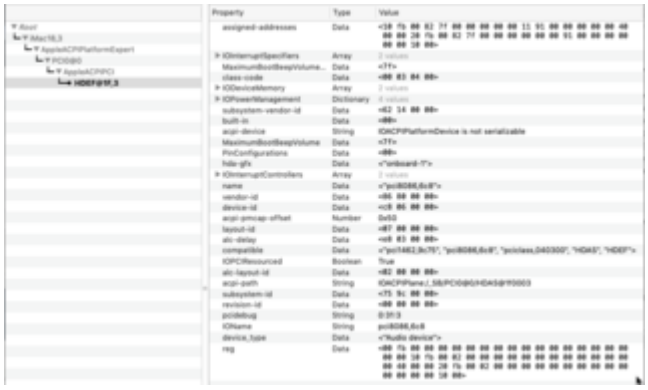
AppleHDA scheint nicht zuverlässig zu laden

Beitrag von „Keksfamilie“ vom 3. April 2021, 14:14

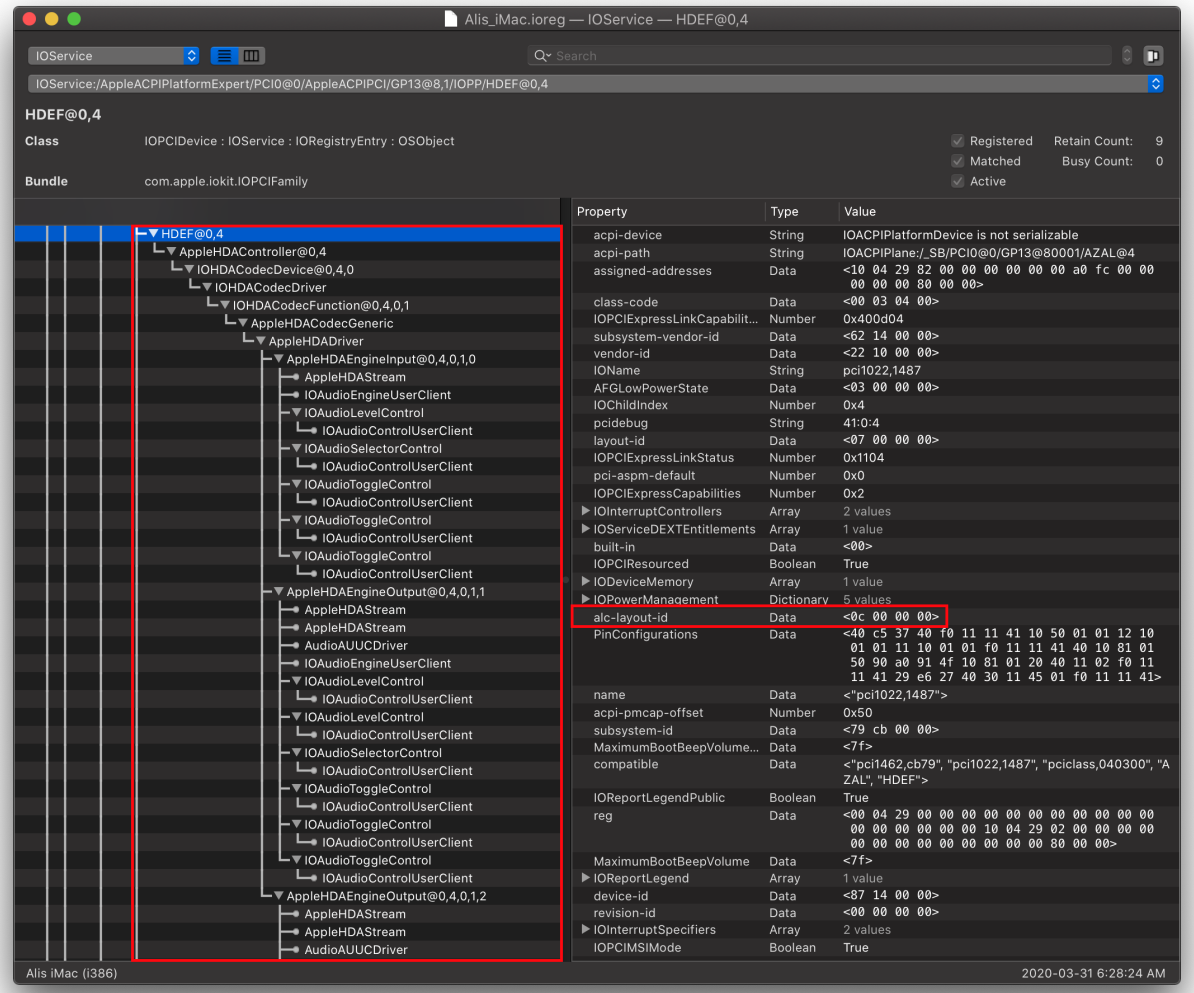
Hallo zusammen,

ich habe Probleme auf meinem Hackintosh Sound zum laufen zu bekommen.

In den Soundeinstellungen taucht kein Ein- oder Ausgang auf und wenn ich einen Blick auf den IORegistryExplorer werfe, sehe ich, dass ein HDEF Device auftaucht, aber kein untergeordnetes AppleALC:



So sollte das wohl eigentlich aussehen (Quelle: Dortanias Guide):



Ich habe auch immer wieder den Fall, dass AppleHDA nicht geladen wurde:

Code

1. $ kextstat | grep -E "AppleHDA|AppleALC|Lilu"
2. 38 6 0xffffff7f83eba000 0x87000 0x87000 as.vit9696.Lilu (1.5.1) 64003D3E-735A-3076-B7D5-A88271D2510B <8 6 5 3 2 1>
3. 39 0 0xffffff7f83f5b000 0x181000 0x181000 as.vit9696.AppleALC (1.5.8) CF2BB91F-73DC-3B69-9AB0-CF454FA77774 <38 13 8 6 5 3 2 1>

Beim nächsten Reboot taucht es mit etwas Glück aber wieder auf, ändert aber nichts daran, dass ich keine Geräte bekomme.

Habe schon mit diversen Werten von alc-delay rumprobiert, selbst bei 3000ms (maximum) tut sich nichts.

Kann mir jemand weiterhelfen?

Beitrag von „HackBook Pro“ vom 3. April 2021, 17:05

~~Hast du nicht schon einen Thread zu diesem Thema?~~

Edit: Sorry, hab dich verwechselt.

Beitrag von „al6042“ vom 3. April 2021, 17:20

Nope.. hat er nicht...

Beitrag von „Keksfamilie“ vom 6. April 2021, 18:24

Habe da noch eine Vermutung entwickelt, das Audiodevice "versteckt" sich ja hinter dem PCH, in meinem Fall ist das ein Comet Lake PCH, welches ich aber unter Mojave betreibe. Während des Bootvorgangs taucht die Warnung "Unknown PCH" auf, was bis jetzt keine Probleme bereitet hatte. Könnte das dafür sorgen, dass sich AppleHDA da nicht attachen kann?

AppleALC kommt augenscheinlich damit klar, hier der Auszug aus dem Hackintool:

6c8 taucht in den Debug Logs als controller 1 wie folgt auf (controller0 ist die iGPU):

Code

1. 2021-04-06 17:29:35.496208+0200 localhost kernel[0]: (Lilu) AppleALC iokit: @ (DBG) getOSData vendor-id has 8086 value
2. 2021-04-06 17:29:35.509522+0200 localhost kernel[0]: (Lilu) AppleALC iokit: @ (DBG) getOSData device-id has 6C8 value
3. 2021-04-06 17:29:35.522647+0200 localhost kernel[0]: (Lilu) AppleALC iokit: @ (DBG) getOSData revision-id has 0 value
4. 2021-04-06 17:29:35.535583+0200 localhost kernel[0]: (Lilu) AppleALC iokit: @ (DBG) getOSData alc-layout-id has 45 value
5. 2021-04-06 17:29:35.548493+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) found 2 audio controllers
6. 2021-04-06 17:29:35.561125+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) validating 0 controller 8086:3E9B:5
7. 2021-04-06 17:29:35.573973+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) comparing to 0 mod 10DE:E0F
8. [...]
9. 2021-04-06 17:29:35.934662+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) validating 1 controller 8086:6C8:0
10. 2021-04-06 17:29:35.943764+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) comparing to 0 mod 10DE:E0F
11. [...]
12. 2021-04-06 17:29:36.097536+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) comparing to 28 mod 8086:6C8
13. 2021-04-06 17:29:36.100597+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) found mod for 1 controller
14. 2021-04-06 17:29:36.103497+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) missing ControllerModInfo for 0 controller
15. 2021-04-06 17:29:36.106742+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) handling 1 controller 8086:6C8 with 1 patches

16. 2021-04-06 17:29:36.110015+0200 localhost kernel[0]: (Lilu) Lilu mach: @ (DBG) found symbol __ZN16IOHDACodecDevice11executeVerbEtttPjb at 0xffffff7f99bcc3b0 (non-aslr 0x13b0), type f, sect 1, desc 0
17. 2021-04-06 17:29:36.115312+0200 localhost kernel[0]: (Lilu) Lilu patcher: @ (DBG) from 0xFFFFFF7F99BCC3B0 to 0xFFFFFF7F9955E420 diff 0xFFFFFFFFFF99206B argument FF99206B
18. 2021-04-06 17:29:36.119977+0200 localhost kernel[0]: (Lilu) Lilu patcher: @ (DBG) from 0xFFFFFF7F994C5BA6 to 0xFFFFFF7F99BCC3B6 diff 0x000000000070680B argument 70680B
19. 2021-04-06 17:29:36.124518+0200 localhost kernel[0]: (Lilu) Lilu patcher: @ (DBG) wrapped __ZN16IOHDACodecDevice11executeVerbEtttPjb
20. 2021-04-06 17:29:36.138505+0200 localhost kernel[0]: (Lilu) Lilu patcher: @ (DBG) invoked at kext loading/unloading
21. 2021-04-06 17:29:36.141704+0200 localhost kernel[0]: (Lilu) Lilu patcher: @ (DBG) last kext is 0xFFFFFF7F99BD7000 and its name is com.apple.driver.AppleHDAController
22. 2021-04-06 17:29:36.146294+0200 localhost kernel[0]: (Lilu) Lilu patcher: @ (DBG) caught the right kext at 0xFFFFFF7F99BD7000, invoking handler
23. 2021-04-06 17:29:36.150390+0200 localhost kernel[0]: (Lilu) Lilu mach: @ (DBG) aslr/load slide is 0xffffff7f99bd7000
24. 2021-04-06 17:29:36.153931+0200 localhost kernel[0]: (Lilu) Lilu mach: @ (DBG) getRunningPosition 0xffffff7f99bd7000 of memory 114688 size
25. 2021-04-06 17:29:36.158080+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) missing ControllerModInfo for 0 controller
26. 2021-04-06 17:29:36.161994+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) handling 1 controller 8086:6C8 with 1 patches
27. 2021-04-06 17:29:36.166087+0200 localhost kernel[0]: (Lilu) AppleALC alc: @ (DBG) checking patch 0 for 2 kext (com.apple.driver.AppleHDAController)

Alles anzeigen

Beitrag von „apfelnico“ vom 6. April 2021, 19:58

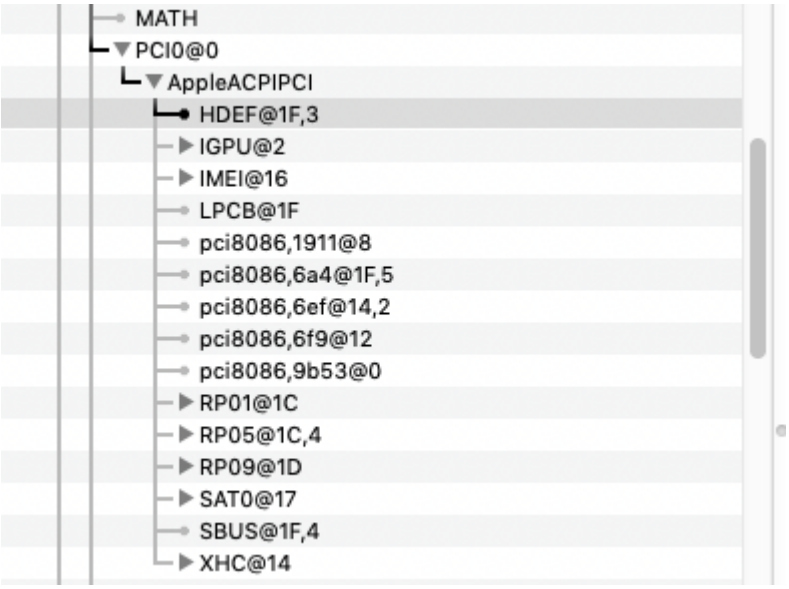
Keksfamilie

Wenn du innerhalb von "IORegistryExplorer" nach "HDEF" suchst, bekommst du auch das von dir verlinkte Bild zu sehen. Nimm die Suche raus und scrolle bis zu "HDEF", dann siehst du auch den folgenden Baum.

Beitrag von „Keksfamilie“ vom 6. April 2021, 20:48

apfelnico Hi, da hast du recht, sorry, da hab ich den Screenshot verhauen.

So hier ein Screenshot ohne Suchfilter, aber auch ohne AppleHDA:



Beitrag von „MacPeet“ vom 6. April 2021, 21:15

Das Problem ist ja, dass diese neuen Rechner alle ein Intel-Device vor dem eigentlichen Realtek-Audio haben, was sich als Audio-Chip ausgibt.

Sieht man sehr gut unter Windows in den Hardware-Eigenschaften.

Es betrifft alle 400Series und neu die 500Series. Letztere haben heute im neuen Release AppleALC auch zwei neue Controller-Patches bekommen.

Diese Sache ist aber lange bekannt und es gibt für Dein Device 6c8 bereits seit AppleALC 1.5.1 einen Controller-Patch. (https://github.com/acidanthera/AppleALC/releases)

Erst heute mit dem neuen Release AppleALC kamen 2 neue Controller-Patches für 500Series (Z590).

Leider habe ich bis dato noch nie einen neuen Codec für Z490 oder Z590 gesehen, halt nur die Controller-Patches.

Dies liegt wohl auch daran, dass selbst Linux damit nicht klar kommt ohne Fake Device-ID und somit keine Codec-Dump's möglich sind.

Dein Controller wird aber seit AppleALC 1.5.1 wohl erkannt. In Deinem ersten Bild Post #1 injectest Du die alc-layout-id 2, welche wohl nicht passt.

Wenn die Knoten nicht passen, dann wird es auch keine Geräte beim Audio geben, selbst wenn der Controller in AppleALC erkannt wird.

Ich kann Dir nur empfehlen, versuche alle layoutID's des entsprechenden Realtek-Chip's durch. Mit viel Glück sind die Knoten gleich oder fast gleich und Du bekommst einige Audio-Geräte.

Ferner, versuche mittels Linux-Live ein Codec-Dump zu machen!

Beitrag von „Keksfamilie“ vom 6. April 2021, 22:54

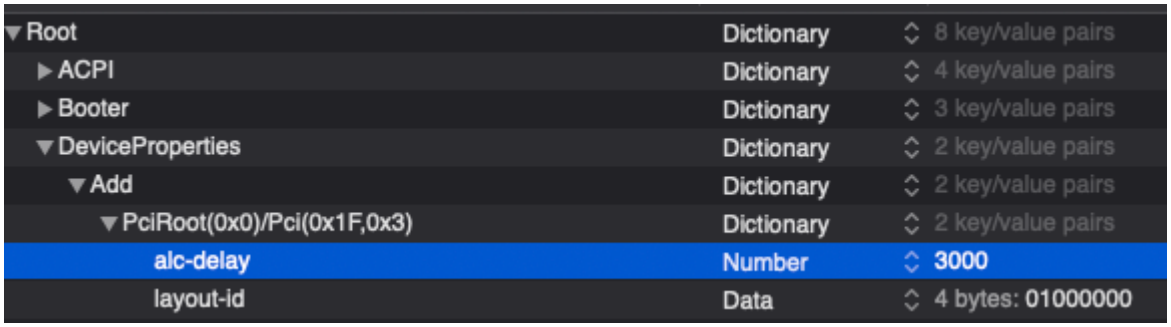
MacPeet Hi, ich habe das neue Update heute schon voller Vorfreude ausprobiert gehabt... ne ☹

Ich habe auch bereits alle Layout IDs die für den ALC892 ausprobiert die gelistet sind, aber ohne Erfolg. Ich dachte, AppleHDA taucht trotzdem auf, auch wenn mit zu wenigen weiteren Children? Unabhängig davon habe ich es aber mit keiner ID geschafft das zum laufen zu bringen (und auch nie Ausgänge/Eingänge bekommen).

Codec Dump(s) (0 ist Realtek, 2 ist Intel HDMI) habe ich soeben angefertigt und hier angehängt ☺ Kann mit denen selbst nicht so viel anfangen, hast du da ne Resource zum nachlesen? Oder kannst mir selbst erklären was man jetzt mit denen macht?

Beitrag von „ozw00d“ vom 7. April 2021, 01:41

Keksfamilie was soll der folgende eintrag in deiner EFI bewirken?
alc-delay --> 3000



Bei Comet Lake sollte dieser auf
PciRoot(0x0)/Pci(0x1b,0x0)

lauten,

nicht auf

(0x1F,0x3)

wie bei dir.

Beitrag von „MacPeet“ vom 7. April 2021, 09:17

Ich habe den Codec_Dump mal gewandelt. Deine Knoten sind im Prinzip mit vielen ID's gängig. Am Besten würde wohl die layoutID 100 passen, welche für einen MSI Z370-A PRO entwickelt wurde. Die Knoten sind dort identisch.

Dein Problem liegt aber vermutlich ganz woanders. Habe mir Deine EFI mal angesehen.

Du hast keine IRQ-Fixes gemacht, ohne die AppleALC keine Geräte lädt.

Nutze dafür das Script SSDTTime von GitHub. Hierfür brauchst Du die Clean-DSDT.

SSDTTime zeigt Dir dann die nötigen Patches, welche Du in der OC config.plist im Bereich ACPI/patch einbinden kannst.

Genauer gesagt, muss bei RTC und TIMR die 0 und die 8 (manchmal auch noch die 11) aus dem irqnoflags gelöscht werden.

Die 2 bei IPIC ist für Audio nicht wichtig, wird bei SSDTTime oft mit angezeigt. Wenn ein Patch für IPIC angeboten wird, kann man den auch setzen.

Die 0, 8 und 11 werden für HPET gesetzt. Evtl. brauchst Du auch noch den HPET-Patch von SSDTTime.

Wenn Du mal im System das Tool MaciASL startest, dann wird die aktuelle System-DSDT geladen. Diese kannst Du ja hier mal posten, bzw. Dir ansehen, wie die Bereiche RTC, TIMR und HPET aktuell aussehen.

Setze Dich mal mit SSDTTime auseinander! Ich denke, danach werden sicher Geräte beim Audio geladen.

Beitrag von „Keksfamilie“ vom 7. April 2021, 15:23

ozw00d Da hast du Recht... was man so online findet, ist CML auf 0x1b,0x0.

Ich bin nach dem gegangen, was mir gfxutil ausgespuckt hatte:

Code

1. $ ./gfxutil-1/gfxutil -f HDEF
2. 00:1f.3 8086:06c8 /PCI0@0/HDEF@1F,3 = PciRoot(0x0)/Pci(0x1F,0x3)

Warum weicht das bei mir ab?

MacPeet Danke für die Erklärung,

einmal Pre-SSDTTime:

RTC hatte die 8, TIMR die 0. Nach kopieren der Patches in ACPI/Patch sieht das wie folgt aus:

Die 0 und 8 wurden also erfolgreich aus den Devices entfernt. Wenn ich dich richtig verstanden habe, sollten die jetzt an das HPET Device gebunden werden mithilfe des HPET Patches. Den habe ich also auch in den ACPI Ordner geschoben und in die config.plist hinzugefügt. Da steht als Kommentar noch dabei "HPET _CRS (Needs _CRS to XCRS Rename)", dieser wurde auch in ACPI/Patches eingetragen. Das HPET Device scheint aber nicht die IRQNoFlags zu übernehmen (auch wenn diese im SSDT-HPET.aml deklariert werden):

```
Scope (_SB.PCI0.LPCB)
{
    Device (HPET)
    {
        Name (_HID, EisaId ("PNP0103") /* HPET System Timer */)  // _HID: Hardware ID
        Name (_UID, Zero)  // _UID: Unique ID
        Name (BUF0, ResourceTemplate ()
        {
            Memory32Fixed (ReadWrite,
                0xFED00000,         // Address Base
                0x00000400,         // Address Length
                _Y41)
        })
        Method (_STA, 0, NotSerialized)  // _STA: Status
        {
            If (HPTE)
            {
                Return (0x0F)
            }

            Return (Zero)
        }

        Method (XCRS, 0, Serialized)
        {
            If (HPTE)
            {
                CreateDWordField (BUF0, \_SB.PCI0.LPCB.HPET._Y41._BAS, HPT0)  // _BAS: Base Address
                HPT0 = HPTB /* \HPTB */
            }

            Return (BUF0) /* \_SB_.PCI0.LPCB.HPET.BUF0 */
        }
    }
```

Habe mal die Ergebnisse die SSDTTime erzeugt hat angehängt (da ist auch meine Vanilla-DSDT dabei) und mein aktualisiertes EFI. Hab ich da noch was übersehen?

Beitrag von „MacPeet“ vom 7. April 2021, 15:52

Auf all Deinen Bildern hier im Thread zeigt Dein Audio auf 1F und nicht auf 1B. Viele neuere Rechner haben 1F.

Welche Adresse steht denn in der clean-DSDT bei HDEF oder HDAS?

Wenn es 1F ist, dann könntest Du in Section UEFI/Audio den Pfad auch in 1F ändern, falls Du mal Startsound haben möchtest. Dort steht noch die 1B.

Die IRQ-Patches hast Du perfekt gemacht.

Hast Du danach schon mal getestet, ob Du Audiogeräte bekommst, z.B. mit der layoutID 100 ?

Ich rate auch bei allen Test's zum Bootflag z.B. alcid=100

Die Sache mit HPET und SSDTTime ist nicht immer ganz perfekt, hatte die gleichen Probleme auf meinem Lenovo.

SSDTTime will es immer in die _CSR legen.

In meiner SSDT für HPET ist es nun in der BUF0-Methode, in dem Fall nur die 0 und 8, aber der Rechner geht perfekt. Beispiel:

```
Scope (_SB.PCI0.LPC)
{
    Device (HPE0)
    {
        Name (_HID, EisaId ("PNP0103") /* HPET System Timer */)  // _HID: Hardware ID
        Name (_UID, Zero)  // _UID: Unique ID
        Name (BUF0, ResourceTemplate ()
        {
            IRQNoFlags ()
                {0,8}
            Memory32Fixed (ReadWrite,
                0xFED00000,         // Address Base
                0x00000400,         // Address Length
                )
        })
        Method (_STA, 0, NotSerialized)  // _STA: Status
        {
            If (_OSI ("Darwin"))
            {
                Return (0x0F)
            }
            Else
            {
                Return (Zero)
            }
        }

        Method (_CRS, 0, Serialized)  // _CRS: Current Resource Settings
        {
            Return (BUF0) /* \_SB_.PCI0.LPC_.HPE0.BUF0 */
        }
    }
```

Als Experte fällt mir hier grt ein.

Beitrag von „grt“ vom 7. April 2021, 16:09

guckst du mal da: KLIKK

wir hatten das problem, dass zwar die ssdt-hpet (nicht die von ssdt-time, ist in dem thread verlinkt) sehr gut und überzeugend aussah, aber das renaming mit OC nicht so richtig reibungslos funktionierte.

das musste quasi zu fuss erledigt werden:

erstmal zählen, wieviele _CRS in der dsdt vor dem umzubenennenden vorkommen, und zwar vor dem letzten, was zu ändern ist (hier im TIMR). die zahl in das feld "skip" eintragen, reboot, mit maciasl gucken, wo der rename gelandet ist (was da wie gezählt wird, kommt mir höchst spanisch vor, weil der rename nicht dort landet, wo man es nach der zählerei eigentlich erwartet), nachzählen, wie der versatz ist, "skip" entsprechend korrigieren, gucken, korrigieren... bis der rename passt. dann kann man dahinter(!!!) die nächsten renames setzen, von unten nach oben sozusagen. danach griff die ssdt-hpet, und mit dem richtigen layout gab es dann auch ton.

Beitrag von „apfelnico“ vom 7. April 2021, 16:26

Den Zirkus muss man jetzt nicht mehr tun, im neuen OoenCore ist dafür eint tolle Funktion

eingebaut, um ein ACPI-Patch exklusiv auf ein bestimmtes Device ausführen zu lassen. Einfach mal reinschauen, die ersten beiden Einträge studieren.

Beitrag von „MacPeet“ vom 7. April 2021, 16:30

> Zitat von apfelnico
>
> ..., die ersten beiden Einträge studieren.

Klingt interessant, aber die ersten beiden Einträge wovon? Stehe gerade auf dem Schlauch.

Beitrag von „apfelnico“ vom 7. April 2021, 16:39

So sieht's zum Beispiel aus:

▼ACPI	Dictionary	4 Schlüssel/Wert-Paare
▶ Add	Array	17 geordnete Elemente
▶ Delete	Array	2 geordnete Elemente
▼ Patch	Array	1 geordnete Elemente
▼ 0	Dictionary	14 Schlüssel/Wert-Paare
Base	String	_SB.PCI0.LPCB.HPET
BaseSkip	Zahl	0
Comment	String	HPET _CRS to XCRS
Count	Zahl	1
Enabled	Boolean	NO
Find	Daten	4 Bytes: 5F435253
Limit	Zahl	0
Mask	Daten	0 Bytes:
OemTableId	Daten	0 Bytes:
Replace	Daten	4 Bytes: 58435253
ReplaceMask	Daten	0 Bytes:
Skip	Zahl	0
TableLength	Zahl	0
TableSignature	Daten	0 Bytes:

Beitrag von „grt“ vom 7. April 2021, 16:44

das hatten wir stehen, ich hab mich so richtig gefreut, weil ich darauf spekulierte, die zählerei bleibenlassen zu können, und es hat definitiv nicht funktioniert (pfade hab ich kontrolliert). destawegen dann doch zählen wie der teufel... OC 0.6.8

Beitrag von „apfelnico“ vom 7. April 2021, 17:11

Man kann natürlich auch die DSDT in einen Hexeditor werfen, die benötigte Stelle suchen und bissel Bits davor mitnehmen, damit man ein exklusives Ergebnis bekommt.

Beitrag von „grt“ vom 7. April 2021, 17:13

ganz schön umständlich..

aber klar, ginge auch, aber mit dem zählen hats ja auch geklappt. und prüfen muss man ja sowieso.

Beitrag von „Keksfamilie“ vom 7. April 2021, 17:46

So viel Hilfe, so viel zu lesen Fettes Danke euch allen schon mal.

Jetzt komme ich erstmal wieder mit den dummen Fragen:

Ihr schreibt ja davon, dass der Rename von _CRS zu XCRS nicht an der richtigen Stelle ausgeführt wird. Wenn ich mir das dann mal in live am System anschaue, sieht das derzeit so

aus:

```
Scope (_SB.PCI0.LPCB)
{
    Device (HPET)
    {
        Name (_HID, EisaId ("PNP0103") /* HPET System Timer */)  // _HID: Hardware ID
        Name (_UID, Zero)  // _UID: Unique ID
        Name (BUF0, ResourceTemplate ()
        {
            Memory32Fixed (ReadWrite,
                0xFED00000,         // Address Base
                0x00000400,         // Address Length
                _Y41)
        })
        Method (_STA, 0, NotSerialized)  // _STA: Status
        {
            If (HPTE)
            {
                Return (0x0F)
            }

            Return (Zero)
        }

        Method (XCRS, 0, Serialized)
        {
            If (HPTE)
            {
                CreateDWordField (BUF0, \_SB.PCI0.LPCB.HPET._Y41._BAS, HPT0)  // _BAS: Base Address
                HPT0 = HPTB /* \HPTB */
            }

            Return (BUF0) /* \_SB_.PCI0.LPCB.HPET.BUF0 */
        }
    }
```

Sieht für mich als Dummie so aus, als ob das schon renamed wurde? Nach was muss ich denn suchen und dann mithilfe der Skips nach hinten verschieben?

Habe den SSDT-HPET auch mal versucht an MacPeet 's Variante anzugleichen (einfach _CRS durch BUF0 ersetzt), also das Zeug in BUF0 schreiben zu lassen anstatt in _CRS. Aber mit wenig Erfolg, es tat sich am BUF0 genau: nix. Falls ich das alá MacPeet mache, müsste ich aber den rename auch rausnehmen oder?

Soweit ich das verstehe, will ich, dass am Ende beim aufrufen der _CRS Methode die IRQNoFlags rausfallen. Da _CRS in MacPeets Fall einfach das BUF0 Objekt (? sind wir hier in der OOP? :D) returned und das BUF0 die IRQNoFlags hat, funktioniert das.

Der Ansatz von SSDTTime ist es die "alte" _CRS Methode umzubenennen damit die "aus dem Weg" ist und dann seine eigene _CRS Methode einzufügen die direkt die IRQNoFlags besitzt, ohne auf BUF0 zurückzugreifen.

Versteht das mein Gehirn soweit richtig?

Beitrag von „apfelnico“ vom 7. April 2021, 17:50

Das mit dem Zählen klappt. Problematisch finde ich, wenn man mehrere gleiche Einträge an verschiedenen Stellen ändern möchte und dafür einzelne Einträge erstellt. Dann funktionieren diese in "Summe", weil man die vorangegangenen gepatchten im nächsten "Durchlauf" nicht

mehr mitzählen darf, da diese ja nicht mehr enthalten, da bereits geändert. Möchte man jetzt aber einzelne zum testen ein/ausschalten, dann stimmt die darauffolgende Zählung nicht mehr. Besser also, eine absolute Adressierung. Genau dafür sind die neuen zusätzlichen Einträge. Wenn diese nicht funktionieren, dann Fehlerbeschreibung an zum Beispiel mhaeuser

---

Beitrag von „grt“ vom 7. April 2021, 18:00

> Zitat von apfelnico
>
> weil man die vorangegangenen gepatchten im nächsten "Durchlauf" nicht mehr mitzählen darf

man fängt von hinten an. dann passen die zahlen beim nächsten durchgang noch.

Beitrag von „apfelnico“ vom 7. April 2021, 18:02

Das ist schlau. 🙂

Aber die Zählerei hätte ein Ende, wenn man direkt adressieren könnte, wie angedacht …

Beitrag von „grt“ vom 7. April 2021, 18:03

> Zitat von apfelnico
>
> Aber die Zählerei hätte ein Ende, wenn man direkt adressieren könnte, wie angedacht …

womit du recht hast.. 🙂👍

Beitrag von „MacPeet“ vom 7. April 2021, 18:38

echt spanische Dörfer für mich, über was Ihr Euch da unterhaltet, lach

fakt ist aber, der User hat noch immer kein Audio

Als diese neuen Rechner mit dem vorgesetzten Intel-Device vorm eigentlichen Realtek-Chip erstmalig rauskamen, da war der User fewtarius auf InsanelyMac im Thread https://www.insanelymac.com/forum/topic/311293-applealc-—-dynamic-applehda-patching/ der erste User, der die ersten Lösungen dafür gefunden hat. Damals arbeitete er mit FakePCI.kext und hat eine Device dafür gefixt. Inzwischen sind viele Controller-Patches diesbezüglich in AppleALC eingeflossen, von dessen tatsächliche Funktion ich aber keine Kenntnis habe, auf Grund der fehlenden Hardware hier. Vielleicht ist aber dieser Device-Patch noch immer nötig.

Die entsprechenden Beiträge liegen von der letzen Seite aus gesehen, aber etliche Seiten zurück. Vielleicht mal die Suche im Thread verwenden nach "fewtarius".

Vielleicht auch mal fewtarius dort anschreiben!

Beitrag von „grt“ vom 7. April 2021, 18:50

das war jetzt nur der IRQ-fix quasi zu fuss... wenn man die "komfortablen" cloverhäkchen nicht mehr zur verfügung hat, das ssdttimezeugs nicht funktioniert, und ebenso irgendwie die neue OC-funktion zum gezielten umbenennen in der dsdt auch noch nicht richtig tut (ich guck mir das aber noch mal ganz genau an, vielleicht sass der fehler ja auch vor dem rechner..), man aber trotz allem den patch einbauen muss. also die version "ich schnitz mir dann eben selbst zurecht, was ich brauche" 

Beitrag von „MacPeet“ vom 7. April 2021, 19:00

grt

ok, verstehe so halbwegs

übrigens, eine mega Arbeit in dem anderen Thread, was ja zu Audio führte. Hier hat Eure FaceTime-Sitzung mal richtig gefruchtet, prima.

Beitrag von „grt“ vom 7. April 2021, 19:04

und hat auch noch so richtig spass gemacht. war super spannend: recherchieren, überlegen, hoffen, richtig verstanden zu haben, umsetzversuch, erstes kleines erfolgserlebnis.. nächsten schritt recherchieren... und am ende sprach der klapptopf dann tatsächlich. *jubel*

Beitrag von „Keksfamilie“ vom 7. April 2021, 20:41

Wenn Änderungen durch eine SSDT vorgenommen werden, sind die dann auch in der aktuellen System DSDT sichtbar (die, die sich öffnet wenn man MaciASL aufmacht)? Oder tauchen da nur Renames auf.

Beitrag von „MacPeet“ vom 7. April 2021, 20:55

Nein, in der System-DSDT tauchen tatsächlich nur Änderungen auf, wie z.B. die IRQ-Patches von Dir, welche tatsächlich die DSDT selbst verändern.

External-SSDT ändern die DSDT nicht, es sei denn der Code ist explizit so geschrieben, was in der Regel so nicht gemacht wird.

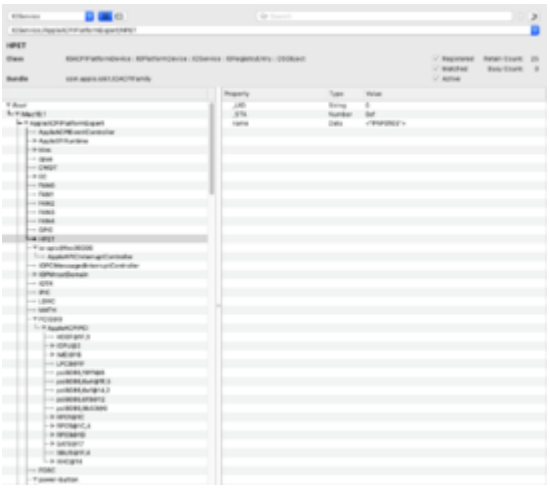
Diese geben dem System Zusatz-Info's, welche aber in der System-SSDT nicht auftauchen.

Daher ist es oft schwer erkennbar, oben eine SSDT nun wirklich fruchtet und das tut, was sie tun soll.

Manchmal hilft hier das Ergebnis vom ioreg (IORegistryExplorer).

Beitrag von „Keksfamilie“ vom 7. April 2021, 21:27

Ah ok. Habe grade gesehen, dass die SSDT-HPET als eigener Table in MaciASL gefunden wird, scheint also zumindest geladen zu werden. Im IORegistryExplorer finde ich das HPET device:



EDIT:

So inzwischen eine ganze Menge schlauer geworden und Audio geht endlich Die HPET fixes brauche ich mit meiner Platform anscheinend nicht und die IRQs waren nicht das Problem. In meinem Fall reicht es, die PCIID für das HDEF device mithilfe von rehabmans FAKEPCIID.kext zu faken. Habe mit CorptNewts Unterstützung einen Injector Kext für FAKEPCIID gebaut der den oft benutzten Rundumschlag von FAKEPCIID + FAKEPCIID_Intel_HDMI_Audio + fake der device-id in den DeviceProperties etwas eleganter mit weniger bloat löst. Werde dazu morgen noch was ausführlich schreiben um das dokumentiert zu haben Aber jetzt gehts erstmal in die Heia für mich

Beitrag von „MacPeet“ vom 8. April 2021, 08:03

Ok, dann war mein Verdacht mit der FakePCI schon richtig. Feine Sache, wenn's jetzt geht.

Auf welche ID bist Du nun gegangen?

Beitrag von „asr10“ vom 8. April 2021, 08:18

Herzlichen Glückwunsch

Keksfamilie hast Du auch Audio über HDMI hinbekommen?

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

So, jetzt kann ich mal in Ruhe tippen

MacPeet 709D0000

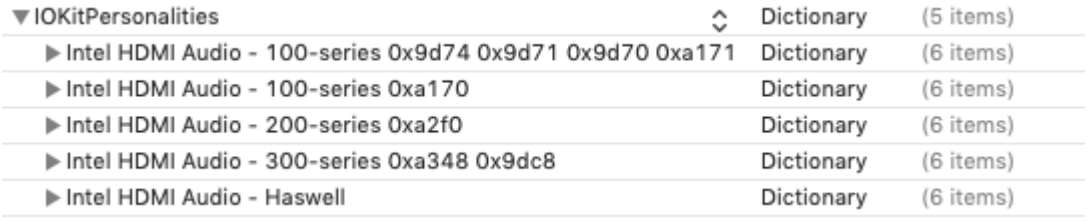
asr10 gute Frage... ich habe leider keinen Monitor mit Lautsprechern oder Kopfhörerausgängen Kanns also überhaupt nicht testen

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>
```

13. <key>FakeProperties</key>
14. <dict>
15. <key>RM,device-id</key>
16. <data>cJ0AAA==</data>
17. </dict>
18. </dict>

Alles anzeigen

und geändert in:

Code

1. <key>Intel CML PCH (device-id c8060000)</key>
2. <dict>
3. <key>CFBundleIdentifier</key>
4. <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.

Beitrag von „asr10“ vom 9. April 2021, 14:47

Klingt sehr interessant. Insbesondere als ich Intel HDMI Audio gelesen habe, wurden meine Augen groß: Mein Problem ist, dass ich zwar unter BigSur von Anfang an Ton an den Ausgängen, aber nicht über HDMI hatte; letzteres erst, wenn ich ein 2. Anzeigegerät angeschlossen habe - erst dann wird ein 2. sound device "Intel Kabylake HDMI" im Hackintool angezeigt.

Da ich auch das Comet Lake PCH cAVS (0x06C8) Problem habe, hatte ich gerade Deinen Weg gerade ausprobiert, leider ohne Erfolg.

Vielleicht - falls Du magst, kannst Du ja mal einen TV über HDMI anschließen und die Tonausgabe überprüfen. Bei nur einer iGPU haben auch andere mit Gigabyte und MSI z490 boards das Problem.

Beitrag von „MacPeet“ vom 9. April 2021, 15:41

> Zitat von Keksfamilie
>
> MacPeet 709D0000

ja, schon klar, dies ist die DeviceID für den Fake, ich meinte aber die dann verwendete layoutID, welche Du mit AppleALC injectest und ob dann alle Geräte des Onboard-Audio's arbeiten.

asr10

Seine Beschreibung bezieht sich aber auf den Fake des Onboard-Audio-Chip's, auch wenn er hier einen Intel HDMI Eintrag im FakePCIID.kext dafür missbraucht. Hatte man natürlich auch

duplizieren können, somit als Ergänzung.

Wenn Du auf dem 2. Anzeigegerät HDMI-Audio bekommst, dann ist dieser Anschluss vermutlich richtig konfiguriert, der erste Anschluss scheinbar nicht, wo Du kein HDMI-Audio bekommst.

Was zeigt Hackintool denn bei Sektion Patch im Tab Connectors ?

Beitrag von „asr10“ vom 9. April 2021, 15:57

MacPeet mein Problem ist hier

Kein Ton über HDMI via iGPU (Asrock z490 PGITX/TB3 u.a.)

Ich war hier gespannt, weil Chipsatz und solo iGPU zu meinem Problem passen und dieses wohl nicht bei Leuten mit dGPU und/oder mehreren Displays auftritt.

Beitrag von „Keksfamilie“ vom 9. April 2021, 16:01

MacPeet ah so  layout-id ist die 100

Es geben alle Ports ton aus/empfangen welchen, ich kann dir leider nicht sagen, ob die ganzen Ports fürs 5.1 Setup korrekt belegt sind, sound kommt auf jeden fall raus.

Beitrag von „MacPeet“ vom 9. April 2021, 16:46

Keksfamilie

prima, aber es beantwortet noch nicht meine Frage, ob alle Geräte (Outputs/Inputs) nun gehen.

asr10

ist auch ein völlig anderer Rechner: Asrock Z490 Onboard-Audio ALC1220 vs. MSI Z490-A Pro Onboard-Audio ALC892

jeder Hersteller kocht seine eigene Suppe

Habe mir Deinen Link zu Deinem Thread angesehen.

Du verwendest Adapter USB-C auf HDMI, die dies vermutlich nicht liefern können. Nicht jeder Adapter ist dafür ausgelegt.

Damit kannst Du die Fehlerquelle nicht finden.

Ferner, unter BigSur zeigt Hackintool selbst auf meinem M1 Mini die Connectors als DP an und nicht als HDMI, obwohl HDMI-Anschluss ja vorhanden ist.

Vielleicht solltest Du die Anschlüsse laut Deinem Bild im Thread mit HDMI mal auf DP patchen, bringt aber nix, wenn der Adapter HDMI-Audio gar nicht unterstützt.

Beitrag von „asr10“ vom 9. April 2021, 17:11

MacPeet lieben Dank für Dein Feedback. Vielleicht sollten wir in meinem thread weiterschreiben, soweit es um Ton via HDMI geht.

Aber um kurz zu antworten: Ja, ALC ist anders, aber sitzt, soweit ich das verstanden habe, doch auch hinter einem Intel Gerät. Ich hab ein normales HDMI Kabel an ein Display geschlossen. Hackintool zeigt nur den ALC an, gleichwie ich den framebuffer konfiguriere. Erst wenn ich eine weitere Videoverbindung dazu schalte - also bei mir ein usb-c zu HDMI Kabel, erscheint ein zweites (Intel Kabylake HDMI) device und ich kann dann sowohl über das HDMI Kabel, also auch über den usb-c Adapter Ton ausgeben. In englischen Foren wird neben dem HDMI Kabel ein zusätzliches DP Kabel benötigt, also auch zwei Anschlüsse. Ich hab zwar kein DP Kabel, gehe aber davon aus, dass es analog zu den anderen Berichten und meinem usb-c Adapter läuft. Ich kann dann auch eine Verbindung kappen und immer noch Ton ausgeben, je nachdem über HDMI oder den Adapter...bis zum nächsten Neustart.

Beitrag von „MacPeet“ vom 9. April 2021, 17:21

asr10

Klingt interessant, hast also genau mit diesem Adapter Erfolg, beim Direktanschluss nicht.

Allerdings ist es unter BigSur aktuell bekannt, dass Apple Probleme hat mit der externen Monitorerkennung, kann ich jeden Tag ein Lied von Singen.

Hat sich bis gestern mit 11.3 DP7 nicht geändert.

Wie ist denn dieses Monitorverhalten unter Catalina? Kann man ja parallel installieren und es würde nicht nicht mal wundern, wenn dort alles geht.

Beitrag von „asr10“ vom 9. April 2021, 17:33

Vielleicht hab ich mich unglücklich ausgedrückt. Es ist ein usb-c zu HDMI Kabel. Also mein z490 Board braucht zwei Verbindungen - entweder an zwei Displays oder wie ich’s nun habe, an 2 HDMI Buchsen an einem Display - um das Intel Kabylake HDMI, also den AppleHDAHDMI_DPDriver zu „triggern“.

Beitrag von „Keksfamilie“ vom 14. April 2021, 21:40

asr10 hatte heute kurz die Möglichkeit das mit einem Monitor der Ton über HDMI macht zu testen: Es taucht kein Audio Device für HDMI auf.

Hatte leider nicht genug Zeit, das genauer zu debuggen und hatte auch noch HDMI -> DVI-D Adapter dazwischen, das find ich auch immer super unsympathisch

Ich schaue, ob ich da nochmal Zugang zu dem Monitor bekomme die nächsten Tage

---

Beitrag von „asr10“ vom 21. April 2021, 19:17

Keksfamilie Das tut mir leid - aber ehrlich nicht wirklich :). Ich hoffe, Du machst gut voran und findest vielleicht eine Lösung, ein HDMI Audio device der iGPU mit nur einer Verbindung zu triggern.

Das dies nichts mit framebuffer etc. zu tun zu haben mag, scheinen nun auch mehr und mehr Leute zu bestätigen:

I have the same problem. I wasn't aware of it until my dGPU died and now I've been forced to use the iGPU for the foreseeable future. My monitor populates on the Audio Devices Output list after I plug it in via both DisplayPort and HDMI. Only one or the other produces no audio.

Same result if I run absolutely nothing but framebuffer + device-id or the full Hackintool patch listed at the beginning of this guide. Totally weird.

i9-10900K + ASRock Z490 Phantom Gaming ITX/TB3