AppleHDA scheint nicht zuverlässig zu laden

Beitrag von „Keksfamilie“ vom 8. April 2021, 21:57

So, jetzt kann ich mal in Ruhe tippen

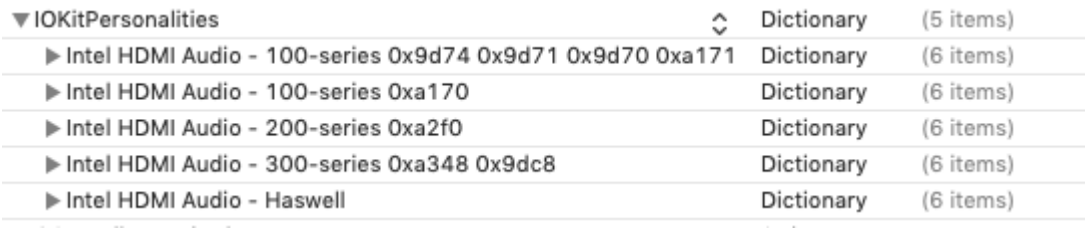
MacPeet 709D0000

asr10 gute Frage... ich habe leider keinen Monitor mit Lautsprechern oder Kopfhörerausgängen Kanns also überhaupt nicht testen

Bei Comet Lake wird ja wie schon festgestellt das Audiogerät hinterm PCH "versteckt" und gibt ne Device-ID vom PCH aus. Die kennt MacOS nicht und was der Bauer nicht kennt frisst er nicht Daher muss also die ID gefaked werden. Da reicht es aber nicht, das über die DeviceProperties zu machen, ich zitiere CorpNewt:

> Zitat
>
> When you fake a device-id in DeviceProperties, it fakes it "at the surface" - i.e. it can be used to get another kext to load by matching an existing IOPCIMatch within the target kext
>
> Sometimes there are further checks within the kext though - which would either need to be patched, or worked around.

Daher ist der Weg, der ganz oft zu finden ist, die ID auf einem "tieferen" Level zu faken, das geht mithilfe des FakePCIID.kext von RehabMan. Das alleine tut erstmal nichts, bietet aber eine Schnittstelle an die man sich mit anderen kexts hängen kann. Da gibts z.B. den bekannten FakePCIID_Intel_HDMI_Audio.kext. Dieser matched auf verschiedene IOKitPersonalities:



und injected dann entsprechend die gefakte ID.

Das matchen auf die IOKitPersonalities klappt halt bei Comet Lake (und anderen neuen Plattformen dann wohl auch) nicht, weil die natürlich andere IDs haben.

Nun gehen dann viele hin, setzen eine device-id in den DeviceProperties wie z.B. 70A10000 (wäre im Bild die zweite), was dann dafür sorgt, dass die IOKitPersonality im FakePCIID matched, damit dann wieder eine device-id gefaked werden kann Doppelt gemoppelt. Man kann das ganze dann etwas sauberer machen, in dem man keine DeviceProperty nutzt um die device-id zu faken, sondern einen kext baut, der direkt auf die vanilla/originale device-id matched.

Als Template kann man einen der vorhandenen IOKitPersonality Einträge nehmen, alle anderen rauslöschen (werden ja nicht benötigt) und dann IDs austauschen.

In meinem Fall habe ich folgendes als Vorlage genommen, meine vanilla device-id war die c8060000:

Code

```
<key>Intel HDMI Audio - 100-series 0xa170</key>
<dict>
<key>CFBundleIdentifier</key>
<string>org.rehabman.driver.FakePCIID</string>
<key>IOClass</key>
<string>FakePCIID</string>
<key>IOMatchCategory</key>
<string>FakePCIID</string>
<key>IOPCIPrimaryMatch</key>
<string>0xa1708086</string>
<key>IOProviderClass</key>
<string>IOPCIDevice</string>
<key>FakeProperties</key>
<dict>
<key>RM,device-id</key>
<data>cJ0AAA==</data>
</dict>
</dict>
```

Alles anzeigen

und geändert in:

Code

```
<key>Intel CML PCH (device-id c8060000)</key>
<dict>
<key>CFBundleIdentifier</key>
<string>org.rehabman.driver.FakePCIID</string>
```

5. <key>IOClass</key>
6. <string>FakePCIID</string>
7. <key>IOMatchCategory</key>
8. <string>FakePCIID</string>
9. <key>IOPCIPrimaryMatch</key>
10. <string>0x06c88086</string>
11. <key>IOProviderClass</key>
12. <string>IOPCIDevice</string>
13. <key>FakeProperties</key>
14. <dict>
15. <key>RM,device-id</key>
16. <data>cJ0AAA==</data>
17. </dict>
18. </dict>

Alles anzeigen

cJ0AAA== entspricht der id 709d0000.

Bei dem IOPCIPrimaryMatch dran denken, dass der Vendor mit in den String kommt (weil Intel hier 8086) und die byte-order einhalten.

Die beiden Kexts dann nicht vergessen in der OC Config zu aktivieren 

Ich hoffe ich habe das jetzt so alles korrekt wiedergegeben, ist eigentlich kein Hexenwerk.