Geräte Eigenschaften (Device Properties) ohne DSDT Patch ändern.

Beitrag von „Brumbaer“ vom 12. Dezember 2017, 14:28

Nichts währt ewig
Die Vega lief OOB von Anfang an. Sie brauchte keine Hilfs-Karte, Sound funktionierte und sogar Sleep ging ohne Patches.
Die Performance war allerdings schlechter als erwartet und die OpenGL Implementierung lies eine Menge Raum für Verbesserungen.
Seit der ersten 10.13.2 Beta waren Leistung und OpenGL Unterstützung zufrieden stellend, aber der Sleep Modus ging nicht mehr.
Jetzt mit der 10.13.3 Beta 1 geht der Sleep Modus wieder, wenn auch nur mit bestimmten Framebuffern.
Dies ist der Anlass eine alternative Methode zum Verändern von Geräteeigenschaften zu beschreiben.

Häh ?
Die Hardware des Hacks besteht aus vielen Geräten (Devices) CPU, PCIe Karten, USB Ports, Kartenleser usw. Manche Geräte haben wieder Untergeräte.
Alle Geräte, die das System kennt werden in der IORegistry zusammengefasst. Jedes Gerät hat Eigenschaften (Properties), manche tauchen bei verschiedenen Geräten auf und manche sind gerätespezifisch.

Aaah
Welcher und wieviele Framebuffer verwendet werden wird anhand solcher Geräteeigenschaften bestimmt. Diese spezielle Eigenschaft findet man natürlich nur bei Graphikgeräten.

Ja, wo kommst du denn her ?
Einige Geräteeigenschaften setzt der Geräte Treiber andere stammen aus der DSDT oder werden per Software gesetzt.
Clover z.B. kann für einige Geräte bestimmte Eigenschaften und für manche Geräte beliebige Eigenschaften setzen.
Man kann Eigenschaften auch über die DSDT oder eine SSDT setzen - vorausgesetzt das Gerät taucht dort auf.
Die Eigenschaften setzt man in der _DSM Methode des Gerätes.

Für die Grafikkarte sähe das so aus:

Code

```
Device (PEGP)
{
Name (_ADR, Zero) // _ADR: Address
Device (LTRE)
{
Name (_ADR, Zero) // _ADR: Address
Device (GFX0)
{
Name (_ADR, Zero) // _ADR: Address
Method (_DSM, 4, NotSerialized) // _DSM: Device-Specific Method
{
If (LEqual (Arg2, Zero))
{
Return (Buffer (One)
{
0x03
})
}


Return (Package (0x10)
{
"AAPL,slot-name",
Buffer (0x07)
{
"Slot-1"
},


"@0,name",
Buffer (0x0D)
{
"ATY,Kamarang"
},


"@1,name",
Buffer (0x0D)
```

```
{
"ATY,Kamarang"
},


"@2,name",
Buffer (0x0D)
{
"ATY,Kamarang"
},


"@3,name",
Buffer (0x0D)
{
"ATY,Kamarang"
},


"device_type",
Buffer (0x13)
{
"ATY,KamarangParent"
},


"model",
Buffer (0x13)
{
"Radeon Pro Vega 64"
},


"hda-gfx",
Buffer (0x0A)
{
"onboard-2"
}
})
}
}
}
}
```

```
82.  }
```

Alles anzeigen

Hier werden 4 Ausgänge festgelegt, die jeweils den ATY, Kamarang Framebuffer verwenden.
Zusätzlich werden Slotname, Modelbezeichnung und Soundausgaabe festgelegt.

Immer *d*ie *s*elbe *d*röge *T*abelle
Ich ändere ungerne die DSDT.
Uns liegen die DSDT Quellen nicht vor, sondern wir verwenden ein Tool (MaciASL), dass aus dem vorliegenden Code die Quellen erzeugt.
Das funktioniert nicht perfekt und man verschwendet erst einmal einige Zeit damit die Umwandlungsfehler zu beseitigen.
Oft werden einfach Patches angewandt ob man sie braucht oder nicht und die Patches überschreiben oft was existiert ohne Teile, die man vielleicht braucht zu erhalten. Kürzlich kam ich zu einem System bei dem USB nicht funktionierte und es stellte sich heraus, dass die DSDT zwei verschiedene USB Chipsets unterstützt und die Auswahl des richtigen Chipsatzes aud der DSDT entfernt worden war.

Ohne vernünftige Dokumentation fängt man beim Wechsel des MoBos von vorne an - nachdem man erstmal die ganzen Patches zusammengesucht hat.
Es gibt aber auch Gründe die DSDT Patch Methode anderen vorzuziehen und wer glücklich damit ist soll auch dabei bleiben.

Alternative Liste
Die hier vorgestellte Alternative ist ein Kext, dass die Properties aus einer Liste kopiert und einträgt.
[Edit]
Es sei nicht verschwiegen, dass es Fälle gibt in denen die Methode im Gegensatz zum Patchen der DSDT nicht funktioniert. Ein Beispiel sind FakeIDs[/Edit]

Man editiert die Info.plist des Kext und kopiert es in den EFI/CLOVER/kexts/Other Ordner der EFI Partition.

Zum Editieren der Info.plist sollte man XCode verwenden, es macht das Leben einfacher.

Nicht schon wieder Kexte

Aufbau und prinzipielle Funktionsweise eines Kextes sind in Kext as Kext Can beschrieben.

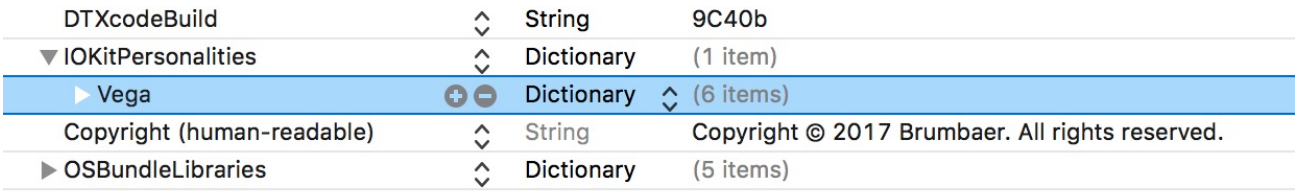
Das Kext heißt PropertyInjector.kext. Es enthält eigenen Programmcode.
Wenn man dessen Info.plist in XCode öffnet sieht man:

Key	Type	Value
▼ Information Property List	Dictionary	(20 items)
BuildMachineOSBuild	String	17C88
Localization native development re...	String	en
Executable file	String	PropertyInjector
Bundle identifier	String	de.Brumbaer.PropertyInjector
InfoDictionary version	String	6.0
Bundle name	String	PropertyInjector
Bundle OS Type code	String	KEXT
Bundle versions string, short	String	1.0
▶ CFBundleSupportedPlatforms	Array	(1 item)
Bundle version	String	1
DTCompiler	String	com.apple.compilers.llvm.clang.1_0
DTPlatformBuild	String	9C40b
DTPlatformVersion	String	GM
DTSDKBuild	String	17C76
DTSDKName	String	macosx10.13
DTXcode	String	0920
DTXcodeBuild	String	9C40b
▶ IOKitPersonalities	Dictionary	(3 items)
Copyright (human-readable)	String	Copyright © 2017 Brumbaer. All rights reserved.
▶ OSBundleLibraries	Dictionary	(5 items)

Bis auf IOKitPersonalities braucht uns das Alles nicht zu kümmern.
Im Folgenden wird deshalb nur noch der Ausschnitt mit den IOKitPersonalities gezeigt.

Man legt für jedes Gerät für das man Properties ändern oder hinzufügen will eine Personality an.

Ich will die Framebuffer meiner Vega ändern also nenne ich die Personality ... Vega, jeder andere Name hätte es auch getan, aber ich will ja ohne viel nachzudenken wissen worum es geht.



Jede Personality muss wissen welches Programmstück ausgeführt werden soll deshalb bekommt sie Einträge für IOClass und CFBundleIdentifier.

Die Werte müssen die dort angegebenen sein.

Als nächstes müssen wir das Gerät identifizieren, dessen Properties ergänzt oder geändert werden sollen.

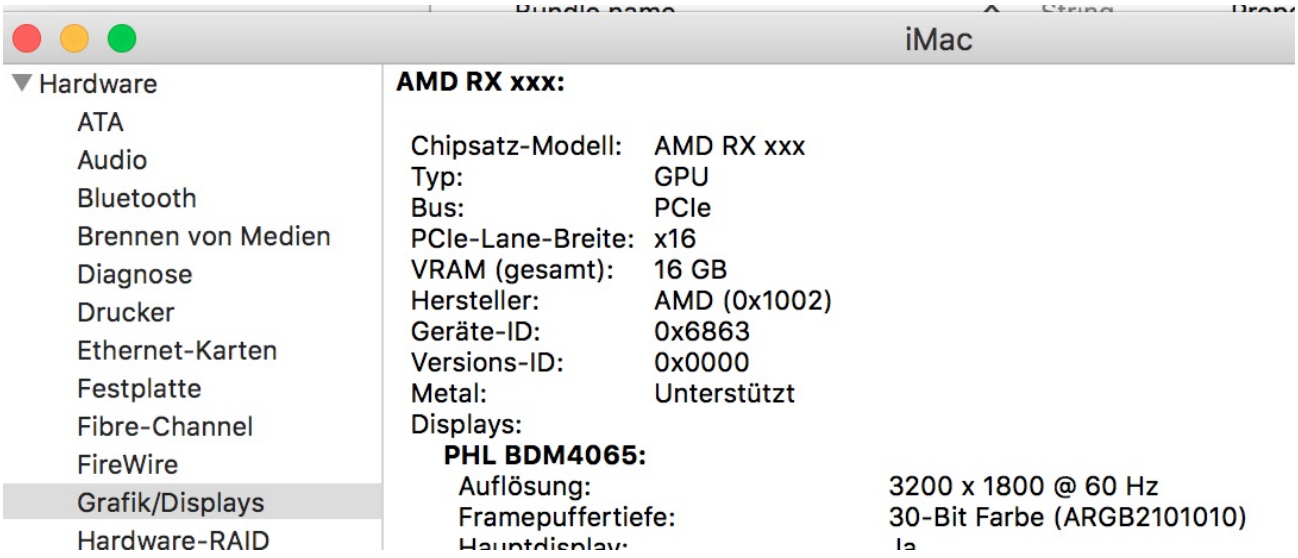
Nichts ist unmöglich

Es gibt eine Reihe von Parametern die man dazu verwenden kann und hier liegt ein Vorteil dieser Methode Properties zu editieren, man kann sie nicht nur auf Geräte, die in der DSDT vorkommen anwenden, sondern alle Geräte, die in der IORegistry auftauchen, wie z.B. USB Geräte.
Die Optionen zur Gerätewahl wären einen eigenen Artikel Wert - ein andermal.
Wir beschränken uns jetzt hier auf die Auswahl eines bestimmten PCIe Gerätes.

▼ IOKitPersonalities	Dictionary	(1 item)
▼ Vega	Dictionary	(4 items)
CFBundleIdentifier	String	de.Brumbaer.PropertyInjector
IOClass	String	PropertyInjector
IOProviderClass	String	IOPCIDevice
IOPCIPrimaryMatch	String	0x68631002
Copyright (human-readable)	String	Copyright © 2017 Brumbaer. All rights reserved.
▶ OSBundleLibraries	Dictionary	(5 items)

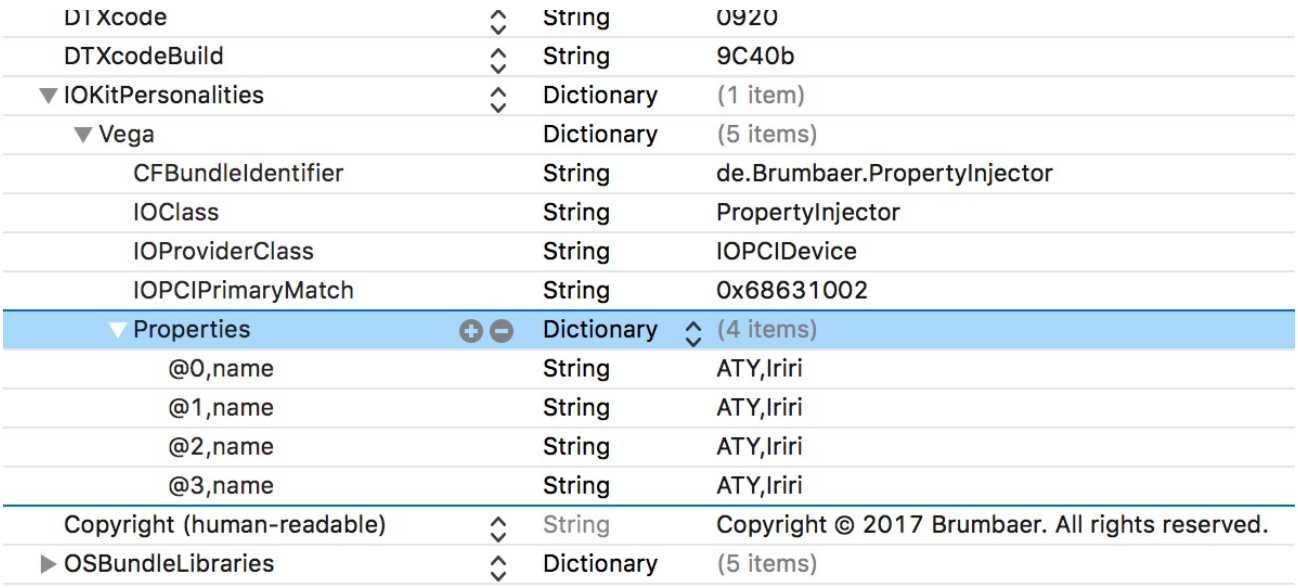
In diesem Fall wollen die Werte für ein PCI Gerät ändern, deshalb die IOProviderClass IOPCIDevice.
IOPCIPrimaryMatch sagt uns mittels Device und Vendor Id welches PCI Gerät gepatched werden soll..

Es gibt verschiedene Möglichkeiten diese herauszufinden.
Im Systembericht unter Grafik/Displays zum Beispiel



GeräteId 0x6863 und Hersteller 0x1002 zusammengefasst zu einem Wert 0x68631002.

Eigenes Schaffen
Jetzt fehlen nur noch die Properties.
Die Properties werden in ein Dictionary mit Namen Properties geschrieben.



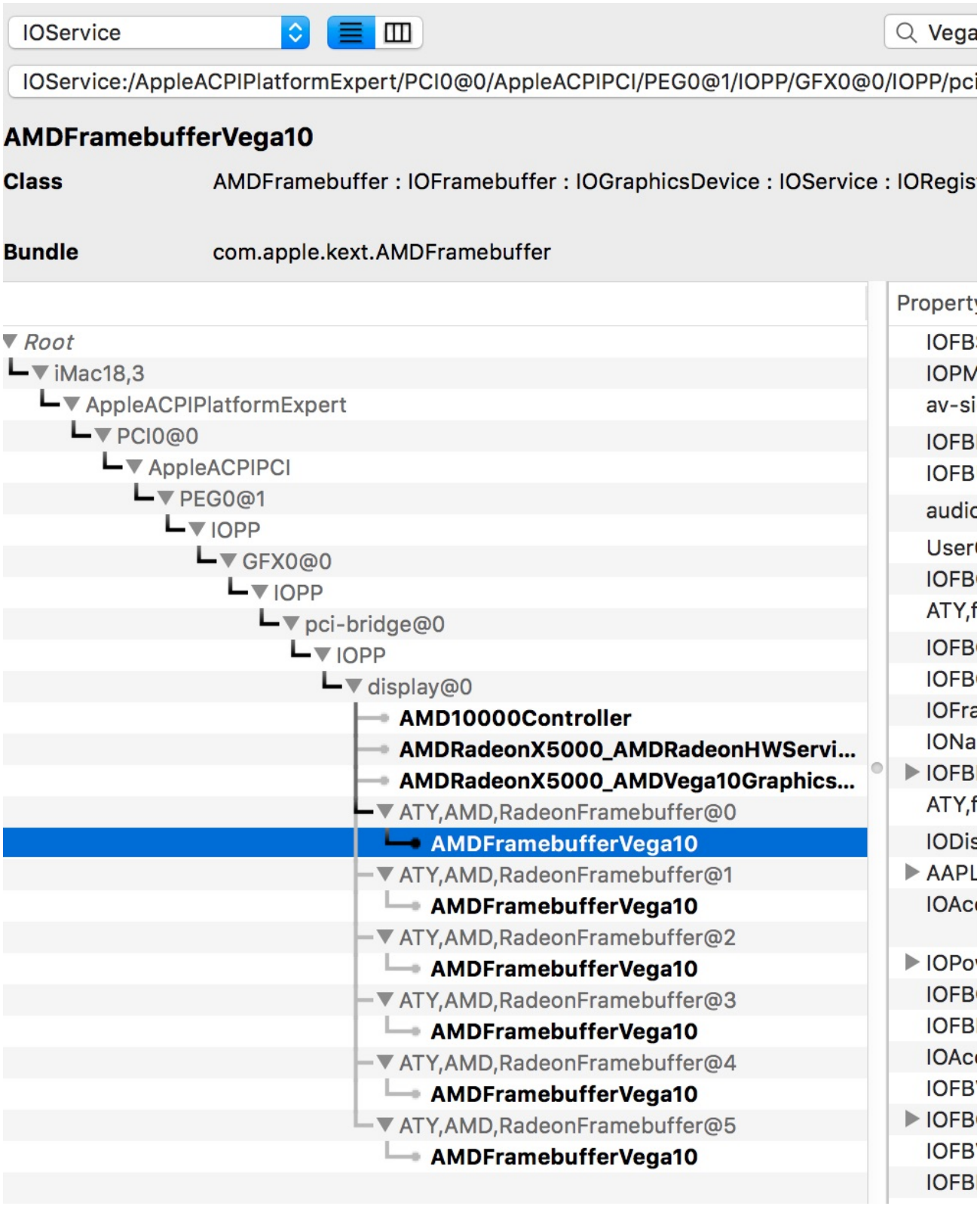
DTXcode	String	0920
DTXcodeBuild	String	9C40b
▼ IOKitPersonalities	Dictionary	(1 item)
▼ Vega	Dictionary	(5 items)
CFBundleIdentifier	String	de.Brumbaer.PropertyInjector
IOClass	String	PropertyInjector
IOProviderClass	String	IOPCIDevice
IOPCIPrimaryMatch	String	0x68631002
▼ Properties	Dictionary	(4 items)
@0,name	String	ATY,Iriri
@1,name	String	ATY,Iriri
@2,name	String	ATY,Iriri
@3,name	String	ATY,Iriri
Copyright (human-readable)	String	Copyright © 2017 Brumbaer. All rights reserved.
▶ OSBundleLibraries	Dictionary	(5 items)

Dass die Properties @0,name bis @3,name heißen haben wir oben festgestellt. Der Framebuffer heißt ATY,Iriri und ist vom Typ String.

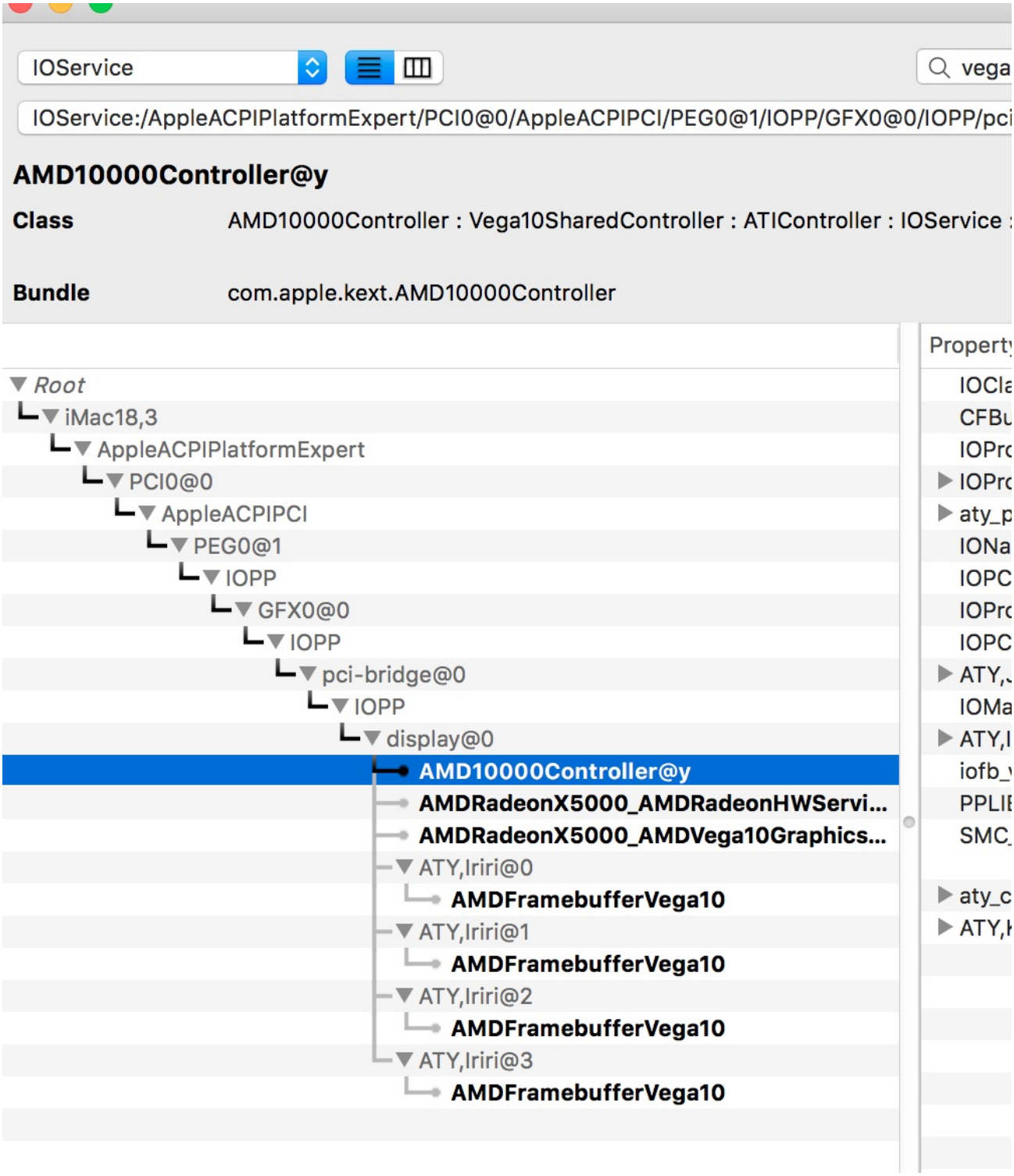
Neustart und fertig.

Alles roger ?

Mal sehen. Startet man den IORegistryExplorer und sucht nach Vega sieht man vor der Änderung 6 Framebuffer vom Typ ATY,AMD,RadeonFramebuffer:



Und nach der Änderung 4 vom Typ ATY,Iriri:



Das soll es schon gewesen sein ?
Weiter oben sieht man, dass die Karte als AMD RX xx erscheint.

Auch das kann man mit einem Property ändern.

DTXcodeBuild	String	9C40b
▼ IOKitPersonalities	Dictionary	(1 item)
▼ Vega	Dictionary	(5 items)
CFBundleIdentifier	String	de.Brumbaer.PropertyInjector
IOClass	String	PropertyInjector
IOProviderClass	String	IOPCIDevice
IOPCIPrimaryMatch	String	0x68631002
▼ Properties	Dictionary	(5 items)
model	Data	Radeon Vega Frontier
@0,name	String	ATY,Iriri
@1,name	String	ATY,Iriri
@2,name	String	ATY,Iriri
@3,name	String	ATY,Iriri
Copyright (human-readable)	String	Copyright © 2017 Brumbaer. All rights reserved.
▶ OSBundleLibraries	Dictionary	(5 items)

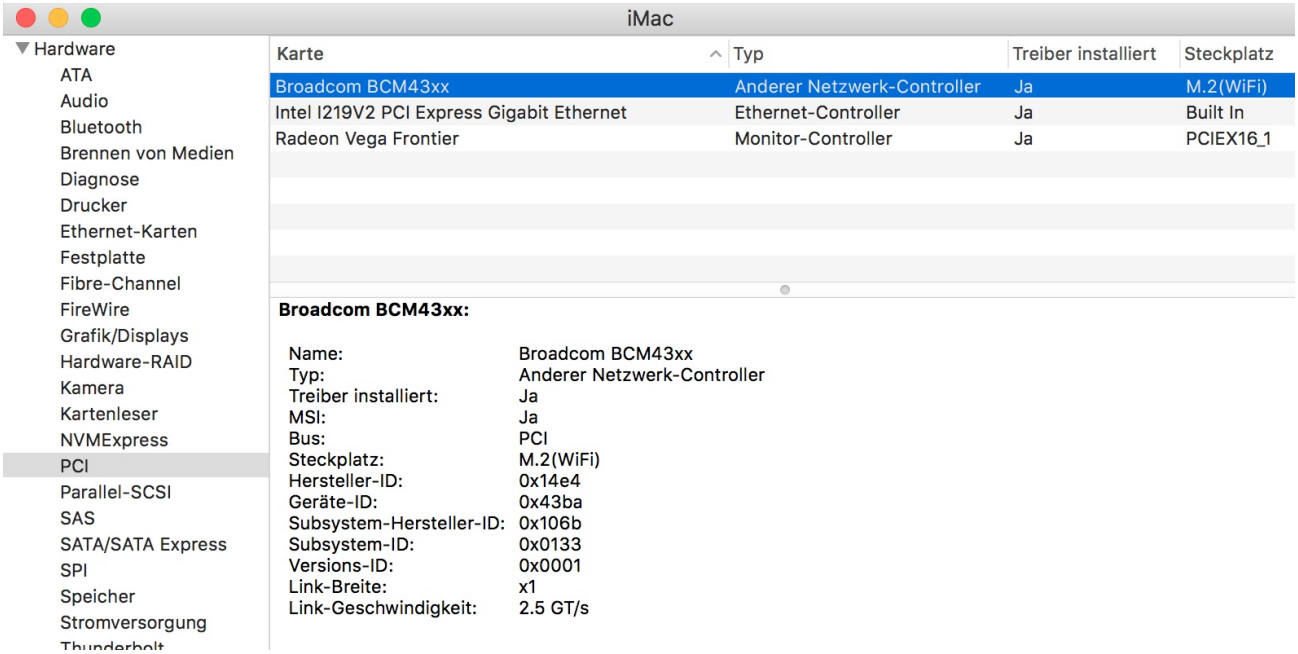
Dann erscheint die Grafikkarte als

War's das jetzt ?
Mmmmh, fast.

Im Systembericht erscheint unter PCI nur eine Fehlermeldung.
Möchte man, dass Geräte erscheinen benötigen sie einen AAPL,slot-name, name und model Eintrag. Je nach Gerät ist der eine oder andere schon definiert.
Wichtig ist, dass diese Einträge vom Typ Data sind.
Hier ist das Kext mit dem Framebuffer Patch, den Vega Frontier Patch und den "PCI Patches" für LAN, WLAN und Vega.
Da LAN und WLAN andere PCI Geräte sind benötigen sie eigene Einträge unter IOKitPersonalities.

DTXcode	String	0920
DTXcodeBuild	String	9C40b
▼ IOKitPersonalities	Dictionary	(3 items)
▼ LAN	Dictionary	(5 items)
CFBundleIdentifier	String	de.Brumbaer.PropertyInjector
IOClass	String	PropertyInjector
IOPCIPrimaryMatch	String	0x15b88086
IOProviderClass	String	IOPCIDevice
▼ Properties	Dictionary	(2 items)
AAPL,slot-name	Data	Built In
name	Data	Intel I219V2
▼ WLAN	Dictionary	(5 items)
CFBundleIdentifier	String	de.Brumbaer.PropertyInjector
IOClass	String	PropertyInjector
IOPCIPrimaryMatch	String	0x43ba14e4
IOProviderClass	String	IOPCIDevice
▼ Properties	Dictionary	(3 items)
model	Data	Broadcom BCM43xx
AAPL,slot-name	Data	M.2(WiFi)
name	Data	Broadcom BCM43xx
▼ Vega	Dictionary	(5 items)
CFBundleIdentifier	String	de.Brumbaer.PropertyInjector
IOClass	String	PropertyInjector
IOPCIPrimaryMatch	String	0x68631002
IOProviderClass	String	IOPCIDevice
▼ Properties	Dictionary	(6 items)
AAPL,slot-name	Data	PCIEX16_1
model	Data	Radeon Vega Frontier
@0,name	String	ATY,Iriri
@1,name	String	ATY,Iriri
@2,name	String	ATY,Iriri
@3,name	String	ATY,Iriri
Copyright (human-readable)	String	Copyright © 2017 Brumbaer. All rights reserved.
▶ OSBundleLibraries	Dictionary	(5 items)

Das sieht dann im Systembericht so aus:



Is gut jetzt, lass es
Ok, das war's.

Falls Fragen bitte nur zur Methode, nicht welches Property wozu dient oder welches Property welches Feature beim Laptop xyz frei schaltet.
Solche Fragen gehören in den jeweiligen Problem Thread.

P.S.
Das Beispiel Kext
PropertyInjector.zip
Für eigene Projekte die IOKitPersonalities löschen, ändern oder ergänzen.