CPUFriend Guide, HWP & Speedstep: X86PlatformPlugin vs ACPI_SMC_PlatformPlugin

Beitrag von „kuckkuck“ vom 26. April 2020, 00:49

CPU Friend Guide

X86PlatformPlugin Customization und Anpassung – Perfektes Speedstepping

Vor und Nachteile des X86PlatformPlugin vs ACPI_SMC_PlatformPlugin in der Diskussion: plugin-type=1 vs plugin-type=0

In diesem ausführlichen Guide geht es um absolutes Feintuning – um das Ziel perfektes Apple-konformes CPU-Speedstepping und Frequenz-/Powermanagement zu erreichen. Ich will ein paar Anpassungsmöglichkeiten und Herangehensweisen für Speedstepping erklären und durchgehen, mitunter wird es aber nicht besonders unkompliziert und nutzerfreundlich.

Das am häufigsten genannte Indiz für korrektes PowerManagement ist meistens das TaktFrequenz Verhalten der CPU. Dies kann unter macOS beispielsweise mit dem Intel Power Gadget ausgelesen werden, hierbei gilt jedoch zu beachten, dass der angezeigte Graph häufig und schnell die Realität ein wenig verzerrt. Deswegen ist es wichtig beim Testen von Speedstep im Intel Power Gadget das Log zu aktivieren und die dabei erzeugte Tabelle im Nachhinein auszuwerten. Hier werden die "echten" Frequenzen angezeigt.

Vereinfacht gesagt stellt macOS zwei Treiber zur Verfügung, die genutzt werden können um allgemein das Powermanagement zu kontrollieren: ACPI_SMC_PlatformPlugin und X86PlatformPlugin. Beide Kexts liegen unter /System/Library/Extensions/IOPlatformPluginFamily.kext/Contents/PlugIns.

Das ACPI_SMC_PlatformPlugin stammt zwar aus Zeiten vor Ivy Bridge Prozessoren, hat aber die schöne Eigenschaft, dass es die benötigten Daten über die CPU größtenteils aus dem ACPI ließt, was in den meisten Fällen dazu führt, dass viele CPU Frequenzen verfügbar sind und die CPU "schön" taktet. Das ACPI_SMC_PlatformPlugin ist für aktuelle Prozessoren und Hardware jedoch nicht wirklich konzipiert, sondern nur ein Fallback. Das X86PlatformPlugin ist das

korrekte Plugin für aktuelle CPUs und Chipsets. Das Plugin sowie die bekannte Technologie XCPM (Xnu Power Management) ist tief in den Kernel Integriert und hat Einfluss auf viele Verschiedene Bereiche:

> Zitat von kuckkuck
>
> Xcpm ist tief in den Kernel Integriert und hat weitreichende Auswirkungen und Dependenzen. Das X86PlatformPlugin sorgt (von seiner Funktion her) für korrektes Speedstepping, aber hat Einfluss auf viel mehr als nur das, wie zB System Capabilities, Sleep, Wake, DVFS, HWP, (AGPM), (Temp-/FanManagement), Shutdown-, Reboot-, Reset- und PowerFailure Monitoring, ME Events, Energiehaushalt/PowerProfiles, Boot Geschwindigkeit, (Systemeinstellungen), etc.

GUIDE:

Das X86PlatformPlugin lädt nur, wenn macOS für die installierte CPU das Property `plugin-type=1` bereitgestellt wird. (`plugin-type=0` sorgt dafür, dass das ACPI_SMC_PlatformPlugin lädt)

Um `plugin-type=1` zu injecten können wir eine SSDT benutzen, ein gutes Template hierfür ist diese SSDT-PLUG. (Für IvyBridge CPUs hier entlang)

Im Gegensatz zum ACPI_SMC_PlatformPlugin holt sich das X86PlatformPlugin seine Informationen nicht aus dem ACPI, sondern aus SMBios-abhängigen Profilen. In /System/Library/Extensions/IOPlatformPluginFamily.kext/Contents/PlugIns/X86PlatformPlugin.kext/Contents/R sind für sämtliche Board-IDs Profile mit Properties und Daten hinterlegt. Zu diesen Daten gehören ebenfalls die `FrequencyVectors` (siehe Plists), welche das Speedstepping der CPU beeinflussen, und mit genau diesen Frequency-Vectors wollen wir uns im Laufe des Guides beschäftigen.

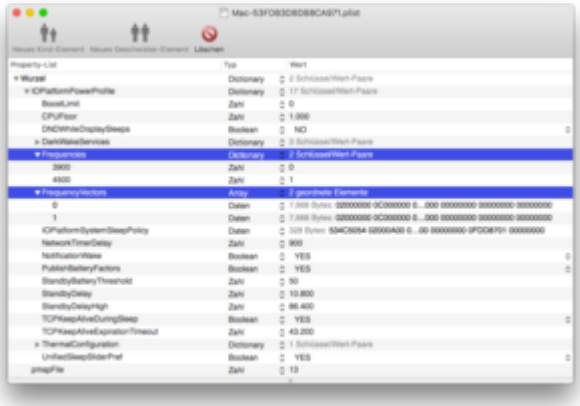
Da die Profile also SMBios abhängig sind, müssen wir jetzt erstmal ein SMBios finden, dass am besten zu unserer CPU passt (wir werden unser per Clover/OC/Oz injectetes SMBios nicht ändern! Bitte weiterlesen). Es empfiehlt sich MacTracker zu nutzen. Ihr solltet unbedingt einen Mac finden, der die gleiche CPU Generation besitzt (zB Kaby Lake). Die genauen CPU-Modell Bezeichnungen können von Mac zu eigener Hardware variieren, also am besten etwas möglichst Nahes wählen. Bei zB einem i5-8265U bietet sich das MacBookPro15,4 SMBios mit i5-8257U an, denn beide Prozessoren haben laut Intel Website die gleiche maximale Taktrate von 3,9 GHz (Wichtig! Wenn nicht möglich, dann möglichst nah) und gehören der gleichen Prozessor Generation an. Wir merken uns "MacBookPro15,4" und/oder die passende Board-ID von zB hier.

Pike R. Alpha hat Teile der Frequency-Vectors entschlüsselt und ein Skript namens freqVectorsEdit geschrieben. Dieses Script wollen wir jetzt erstmal für den Anfang benutzen, da es uns hilfreiche Informationen gibt und erste sinnvolle Anpassungen vornimmt, auf die ich jetzt nicht explizit eingehen will.

Also, Script herunterladen, ggf ausführbar machen (chmod, chown) und auf ein geöffnetes Terminal-Fenster ziehen. Dahinter setzen wir noch die Debug Anweisung, sodass es so aussieht: `./freqVectorsEdit.sh -d 1` und führen den Befehl aus.

Das Script analysiert jetzt alle "Mac-... .plists" aus dem X86PlatformPlugin und stellt uns verschiedene Informationen bereit, Mac-Modelle mit passender maximal Frequenz werden automatisch blau markiert. Warum ist das wichtig?

In den Profilen aus dem X86PlatformPlugin sind häufig mehrere FrequencyVectors hinterlegt. macOS wählt die entsprechende Spalte je nach Maximalfrequenz der CPU aus. Beispiel /System/Library/Extensions/IOPlatformPluginFamily.kext/Contents/PlugIns/X86PlatformPlugin.kext/Contents/R
53FDB3D8DB8CA971.plist



Hat unsere CPU einen MaxClock von 3900mHz wird Eintrag 0 unter FrequencyVectors benutzt, bei 4500mHz der andere.

Mit den gegebenen Daten haben wir jetzt also das korrekte CPU SMBios ausgewählt. Jetzt machen wir folgendes:

1. SIP deaktivieren (per config.plist Eintrag oder Terminal) -> Neustarten
2. Aus /System/Library/Extensions/IOPlatformPluginFamily.kext/Contents/PlugIns/X86PlatformPlugin.kext/Cont folgende Plists finden und als Backup auf dem Schreibtisch sichern: 1. die Plist unserer

aktuell gesetzten SMBios Board ID; 2. die Plist unserer für die CPU gewollten Board ID (unter Catalina+ mit KextUpdater -> Werkzeuge die Systempartition als read/write mounten)
3. Danach freqVectorsEdit im Terminal ausführen (siehe oben) und die Plist die wir für die CPU wollen per Nummer auswählen (entsprechend unseren Recherchen)
4. freqVectorsEdit patcht jetzt die entsprechende Plist im X86PlatformPlugin und überschreibt die Alte (welche wir gebackupt haben)
5. Danach unsere Plist aus /System/Library/Extensions/IOPlatformPluginFamily.kext/Contents/PlugIns/X86PlatformPlugin.kext/Cont kopieren und sichern. Wir wollen genau das Produkt von freqVectorsEdit, wir wollen aber nicht, dass das X86PlatformPlugin modifiziert wurde
6. Also kopieren wir unsere gebackupte Plist von Schritt 1 wieder zurück an ihren alten Ort, sodass dort alles wie eh und jeh ist
7. Mit KextUpdater unter Werkzeuge Kextcache neu aufbauen und Rechte reparieren

Wir haben jetzt also eine X86PlatformPlugin Plist – von freqVectorsEdit gepatcht – in einem eigenen Ordner vorliegen. Weiter gehts.

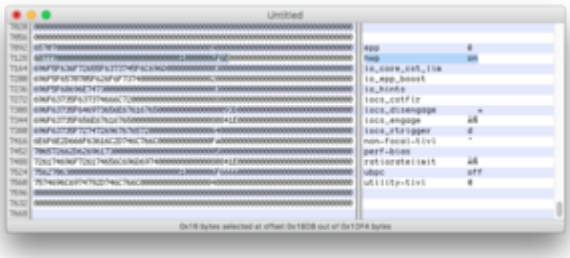
Wir öffnen diese Plist mit einem Plist Editor und beachten 2 Dinge:

1. Unter Frequencies steht ein Eintrag mit genau der Maximalfrequenz unserer installierten CPU. Wenn nicht, passen wir den nähesten Eintrag auf unseren MaxClock an und merken uns den Index.
2. Wer gerne 2 (statt einem) Slider in Systemeinstellungen -> Energie Sparen haben will, setzt ganz unten `UnifiedSleepSliderPref` auf `false`

Jetzt suchen wir unter FrequencyVectors den Eintrag raus, der unserem Index/Takt bei Frequencies entspricht. Der i5-8265U MaxClock ist 3900 und das steht auf Index 0 (siehe Bild oben), also wähle ich die Daten bei FrequencyVectors Index 0 aus.

Wir kopieren die FrequencyVectors-Daten in einen Hex Editor.

1. Wenn wir eine CPU größer, gleich Skylake haben, unterstützt sie wahrscheinlich HWP (Hardware P-States) (am besten nochmal für die CPU googeln!) und deswegen ist es wichtig, dass unsere FrequencyVectors dies auch tun. Am Ende der Hex Daten können wir evtl folgendes sehen:



(Mac-53FDB3D8DB8CA971) "hwp on"

und das zeigt uns, dass HWP supported ist. Ist der Eintrag nicht vorhanden und wir benutzen eine HWP CPU, müssen wir uns eine neue Plist suchen! In Hex lautet die Folge: `687770` [...] `010000006F6E`, also einfach danach suchen.

2. Alle FrequencyVectors beginnen mit `02000000`, danach folgt beispielsweise `0C000000`. Diese 4 byte (8 Zahlen) sind der LFM (Low Frequency Mode), welchen wir anpassen müssen. Also googeln wir den LFM oder TDP-down für unsere CPU. In unserer Plist entspricht `0C 00 00 00` = `0C` (hex) was in dezimal 12, also 1200MHz ist. Der i5-8265U hat einen 800MHz LFM, also tragen wir 8 (dec) -> `08000000` (hex) anstatt `0C000000` ein.

Manche CPUs vertragen sogar noch weniger, beispielsweise hat sich bei meinem i5-8265U herausgestellt, dass selbst 500MHz möglich sind. Also evtl ausprobieren.

3. Wir kommen zum EPP, dem Energy Performance Preference (im Screenshot oben auch sichtbar). Dies ist ein Wert für HWP CPUs, der die Aggressivität des Taktverhaltens steuert und wie eine Präferenz festgelegt werden kann. Der EPP liegt zwischen 0x00 (0) und 0xFF (255) und entspricht grob folgenden Richtlinien:

Richtlinie	**EPP**	**EPB (siehe 4.)**
performance	0x00 - 0x40	0 - 4
balance-performance	0x40 - 0x80	4 - 8
normal, default	0x60	6
balance-power	0x80 - 0xC0	8 - 12
power	0xC0 - 0xFF	12 - 15

Wir müssen uns für einen Wert zwischen 0x00 (0) und 0xFF (255) entscheiden, je niedriger desto schneller taktet die CPU nach oben. Wenn wir ihn zu niedrig setzen kann es sein, dass die CPU später praktisch nie im Idle die LFM Frequenz erreichen wird. Wenn wir den Wert zu hoch setzen kann es sein, dass unsere CPU praktisch garnicht mehr in den Turbo schaltet. Hier ist also evtl später etwas probieren gefragt. Für Desktops eignet sich meist ein Wert zwischen

performance und balance-performance, zB 0x40. Für Laptops empfiehlt sich meist ein Wert im balance-power bis power Bereich, zB 0xC0.

Sobald wir uns entschieden haben suchen wir im Hex Editor die Folge `65707000` (epp) und passen 16 bytes später den Wert an, also zB von 657070000000000000000000000000000000000000**60**00 zu 657070000000000000000000000000000000000000**C0**00 (normal -> stromspar)

4. EPB ist eine Art Fallback zu EPP, wenn EPP nicht unterstützt ist. Also sollten wir EPB (IA32_ENERGY_PERF_BIAS) auch noch passend zum gewählten EPP setzen (siehe Tabelle). Der Wert findet sich 16 bytes hinter `perf-bias`, eine mögliche Anpassung wäre zB 706572662D6269617300000000000000000000000**05**00 zu 706572662D6269617300000000000000000000000**12**00.

Unsere angepassten FrequencyVectors können wir jetzt aus dem Hex Editor wieder in das entsprechende Feld des PlistEditors kopieren und unsere Custom X86Platform Plist speichern.

Jetzt laden wir als letztes CPU-Friend herunter und ziehen das Tool ResourceConverter.sh ins Terminal. Der Befehl lautet `./ResourceConverter.sh -a "Mac-... .plist"` wobei wir anstatt "Mac-... .plist" unsere Custom X86Platform Plist ins Terminal ziehen. Es wird eine ssdt im Ausgangsordner erstellt.

Wir öffnen MaciASL, lassen uns die DSDT anzeigen und suchen nach CPU0 oder PR00 um die ACPI ID unserer CPU herauszufinden. Sobald wir etwas finden wie

Code

```
1. Scope (_PR)
2. {
3. Processor (CPU0, 0x01, 0x00001810, 0x06) {}
```

wissen wir, dass unsere CPU zB bei _PR_.CPU0 liegt.

Als letztes öffnen wir die von ResourceConverter erstellt ssdt_data.dsl mit MaciASL und stellen sicher, dass als External und Scope jeweils die gerade gefundene ACPI ID eingetragen ist. Danach Save As -> File Format: ACPI Machine... aml, speichern. Die SSDT wird normal über die EFI eingebunden (nicht parallel mit SSDT-PLUG nutzen!) und CPUFriend normal als Kext eingebunden.

Neustart -> testen -> hoffentlich freue! Voraussetzung ist, dass das X86PlatformPlugin

jetzt geladen wird. Dies lässt sich prüfen mit `kextstat | grep -R X86` im Terminal.

Und für die ganz Harten gibts dann auch noch VoltageShift, aber das wird jetzt wirklich zu viel.

Älterer Lesestoff: SMBIOS iMac17,1 / Skylake i76700K und Powermanagement - wie funktioniert es richtig?

Guide wurde eingefügt. Ehemaliger Beginn des Threads ohne Guide:

> Zitat von karacho
>
> *Du kannst den PluginType in der SSDT-PLUG.aml auf 0 setzen*

Darf ich fragen wieso?

Beitrag von „karacho“ vom 26. April 2020, 15:00

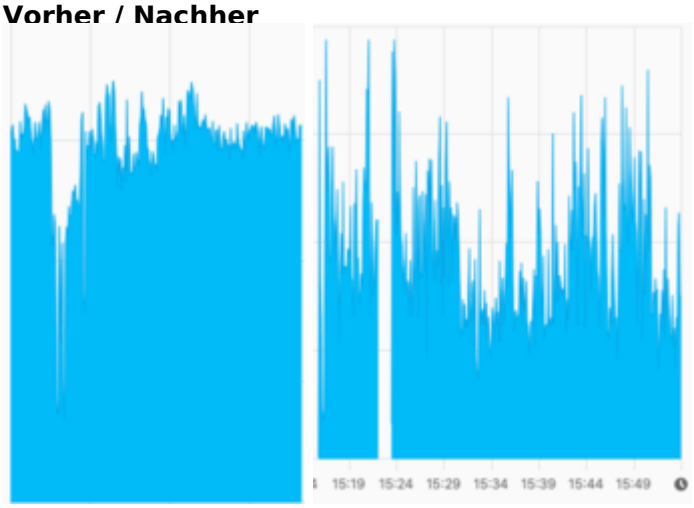
Weil sein Powermanagement mit 1 nicht funktionierte. -> OpenCore Sammelthread (Hilfe und Diskussion)

Beitrag von „revunix“ vom 26. April 2020, 15:26

> Zitat von karacho
>
> Nicht deaktivieren, den PluginType in der SSDT-PLUG.aml auf 0 setzen

Hab ich jetzt auch mal gemacht. Meine CPU Tacktet mit PluginType = 1 immer über 3GHz und kommt nie unter 2GHz. Warum auch immer.

Vorher / Nachher



Beitrag von „JimSalabim“ vom 26. April 2020, 16:02

Ja, bei mir ist das auch so. Mich würde mal sehr interessieren, wie sich das an nem echten iMac verhält, wenn Power Nap etc. aktiviert ist.

Beitrag von „kuckkuck“ vom 26. April 2020, 16:23

karacho Verstehe, aber ich würde nicht unbedingt dazu raten das ACPI_SMC_PlatformPlugin zu benutzen. Ist vielleicht eine Notlösung, aber im Regelfall sollte das per X86PlatformPlugin überall irgendwie möglich sein und ist imo die bessere Variante

---

Beitrag von „CMMChris“ vom 26. April 2020, 17:33

kuckkuck Bringt keine Nachteile. Teils ist sogar das Gegenteil der Fall. Bei IGPU Nutzung sollte man aber tatsächlich das X86PlatformPlugin nutzen wegen AGPM.

Beitrag von „kuckkuck“ vom 27. April 2020, 00:26

dutch64 Per CPUFriend lässt sich die Funktion des X86PlatformPlugins anpassen, Voraussetzung ist dass das X86PlatformPlugin lädt.

> Zitat von CMMChris
>
> Bringt keine Nachteile. Teils ist sogar das Gegenteil der Fall.

Das musst du mir jetzt erklären, sowohl warum das keine Nachteile haben sollte und wo die Vorteile liegen. Ich sehe bei Xnu Power Management wenige Nachteile, xcpm ist tief in den Kernel Integriert und hat weitreichende Auswirkungen und Dependenzen. Das X86PlatformPlugin sorgt (von seiner Funktion her) für korrektes Speedstepping, aber hat Einfluss auf viel mehr als nur das, wie zB System Capabilities, Sleep, Wake, DVFS, HWP, (AGPM), (Temp-/FanManagement), Shutdown-, Reboot-, Reset- und PowerFailure Monitoring, ME Events, Energiehaushalt/PowerProfiles, etc. Ich kenne garnicht alle Abhängigkeiten, aber in meinen Augen spricht alles dafür, dass das X86PlatformPlugin für neue CPUs laden sollte, um korrektes PowerManagement im Allgemeinen zu ermöglichen, auch spricht imo oberflächlich gesehen die häufig deutlich bessere Bootzeit dafür. Würde mich interessieren wie du zu deinem Schluss kommst

---

Beitrag von „karacho“ vom 27. April 2020, 07:29

> Zitat von kuckkuck
>
> ACPI_SMC_PlatformPlugin zu benutzen.

Sieh es positiv. Dafür hat er ja jetzt bei 'Energie sparen' einen Slider mehr...

Beitrag von „kuckkuck“ vom 27. April 2020, 15:35

> Zitat von karacho
>
> Dafür hat er ja jetzt bei 'Energie sparen' einen Slider mehr...

Und selbst den kriegt man wenn man ihn unbedingt haben will per X86PlatformPlugin 

Beitrag von „NoBody_0“ vom 27. April 2020, 15:51

du Eierwerfer kuckkuck wie denn?!

Beitrag von „CMMChris“ vom 27. April 2020, 17:05

> Zitat von kuckkuck
>
> Das musst du mir jetzt erklären, sowohl warum das keine Nachteile haben sollte und wo die Vorteile liegen.

Zunächst mal grundlegend: Natürlich ist das X86PlatformPlugin (also PluginType=1) die zu bevorzugende Variante. ACPI_SMC_PlatformPlugin wird wahrscheinlich auch nicht mehr lange in macOS erhalten bleiben da Vintage Überrest.

Es gibt aber halt durchaus Fälle wo man einen Rechner als schnellen Fix durchaus mit PluginType=0 / ACPI_SMC_PlatformPlugin laufen lassen kann. Ich hatte nun schon ein paar Fälle wo das Speed Stepping einfach nicht richtig wollte und da halfen auch die üblichen verdächtigen Fixes inklusive CPUFriend nichts. Mit ACPI_SMC_PlatformPlugin lief es dann aber einwandfrei.

Ich selbst lasse meinen Hacki fast durchgehend mit dem ACPI_SMC_PlatformPlugin laufen und

das hat folgende Gründe:

- CPU geht im Idle auf bis zu 700MHz runter und verbleibt auch länger im unteren Taktbereich als mit PluginType=1 + CPUFriend was mir im Idle 1 bis 2 Grad weniger an der CPU bringt und den Idle Verbrauch vom System nochmal ein wenig verringert
- Die Takt Kurve im Intel Power Gadget sieht wesentlich schöner (gleichmäßiger) aus wenn man sie z.B. bei einem, klassischen Office Workload beobachtet
- CPU Benchmarks werden davon im übrigen nicht negativ beeinflusst, teils ist sogar das Gegenteil der Fall
- Gelegentlich auftretendes Coil Whine vom Mainboard bei Anwendungen mit schnellem Lastwechsel auf der CPU verschwindet
- AGPM wird nicht geladen was zumindest bei mir nachweislich die Performance in manchen Games stört. Ein paar Games laufen ohne AGPM deutlich runder egal ob nun mit PluginType=1 oder PluginType=0. Auf den Stromverbrauch der GPU hat das ganze interessanterweise keinerlei Einfluss.
- PowerNap nervt! Seit Catalina kann man den Rotz deaktivieren wie man will, der Rechner wacht trotzdem sporadisch nachts mit nem RTC Alarm auf und reißt mich aus dem Schlaf. Passiert im übrigen auch an meinen echten Macs. Auf PluginType=0 hat der Spuk ein Ende und der Rechner bleibt im Standby wenn ich ihn schlafen schicke.

Für mich ist das alles jedenfalls Grund genug das ACPI_SMC_PlatformPlugin zu nutzen wenn es schonmal da ist. Mag ja sein, dass da noch irgendwelche anderen Sachen außer PowerNap und AGPM in macOS außer Funktion gesetzt werden. Negative Auswirkungen habe ich dadurch jedoch noch nicht bemerkt. Insofern ist mir das auch völlig wurscht.

Beitrag von „JimSalabim“ vom 27. April 2020, 17:19

apfelnico OpenUsbDxe.efi hatte ich eh schon drin. Ich hab jetzt mal noch XhciDxe.efi mit rein, macht aber keinen Unterschied. Dafür denkt OpenCore jetzt, wenn ich (eben mehrfach) Cmd-Alt-P-R drücke, ich würde einfach nur R drücken und startet dann von der Recovery 😆

Naja. Kann man wohl nix machen.

CMMChris Ich hab bei mir jetzt mal CPUFriend mit rein und mir mit dem CPUFriendFriend-Skript eine ssdt_data.aml erstellen lassen, die (da sie Plugin-Type One eh schon enthält) jetzt bei mir statt der SSDT-PLUG drin ist.

`External (_PR_.PR00, DeviceObj)` hab ich noch durch `External (_SB_.PR00, ProcessorObj)`

ersetzt. Außerdem noch `Scope (\_PR.PR00)` durch `Scope (\_SB.PR00)`. Das war in der selbst generierten SSDT-PLUG nämlich auch so.

Im Idle liege ich in der Regel nach wie vor bei etwas über 2 GHz, aber dafür liegt das Minimum nun bei 800 MHz (so wie ich es in eingestellt habe).

Wenn ich darauf jetzt keinen Bock habe und auch Power Nap nicht haben will: Hab ich es jetzt richtig verstanden, dass das ACPI_SMC_PlatformPlugin automatisch genutzt wird, wenn ich nun in der ssdt_data.aml Plugin-Type auf Zero setze?

Beitrag von „CMMChris“ vom 27. April 2020, 18:25

> Zitat von JimSalabim
>
> Hab ich es jetzt richtig verstanden, dass das ACPI_SMC_PlatformPlugin automatisch genutzt wird, wenn ich nun in der ssdt_data.aml Plugin-Type auf Zero setze?

Ja aber man kann sich die SSDT auch einfach sparen weil das ACPI_SMC_PlatformPlugin der Default ist wenn kein PluginType übergeben wird. Das ACPI_SMC_PlatformPlugin nutzt die vom BIOS generierten CPU Power Management Daten (CPU SSDTs).

Beitrag von „revunix“ vom 27. April 2020, 19:03

CMMChris Wie sieht das eigentlich aus mit CPUFriend und PluginType 1 wenn man Daten von einem anderen Mac verwendet die z.B die CPU verbaut hat die ich habe ?!

Mal angenommen ich habe das SMBIOS: iMacPro1,1 und verwende mit CPUFriend die Daten vom iMac19,1 denn der hat den i5 8600 verbaut. Macht das sinn bzw. funktioniert das so überhaupt?

Beitrag von „JimSalabim“ vom 27. April 2020, 19:48

revunix Die Daten für CPUFriend kannst du dir einfach selber generieren:

https://github.com/corpnewt/CPUFriendFriend

"*This Py script will inspect the frequency vectors of the X86PlatformPlugin plist matching your SMBIOS configuration and leverage acidanthera's CPUFriend ResourceConverter to help you optimize your power management configuration.*"

Du musst nur den LFM-Wert für deine CPU im Datenblatt nachschauen (bei mir beispielsweise 800 MHz). Danach ist es am einfachsten, die generierte CPUFriendDataProvider.kext zusätzlich zu CPUFriend.kext zu verwenden. Und das war’s dann auch schon.

Beitrag von „kuckkuck“ vom 27. April 2020, 20:50

Könnte ein Mod die ganze PowerManagement Diskussion mal in einen extra Thread schieben? Hat ja nicht nur was mit OC zu tun

CMMChris Ich stimme dir definitiv zu, wenn es um einen "schnellen Fix" geht kann man das ACPI_SMC Plugin definitiv mal aktivieren, meist hat das ja direkt die Auswirkung, dass zumindest Speedstepping erstmal funktioniert. Aber nichts anderes habe ich ja geschrieben

> Zitat von kuckkuck
>
> Ist vielleicht eine Notlösung,

Da das X86PlatformPlugin aber nunmal die zu bevorzugende Variante ist, finde ich sollte man vorsichtig sein, wenn man Neuankömmlingen zu plugin-type=0 rät und mitgibt, dass alles in Butter ist und keinerlei negative Effekte auftreten können. Das ist alles um was es mir ging.

Als Gegenbeispiel kann ich von meinem neu eingerichteten Laptop berichten, hier läuft Speedstep CPU-gesteuert über das X86PlatformPlugin und HWP was entsprechende Performance Verbesserungen mit sich bringt, hat einen LFM von variabel 400-800Mhz wenn ich das will und geht nur bei lang andauerndem Workload überhaupt in den Turbo. Die Energieeffizienz ist dementsprechend hoch, die Frequenzen sind aktuell von 500Mhz bis 3900Mhz praktisch im ganzen Spektrum verfügbar und zusätzlich habe ich auch noch 2 Slider in den Systemeinstellungen -> Energie Sparen. PowerNap tut soweit trotz Catalina auch das was es soll, und zwar deaktiviert sein, aber das werde ich mir nach deinem Tipp nochmal bisschen genauer anschauen.

Ich kann gerne mal in einem extra Thread dazu etwas genauer erläutern, wie man das X86PlatformPlugin möglichst auf die vorhandene HW optimiert, vielleicht waren da ja einfach nur ein paar Einstellungen blöd gesetzt bei manch einem. Was AGPM angeht frage ich mich, ob es nicht schlauer wäre den Treiber einfach anderweitig zu deaktivieren, da kenne ich aber die aktuelle Lage nicht so ganz.

muster48 Per CPUFriend kann man eine entsprechend angepasste X86PlatformPlugin Plist übergeben. Mehr dazu kann ich ja mal in einem anderen Thread ausführen.

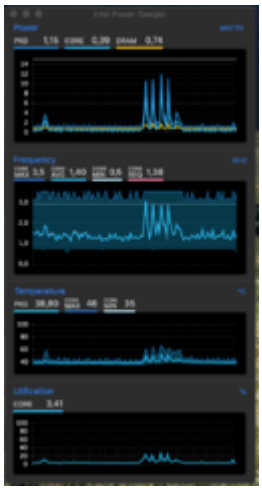
Beitrag von „anonymous_writer“ vom 27. April 2020, 21:05

Wir hatten hier über das Thema diskutiert.

SMBIOS iMac17,1 / Skylake i76700K und Powermanagement - wie funktioniert es richtig?

Bin auch immer gerne bereit hier mit zu diskutieren da es ein sehr interessantes Thema ist. Mein I7-7500U ist da auch sehr heikel und man kann da richtig Spielen mit den Einstellungen.

Hier aktuelle Ergebnis mit im Moment verwendeter CPU ssdt-i7-7500U + CPUFriend.kext + CPUFriendDataProvider.kext.



Eventuell kann einer der Mods die Einträge verscheiben und das Thema etwas anpassen.

Beitrag von „NoBody_0“ vom 27. April 2020, 21:51

kuckkuck

dann werde ich mich auf deinen Thread sehr freuen, sogar jede hier im Forum

---

Beitrag von „schmalen“ vom 27. April 2020, 21:54

pstr

> Zitat von pstr
>
> Auch OC 0.58 mit OpenCanopy bootet seit kurzem nach Timeout durch .

Habe auch 0.5.8 und timeout auf 0 bei mir bootet der hacki nicht durch

CMMChris

> Zitat von CMMChris
>
> Ich selbst lasse meinen Hacki fast durchgehend mit dem ACPI_SMC_PlatformPlugin laufen und das hat folgende Gründe:
>
> CPU geht im Idle auf bis zu 700MHz runter und verbleibt auch länger im unteren Taktbereich als mit PluginType=1 + CPUFriend was mir im Idle 1 bis 2 Grad weniger an der CPU bringt und den Idle Verbrauch vom System nochmal ein wenig verringert

Trotz PluginType=0 dümpelt meiner nicht mal unter 1.2 Ghz, da müsste doch was anderes im Busch sein?

Beitrag von „kuckkuck“ vom 28. April 2020, 17:53

@Mods Kann jemand diese Beiträge mal bitte in einen eigenen Thread packen?

3373 - 3377, 3380, 3382, 3384, 3389, 3391, 3398 - 3399, 3402 - 3406, 3409 - 3410

Wäre sehr lieb, dann können wir dort weiterforschen

Beitrag von „al6042“ vom 28. April 2020, 18:42

kuckkuck

Das müsste es gewesen sein... hoffe ich...

Beitrag von „luxus13“ vom 28. April 2020, 19:51

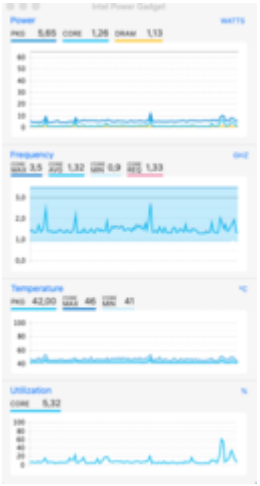
Hallo CMMChris!

Kannst Du mir einen Link zukommen lassen damit ich die "ACPI_SMC_PlatformPlugin" anpassen oder austauschen kann? Möchte lesen und mich informieren.

Habe das selbe verhalten wie die Forumsmitglieder schreiben, egal wie ich es anstelle meine CPU taktet zwischen 2GHz und 3,xGHz.(mit SSDT-Plug.aml oder mit SSDT-DATA.aml (selber erstellt) oder mit beiden zusammen)

Edit:

Derzeit verwende ich gar keine aml oder kext und die CPU taktet bis auf ca. 1,3GHz



LG

Beitrag von „anonymous_writer“ vom 28. April 2020, 20:22

Hallo luxus13 ,

Das ist doch perfekt für einen Desktop. Daran würde ich nichts ändern.

Beitrag von „luxus13“ vom 28. April 2020, 20:31

Danke **anonymous writer** für die Info.

werde es so lassen, aber ein wenig der Spieltrieb ob ich unter 1 GHz kommen könnte.

LG

Beitrag von „anonymous_writer“ vom 28. April 2020, 20:36

Wenn dein CPU bei diesem Script mit drin ist bekommst denn damit sehr wahrscheinlich runter.

https://github.com/stevezhengshiqi/one-key-cpufriend

Ich würde es jedoch bei einem Desktop so lassen da hier Power gefragt ist und keine Batterie geschont werden muss.

Beitrag von „luxus13“ vom 28. April 2020, 20:45

Leider nicht möglich, auch gut, Danke für den Tipp.

LG

Hmm, nichts zu verschlimmbessern ....

Beitrag von „kuckkuck“ vom 1. Mai 2020, 22:39

Sooo, ich habe jetzt mal einen ausführlichen Guide geschrieben und in den ersten Post gepackt: CPUFriend Guide und Speedstepping: X86PlatformPlugin vs ACPI_SMC_PlatformPlugin

Ich hoffe es hilft dem ein oder anderen, wenn man etwas tüftelt kriegt man wirklich super Ergebnisse hin. Mein Hackbook taktet hervorragend und hat im Vergleich zum Anfang eine immens bessere Akkulaufzeit. Für Laptops macht es glaube ich wirklich Sinn die Zeit zu investieren. Für alle anderen ist es trotzdem sinnvoll und vielleicht ist ja auch so die ein oder andere interessante allgemeine Information für manch Einen im Guide zu finden

---

Beitrag von „NoBody_0“ vom 2. Mai 2020, 13:46

kuckkuck danke für deine Zeit, die du genommen hast, um das hier zu erklären.

Darf ein paar Fragen stellen

1- SIP lässt sich nicht deaktivieren, ich habe dafür die Werte 67000000, 03000000 und 07030000 in d. config.plist eingetragen... was mache ich falsch!!, dadurch lässt sich auch read/write nicht mit kextupdater sowie mit Hackinttol aktivieren!!

2- Welches Programm nutzt du als Hex Editor, um die Werte zu lesen und auch zu ändern

---

Beitrag von „kuckkuck“ vom 2. Mai 2020, 15:11

Na sicher.

1. Die SIP bzw csr-active-config ist wie so vieles eine Bitmaske. Sprich die einzelnen Bits der 4 Byte Hex-Zahl toggeln bestimmte SIP (Deaktivier-) Funktionen an oder aus. Wir haben es mit einer 4 Byte Hex-Zahl zu tun, 00000000 bedeutet, dass kein einziges (Deaktivierungs-) Bit gesetzt ist -> SIP ist vollständig aktiviert. Die einfachste und zukunftssichere Art die SIP komplett zu deaktivieren ist einfach alle Bits zu aktivieren. Das entspricht also FFFFFFFF -> SIP ist vollständig deaktiviert, oder konvertiert zu Dec: 4294967295.

csr-active-config ist ein NVRam Eintrag, NVRam Einträge werden nicht überschrieben, wenn sie bereits vorhanden sind. Änderungen an der csr-active-config werden also erst übernommen, wenn der alte Wert gelöscht wurde. Dies kann man durch einen NVRam reset machen (zB nvram -c im Terminal), oder der Bootloader übernimmt dies. Unter OpenCore gibt es die NVRam -> Block Sektion, welche NVRam Einträge automatisch "löscht", will man also, dass Änderungen an einem bestimmten NVRam Eintrag immer direkt übernommen werden, setzt man diesen NVRam Eintrag (zB csr-active-config) in die Block-Sektion unter der entsprechenden GUID (hier 7C436110-AB2A-4BBB-A880-FE41995C9F82).

2. In den Screenshots ist Hex Fiend und PlistEdit Pro zu sehen.

Beitrag von „NoBody_0“ vom 2. Mai 2020, 15:17

vielen Dank für die Rückmeldung...

> Zitat von kuckkuck
>
> sodass es so aussieht: ./freqVectorEdit.sh -d 1 und führen den Befehl aus.

es fehlt s bei freqVectorsEdit.sh

Beitrag von „kuckkuck“ vom 2. Mai 2020, 17:16

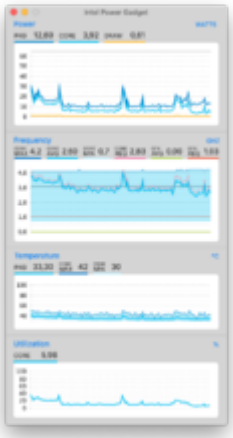
Danke, wenn dir noch was ins Auge fällt immer her damit

Hat das Einstellen der SIP geklappt?

Beitrag von „revunix“ vom 2. Mai 2020, 18:10

> Zitat von anonymous writer
>
> Wenn dein CPU bei diesem Script mit drin ist bekommst denn damit sehr wahrscheinlich runter.
>
> https://github.com/stevezhengshiqi/one-key-cpufriend

Nettes Script. Verwende allerdings den `iMac19,1 - Mac-AA95B1DDAB278B95` und hab den einfach mal hinzugefügt. Hm aber irgendwie ... hm hat sich da nichts getan.



Beitrag von „luxus13“ vom 2. Mai 2020, 19:43

Hallo,

Hast Du die SSDT-PLUG.aml in Deiner EFI aktiv?

LG

Beitrag von „revunix“ vom 2. Mai 2020, 20:04

Ja die ist wieder aktiv.

Beitrag von „luxus13“ vom 2. Mai 2020, 20:20

Bei mir ist es so, sobald ich die SSDT-PLUG.aml aktiv habe bewegt sich die Leerlaufrequenz so wie bei Dir, nehme ich sie raus fällt die Leerlauffrequenz auf 1,3GHz und alle funktioniert wie es soll.

LG

Beitrag von „kuckkuck“ vom 2. Mai 2020, 20:25

Ich denke im ersten Post sollten genug Ansatzpunkte zu finden sein, um das Verhalten weiter zu verstehen oder zu beheben. Ein i5 8600 ist da ja jetzt auch nicht so der außergewöhnlichste Prozessor...

Beitrag von „revunix“ vom 2. Mai 2020, 20:49

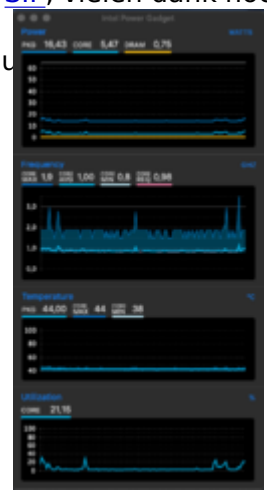
> Zitat von luxus13
>
> Bei mir ist es so, sobald ich die SSDT-PLUG.aml aktiv habe bewegt sich die Leerlaufrequenz so wie bei Dir, nehme ich sie raus fällt die Leerlauffrequenz auf 1,3GHz und alle funktioniert wie es soll.

Habe ich schon mehrfach ausprobiert, ist aber nicht gut denn dann laggen Videos wenn ich im Chrome den Tab Wechsel... läuft also nicht so doll.

Beitrag von „NoBody_0“ vom 2. Mai 2020, 20:53

ja es hat geklappt mit SIP, vielen dank noch einmal

ir au



weiß war ein Film am laufen und schwarz war Leerlauf

und mit dem Slider auch

Beitrag von „luxus13“ vom 2. Mai 2020, 21:32

welches SMBIOS hast Du gewählt für Deinen i5-7400

LG

Beitrag von „NoBody_0“ vom 2. Mai 2020, 21:35

wenn du bei mir meinst!

iMacPro1,1 mit OpenCore bootloader

Beitrag von „kuckkuck“ vom 2. Mai 2020, 21:54

Frage ist wahrscheinlich eher welche Board ID du passend zu deiner CPU gewählt hast um passende FrequencyVectors zu erhalten.

Beitrag von „NoBody_0“ vom 2. Mai 2020, 22:00

das dachte ich auch aber nun steht SMBIOS!!

bei mir läuft Catalina 15.4 und habe das hier Mac-87DCB00F4AD77EEA.plist genommen

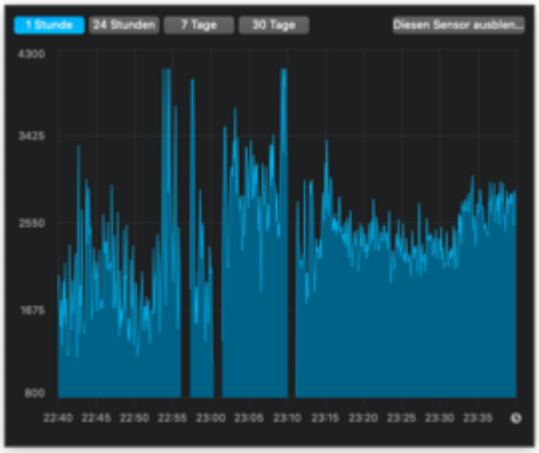
---

Beitrag von „revunix“ vom 2. Mai 2020, 23:41

> Zitat von Un!x
>
> Nettes Script. Verwende allerdings den iMac19,1 - Mac-AA95B1DDAB278B95 und hab den einfach mal hinzugefügt. Hm aber irgendwie ... hm hat sich da nichts getan.

Hab es nun mal nach der Anleitung gemacht ... scheint nun funktioniert zu haben.

Perfekt.



Beitrag von „kuckkuck“ vom 3. Mai 2020, 00:33

Wunderbar, auf was hast du EPP gesetzt?

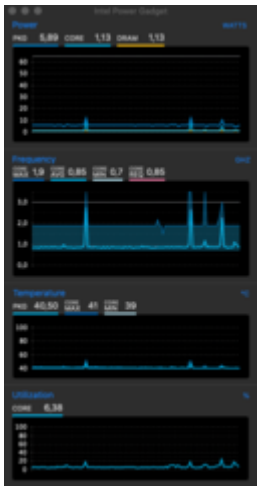
Beitrag von „luxus13“ vom 3. Mai 2020, 00:41

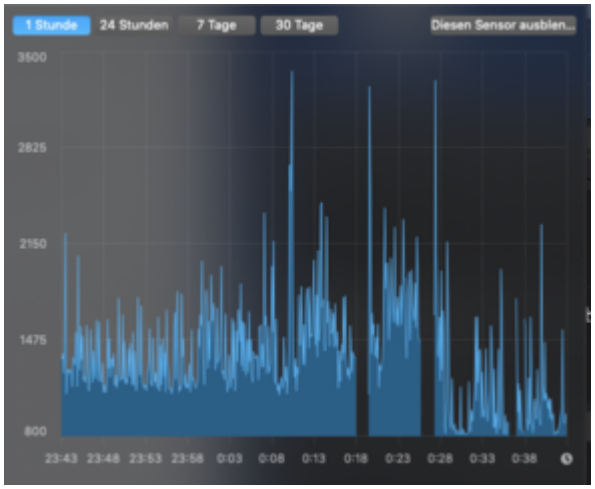
Danke an kuckkuck und muster48,

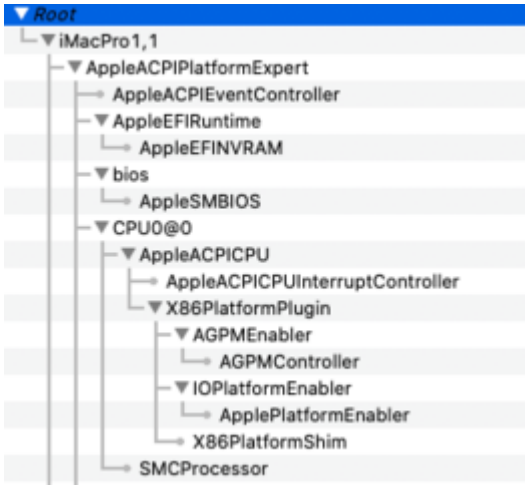
habe mich endlich einmal durchgerungen auch diese "Baustelle" zu erledigen.

Einfach Top.

LG







Beitrag von „revunix“ vom 3. Mai 2020, 00:54

> Zitat von kuckkuck
>
> Wunderbar, auf was hast du EPP gesetzt?

Ich hab es erstmal exakt wie in der Anleitung gemacht: 0xC0.

Beitrag von „kuckkuck“ vom 3. Mai 2020, 01:08

Ich habe mal noch einen kurzen Kommentar zur Anleitung hinzugefügt:

> Zitat von kuckkuck
>
> Für Desktops eignet sich meist ein Wert zwischen performance und balance-performance, zB 0x40. Für Laptops empfiehlt sich meist ein Wert im balance-power bis power Bereich, zB 0xC0.

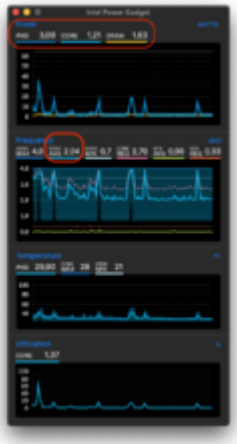
Bei Desktops steht meist die Leistung im Vordergrund und Stromverbrauch ist aufgrund der direkten Stromversorgung nicht so wichtig --> EPP ~0x60.

Für Laptops ist hingegen Stromverbrauch und Akkulaufzeit häufig die höchste Priorität, deswegen eignet sich ein höherer EPP wie zB 0xC0.

Beitrag von „cobanramo“ vom 3. Mai 2020, 11:26

Vielleicht sollte man auch erwähnen das man gerade bei den Laptop's & Frequenzvectoring auch auf die Power PKG achten sollte. Tiefste Frequenz heisst nicht immer das es auch weniger konsumiert. Man kann gerade bei Mobile Systeme das ganze auch völlig vermurksen.:-)

Man kann auch mit "höheren" Cpu Frequenzen "reaktionsfreudigere" Systeme die aber wenig "konsumieren" erreichen.



Gruss Coban

Beitrag von „kuckkuck“ vom 4. Mai 2020, 13:04

Du meinst bezüglich EPP? Was sind deine Erfahrungen?

Beitrag von „luxus13“ vom 6. Mai 2020, 12:13

Hallo

Möchte für meinen iMac Pro II (siehe Sig.)

mit dem i5-6400 Prozessor

ebenfalls die Taktrate anpassen hat jemand einen Vorschlag welchen ich nehmen soll in der Tabelle von 1 bis 53?

LG

Beitrag von „JimSalabim“ vom 6. Mai 2020, 14:05

Ich habe es zuletzt so gemacht, dass ich mir mit dem CPUFriendFriend-Skript (https://github.com/corpnewt/CPUFriendFriend) eine CPUFriendDataProvider.kext habe erstellen lassen. Mein SMBIOS (iMac19,1) und die vom Skript erkannte Board-ID Mac-AA95B1DDAB278B95 entsprechen auch dem Modell, das ich im freqVectorsEdit-Skript manuell wählen würde. Die Minimalfrequenz stelle ich auf 800 Hz und EPP bisher immer auf 0x00 für meinen i9 9900K. Die SSDT-PLUG hab ich drin gelassen. Hat die Methode aus dem ersten Post hier in meinem Fall nun noch Vorteile oder fahre ich so schon ganz gut?

Beitrag von „revunix“ vom 6. Mai 2020, 15:15

JimSalabim Die Methoden sind identisch. Hier wird es allerdings detailliert erklärt, was bei CPUFriendFriend nicht der Fall ist. Ich selbst habe beide Methoden ausprobiert und das gleiche gewünschte Ergebnis erzielt.

Beitrag von „kuckkuck“ vom 6. Mai 2020, 17:15

Unix Das haben wir doch schon im Discord besprochen?

Nein, die Methoden sind nicht identisch. In bestimmten Fällen (wenn zB das für PM bevorzugte SMBios das bereits gewählte SMBios ist) kann das Ergebnis das gleiche sein.

Abgesehen davon, dass CPUFriend Friend nicht auf die Wichtigkeit der Wahl eines korrekten CPU-SMBios eingeht und somit beispielsweise die komplette HWP Funktionalität wegfallen

kann, sind auch die letztendlichen Ergebnisse unterschiedlich. Eine X86PlatformPlugin Mac-...plist sieht nach den Bearbeitungen aus der Anleitung im ersten Post anders aus, als nach der Benutzung von CPUFriend Friend. Dazu zählen: Inject der FrequencyVectors eines anderen SMBios (eigentlicher Sinn von CPUFriend, CPUFriend sollte nicht nur zur Anpassung des LFM genutzt werden), Anpassung von EPB, Matching durch "Frequencies", Verändungen durch FreqVectors, Anpassung von zB SleepSlider und natürlich die Wahl von sinnvollen/präzisen Werten und Einstellungen.

CPUFriend Friend ist in manchen Fällen eine gute Möglichkeit den LFM oder EPP schnell zu patchen. JimSalabim du fährst wahrscheinlich schon ganz gut, wenn du mit dem Taktverhalten deiner CPU zufrieden bist.

Beitrag von „JimSalabim“ vom 6. Mai 2020, 17:22

Super, danke kuckkuck für die Erklärung! Das leuchtet ein und ist sehr erhellend

---

Beitrag von „revunix“ vom 6. Mai 2020, 21:49

> Zitat von kuckkuck
>
> Unix Das haben wir doch schon im Discord besprochen?

Ich habe beide SSDTs länger getestet und konnte keinen unterschied feststellen. Von daher, sage ich einfach das es gleich ist

Die Werte sind auch identisch. Teste es mal selbst, falls du es noch nicht gemacht hast.

Beitrag von „JimSalabim“ vom 6. Mai 2020, 21:57

Dann wird wohl (wie auch bei mir) das hier zutreffen:

> Zitat von kuckkuck
>
> In bestimmten Fällen (wenn zB das für PM bevorzugte SMBios das bereits gewählte SMBios ist) kann das Ergebnis das gleiche sein.

Das Ergebnis mit dem CPUFriendFriend-Skript wäre (mangels manueller Auswahlmöglichkeit des SMBIOS-Settings) für Nutzer des iMacPro1,1-SMBIOS bei der gleichen Hardware, die ich verwende, dann mit hoher Wahrscheinlichkeit ja nicht dasselbe und möglicherweise sogar komplett ungeeignet, weil die CPU ja eben ganz und gar nicht übereinstimmen würde (Xeon vs i9).

Beitrag von „kuckkuck“ vom 7. Mai 2020, 16:57

Unix Kann ja sein (auch wenn identisch schwer zu ermitteln ist, wenn es zB um CPU Taktverhalten geht), ging nur darum, dass nur weil es in bestimmten Fällen ein praktisch gleiches Ergebnis hervorrufen kann, kann der Guide nicht allgemein mit CPU Friend Friend gleichgesetzt werden.

Wollte nur verhindern, dass plötzlich alle anfangen CPUFriendFriend zu benutzen und sich dann hier beschweren, dass es nicht funktioniert

---

Beitrag von „revunix“ vom 7. Mai 2020, 17:56

kuckkuck Liegt wahrscheinlich an bestimmten SMBIOS-Typen.

Beitrag von „dutch64“ vom 21. Mai 2020, 16:09

Hi,

kuckkuck , vielen Dank für diesen Beitrag und die Arbeit.

Für mich war es ein langer und mitunter steiniger Weg aber jetzt bin ich mit dem Ergebnis mehr als zu Frieden und gelernt habe ich auch wieder was.

Danke!

Beitrag von „kuckkuck“ vom 24. Mai 2020, 17:53

Freut mich zu hören, dass es funktioniert hat! Und das Ergebnis sieht auf den schnellen Blick sehr gut aus, gratuliere

---

Beitrag von „dutch64“ vom 24. Mai 2020, 18:57

Hi,

inzwischen ist es mir auch gelungen auch den IGPU Takt zu senken.

Um eine KP nach Sleep mit den Einstellungen zu unterdrücken musste ich bei OC,

bei Kernel - Quirks- PowerTimeoutKernelPanic aktivieren.

Läuft, bin sehr zufrieden.

Beitrag von „kuckkuck“ vom 5. Juni 2020, 15:21

Was hast du für die iGPU noch unternommen?

Und die KP nach dem Sleep kam aber erst nachdem du das X86Platform Plugin aktiviert hattest? Ist ja interessant, kann sein, dass setPowerState da auch irgendwas mit zu tun hat. Mit PowerTimeoutKernelPanic hast du ja aber für Catalina auf jeden Fall eine Lösung

---

Beitrag von „dutch64“ vom 5. Juni 2020, 16:49

Hi,

am besten lässt sich mein Vorgehen mit „try and error“ beschreiben.

Wie das halt so ist mit, unzureichendem Halbwissen.

Nachdem ich mit Hilfe Deiner Anleitung den CPU Takt senken konnte gab es nach Sleep immer eine KP nach dem aufwachen des Rechners, verantwortlich war dafür meine DSDT. Ohne diese war alles in Ordnung nur eben der IGPU Takt, wie auf meinem ersten Screenshot zu sehen, konstant hoch.

Dann habe ich mir mit MaciASL eine SSDT erstellt und hatte wieder eine KP nach Sleep.

Nach intensiver Suche im Netz und mehrmaliger Lektüre der OC PDF bin ich dann auf den „PowerTmeoutKernelPanic“ gestossen, damit ging es dann.

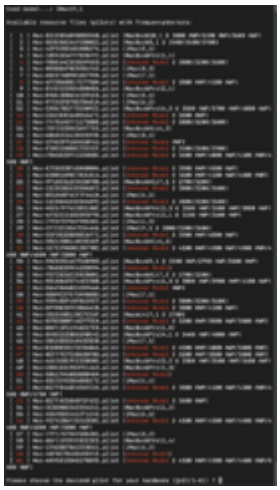
Das hört/ liest sich gerade so logisch und einfach an, war es nicht, viele viele Neustarts und eben „try and error“.

Wahrscheinlich wäre das ganze auch einfacher und eleganter zu lösen aber egal.

Funzt, selbstgebastelt und was gelernt.

Beitrag von „badbrain“ vom 6. Juni 2020, 12:08

kuckkuck Nach der Ausführung des *freqVectorsEdit*-Skripts erhalte ich in einem Z490er System mit i9 10900k und SMBios iMac19,1 die folgende Ausgabe:



Habe ich da etwas falsch gemacht oder liegt das am Comet Lake S Prozessor?

Beitrag von „anonymous_writer“ vom 6. Juni 2020, 12:15

Was soll daran falsch sein? Du wirst aufgefordert eine Auswahl zu treffen. Auswahl 52 sieht ganz gut aus.

Beitrag von „badbrain“ vom 6. Juni 2020, 12:34

Ich habe blau markierte Einträge erwartet, was hier jetzt nicht der Fall ist 

EDIT: Ich vermute, dass das Skript eine Aktualisierung benötigt

> Zitat von kuckkuck
>
> Das Script analysiert jetzt alle "Mac-... .plists" aus dem X86PlatformPlugin und stellt und verschiedene Informationen bereit, Mac-Modelle mit passender maximal Frequenz werden automatisch blau markiert.

Beitrag von „kuckkuck“ vom 6. Juni 2020, 14:29

Der Befehl lautet `./freqVectorsEdit.sh -d 1` für d = 1 was debug = true entspricht.

Beitrag von „badbrain“ vom 6. Juni 2020, 14:49

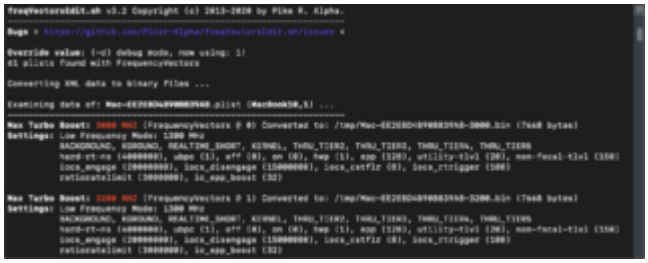
Danke, aber ich habe das genau so ausgeführt.

Beitrag von „kuckkuck“ vom 6. Juni 2020, 14:52

Schick von dem Beginn der Ausführung des Scripts mal ein Screenshot vom Terminal. Benutzt du freqVectorsEdit.sh v3.2? Hast du mal hoch gescrollt?

Beitrag von „badbrain“ vom 6. Juni 2020, 15:00

Ja, das ist Version 3.2. Hier der Screenshot:



Beitrag von „kuckkuck“ vom 6. Juni 2020, 15:00

Da ist doch die Debug Ausgabe darunter, ist doch alles wunderbar...

Beitrag von „badbrain“ vom 6. Juni 2020, 15:07

Ich war ein wenig verwirrt und verunsichert, weil du in der Anleitung die blau markierten Einträge erwähntest und ich diese bei mir nicht sah.

Beitrag von „kuckkuck“ vom 6. Juni 2020, 15:29

> Zitat von kuckkuck
>
> Mac-Modelle mit passender maximal Frequenz werden automatisch blau markiert

Das werden sie ja auch, wenn du die Debug Ausgabe, also das was am Ende deines zweiten Screenshots anfängt und in den Zeilen vor deinem ersten Screenshot aufhört, durchliest. Wenn es keine Mac-Modelle mit passender Frequenz gibt, wird nichts blau markiert. Anyway musst du dich ein bisschen durch die Intel Website zu deinem Prozessor und den Prozessoren der ausgegebenen MacModelle klicken, um ein möglichst gutes Match zu finden.

Beitrag von „schmalen“ vom 13. Juni 2020, 10:04

kuckkuck hatte auch das Script ausgeführt, aber meine Board ID wird nicht aufgeführt (iMac Pro (2017)

Ich weiss das die CPU welche ich im betrieb habe dort nicht aufgelistet ist (iMac Pro (2017) wird aber wohl nichts damit zu tun haben?

Beitrag von „kuckkuck“ vom 13. Juni 2020, 11:01

Die Frage ist, ob irgendein Apple Gerät deine CPU besitzt. Wenn nein, dann musst du das Gerät wählen, dass am nähesten an deine CPU kommt, ist in den seltensten Fällen der Mac Pro.

Beitrag von „ozw00d“ vom 13. Juni 2020, 11:19

Ich bin da anders vorgegangen:

Seit OC 058 gibts ja in der Debug Variante die Möglichkeit sich auch Dumps von der DSDT etc. zu ziehen.

Bin mal den weg gegangen und hab diese DSDT durch SSDTTime gejagt.

Danach die SSDT-PLUG in meiner config aktiviert.

Seitdem schnurrt alles wie es soll, ich kann dabei zuschauen wie meine CPU im IDLE wesentlich öfters auf 800Mhz runtertaktet, überhaupt ein riesiger unterschied.

Wo liegt denn jetzt der Unterschied zwischen der Methode und der beschriebenen?

Zumal, insofern ich das richtig verstehe, laufe ich auf einem iMacPro1,1, da gibts keinen i7 6700k soweit ich das nach der Prozedur sehen konnte.

Beitrag von „JimSalabim“ vom 13. Juni 2020, 12:23

schmalen Wenn es sich um den i9 9900K handelt, der in deinem Profil gelistet ist, dann wäre es folgende Board ID:

[61] Mac-AA95B1DDAB278B95.plist (Unknown Model @ 4100 HWP/4300 HWP/4600 HWP/5000 HWP)

Das ist der iMac19,1, den es mit i9 9900K gibt.

Dein SMBIOS kannst du auf iMacPro1,1 lassen, es geht hier ja nur im die tatsächlich von dir verwendete CPU und den entsprechenden echten Mac.

Beitrag von „schmalen“ vom 13. Juni 2020, 13:23

JimSalabim wenn ich diesen nehme [61] Mac-AA95B1DDAB278B95.plist 19,1. würd mir das aber nichts bringen, da iMacPro diese hat:

"board-id" = <"Mac-7BA5B2D9E42DDD94">

"IOPropertyMatch" = ({"board-id"="Mac-F60DEB81FF30ACF6"},{"board-id"="Mac-7BA5B2D9E42DDD94"},{"board-id"="Mac-27AD2F918AE68F61"})

oder wechsle zu 19,1 und müsste aber dann die interne GPU aktivieren oder auf connectorless setzen?

Beitrag von „JimSalabim“ vom 13. Juni 2020, 13:55

schmalen Wenn es um deinen Hackintosh mit Z390 Aorus Master und i9 9900K geht, ist die von mir genannte Board ID genau die richtige. Es geht hier wie gesagt nur um das CPU-Speedstepping und -Powermanagement! Dass du das iMacPro1,1-SMBIOS verwendest und hierfür eine andere Board-ID existiert, hat hiermit nichts zu tun und steht dem auch nicht im Wege. Du brauchst dein SMBIOS nicht auf iMac19,1 zu ändern.

Beitrag von „kuckkuck“ vom 13. Juni 2020, 14:05

ozw00d Lies am besten den Guide durch. SSDT-Plug hat damit nur bedingt was zu tun und dient nur der Aktivierung des X86PlatformPlugins, siehe Guide. Der Guide geht hier wesentlich weiter und beschreibt wie sich das X86PlatformPlugin-abhängige Speedstepping beeinflussen und anpassen lässt und wie sich ein passendes CPU-Speedstepping-Profil per CPUFriend für den verbauten Prozessor injecten lässt.

> Zitat von kuckkuck
>
> In diesem ausführlichen Guide geht es um absolutes Feintuning – um das Ziel perfektes Apple-konformes CPU-Speedstepping und Frequenz-/Powermanagement zu erreichen.

schmalen Im Guide wird beschrieben, wie du ein CPU-Profil eines SMBios injectest, das nicht deinem aktuellen SMBios entspricht. Es geht also in deinem Fall darum den iMacPro1,1 weiterhin zu nutzen, jedoch das CPU-Profil (that's it) für den i9 9900K (welcher im iMac19,1 verbaut ist) zu injecten.

Beitrag von „schmalen“ vom 13. Juni 2020, 15:11

kuckkuck ich habe das mal soweit gemacht, wie du es im Post 1 beschrieben hast, habe die nächst ähnliche Plist für meine CPU genommen 19,1. Mac-AA95B1DDAB278B95. und diese mit dem Script durchlaufen lassen, verändert wird aber dann die

Mac-7BA5B2D9E42DDD94 für meinem iMacPro 2017, hier wurde aber nur das Datum der Änderung geändert, selbst in der Plist nichts! was mach ich falsch...?

Beitrag von „kuckkuck“ vom 13. Juni 2020, 15:39

Das Script verändert die zum aktuell gewählten SMBios passende Plist und ersetzt diese durch die überarbeitete Plist der gewählten Board-ID. Sicher, dass sich der Inhalt, insbesondere die FrequencyVectors, nicht verändert hat?

Beitrag von „cobanramo“ vom 13. Juni 2020, 17:09

Betriebssystem: Catalina aktuell !

Vermutlich kann da nichts verändert werden, SIP deaktivieren & System Partition schreibbar Mounten nicht vergessen.

Gruss Coban

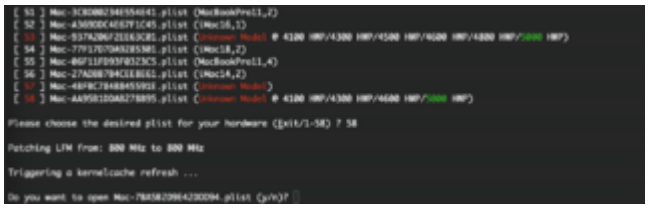
Beitrag von „schmalen“ vom 13. Juni 2020, 17:11

kuckkuck Nachdem ich das Script gestartet habe, habe ich Position 58 gewählt 19,1. also die Mac-AA95B1DDAB278B95 Siehe Bild es zeigt auch an, das was geändert wurde.

Ich habe die Mac-AA95B1DDAB278B95 für 19.1 kontrolliert, darin wurden keine Veränderungen vorgenommen. Währenddessen nach Überprüfung wohl in Mac-7BA5B2D9E42DDD94. iMacPro 1.1 was geändert wurde, was genau kann ich nicht sagen.

Welchen HexEditor würdest du empfehlen..?

möchtest dir mal die Plist anschauen?



Beitrag von „kuckkuck“ vom 13. Juni 2020, 18:40

cobanramo Guter Hinweis!

schmalen Ja, schick mal bitte die original Mac-7BA5B2D9E42DDD94.plist und die veränderte

Mac-7BA5B2D9E42DDD94.plist

Beitrag von „schmalen“ vom 13. Juni 2020, 19:05

kuckkuck Bitte sehr

Geändert_Mac-7BA5B2D9E42DDD94.plist

Original_Mac-7BA5B2D9E42DDD94.plist

Beitrag von „kuckkuck“ vom 13. Juni 2020, 19:14

Die FrequencyVectors sind sehr unterschiedlich:

Hier zB ein Online Tool für dich: http://www.mergely.com/editor?wl=1

Beitrag von „schmalen“ vom 13. Juni 2020, 19:41

kuckkuck sollten die denn nicht mit dem Script eingetragen werden?

Wie ich sehe hast du die in HEX hochgeladen im Onlinetool, kannst du mir noch verraten welchen Hex Editor du benutzt?

Und Danke für das nachschauen!

Beitrag von „kuckkuck“ vom 13. Juni 2020, 21:27

> Zitat von schmalen
>
> kuckkuck sollten die denn nicht mit dem Script eingetragen werden?

Wie meinst du, das wurden sie doch? Ich habe doch die beiden Plists von dir verglichen.

Als simplen Hex Editor kannst du zB Hex Fiend nutzen. Komplexer wird's dann zB mit dem 010 Editor.

Beitrag von „schmalen“ vom 13. Juni 2020, 21:41

kuckkuck ok war wegen der Aussage „ Die FrequencyVectors sind sehr unterschiedlich:„

irritiert.

Super läuft auch im Idle bei ca. 1000-1150

kann ich mit leben, Danke!

Beitrag von „kuckkuck“ vom 13. Juni 2020, 22:58

Die FrequencyVectors wurden durch das script angepasst, das gehört so. Die angepassten Daten extrahierst du dann und verarbeitest sie weiter, siehe Guide. Dadurch kannst du sowohl den LFM (bei dir vielleicht noch auf ~1000Mhz?) senken, als auch das Taktverhalten steuern (EPP) sodass du sicherlich noch angenehmere Taktraten erzielen kannst

Beitrag von „revunix“ vom 15. Juni 2020, 11:35

Also irgendwie funktioniert das bei dem aktuellen Dev Update nicht mehr so richtig.

Donnerstag hab ich bei mir das Update gemacht.

EDIT: Ich hab das ganze nochmal gemacht, und es scheid wieder zu funktionieren. Wird sich wohl was geändert haben. idk

Beitrag von „ozw00d“ vom 15. Juni 2020, 18:03

So schaut's bei mir nur mit ner richtigen SSDT aus, ohne das ganze CPUFriend gedöhns. Eventuell eine Möglichkeit für dich revunix ?

Last ist wegen Arbeit

Beitrag von „revunix“ vom 15. Juni 2020, 18:09

Bei mir läuft das ganze auch mit einer SSDT und nicht mit einem Kext. Oder hast du noch etwas anderes gemacht?

Beitrag von „ozw00d“ vom 15. Juni 2020, 18:13

revunix ich bin vorgegangen wie im dorthania How To für Skylake.

Damit meine ich folgende Punkte:

1.) DEBUG Variante OC nutzen und damit eine DSDT erstellen,

2.) DSDT durch SSDTTime jagen,

3.) erstellte SSDT-PLUG in den ACPI Settings von OC hinterlegen.

mehr nicht.

Alle anderen DSDT Varianten (MaciASL, Clover etc.) hatte ich auch probiert aber die DSDTs waren immer fehlerhaft.

Meine SSDT:

Code

```
/*
* Intel ACPI Component Architecture
* AML/ASL+ Disassembler version 20200110 (64-bit version)
* Copyright (c) 2000 - 2020 Intel Corporation
*
* Disassembling to symbolic ASL+ operators
*
* Disassembly of iASLNxAWzF.aml, Mon Jun 15 18:14:13 2020
*
* Original Table Header:
* Signature "SSDT"
* Length 0x000000A7 (167)
* Revision 0x02
* Checksum 0xA6
* OEM ID "CORP"
* OEM Table ID "CpuPlug"
* OEM Revision 0x00003000 (12288)
* Compiler ID "INTL"
* Compiler Version 0x20180427 (538444839)
*/
DefinitionBlock ("", "SSDT", 2, "CORP", "CpuPlug", 0x00003000)
{
External (_PR_.CPU0, ProcessorObj)

```

```
Scope (\_PR.CPU0)
{
Method (DTGP, 5, NotSerialized)
{
If ((Arg0 == ToUUID ("a0b5b7c6-1318-441c-b0c9-fe695eaf949b")))
{
If ((Arg1 == One))
{
If ((Arg2 == Zero))
{
Arg4 = Buffer (One)
{
0x03 // .
}
Return (One)
}

If ((Arg2 == One))
{
Return (One)
}
}
}

Arg4 = Buffer (One)
{
0x00 // .
}
Return (Zero)
}

Method (_DSM, 4, NotSerialized) // _DSM: Device-Specific Method
{
Local0 = Package (0x02)
{
"plugin-type",
One
}
DTGP (Arg0, Arg1, Arg2, Arg3, RefOf (Local0))
Return (Local0)
}
}
```

```
67. }
```

Alles anzeigen

Beitrag von „kuckkuck“ vom 15. Juni 2020, 18:18

> Zitat von ozw00d
>
> ohne das ganze CPUFriend gedöhns.

Deine Antwort geht etwas am Thema vorbei, wenn ihr euch über verschiedene Möglichkeiten austauschen wollt, wie man das X86PlatformPlugin aktivieren kann, dann tut das bitte nicht in einem Thread in dem es nicht um das Aktivieren, sondern um das Anpassen des X86PlatformPlugins geht. Um zu verstehen was CPUFriend macht und wann die Benutzung sinnvoll ist, kann ich nur auf den Eingangspost und die Anleitung in diesem Thread verweisen.

> Zitat von ozw00d
>
> mit ner richtigen SSDT

Die SSDT ist für die Nutzung von CPUFriend die gleiche, sie stellt die Basis zur Aktivierung des X86PlatformPlugins dar. Das finden der ACPI ID wird ebenfalls im Guide erwähnt, bei falschem ACPI CPU Namen wird das Property aus der _DSM Methode nicht für die CPU injected und das X86PlatformPlugins lädt somit nicht. Deswegen ist deine eingefügte SSDT ozw00d auch nicht besondern sinnvoll für jedermann, da sie nicht allgemeingültig ist (vgl. SSDT-PLUG von OpenCore). Um eine SSDT zur Aktivierung des X86PlatformPlugins zu erstellen, kann man entweder eine Vorlage finden und die ACPI ID (CPU0 Name) mit der ACPI ID aus der DSDT ersetzen, SSDT-PLUG von acidanthera nutzen, oder sich von Tools wie SSDTTime eine angepasste Variante ausgeben lassen.

Beitrag von „ozw00d“ vom 15. Juni 2020, 18:20

kuckkuck alles klar.

Beitrag von „Inspector42“ vom 17. August 2020, 19:02

kuckkuck

Bin gerade über diesen Thread gestolpert. Ich hatte mir das Skript von Pike R. Alpha FreqVectorsEdit.sh vor etlicher Zeit mal angeschaut, aber wegen der Tatsache, dass es direkt im System "rumfuscht" wieder Abstand davon genommen.

Nach einem erneuten Blick auf das Skript ist mir dann aufgefallen, dass der Pfad zu den System Extensions separat in einer Variablen definiert wird

Code

```
122 #
123 #
124 #
125 gExtensionsDirectory="/System/Library/Extensions"
126
127 #
128 # Path to kext.
129 #
130 gResourcePath="${gExtensionsDirectory}/IOPlatformPluginFamily.kext/Contents/PlugIns/X86PlatformP
131
```

Hier kann man vermutlich einen anderen Pfad im eigenen Home-Verzeichnis eintragen und spart sich damit das Ganze Drama mit SIP aus und wieder an sowie dem Zurückkopieren der plist.

Habe leider im Moment keine Zeit, das mal in Ruhe zu testen.

Beitrag von „kuckkuck“ vom 17. August 2020, 19:24

Inspector42 Da das script open-source ist, sollte das auf jeden Fall irgendwie möglich sein. Wenn du das script so anpassen solltest, dass es für User angenehmer zu benutzen ist, dann lass es mich gerne wissen und ich füg es entsprechend in den Guide ein

---

Beitrag von „Inspector42“ vom 18. August 2020, 18:08

kuckkuck

schaue ich mir definitiv an, kann nur einen Moment dauern, ich bin gerade beruflich ziemlich eingespannt.

Da sich Pike nach dem Tod seiner Frau aus der Szene zurückgezogen hat, muss ich dann dafür wohl mindestens einen neuen Branch in Github anlegen. Da muss ich auch erstmal rantasten, aber im Zweifel hängen wir die Version einfach hier im Forum dran. Mir schwebt idealerweise vor, dass man dem Script den Pfad als Parameter übergeben kann.