AMD Radeon 6650 XT, 6950 XT und 6900 XTXH ohne GPU-ID Spoofing

Beitrag von „hObelware“ vom 26. Februar 2023, 13:28

Hallo allerseits,

ich stelle hier mal meine Kernel Extension für die eigentlich nicht unterstützten Ausbaustufen von Navi21 (RX 6950 XT und RX 6900 XTXH) und Navi23 (RX 6650 XT) zur Verfügung.

Ein Vorwort

Warum weg vom DeviceID Spoofing? ..

DeviceID Spoofing hat meines Erachtens nach "unsaubere" und sogar "dreckige" Lösungsqualitäten und sollte eigentlich nur als temporärer Workaround in Betracht gezogen werden.

unsauber, weil das GPU-ID Spoofing offensichtlich nicht bis zu letzten (untersten) Ebene greift. Wer mit gespoofter GPU-ID mal im IORegistryExplorer im GFX0 Device den "compatible" Key ansieht, wird dort nur die originale und nicht die gespoofte ID vorfinden. .. Bisher hatte ich zwar noch nie das Problem, daß das relevant wird, .. richtig konsistent ist dieser Zustand aber auch nicht.

dreckig, weil hier Device-unkritisch in die Geräte-/Treiberinstanzierung eingegriffen wird, und mit der Festlegung der DeviceID nach PCI-Pfad alle Geräte in diesem Steckplatz die gleiche DeviceID (und damit den gleichen Teiber) aufgedrückt bekommen, .. ganz egal, was da drin steckt. Die Flexibilität der Hardwarekonfiguration ist damit (zumindest für die Grafikkarte) dahin.

Spätestens wenn man mal - z.B. im Notfall - ne andere Grafikkarte verbauen muß, holt einen das vielleicht wieder ein .. mit Sicherheit aber, wenn das Tauschgerät ne komplett andere Architektur aufweist.

Außerdem finde ich DeviceProperties per config.plist oder SSDT/DSM nur für Geräte angemessen, die man nicht austauschen kann, weil fest aufm Board verbaut (Ethernet, Onboard Audio, etc.) Bei allem, was tauschbar ist, sollte eine dynamische Lösung angestrebt werden ..

Das Ziel

Ein PersonalityProvider, der wie jeder normale Treiber gerätesensitiv die Treiberinstanzen erzeugt, und alle zusätzlichen DeviceProperties on-demand-only als All-In-One Lösung vergibt.

Das Ergebnis

Ursprünglich hatte ich das Projekt nur für die RX 6650 XT umgesetzt, da das Konzept im Grunde aber für jedes Device funktioniert, dass über ID Spoofing läuft (nicht nur Grafik), habe ich diese Kext mal auf alle mir bekannten, zu spoofenden Navi20 erweitert.

Die IOPersonalitiy zu erzeugen ist ja nicht sonderlich kompliziert und bei GitHub habe ich am Wochenende noch eine Lösung gefunden, die DeviceProperties via IOService zu übergeben (vgl. hier). Ich nutze eine leicht abgewandelte Version diese Codes, der Credit hierfür gebührt in jedem Falle Voight-Kampf. .. Danke

Installation/ggf. Anpassungen

Die Kernel Extension muß wie jede andere auch in Kernel->Add eingebunden werden.

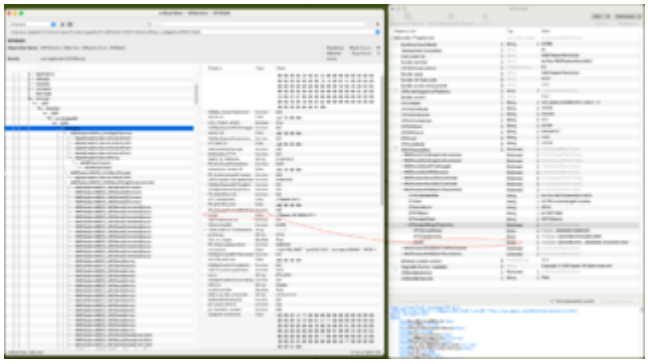
Mindestvoraussetzung ist Navi23 Support, also macOS 12.1 ..

Die Bridge SSDT(SSDT-BRG0.aml oder vergleichbares) ist nicht mehr nötig (zumindest hier unter macOS 13.2.1 .. gerne testen und hier Bescheid geben ..) und alle DeviceProperties für die Navi20 .. bei mir: PciRoot(0x0)/Pci(0x1,0x0)/Pci(0x0,0x0)/Pci(0x0,0x0)/Pci(0x0,0x0) .. müssen auch weg (wir wollen ja explizit davon weg).

Unmodifiziert gibt die Kext nur die Properties model (Graka-Name), ATY,FamilyName und ATY,DeviceName (die letzteren für MetalDeviceName) weiter. Ich habe bei model übrigens "AMD" weggelassen, damit bei "über diesen Mac" kein Zweizeiler draus wird, wer will kann das je wieder hinzufügen.

Wenn noch FramebuffersEdits, PowerPlayTables oder ggf. weitere Kosmetika etc. mitgegeben werden sollen, können diese in der info.plist unter IOKitpersonalities->entsprechendes Root Device->IOProviderMergeProperties als Kind-Elemente eingetragen werden.

Achtung!: Hierbei müssen die erwarteten Datentypen passen (meist Data, .. vermute ich) .. am besten vorher mit gespoofter ID mit IORegistryExplorer das Device GFX0 abgleichen. Strings sind z.B. Null-terminated in Mac Roman Encoding zu übergeben (vgl. hier)



fertig .. echter Device Support bei vollständiger Flexibilität!

Zum Testen habe ich hier nur macOS 13.2.1, da funktioniert es mit WhateverGreen /agdpmod=pikera bislang ohne Einschränkungen

EDIT 07.04.23

DRM Decoder funktioniert mit dieser Lösung leider nicht. Das ist mir bedauerlicherweise nicht aufgefallen, da Trailer allgemein funktionieren - HD Content aber nicht .. und damit (inkl. Netflix) beriesele ich mich eigentlich nur übers Apple TV am Fernseher.

Mein Plan ist das Ganze nochmal als .dext (DriverExtension) anzusetzen, so wie Apple das vorsieht. Leider weiss ich noch nicht genau, wann ich Zeit dafür finde und wie das mit den Developerlizenzen dafür aussehen muß, damit das auch überall geht (und nicht nur lokal bei mir).