

Erledigt

AMD HD 7970 - Treibermysterien des OSX

Beitrag von „Griven“ vom 6. Mai 2015, 00:31

Puh, so nun das versprochene Update. Vorab sei gesagt es wird ein wenig mehr Text und geht ein bisschen mehr in die Tiefe...

Ich habe mich gestern Abend und auch Heute noch intensiv mit dem Thema beschäftigt und einige Interessante Dinge über OS-X und dessen Umgang mit dem GraphicsPowerManagement sowie der Vergleichbarkeit von Benchmark Ergebnissen zwischen OS-X und Windows gelernt. Fast überall wo es um das Thema geht steht zu lesen, dass Benchmark Ergebnisse zwischen OS-X und anderen Plattformen nur bedingt miteinander vergleichbar sind was in der Eigenart begründet liegt, dass bei OS-X VSYNC prinzipiell aktiv ist. Apple nutzt sein QuartzExtreme Framework auch massiv für die Darstellung der GUI und wendet hier zum Rendern im großen und ganzen die selben Techniken an, wie sie zum Beispiel auch in Spielen oder Benchmarks Verwendung finden. Um nun zu verhindern, dass es innerhalb der GUI hässliche Tearing Effekte gibt aktiviert der WindowsServer von OS-X VSYNC also per default. Aber was bedeutet dies nun?

VSYNC ist eine Technik, welche die Bildwiederholrate der GPU mit der des Monitors synchronisiert. Sprich wenn ein Monitor eine Wiederholrate zwischen 60 und 75hz hat (typischer Wert für LCD´s) berechnet die Grafikkarte auch nur 60 bis 75 Frames/Sek. sofern sie schneller ist als die Wiederholrate des Monitors. Das Ergebnis ist ein schlechterer Benchmark Wert da ja maximal so viele Frames erreicht werden wie sie der Monitor auch darstellen kann und im Gegenzug erhält man eine kühlere GPU und geringeren Stromverbrauch da ja weniger berechnet werden muss.

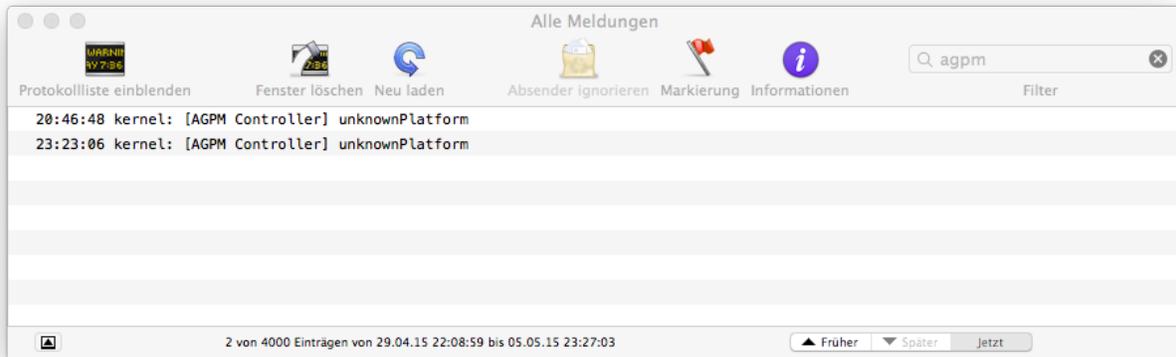
Das Ganze gilt im Übrigen nur, wenn es darum geht auch etwas auf dem Monitor darzustellen. Geht es darum die GPU zum Rendern zum Beispiel in FinalCut zu benutzen lässt sich für diese Tasks ohne direkte Ausgabe auf dem Bildschirm VSYNC gezielt deaktivieren. Hier kann die GPU nun also mit ihrer vollen Power zu Werke gehen und genau das ist auch der Punkt an dem das AGPM ins Spiel kommt denn bei allen einigermaßen aktuellen GPU´s (meine HD5570 Möhre lasse ich da mal ra ) dürfte die Power je nach Detailstufe und Qualität auch ohne Taktänderungen ausreichen um die geforderten 60 Frames zu rendern...

Sucht man im Internet mal nach AGPM Patch wird man reichlich fündig allerdings bezieht sich der allergrößte Teil von dem was man findet auf NVIDIA Karten die AMD Fraktion steht hier

ziemlich im Regen sprich so richtig schlüssiges zum AGPM und AMD Karten findet man nicht wirklich. Natürlich habe ich mich gefragt warum ist das so und die Antwort ist eigentlich ziemlich einfach die AMD Karten regeln diese Dinge ganz offensichtlich über den jeweiligen framebuffer sprich alle nötigen Informationen sind dort schon enthalten man muss der AppleGraphicsPowerManagent.kext also nur noch bebiegen wie die Karte zu handhaben ist. Wie und ob der Kext überhaupt geladen wird hängt sehr von der gewählten SystemDefinition ab und von der Art wie die AMD Karte dem System "nahegebracht" wurde. Schaut man sich die Info.plist in der AGPM.kext an findet man eine ziemlich lange Liste von Maschinentypen:

▼ IOKitPersonalities	Dictionary	1 Schlüssel/Wert-Paare
▼ AGPM	Dictionary	5 Schlüssel/Wert-Paare
CFBundleIdentifier	String	com.apple.driver.AGPM
IOClass	String	AGPMController
IONameMatch	String	AGPMEnabler
IOProviderClass	String	IOPlatformPluginDevice
▼ Machines	Dictionary	65 Schlüssel/Wert-Paare
▶ Mac-00BE6ED71E35EB86	Dictionary	5 Schlüssel/Wert-Paare
▶ Mac-031AEE4D24BFF0B1	Dictionary	1 Schlüssel/Wert-Paare
▶ Mac-031B6874CF7F642A	Dictionary	1 Schlüssel/Wert-Paare
▶ Mac-189A3D4F975D5FFC	Dictionary	1 Schlüssel/Wert-Paare
▶ Mac-27ADBB7B4CEE8E61	Dictionary	5 Schlüssel/Wert-Paare
▶ Mac-2BD1B31983FE1663	Dictionary	2 Schlüssel/Wert-Paare
▶ Mac-2E6FAB96566FE58C	Dictionary	1 Schlüssel/Wert-Paare
▶ Mac-35C1E88140C3E6CF	Dictionary	1 Schlüssel/Wert-Paare
▶ Mac-35C5E08120C7EEAF	Dictionary	1 Schlüssel/Wert-Paare
▶ Mac-3CBD00234E554E41	Dictionary	1 Schlüssel/Wert-Paare
▶ Mac-42FD25EABCABB274	Dictionary	2 Schlüssel/Wert-Paare

Apple unterscheidet hier zwischen neueren und älteren Maschinen. die älteren Maschinen stehen noch mit dem Klarnamen da, die neueren nur noch mit dem Produktidentifizier zusätzlich zur Unterscheidung in der Namensgebung ändert sich auch die Art und Weise wie AGPM angesteuert wird was uns aber jetzt erstmal nicht kümmern muss, da die meisten von uns doch eine der aktuelleren SystemDefinitionen verwenden dürften. In meinem Fall wäre das zum Beispiel ein iMAC 13.1 was dem Identifizier Mac-00BE6ED71E35EB86 entspricht (einfach im CloverConfigurator in der Sektion SMBIOS nachgesehen 😊) für die Mülltonne wäre es Mac-F60DEB81FF30ACF6. Ob Ihr nun überhaupt tätig werden müsst hängt davon ab ob AGPM mit der Systemdefinition und der eingesetzten Grafikkarte was anfangen kann. Prüfen kann man das einfach indem man sich mal die Konsole öffnet und oben im Filter AGPM eingibt falls das Ergebnis so aussieht ist Handlungsbedarf gegeben:



[AGPM Controller] unknownPlatform bedeutet, dass AGPM nichts mit Eurer Kombination anfangen kann im schlimmsten Fall kann das bedeuten, dass Eure GPU irgendwo im tiefsten IDLE rum dümpelt was zwar unfassbar Energieeffizient ist allerdings auch alles andere als performant. Seht Ihr gar nichts aber die AGPM ist geladen, dann braucht Ihr eigentlich nichts weiter tun, AGPM ist aktiv und taktet die Karte je nach Bedarf hoch oder runter. Gut, aber was tun, wenn es nicht wirklich läuft? Wenn es nicht wirklich laufen will solltet Ihr zunächst mal eure gewählte SystemDefinition überprüfen und nachsehen welche Geräte eigentlich für AGPM in der gewählten Definition zur Verfügung stehen. Hierzu eignet sich ein plist Editor bestens. In meinem Fall (iMac 13.2) sieht das Ganze so aus (der Inhalt der Key´s ist ausgeblendet):

Code

1. `<key>Mac-00BE6ED71E35EB86</key>`
2. `<dict>`
3. `<key>GFX0</key>`
4. `<key>IGPU</key>`
5. `<key>Vendor10deDevice0fd8</key>`
6. `<key>Vendor10deDevice0fd5</key>`
7. `</dict>`

Gut zu erkennen ist, dass neben den beiden Standard Geräten (GFX0,IGPU) noch 2 NVIDIA Karten definiert sind anders sieht es bei der Mülltonne aus:

Code

1. <key>Mac-F60DEB81FF30ACF6</key>
2. <dict>
3. <key>GFX1</key>
4. <key>GFX2</key>
5. </dict>

Ihr fehlt das Gerät GFX0 was für unsere Zwecke eher suboptimal ist denn ein schneller Blick in den IORegExplorer zeigt uns doch recht deutlich, dass sich die Grafikkarte im Hackintosh als GFX0 ausgibt sofern es keine iGPU ist das Resultat ist AGPM lädt entweder gar nicht oder aber meldet "unknown Plattform". Das Ganze lässt sich aber umschiffen indem man den Eintrag an seine Gegebenheiten anpasst. Wie oben schon geschrieben benötigen wir eigentlich nur den passenden Framebuffer (-> AMD Karte muss korrekt erkannt werden, lässt sich unter Systemprofiler->PCI prüfen) bei mir sieht es so aus:

Code

1. ATI Radeon HD 5570 Series:
2. Name: ATY,Eulemur
3. Typ: Monitor-Controller
4. Treiber installiert: Ja
5. MSI: Ja

Steht da bei Euch nur ATY,Framebuffer oder gar "Es wurden keine PCI Geräte gefunden" müsst Ihr dringend erstmal Eure device Injection auf die Reihe kriegen denn aktuell läuft Eure AMD Graka mit einem generischen Treiber ohne jedes PowerManagement. Sieht es bei Euch so ähnlich aus, wie bei mir kann es jetzt ans Eingemachte gehen. Dazu wird nun die info.plist der AGPM bearbeitet. Wenn wir uns die bestehenden Einträge der Mülltonne ansehen, dann sehen wir das hier 2 Grafikkarten definiert sind (nicht verwunderlich die Tonne hat 2 GPU's) jede Karte besitzt eine eigene ControlID (16 und 17) die dem AGPM sagt wie die Karte zu steppen ist. Vergleichen wir diese Einträge nun mit dem Gerät GFX0 zum Beispiel aus dem iMac 13.1 fällt auf, dass auch hier die ControlID 17 zum Einsatz kommt und genau hier können wir ansetzen. Als erstes löschen wir mal die beiden vorhandenen Einträge aus dem MacPro 6.1 Profil und dann kopieren wir uns das GFX0 Gerät aus dem iMac 13.1 Eintrag. Ist alles erledigt sollte das jetzt in etwa so aussehen:

Code

1.]<key>Mac-F60DEB81FF30ACF6</key>
2. <key>GFX0</key>
3. <dict>
4. <key>Heuristic</key>
5. <dict>
6. <key>ID</key>

7. `<integer>4</integer>`
8. `</dict>`
9. `<key>control-id</key>`
10. `<integer>17</integer>`
11. `</dict>`

Alles anzeigen

Damit ist erstmal der Grundstein dafür gelegt, dass das GFX0 Device angesprochen wird. Im nächsten Schritt fügen wir jetzt noch unsere Grafikkarte hinzu, die erledigen wir mit Hilfe der Device und Vendor ID. In meinem Fall sieht das so aus:

Code

1. `<key>Vendor1002Device68d9</key>`
2. `<dict>`
3. `<key>LogControl</key>`
4. `<integer>1</integer>`
5. `<key>control-id</key>`
6. `<integer>17</integer>`
7. `</dict>`

Dieser Eintrag sorgt dafür, dass AGPM Eure Karte richtig erkennt und nach der unter der control-id im jeweiligen Frambuffer hinterlegten Regeln taktet. Hat alles geklappt sieht das Ergebnis dann so aus:



Fazit: viel Aufwand für wenig Effekt zumindest bei meiner Karte denn ein mehr an Leistung spüre ich nicht wirklich dafür aber ein weniger an warm sprich die Karte geht im idle deutlich runter (leider liest der Sensor bei meiner GPU die Taktraten nicht, aber man sieht es dann doch recht deutlich an der Temperatur). Ich denke der X-Bench Wert spiegelt so ziemlich das wieder, was meine Karte auch zu leisten vermag:

Code

1. Results 84.37
2. System Info
3. Xbench Version 1.3
4. System Version 10.10.4 (14E11f)
5. Physical RAM 8192 MB
6. Model iMac13,1
7. Drive Type OCZ-AGILITY3
8. Quartz Graphics Test 338.78
9. Line 364.13 24.24 Klines/sec [50% alpha]
10. Rectangle 313.19 93.50 Krechts/sec [50% alpha]
11. Circle 271.11 22.10 Kcircles/sec [50% alpha]
12. Bezier 263.66 6.65 Kbeziers/sec [50% alpha]
13. Text 747.18 46.74 Kchars/sec
14. OpenGL Graphics Test 240.04
15. Spinning Squares 240.04 304.50 frames/sec
16. User Interface Test 35.16
17. Elements 35.16 161.38 refresh/sec

Alles anzeigen