemnahe Funktionen bereit, fungieren als Hardwaretreiber oder stellen Funktionen für die grafische Darstellung, das Fenstermanagement usw. bereit.

Hö, Ebene des Kernels wie jetzt? Stellt Euch den Kernel wie eine Zwiebel vor, genau wie eine Zwiebel hat auch ein Kernel verschiedene Schichten mit verschiedenen Aufgaben. Bei einem Kernel nennt man diese Schichten auch Layer und je näher ein solcher Layer am Kern ist um so näher sind seine Funktionen an der Hardware.

Einige in der Hackintosh Welt zum Standard Repertoire gehörende Extensions sind:

- FakeSMC.Kext (Unterste Ebene, simuliert dem Kernel einen Hardwarebestandteil der nur in MAC´s vorkommt)

- NullCPUPowermanagment.kext (Untere Ebene, Verhindert das Laden des Apple CPU Powermanagements)

Genau genommen ist eine kext eigentlich ein Verzeichnis in dem sich neben dem zur eingesetzten Kernelversion binären Anteil meist auch noch Konfigurationsdateien befinden. Diese Konfigurationsdateien liegen entweder ebenfalls binär vor (selten) oder in Form einer recht gut les- und editierbaren XML Datei. Man erkennt die Konfigurationsdateien sehr gut daran, dass sie auf das Kürzel .plist enden, was für

"PropertyList" steht.

PropertyLists (plist) - Kleine Datei, Große Schaltzentrale

Die plist ist ein wahres Multitalent und wird auch genau so eingesetzt, neben allgemeinen Informationen zur Systemkonfiguration können sie auch Informationen zur unterstützen Hardware eines Treibers oder einfach auch sonst nur Konfigurationseinstellungen für alles Mögliche enthalten.

Über 2 Vertreter der Gattung plists stolpert jeder Hackintoshler unweigerlich irgendwann in seinem Leben nämlich über die com.apple.boot.plist (neuerdings auch org.chameleon.boot.plist) und die smbios.plist. Propertylists sind immer gleich aufgebaut und enthalten immer eine Sammlung von Schlüsseln und Werten, hier sind sie durchaus ein Stück weit vergleichbar mit der Registry unter Windows. Hier ein typisches Beispiel für den Aufbau einer plist:

Code

```
1. <dict>
2. <key>DSDT</key>
3. <string>/Extra/dtdt.aml</string>
4. <key>EHCIacquire</key>
5. <string>Yes</string>
6. <key>EthernetBuiltIn</key>
7. <string>Yes</string>
8. </dict>
```

Im Beispiel oben fehlt jetzt noch der einleitende XML Tag, da aber die Forensoftware das zerballert lass ich den weg, es ist denke ich auch so gut möglich einen Eindruck zu gewinnen wie die Dinger aufgebaut sind.

Für die Basics soll es das erstmal gewesen sein, aber es geht hier in den nächsten Tagen weiter und ich freue mich auf Eure Wünsche, Kritik oder Anmerkungen zum Thema in DIESEM Thread

Nachdem der erste Teil sich ja eher mit den Basics rund um MAC OS und Unix Systeme beschäftigt hat soll es ab jetzt dann doch zumindest ein wenig ums Eingemachte gehen wobei ich hier klar sage, alles was ich schreibe ist Backgroundwissen und soll helfen zu verstehen, warum eigentlich manche Dinge bei einem Hackintosh nötig sind und warum man sie braucht bzw. auch aufzeigen, auf welche Dinge man verzichten kann, wenn man weiß was man tut.

Der Weg zu einem Hackintosh ist oft kein einfacher, besonders dann nicht wenn man relativ unbedarft an die Sache rangeht. Klar eine MAC OS DVD kaufen kann jeder, dazu bedarf es ausser dem Weg in den nächsten MediaMark, AppleStore oder Gravis Laden nicht viel und dies sollte ausdrücklich (egal was Ihr auch immer gelesen haben mögt) Euer erster Schritt sein! Kauft EUCH das Original, denn nur so ist es in Deutschland zumindest halbwegs legal möglich MAC OS X auf einem Computer laufen zu lassen, der nicht von Apple kommt. Aber was nun, ich habe die DVD hier liegen, packe sie ins Laufwerk und mein PC ignoriert sie ganz egal was ich mache, warum ist das so? Warum kann ich MAC OS nicht einfach wie Windows oder Linux installieren?

Kurz und knapp gesagt, weil ein PC eben nunmal kein MAC ist und damit mit dem Zeugs, dass da auf der DVD drauf ist nichts anfangen kann. Die meisten im Handel befindlichen PC´s sind Windows optimiert und daher mit einem BIOS ausgestattet, dass beim Start des PC´s die installierte Hardware abklappert und alle nötigen Informationen über die installierte Hardware in Form einer mehr oder weniger standarisierten Tabelle (dsdt) an das zu startende Betriebssystem übergibt. Während DOS, Windows und die diversen Linux Distributionen darauf optimiert sind und somit damit arbeiten können geht MAC OS als Unix Derivat hier einen anderen Weg und so bedarf es der Hilfe eines speziellen Bootloaders um MAC OS zu starten.

Und warum ist das so?

Um diese Frage zu beantworten bedarf es einer kleinen Geschichtsstunde. Damit ein Computer überhaupt funktionieren kann benötigt er ein Minisystem, dass vor dem eigentlichen Start des Betriebssystems abläuft und Informationen über die im Computer enthaltene Hardware sammelt. Hier haben sich zwei verschiedene Systeme etabliert.

BIOS:

Für den breiten Bereich der Personalcomputer hat sich mit Einführung der CPU 8086 (bis 4 Mhz Takt auf 16 Bit Bus) durch Intel das sogn. "Basic Input Output System" etabliert und bis heute durchgesetzt. Das BIOS ist ein alter Bekannter in der PC Welt und wird weitestgehend bis heute allen nötigen Anforderungen gerecht. Freilich war die Entwicklung der PC´s so wie wir sie heute kennen ein langer Prozess auf den ich nicht weiter eingehen mag, da es den Rahmen sprengen würde, aber wer mag kann hier Infos finden. Anfangs war auch Apple auf der x86 Linie und somit auch auf der "BIOS" Linie unterwegs, der Apple II und diverse andere Produkte aus dem Hause Apple waren eng genommen nichts anderes als PC´s. *Gutes braucht Veränderung* und so wurde der ewige Querdenker Jobs inkl. seiner Visionen von einem Macintosh Rechner regelrecht demontiert und letztlich von Apple sogar rausgeschmissen. Schlecht für Apple, gut für uns denn Jobs hat seine Visionen mit seiner Firma "NEXT" weiterleben lassen und mit "NEXT STEP" den grafischen Aufsatz für Unix Betriebssysteme geschaffen, dass wir heute als MAC OS kennen. Wer mehr darüber lesen mag, dem sei das Buch "Die Apple Story" ans Herz gelegt.

EFI:

Während die PC Welt glücklich wurde mit den X86 Prozessoren und damit einhergehend zunächst mit MS DOS (DR DOS) und später dann auch Windows als grafischen Aufsatz für DOS (wir erinnern uns Windows lies sich früher mal mit dem Befehl "win" vom der Kommandozeile aus starten) waren freilich an Unix ganz andere Anforderungen gestellt als an das Dos/Windows Gespann. Unix Rechner egal ob Server oder Workstation waren eigentlich immer einem bestimmten sehr eng definierten Einsatzgebiet zugedacht und so ist es nicht weiter verwunderlich, dass sich neben der Software (UNIX) auch die Hardware in einer andere Richtung bewegt hat als die der PC Konkurrenz und mit ihr auch das aus dem PC Bereich bekannte BIOS. Wo in der "Wintel" Welt eine Generation der x86 Prozessoren die nächste ersetzt hat hat sich in der UNIX Welt eine heute als RISC bekannte Bauform für Prozessoren durchgesetzt. Bekannte Vertreter dieser Bauform sind unter anderem die 68000er CPU´s von Motorola, die auch schon in den Mac´s der ersten Stunde eingesetzt wurden. Zusammen mit dieser CPU Generation hat sich (unter anderem getrieben von Intel) ein BIOS Standart entwickelt der bis heute als EFI (Extensible Firmware Interface oder auch UEFI) bekannt ist und der schon in den MAC´s der ersten Stunde zum Einsatz gekommen ist.

Fassen wir also mal kurz zusammen:

PC (bis heute) = WinTel = Bios

MAC = Bis G5 = Motorola/IBM = UNIX = EFI

MAC = Ab Intel = INTEL = immer noch UNIX = (u)EFI

Jetzt wissen wir, warum wir nicht einfach eine MAC OS DVD in unser Laufwerk packen können und

erwarten dürfen, dass das dann am Ende sogar funktioniert, denn unser "unwissender" mit einem Bios ausgestatteter PC, kann nunmal mit den auf ein vorhandenes EFI abgestimmten Inhalten einer MAC OS Installations DVD,Sticks nichts anfangen. Aber zum Glück kann man ja hier tricksen. Was ist also nötig um das MAC OS auf den PC zu bekommen? Richtig ein Bootloader, der MAC OS vorgaukelt ein Apple konformes EFI zur Verfügung zu haben und da gibt es eine schier endlose Auswahl aus der man sich das passende aussuchen kann/darf (Der Downloadbereich hier im Forum ist da ein ganz heißer Tipp).

Bootloader - Mittler zwischen den Welten:

Die beiden verbreitetsten und bekanntesten Vertreter Ihrer Art dürften wohl Chameleon und Chimera sein. Wobei Chimera den selben Quellen entstammt wie Chameleon, allerdings durch TonyMac einige "Zusätze" erfahren hat, die den Bootloader aufwerten. Beide Bootloader eignen sich dazu ein installiertes MAC OS von der Festplatte aus zu starten und bieten verschiedene Möglichkeiten MAC OS eine möglichst perfekte EFI Schnittstelle zu liefern.

Seit einiger Zeit gibt es auch noch einen dritten im Bunde: Clover. Clover ist um einiges flexibler als die vorgenannten aber auch schwieriger optimal einzustellen.

Aber was machen die Dinger nun genau?

Im Großen und Ganzen schlagen sie die Brücke zwischen PC BIOS auf der einen und dem Wunsch von MAC OS nach einem vorhandene EFI auf der anderen Seite indem sie die Informationen aus dem PC BIOS sammeln und in einem EFI kompatiblen Format an MAC OS übergeben, so dass MAC OS eben glaubt, es würde in Wirklichkeit auf einem MAC laufen. Der Bootloader führt also Quasi zur Laufzeit einen Biospatch durch und stellt am Ende dieses Prozesses MAC OS ein "BIOS/EFI" an die Seite mit dem MAC OS leben kann. Nun eigenen sich nicht alle Bootloader gleichermaßen dazu eine MAC OS Installtion von DVD aus zu starten, zumal hier oftmals neben dem eigentlichen Loader auch noch hardwarespezifische Treiber nötig sind die MAC OS freilich so nicht mitbringt. Aber auch für diesen Zweck gibt es Lösungen die entweder mittels USB Stick oder CD/DVD einen preboot durchführen und alles nötige laden um die Installtion zu starten. Bekannte Vertreter sind hier 1,2,3boot oder EmpireEFI (beide auch im Download zu finden) die eine Bootbare CD generieren und wo nach Start des Bootloaders dann die CD/DVD gegen das Installationsmedium getauscht wird.

Fazit:

Es kommt also auf die inneren Werte an. Wie in jeder guten Beziehung erwartet eben MAC OS von seinem Partner (in unserem Fall dem PC), dass ein Mindestmaß an Grundsympathie vorhanden ist bevor es sich dazu herablässt sich auf die Beziehung PC <-> MAC OS einzulassen. Da diese "Inneren Werte" bei den meisten PC´s nun mal einfach von Haus aus schon nicht vorhanden sind, bedarf es einer Paartherapie (Bootloader, EFI Emulation) um das ungleiche Paar doch aneinander zu binden. Wie man nun diese Paartherapie möglichst elegant und effizient gestaltet lernen wir im nächsten Artikel zur Serie wo es dann unter anderem um "Differenzierte System Definitions Tabellen" (dsdt) gehen wird.

Nach dem letzten Artikel wissen wir nun, dass sich PC´s und MAC´s nicht nur äußerlich sondern auch in den „inneren Werten“ gehörig unterscheiden. Ebenso haben wir gelernt, dass sich durch entsprechende Emulatoren und Bootloader diese Unterschiede in den „inneren Werten“ weitestgehend umschiffen lassen und es so möglich wird MAC OS auf dem PC zu installieren.

Wenn wir also nun ausgerüstet mit einem originalen MAC OS und entsprechendem Bootloader bzw. entsprechender Boot CD tatsächlich den PC dazu bewegen konnten die Installations DVD zu starten,

können wir entweder viel Glück haben und der Installer begrüßt uns freundlich mit seinem Startscreen oder aber wir finden einem Bildschirm vor, der wie folgt aussieht:

kernelpanic102.jpg

Image not found or type unknown

oder enden bei einem "Spinning Wheel of Death"

spining-wheel.jpg

Image not found or type unknown

Ein doofes Gefühl, so nahe davor und doch noch immer so meilenweit davon entfernt, aber warum klappt das jetzt nicht?

Na ja, PC Hardware ist eben trotz Bootloader und EFI Emulation noch immer PC Hardware und während naturgemäß die Firmware der Apple Rechner (EFI) optimal an MAC OS angepasst ist, ist das beim PC BIOS noch lange nicht der Fall, hier muss schon noch einiges beachtet werden damit man das Objekt der Begierde auf die Platte kriegt. Fassen wir also mal kurz zusammen:

MAC :

Hard und Software eng aufeinander abgestimmt, auf Seiten der Software ist keine Unterstützung einer "breiten" Hardwarebasis nötig, ein Minimum an sehr spezialisierten Treibern reicht aus.

PC :

Das PC Bios ist recht flexibel gehalten, da PC Mainboards meist viele unterschiedliche CPU Varianten und eine schier endlose Auswahl an Erweiterungskarten unterstützen. Die PC Welt verlässt sich bei der Unterstützung der Hardware ganz auf das Betriebssystem und dessen Treiberbasis und liefert über die Firmware nur rudimentäre Informationen an das OS die ausreichen müssen um den richtigen Treiber zu identifizieren. Hier zählt klar die Devise, läuft oder nicht!

Also, was braucht es jetzt noch um MAC OS an den Start zu bringen?

Zuerst geht es mal dem BIOS buchstäblich an den Kragen und keine Sorge, viel könnt Ihr hier nicht kaputt machen, also frei ans Werk

Damit MAC OS überhaupt was mit unserer Hardware anfangen kann müssen wir mit einigen Einstellungen im BIOS dafür sorgen, dass unser Bootloader/EFI Emulator MAC OS überhaupt brauchbare Werte liefert. Also ab ins BIOS und folgende Einstellungen soweit möglich vornehmen:

-> Integrated Peripherials

sATA auf AHCI einstellen

sATA Mode auf "IDE" oder "Legacy IDE" einstellen

ATA Mode auf AHCI oder IDE einstellen (nur wenn noch alte (e)IDE Platten eingesetzt werden)

-> PowerManagement

ACPI auf Enabled stellen

ACPI auf 64Bit Mode stellen

ACPI Suspend auf S3 stellen (hier probieren, bei einigen Boards muss es S1 sein)

Was sind denn nun wieder AHCI und ACPI und wieso beeinflusst das den Startprozess von MAC OS?

AHCI ist eine Abkürzung und steht für "Advanced Host Controller Interface" Kurz und knapp gesagt nimmt es den Herstellern von Betriebssystemen aus der Pflicht einen, auf die eingesetzte Controller Hardware abgestimmten Treiber zum ansteuern von Festplatten mitzuliefern indem es einen einheitliche offenen Standard zum Ansprechen von Festplatten unabhängig vom eingesetzten Chipsatz definiert. Da Apple in MAC OS voll und ganz auf den AHCI Standard setzt und somit keinerlei spezialisierte Treiber für Festplattencontroller mitliefert hilft die entsprechende Einstellung im BIOS das "Spinning Wheel of Death" (Still waiting for Boot Device...") zu überwinden.

ACPI ist natürlich ebenfalls eine Abkürzung, sie steht für "Advanced Configuration and Power Interface" ähnlich wie AHCI schafft auch ACPI einen offenen und einheitlichen Standard zur Kontrolle und Verwaltung der Energiesparfunktionen eines Computers. Anders als das in der WinTEL Welt bis heute gebräuchliche APM (Advanced Power Management) legt ACPI die Kontrolle der Stromsparfunktionen vollständig in die Hände des Betriebssystems. Da MAC OS ein funktionierendes und eingeschaltetes ACPI vorzugsweise im 64Bit Modus erwartet um nicht zuletzt das Speedstepping der Intel CPU´s zu steuern macht man bei falschen Einstellungen schnell die Bekanntschaft mit einer Kernelpanik (erstes Bild).

Wenn jetzt alles richtig eingestellt ist sollte einer Installation von MAC OS eigentlich fast nichts mehr im Wege stehen, jedenfalls sollte sich nun die DVD bzw. unser Stick booten lassen und der Installer auch starten, wobei eigentlich wäre das ja schon fast zu einfach, oder? Es gibt tatsächlich noch einige Klippen, die es zu umschiffen gilt und die man am besten gleich beim erstellen seines Bootmediums im Auge haben sollte, denn sonst hat man schnell im besten Falle nur viel Zeit investiert, im schlimmsten Falle einen Rohling nutzlos verbrannt. Sofern man sich auf CD/DVD Lösungen mittels Boot 1.2.3, iBoot und Co verlässt sollte alles nötige schon an Bord sein und der Installer einfach so starten. Hat man sich allerdings einen bootfähigen USB Stick gebaut auf dem man Chamelon oder Chimera als Bootloader einsetzt gilt es noch einiges zu beachten, denn es gibt einige "Essentials" die einfach vorhanden sein müssen, damit das System starten kann.

Im Ordner /Extra (direkt auf Root):

- com.apple.boot.plist oder org.chameleon.boot.plist

- smbios.plist

Beide Dateien werden in der Regel direkt bei der Installation des jeweiligen Bootloaders mit installiert. Hierbei empfiehlt es sich den Bootloader über einen Installer wie etwa Multibeast (im Downloadbereich) zu installieren um sicher zu stellen, dass die Dateien vorhanden sind.

Im Ordner /Exta/Extensions:

- FakeSMC.kext

- evtl. je nach Mainboard noch JMICRON.kext

- evtl. NullCPUPowermanagement.kext

Hierbei ist insbesondere die FakeSMC.kext ein absolutes **MUSS** denn ohne diese KEXT werdet Ihr zu 99,5% (es soll Boards geben die ohne auskommen) immer in einer Kernelpanik bzw. bei "Spinning Wheel" hängen bleiben. Neben der in Deutschland unwirksamen EULA (die es verbietet MAC OS auf Hardware zu betreiben, die nicht von Apple ist) hat sich Apple auch einen zugegeben liederlich umgesetzten Hardwareschutz einfallen lassen um zu verhindern, dass MAC OS auf nicht Apple Systemen eingesetzt wird. Alle Intel MAC´s verfügen Hardwareseitig über einen Chip, ein sogenannten SMC (System Management Controller) der von MAC OS beim Bootprozess abgefragt wird und bestimmte verschlüsselte Betriebssystembestandteile entschlüsselt. Freilich fehlt einem 08/15 PC dieser Baustein und die Anfrage läuft ins Leere => KernelPanik. Die FakeSMC.kext wird als "lowlevel extension" vor der Anfrage an diesen Baustein vom Kernel geladen und emuliert den besagten Baustein, ergo MAC OS denkt "Hui ist nen MAC, weiter geht es". Wenn alles passt startet jetzt der Installer und Ihr könnt MAC OS auf die Platte packen.

MAC OS ist installiert und wie jetzt weiter und was ist nun eine DSDT?

Installation gelungen und was folgt ist Ernüchterung? System startet mit 800*600 und die Auflösung lässt sich nicht ändern? Kein Sound? Kein Netzwerk? alles doof? Euch fehlt die Würze aber das kommt im nächsten Artikel, versprochen und da geht es dann auch langsam Richtung DSDT...

Wer die ersten drei Artikel dieser Reihe aufmerksam gelesen hat, dem wird es aufgefallen sein, dies hier ist keine ultimative ***"Ich mache meinen PC zum MAC"*** Anleitung, vielmehr geht es mir darum gewisse Basics die einem im Alltag rund um den Hackintosh, aber auch sonst in der IT Welt immer mal wieder begegnen, in einer auch für Laien verständlichen Form zu vermitteln. Zugegeben, das ganze ist schon ziemlich Unix/MAC OS lastig, aber das muss es auch sein, denn immerhin ist dies hier ein Hackintosh Forum und da geht es nunmal primär um MAC OS

So nun aber genug der einleitenden Worte diesmal gleich zur Tat, ohne viel Gelaber.

- Teil 1 -3 gelesen ? Check !

- MacOS startet? Check !

Also, Euer MAC OS startet und das erste, dass Euch in den meisten Fällen auffallen wird ist, dass weder die Grafikauflösung befriedigend eingestellt, noch irgendein Sound zu hören ist. Besonders fällt dies bei Leopard/SnowLeopard auf die beide beim ersten Start ein kleines Willkommen Video zeigen, dass fröhlich durch die Gegend ruckelt und stumm da steht, wer es noch nie in voller Pracht gesehen hat, so sollte es aussehen und sich anhören:

http://youtu.be/zkZCtPSAGOw

Bei Lion hat Apple zwar auf das Video verzichtet und die Animation lieber in den Installer verfrachtet was zwar nett ist, aber auch nichts an der Tatsache ändert, dass weder die Grafik vernünftig läuft noch Sound zu hören ist. Aber was macht man da nun? Auf jeden Fall ist jetzt mal guter Rat teuer, denn jedes weitere Vorgehen hängt sehr stark davon ab, welche Hardware Ihr wirklich am Start habt und wie nah sie tatsächlich an der Hardware ist, die Apple selbst in den Macintosch Rechnern einsetzt. Fangen wir mal mit der Grafik an....

Apple unterstützt von Haus eigentlich nur eine sehr begrenzte Auswahl an Grafikboards, namentlich sein da

mal NVIDIA und ATI (AMD) genannt, wobei selbst hier nicht jede Karte Unterstützung findet. Sprich es kommt an der Stelle stark auf 1. den GrafikChip an und 2. auf den Bustyp. Auf der sicheren Seite ist erstmal, wer eine Grafikkarte sein Eigen nennt, die in einem PCIe Slot steckt, AGP Karten können teilweise funktionieren, aber das ist immer mit sehr viel Bastelei verbunden und spätestens ab LION ein totales NO GO. Also fassen wir mal zusammen, wir haben eine Graka mit NVIDIA oder ATI Chipsatz und sie steckt idealerweise in einem PCIe Slot...

Methode 1 (boot.plist):

Wir haben eine "Haus und Hof Grafikarte" sprich irgendwas, dass auch in einem echten MAC eingesetzt wird. Wenn wir eine solche Grafiklösung haben, dann machen uns die aktuellen Versionen von CHAMELEON oder CHIMERA das Leben ziemlich leicht, denn es reicht hier vollkommen aus folgendes in die jeweilige boot.plist einzufügen:

Code

1. GraphicsEnabler
2. Yes

Der Bootloader kümmert sich um alles weitere sprich, er gibt MAC OS alle nötigen Informationen zu unsere Grafikkarte mit, so dass sie mit den MAC OS Standard Grafiktreibern angesteuert werden kann. Dieser Eintrag kann Euch auch helfen den unter MAC OS 10.6.8 und unter LION bekannten "PCI Configuration Begin" Fehler zu umschiffen.

Methode 2 (enabler, injektor als KEXT):

Wir haben eine Grafikkarte, die zwar theoretisch mit MAC OS laufen würde, aber praktisch nicht mit Methode 1 zum laufen zu bewegen ist, weil zwar im Grundsatz der Typ unterstützt wird, aber eben das bestimmte Model nicht. Hier helfen oft sogenannte ENABLER oder auch Injektoren aus. Aber was machen die?

Ein Enabler oder Injektor (beides im Grunde das gleiche) macht eigentlich nichts anderes als die vorhandenen Hardwareinformationen in ein Format zu übersetzen, dass es den Apple eigenen Treibern erlaubt die eingesetzte Hardware zu unterstützen. Injektoren oder Enabler kommen immer als KEXT und werden der Ladereihenfolge wegen immer im Ordner /EXTRA/EXTENSIONS installiert, da alle gängigen Bootloader immer erst die KEXT aus /EXTRA/EXTENSIONS laden und dann erst die MAC OS eigenen. Der Enabler oder auch Injektor sammelt dabei Informationen aus dem BIOS und wenn ihm die eingesetzte Grafikkarte bekannt ist, dann "maskiert" er diese als eine passende MACOS bekannte Grafikkarte so, dass der MACOS eigene Grafiktreiber denkt er hätte es physikalisch mit einem Umfeld zu tun, dass auch in echten MAC´s vorkommt. Bekannte Vertreter Ihrer Art sind:

- NVEnabler.kext für NVIDIA Karten und

- ATY_init.kext für ATI Karten.

Beide Kexte können im Übrigen auch dabei helfen den unter LEO 10.6.8 und Lion gerne auftretenden "PCI Configuration Begin" Spaß zu umschiffen.

Methode 3 (EFi String):

Spätestens mit dem Erscheinen von Chameleon und dessen Derivaten (Chimera und Co.) wendet man diese Methode nicht mehr wirklich an, denn die passenden EFI Strings bzw. nötigen Modifikationen an der DSDT werden von den benannten Bootloadern direkt im Bootprozess vorgenommen, so das es einem Eintrag der Geräteeigenschaften (EFI String) in die boot.plist eigentlich nicht mehr bedarf. Wer trotzdem auf diese inzwischen antiquierte Lösung zurückgreifen mag, weil er vielleicht Hardware einsetzt die von den aktuellen Bootloadern nicht mehr als "würdig" anerkannt wird, aber eigentlich trotzdem kompatibel ist, der ist mit EFI Studio (google) gut beraten.

Bis auf sehr wenige Spezialfälle, die immer gesonderter Betrachtung bedürfen, da es hier einfach kein Patentrezept gibt, lassen sich zumindest die meisten "Mainstream" Grafikprobleme so beheben. Fazit, Bild passt nu, aber was ist mit dem Sound oder LAN?

Heute ist nicht aller Tage, da jetzt erstmal die Grafik passt und das Ganze mehr Platz eingenommen hat als ich gedacht hätte, finden die Soundthemen ihren Platz in einem eigenen Artikel, da ich schon jetzt weiß, dass auch dieses Thema umfangreicher wird...