

# OpenCore Sammelthread (Hilfe und Diskussion)

Beitrag von „jgehw“ vom 8. September 2020, 19:54

Hallo,

ich bekomme den Ruhezustand bei mir einfach nicht ans Laufen. Angefangen hab ich mit einem vorkonfigurierten Skylake Clover Image, was hier freundlicherweise im Forum angeboten wurde (damit lief alles auf Anhieb, außer dem Ruhezustand). Dann bin ich auf OpenCore umgestiegen, hab die neuesten Releases von <https://github.com/acidanthera/> verwendet und anhand des Guides <https://dortania.github.io/OpenCore-Install-Guide/> konfiguriert (selbes Ergebnis: es lief alles, außer dem Ruhezustand). Dann hab ich die Lösungsvorschläge <https://dortania.github.io/OpenCore-Install-Guide/universal/sleep.html> komplett abgearbeitet, dabei insbesondere UsbInjectAll.kext durch ein eigenes angepasstes USB mapping ersetzt (<https://dortania.github.io/OpenCore-Post-Install/usb/>), für die iGPU den Parameter `igfxonln=1` hinzugefügt, das CPU Power Mapping angepasst (<https://dortania.github.io/OpenCore-Install-Guide/universal/pm.html>), eine SSDT-HPET.aml gegen mögliche IRQ Konflikte erstellt, eine eigene SSDT-SBUS-MCHC.aml erstellt (<https://dortania.github.io/OpenCore-Install-Guide/universal/smbus.html>), sowie sichergestellt, dass die Voraussetzungen für die beschriebenen möglichen Probleme mit Thunderbolt, NIC, NVMe, Display, NVRAM, RTC, Audio oder TSC nicht vorliegen. Jetzt hab ich zwar eine perfekt angepasste Config und vieles gelernt, aber das Problem mit dem Ruhezustand besteht weiterhin.

Das einzige, was ich während des Anpassungsprozesses auffällig fand, war, dass der Guide <https://dortania.github.io/OpenCore-Install-Guide/universal/pm.html#using-cpu-friend> nicht wie erwartet funktionierte: Obwohl ich eine Skylake CPU habe, wurde der zweite der drei Schritte (EPP) im CPUFriendFFriend übersprungen; und wenn ich mit dem generierten CPUFriendDataProvider.kext gebootet und CPUFriendFFriend nochmal aufgerufen habe, wurden mir als aktuelle Werte immer noch die alten angezeigt. Ich hab mir den Python Source von CPUFriendFFriend daraufhin angeschaut und festgestellt, dass das Tool im ioreg output von IOPlatformPowerProfile -> FrequencyVectors (hex-codierte Binärdaten) nach Teilstrings sucht und teilweise nicht fündig wird. Da ich keine Doku über das Binärformat finden konnte, war mir der Ansatz zu unseriös, und ich bin zu Plan B, das ssdtPRGen Tool zu verwenden (<https://dortania.github.io/OpenCore-Install-Guide/universal/y-bridge-power-management>) übergegangen, da das Tool (entgegen der Überschrift in der Anleitung) mittlerweile offenbar erweitert wurde und auch Skylake unterstützt; diese Anleitung ließ sich dann auch problemlos anwenden.

Im Laufe der vielen Versuche wachte macOS einmal plötzlich aus dem Ruhezustand wieder

auf. Doch zu früh gefreut: mehrfaches wiederholen zeigte, dass es nur ca. jedes 15. Mal klappt. Immerhin eine neue Erkenntnis: es spielt eine race condition / timing / Zufall mit rein.

Schließlich hab ich aus dem Gedanken heraus "vllt geht ja wenigstens suspend-to-disk, wenn suspend-to-ram nicht geht" eine möglicherweise hilfreiche Entdeckung gemacht: wenn ich mittels `sudo pmset -a hibernatemode 25` die Art des Ruhezustands umstelle, scheitert das Aufwachen aus suspend-to-ram und suspend-to-disk nacheinander mit **exakt derselben** kernel panic, wohingegen die kernel panics sich sonst unterscheiden. Ich würde daraus schlussfolgern, dass die zufällige Komponente des Scheiterns / Gelingens vom Zustand vor dem Ruhezustand und nicht vom Aufwachprozess abhängt. Nur bin ich mit meinem Latein leider nun am Ende. Weiß jemand hier Rat?

Ich hänge einige kernel panics an. Man sieht, dass es immer auf eine ähnliche Weise knallt, aber sehr unterschiedliche stack traces dahin führen können.