HowTo: Thunderbolt HotPlug/HotSwap Finetuning für euren Hackintosh

Beitrag von „Mork vom Ork“ vom 26. August 2019, 22:01

Angefixt durch *diesen Beitrag* hier von Alex, habe ich mich die letzten 3 Tage mal nur um das Thema "Thunderbolt HotPlug" gekümmert.

Wie viele Andere (hier als auch in div. anderen Foren) auch, bin ich ebenfalls im Besitz einer Gigabyte GC Titan Ridge PCIe-Karte, welche in meinem ASRock Professional Gaming i7 in Steckplatz PCIe #5 steckt.

Bekanntlich verfügt die Karte über 2 TB3/USB-C Anschlüsse und angeschlossene TB3-Devices werden beim hochfahren des Rechners auch ordnungsgemäß erkannt. Zwar werden diese nicht, wie bei einem echten Mac,

im Bereich "Thunderbolt" der Systeminformation gezeigt, tun jedoch klaglos ihren Dienst. <--- wie gesagt, sofern sie **VOR** dem Start des Rechners angeschlossen wurden.

Nun hat Alex ja bekanntermassen eine Lösung gefunden, bei der das an- und abstecken von TB-Devices auch im laufenden Betrieb funktioniert - gleichzeitig hat er jedoch zu Protokoll gegeben, die gefundene Lösung

nicht öffentlich zu machen, sondern nur in seinen "Custom Build" zum Einsatz kommen zu lassen. Viele fanden das nicht korrekt und haben ihn dafür kritisiert (ob zu Recht steht hier **nicht** zur Debatte) - ich für meinen

Teil habe die Herausforderung für mich angenommen und mich auf die Suche nach "der" Lösung gemacht.

Hier also nun die bereits angekündigte Anleitung, mit der es möglich ist, vollständiges HotPlug/HotSwap für Thunderbolt 2/3 auf einem Hackintosh zu realisieren:

Was wird benötigt?

- Motherboard mit einem Thunderbolt-Header (um eine zusätzliche PCIe-Thunderbolt-Karte zu nutzen) oder Motherboard mit OnBoard Thunderbolt

- auf jeden Fall ein aktuelles BIOS des jeweils genutzten Boards

- SSDT für Thunderbolt (wird dieses Tutorial mitliefern)

- Kenntnisse im anpassen einer SSDT (nicht zwingend Voraussetzung, aber von Vorteil)

- mindestens ein Thunderbolt device und ein USB-C device (um HotPlug/HotSwap zu testen)

Ich schreibe dieses Tutorial auf Grund meiner Erfahrung mit folgenden ASRock Motherboards: ASRock Z270 Super Carrier (OnBoard TB) und ASRock Z370 Professional Gaming i7 (Thunderbolt-Header mit Gigabyte GC Titan Ridge PCIe-Karte)

Ich habe hier auch noch ein 10 Jahre altes ASRock Z170Extreme7+ liegen, weiss aber, das hier die Thunderbolt-Einstellungen im BIOS einfach zu mager sind, um zu garantieren, das der hier gezeigte Weg auch auf diesem Board funktionieren würde.

Warum spreche ich das an dieser Stelle an? Weil ich mir bereits diverse BIOS Dateien der verschiedensten Hersteller auf ihre Thunderbolt-Einstellungen angesehen habe und daher weiss, dass diese im Laufe der letzten Jahre wesentlich an Funktions-

umfang zugelegt und entsprechend "aufgepimpt" worden sind.

Schauen wir uns beispielsweise mal die Standardeinstellungen für Thunderbolt der von mir oben bereits erwähnten Boards ASRock Z270 und Z370 an:

ASRock Z270 Super Carrier: **ASRock Z370 Professional Gaming i7:**


Sieht reichlich dürftig aus, was die Einstellungen für Thunderbolt angeht. ABER: das sind nur die Einstellungen für Thunderbolt, die ASRock für den Enduser freigeschaltet hat. Im BIOS selber befinden sich noch diverse weitere Einstellungen, die es gilt freizuschalten.

ASUS und GIGABYTE bieten hier von Hause aus wesentlich mehr Einstellungen für den Enduser, die standardmässig freigeschaltet sind. Wie also schalten wir die anderen Funktionen für Thunderbolt frei, welche uns von Hause aus vorenthalten bleiben?

Hierzu benötigen wir das Tool "**AMIBCP**". Leider ist dieses Tool kommerziell und auch auf Anfrage nicht bei AMI zu bekommen, da es nur für Entwickler gedacht ist. Somit kann ich es

auch nicht mit diesem Tutorial zur Verfügung stellen. Wer aber in der Lage ist GOOGLE

richtig zu benutzen, wird sicherlich fündig. Ich nutze für dieses Tutorial die Version 5.02.0023. Hiermit öffnen wir unser BIOS-File, wie wir es von der Hersteller Supportseite heruntergeladen haben:

ASRock Z270 Super Carrier BIOS 2.40 default	**ASRock Z370 Professional Gaming i7 BIOS 4.00 default**

WOW, was für eine Möglichkeit, sich hier an den Einstellungen auszuprobieren! Und wie schalten wir diese nun frei, damit wir Sie ggf. alle ändern können? Das wird im nächsten Schritt erklärt.

Wir stellen einfach in der Spalte "Access/Use" jeden "Default" Wert auf "USER" - wie im nachfolgenden Bild:

und speichern das ganze dann über das kleine Diskettensymbol oben links. <--- **Anmerkung von mir:** ich habe es mit div. BIOS Dateien der versch. Hersteller getestet: ASRock BIOS Dateien waren die einzigen, die sich danach auch wieder speichern ließen. BIOS Dateien von Gigabyte, Asus und MSI brachten das Programm "AMIBCP" zum Absturz, ohne das die Änderungen in der BIOS Datei gespeichert wurden.

Wenn die Änderungen erfolgreich gespeichert werden und das so "gepimpte" BIOS erfolgreich geflashed werden konnte, sollten Eure Thunderbolt-Einstellungen beim nächsten Boot wie folgt aussehen:

ASRock Z270 Super Carrier Thunderbolt settings modified:	**ASRock Z370 Professional Gaming Thunderbolt settings modified:**
 	 

Ihr seht übrigens in beiden BIOS Screenshot (Z270 und Z370) die bereits korrekten Einstellungen, die wir benötigen, damit **HotPlug/HotSwap** korrekt funktioniert. **WICHTIG** an dieser Stelle: die Funktion "**GPIO3 Force Pwr**" muss auf "**ENABLED**" stehen! Sollte Euer Board, so wie das hier gezeigte Z270 Super Carrier, auch eine "**Thunderbolt (TM) Force Power**" Funktion haben, so ist diese ebenfalls auf "**ENABLED**" zu setzen.

Bei Boards mit Thunderbolt OnBoard kann es auch nötig sein, die zweite "**AIC Location**" Funktion von ggf. standardmäßig "**NB PCIE D01F0**" auf "**NB PCIE D01F2**" zu setzen. <--- war bei dem ASRock Z270 Super Carrier der Fall - und ich habe mir einen Wolf gesucht, warum **HotPlug/HotSwap** nicht funktioniert ! Das ASRock Z370 Professional Gaming i7 stand standardmäßig auf "**NB PCIE D01F2**", was mir beim direkten Vergleich der Einstellungen aber jedes mal entgangen ist.

Das war der **BIOS**-Teil, welcher ggf. abgearbeitet werden muss, um **HotPlug/HotSwap** zu gewährleisten. Jetzt folgt der leichte Teil: anlegen einer passenden **SSDT** für Thunderbolt. Ihr findet im Anhang zu diesem Tutorial eine von mir vorgefertigte SSDT, die Ihr mit an Sicherheit grenzender Wahrscheinlichkeit für Euer Board anpassen müsst (aber keine Sorge, die Anpassung hält sich wirklich in Grenzen, versprochen).

Beispiel SSDT für Thunderbolt: hackintosh-forum.de/attachment/111617/

Zunächst einmal müssen wir wissen, wo unsere Thunderbolt Hardware eingebunden wird. Dazu sehen wir uns mal die Einträge im IORegistryExplorer an (beide Beispiele erfolgten nach einem Boot **ohne** spezieller SSDT):

ASRock Z270 SuperCarrier (Thunderbolt OnBoard):

ASRock Z370 Professional Gaming i7 (Thunderbolt via GC Titan Ridge PCIe):

Wir sehen beim Z270 (links) den Eintrag "**AppleThunderboltHAL**" unter "**RP01@1C**", beim Z370 (rechts) den Eintrag "**AppleThunderboltHAL**" unter "**PEG2@1,2**". Ebenso sehen wir aber beim Z270 auch, dass hier noch "**PXSX**" und beim Z370 noch "**PEGP**" ins Spiel kommen. Also sind die anzupassenden Werte:

beim Z270: "**RP01**" und "**PXSX**" (wer aufmerksam hinschaut, wird feststellen, daß unsere Beispiel-SSDT offensichtlich vom Z270 stammt)

beim Z370: "**PEG2**" und "**PEGP**"

Wenn wir unsere Beispiel-SSDT öffnen, sehen wir darin den folgenden Code (der Übersichtlichkeit halber verpackt in einen SPOILER):

Spoiler anzeigen

Um die **SSDT** nun für die jeweils eigenen Bedürfnisse anzupassen, müssen von Euch folgende Zeilen geändert werden (ich nenne an dieser Stelle die entsprchenden Zeilennummern):

Zeile 23: External (_SB_.PCI0.**RP01**, DeviceObj) // (from opcode)

Zeile 24: External (_SB_.PCI0.**RP01**.**PXSX**, DeviceObj) // (from opcode)

Zeile 27: Scope (_SB.PCI0.**RP01**)

Zeile 29: Scope (**PXSX**)

Diese Zeilen bestimmen, auf welchem PCI0 Platz Eure Thunderbolt-Hardware sitzt. Die im Beispiel gezeigte Lösung kommt von meinem ASRock Z270 Super Carrier, welches Thunderbolt OnBoard mitbringt und daher unter **RP01/PXSX** zu finden ist.

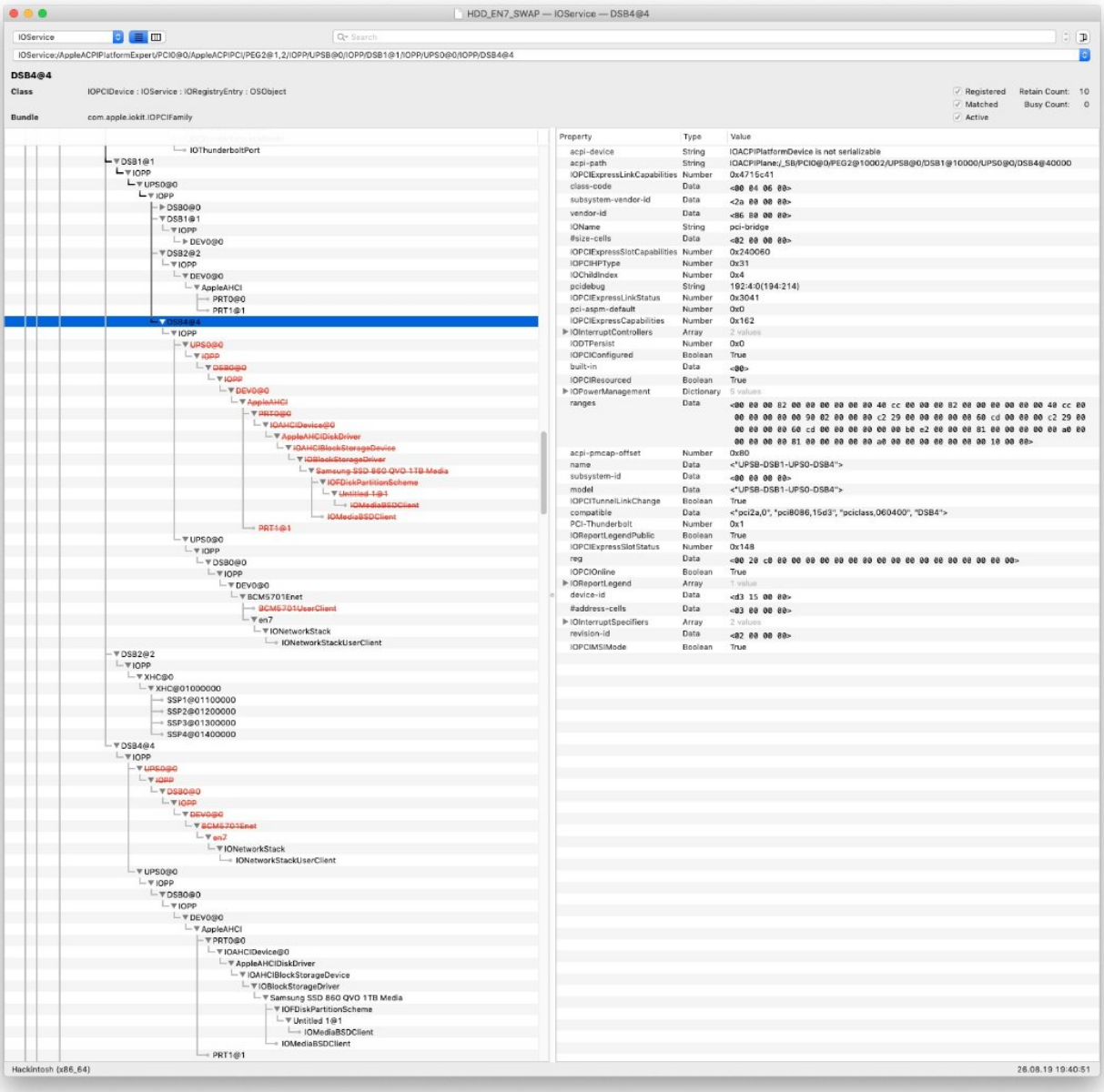
Mein ASRock Z370 Professional Gaming i7 nutzt einen GC Titan Ridge PCIe Karte und sitzt bei mir daher standardmässig auf _SB_.PCI0.PEG2.PEGP, dementsprechend würde die Änderung wie folgt aussehen (wieder im Spoiler):

Spoiler anzeigen

Sprich: wenn Ihr erstmal rausgefunden habt, wo Eure Thunderbolt Hardware sitzt und entsprechend die sechs genannten Zeilen angepasst habt, dann seid Ihr auch schon fertig. Wenn Ihr die Datei anders benennen wollt ist das auch kein Problem. Ich habe sie nur so benannt, weil ich mir das Namensschema an einem original iMac19,1 abgeschaut habe.

Nun müsst Ihr nur noch die fertige SSDT nach "***EFI/CLOVER/ACPI/patched***" kopieren, Rechner neu starten und wenn alles richtig gemacht wurde, Eure Thunderbolt Devices an-, ab- oder umstecken - und alles im laufenden Betrieb!

Im IORegistryExplorer sollte das HotPlug/HotSwap sich dann ähnlich diesem Screenshot bemerkbar machen:



Hier wurde beispielsweise eine TB-SSD und ein TB-to-Ethernetadapter beim hochfahren eingesteckt und während des laufenden Betriebs wurden beide Geräte jeweils auf den anderen Port geHotSwitched. Man sieht es an den **roten** Einträgen, die zeigen, das an jenem Port zuvor mal ein anderes Gerät vorhanden war.

That's **HotPlug/HotSwitch** for Hackintosh. Kein Hexenwerk, wenn man weiss, wie es geht. Aber eins kann ich Euch sagen: der Weg zu einer funktionierenden Lösung war steinig und geprägt von diversen Neustarts, lautstarken Flüchen und einem sehr lauten **YES**, als es dann doch funktioniert hat.

BtW: wer sich fragt, warum die SSDT nur aus den Einträgen UPSB und DSB0 besteht - und nicht aus weit mehr UPS0 und DSBx Einträgen, dem sei gesagt, dass ich versucht habe, weitere Einträge hinzuzufügen (wie es in einer original APPLE TB-SSDT der Fall ist), dann jedoch massiv Probleme beim HotPlug/HotSwap

innerhalb einer längeren TB-Device-Kette bekommen habe. So hatte ich versuchsweise mal alle meine TB-Geräte "in Reihe" gesteckt, und die dafür vorgesehenen UPSB und DSBx Einträge gesetzt, als dann jedesmal nach einem HotPlug an irgendeinem Punkt der Reihe mein Rechner jedesmal prompt neu gestartet

ist. Nutze ich jedoch die SSDT aus diesem Tutorial, kann ich jederzeit an jedem Punkt der Kette ein TB-Device trennen, ohne das mein Rechner neu startet. Natürlich geht so der Weg verloren, über einen _DSM-Eintrag angeschlossene Geräte mit einem "name" und "model" zu versehen, mit welchem es sich dann

innerhalb der Systeminformationen unter dem Punkt "PCI" registrieren würde, aber hier ging mir Funktionalität **vor** Design.

Beitrag von „floris“ vom 26. August 2019, 22:12

Als Du als Experte, kann ich eine Thunderbolt Karte nachrüsten, wenn mein Mainboard bzw. die Systemarchitektur (c612 Chipset) dies nicht unterstützen? Hat ja keinen Thunderbolt-Header. Mir ist bewußt, dass es mit Threadripper+kopierten Intel-Firmware Code geht, aber so unter Windows ist das eine eigene Treiber Architektur. Sowohl im UEFI BIOS des Mainboards (Intel Code) als auch auf High Level in Windows sind da wohl Treiber vorhanden, so wie ich das verstehe.

Beitrag von „Mork vom Ork“ vom 26. August 2019, 22:16

Ganz ehrlich: das Einsetzen einer Thunderboltkarte auf einem Mainboard ohne Thunderbolt-Header habe ich selber **nie** probiert - kann dazu also leider auch **nichts kluges** beisteuern. Sorry.

Beitrag von „OSX-Einsteiger“ vom 26. August 2019, 22:21

Ist hiermit Abonniert weil ist interessant 

Beitrag von „DSM2“ vom 26. August 2019, 22:27

floris Ich nehme mir mal raus diese Frage zu beantworten.

Möglich ist es grundsätzlich aber recht umständlich, sprich du musst dann immer erst Windows booten und anschließend MacOS.

Per "Cold Boot" wird das nicht funktionieren.

Diese Methodik wurde bereits am 5.1 Mac Pro getestet.

Beitrag von „jemue“ vom 27. August 2019, 00:52

Kannst du von deiner TB SSD auch booten?

(... und sie dann via HotPlug abstecken 

)

Beitrag von „DSM2“ vom 27. August 2019, 00:58

Was soll den das rumgetrolle hier? jemue

Beitrag von „Altemirabelle“ vom 27. August 2019, 09:54

@jemue

Du solltest das unbedingt an deinem Rechner testen

Beitrag von „Mork vom Ork“ vom 27. August 2019, 10:15

> Zitat von jemue
>
> Kannst du von deiner TB SSD auch booten?
>
> (... und sie dann via HotPlug abstecken 
>
>)

Jo, funktioniert einwandfrei. Vorher **muss** die Platte natürlich ordnungsgemäß vom Finder abgemeldet werden.

Beitrag von „Altemirabelle“ vom 27. August 2019, 11:52

Echt? Weigert sich macOS nicht, wenn man versucht die TB-SSD mit aktivem OS auszuwerfen? Und stürzt nicht ab?

Beitrag von „Mork vom Ork“ vom 27. August 2019, 12:01

Das von der TB-SSD Platte booten habe ich glatt überlesen. NEIN, das geht natürlich nicht, genausowenig wie mit jeder anderen Platte, von der man das aktive System gebootet hat.

Gehe ich hier gerade echt auf einen Troll-Post ein? 

Beitrag von „Altemirabelle“ vom 27. August 2019, 12:06

Ja sieht so aus.

Beitrag von „jemue“ vom 27. August 2019, 19:25

Es war nur zur Hälfte Troll 

Die erste Hälfte war schon ernst gemeint. (Also booten von der TB SSD)

Beitrag von „Mork vom Ork“ vom 31. August 2019, 19:08

UPDATE:

aus eigener Erfahrung kann ich sagen, dass meine Lösung sowohl auf Boards mit einem Thunderbolt-Header-Anschluss für eine beliebige Thunderbolt-PCIe-Karte,

als auch auf Boards mit OnBoard-Thunderbolt funktioniert. Ich habe gestern und heute mein altes ASRock Z270 Super Carrier Board "reaktiviert", welches TB3

onboard besitzt und auch da konnte ich (nach ein wenig tüfftеln im BIOS) natives HotPlug/HotSwitch zum laufen bringen.

Derzeit warte ich noch auf die Rückmeldung von DSM2 , welcher meine Lösung ebenfalls bei sich testen wollte. So please stay tuned for the detailed Instruction...

Beitrag von „Romsky“ vom 31. August 2019, 19:29

Mal eine Frage, ist diese Lösung auch interessant für Notebooks? Da gibt es leider noch so einige Baustellen in Bezug auf Thunderbolt:/

Beitrag von „DSM2“ vom 31. August 2019, 19:39

Update:

ASUS Prime X299-Deluxe II - läuft

Gigabyte X299 Designare EX - läuft

Edit:

ASUS WS X299 SAGE 10G - läuft

ASRock Fatal1ty Z390 Gaming-ITX/ac - funktioniert nicht

Romsky : spricht in der Theorie nichts gegen, ist ja ebenso Chipset Onboard.

Beitrag von „Mork vom Ork“ vom 31. August 2019, 19:47

ich bereite die Anleitung vor... muss mir aber erst noch benötigte Screenshots und einen Plan zurechtlegen. Ich werde den Initialbeitrag dazu verwenden und den jetzigen Inhalt dann durch die detaillierte Anleitung ersetzen. Gebt mir ein wenig Ziet.

Beitrag von „griven“ vom 31. August 2019, 19:52

Mork vom Ork ich möchte einfach mal Danke dafür sagen, dass Du Dir die Mühe machst die Sache zu recherchieren und in eine Anleitung zu packen das ist ganz großes Tennis 👍

Beitrag von „Mork vom Ork“ vom 31. August 2019, 22:58

griven

! HELP !

Mein Initailbeitrag hat das Limit von 10.000 Zeichen erreicht, ich bin aber noch nicht fertig.

Können wir den Thread später trennen und Ihr fügt Tutorial Teil 1 und Tutorial Teil 2 irgendwie zusammen oder zumindest hintereinander, wenn ich den 2. Teil jetzt einfach in einem neuen Beitrag schreibe?

(in diesem hier zum Beispiel)

DANKE, ich bin dran...

Beitrag von „griven“ vom 31. August 2019, 23:06

Mork vom Ork könnte man so machen fände ich aber nicht sonderlich elegant. Ich habe Für Dich das Limit mal angehoben

---

Beitrag von „Mork vom Ork“ vom 1. September 2019, 00:21

FERTIG mit dem Tutorial. Jemand bereit, es zu testen?

Um Feedback wird gebeten, Hilfestellung gerne geleistet.

WER hotplugged/hotswitched das erste Thunderbolt Device basierend auf diesem Tutorial?

Beitrag von „hitman20“ vom 1. September 2019, 01:53

Danke @Mork vom Ork Ich werde es morgen mal an meinem Dell XPS 9550 testen, ob es dort auch funktioniert.

Beitrag von „griven“ vom 2. September 2019, 16:10

Würde es direkt auch testen habe aber leider keine entsprechende Hardware sprich weder ein Brett mit Thunderbold noch die dazu gehörende Peripherie trotzdem finde ich es klasse das endlich nachhaltig Bewegung in das Thema gekommen ist denn so langsam hat sich das zu eine never Ending Story entwickelt. Ich kann mich erinnern das MacGrummel schon vor Jahren wegen dem Thunderbold Gedönse geflucht hat wie ein Rohrspatz war damals auf der HCKCN schon Thema mit der Kaffeemaschine vom Elch

---

Beitrag von „MacGrummel“ vom 2. September 2019, 16:39

Ja, und der Elch bastelt auch grad an einer Ersatzlösung, denn Gigabyte wollte einfach den vorhandenen Thunderbolt-Anschluss in meinem kleinen (und sonst ganz prima laufenden) Ga Z170n-Gaming 5 nie als solchen aktivieren. Die Hardware ist vorhanden, aber Gigabyte hätte halt die Lizenzgebühr zahlen müssen, um den USB-C-Anschluss als Thunderbolt nutzen zu dürfen..

Das neue Board ist ein ASRock Z390 PhantomGaming itx. Ich bin nur zZt. mit macOS Catalina da noch nicht ganz klar..

Beitrag von „Mork vom Ork“ vom 2. September 2019, 16:51

Die im Tutorial erwähnten Rechner laufen auf folgenden Systemen:

ASRock Z370 Professional Gaming i7: macOS Mojave 10.14.6

ASRock Z270 Super Carrier: macOS Catalina 10.15 beta 7

ich habe heute festgestellt, daß mein altes Gigabyte Z170X SOC FORCE ebenfalls über einen Thunderbolt-Header verfügt und werde dieses Board noch die kommenden Tage "reaktivieren", um auch auf diesem zu testen.

Ich kann mich noch erinnern, daß ich dieses Board "eingemottet" habe, weil ich Thunderbolt damals nicht zum Laufen gebracht habe.

Beitrag von „ResEdit“ vom 2. September 2019, 21:28

Ich teste das sehr gerne. Das ASRock Fatal1ty Z370 Gaming-ITX/ac hatte ich bereits letzte Woche bestellt, es gab aber Stress mit dem Paketdienst und jetzt soll es mit allergrößter Wahrscheinlichkeit morgen ankommen. Ich habe mehrere TB3/USB-C Gerätschaften, um es zu testen.

Im ersten Schritt ist wohl sinnvoll, die BIOS Version zu checken, oder?

Beitrag von „MacGrummel“ vom 2. September 2019, 23:04

Schade, dass ich im Moment so wenig Zeit habe, denn die kleine Ga-Z170er-Kaffee-Maschine wäre schon das richtige Experimentier-Feld. Zwei halbe Jobs sind einfach zu viel..

Geht/ginge das eigentlich ganz ohne vorhandenen Thunderbolt-Eintrag? Bei irgendeinem BIOS gab es das ja mal zwischendurch. Irgendwo in der USB-Abteilung. Nur hatten sie das BIOS dann nach wenigen Tagen wieder zurück gezogen, weil außer TB nicht viel lief..

P.S.:Schade, dass das AMI-Tool nicht unter macOS läuft..

Beitrag von „kuckkuck“ vom 2. September 2019, 23:20

> Zitat von MacGrummel
>
> P.S.:Schade, dass das AMI-Tool nicht unter macOS läuft..

Mal mit Wine probiert?

Beitrag von „ductator“ vom 2. September 2019, 23:58

Würde ich bei Tools, die am BIOS rumhantieren, nicht riskieren. Dann z.B. lieber eine VirtualBox VM mit der Windows Demo Version nutzen.

Wenn das BIOS dann doch nicht so funktioniert wie man will, hat man danach so seine Mühen wieder eine funktionierende Version raufzukriegen.

Beitrag von „kuckkuck“ vom 3. September 2019, 00:00

Das ist wohl war. Meine mich auch dunkel zu erinnern, dass das mit Wine nie lief...

Beitrag von „DSM2“ vom 3. September 2019, 00:27

MacGrummel warum nicht einfach per Parallels Desktop?

Beitrag von „ResEdit“ vom 3. September 2019, 08:50

So hatte ich das für mich spontan auch angedacht. Jetzt mal Butter beim Fisch, Jungs: Besteht wirklich ein *echtes* Risiko, sich dabei das Board zu zerschießen? Die Anleitung lässt doch kaum Raum für Fehler. Oder gehe ich da zu blauäugig dran?

Beitrag von „mhaeuser“ vom 3. September 2019, 08:56

USB Flashback (hardware-basiert!), Dual BIOS, oder 'nen Programmer (ggf. + Klemme) sollte man schon haben

---

Beitrag von „locojens“ vom 3. September 2019, 08:57

Und schlimmer als die damaligen Experimente mit Ozmosis ist es letzten Endes

auch nicht...

Und wenn man schon an Bios / Uefi rumspielt kann man sich ja auch zur Sicherheit so einen billigen Flasher holen (hab meinen wegen dem X230 geholt)

Habe damit sogar mittlerweile eine von einem Freund "zerflashte" ASROCK Z77 Extreme4 wieder von den Toten erweckt.

Beitrag von „ResEdit“ vom 3. September 2019, 09:00

Ich meine – so ein bischen Nervenkitzel gehört ja irgendwo schon dazu, oder? Ist ja schließlich Hobby, bierernst ist hingegen der Job.

Beitrag von „Mork vom Ork“ vom 3. September 2019, 09:59

ich habe gestern erst wieder versucht, das BIOS für das oben genannte Gigabyte Z170X SOC FORCE via AMIBCP zu editieren: und wieder stürzt das Programm beim speichern des midifizierten BIOS ab, OHNE das die Änderungen in der Datei gesichert wurden.

Wei bereits im Tuorial erwähnt, sind ASRock BIOS Dateien, bei der sich die Änderungen unter AMIBCP auch sichern lassen.

Jedoch glaube ich auch immer noch, das bei GIGABYTE und ASUS BIOS die jeweiligen Thunderbolt-Settings von Hause aus freigeschaltet sein sollten, so dass der Endanwender diese auch ändern kann.

Beitrag von „ResEdit“ vom 3. September 2019, 10:01

Ich habe hier eine Anleitung gefunden, eventuell hilft das ja weiter …

https://puissanceled.com/vrac/Bios_modding/EN.html

Beitrag von „Mork vom Ork“ vom 3. September 2019, 10:02

Nichts anderes habe ich in meinem Tutorial geschrieben! Es nutzt aber nichts, wenn AMIBCP dann beim speichern abstürzt.

Beitrag von „ductator“ vom 3. September 2019, 10:10

Wenn man weiß, welche Einstellungen man ändern muss, kann man auch über eine EFI Shell die Einstellung setzen.

Funktioniert analog zu https://www.win-raid.com/t3908...Other-Mod-BIOS-Flash.html, nur dass man dann die entsprechenden Thunderbolt Optionen ändern.

Beitrag von „locojens“ vom 3. September 2019, 10:12

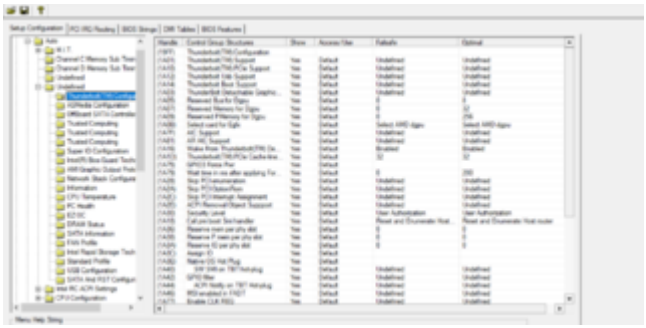
Wenn AMIBCP abstürzt liegt es an der Version des selbigen. Das gleich Problem besteht auch bei der Software mit welcher man bei den verschiedenen lenovos einen Biosdump erstellt (anderes Bios ich weiß). Aber da man ja früher mal des öfteren an Biosdateien gebastelt hat ist das Problem mit den verschiedenen Bios'en und dazugehörigen AMIBCP-Versionen noch im Hinterkopf gespeichert (ich möchte nur das "böse" SLIC einfügen erwähnen).

Beitrag von „DSM2“ vom 3. September 2019, 12:22

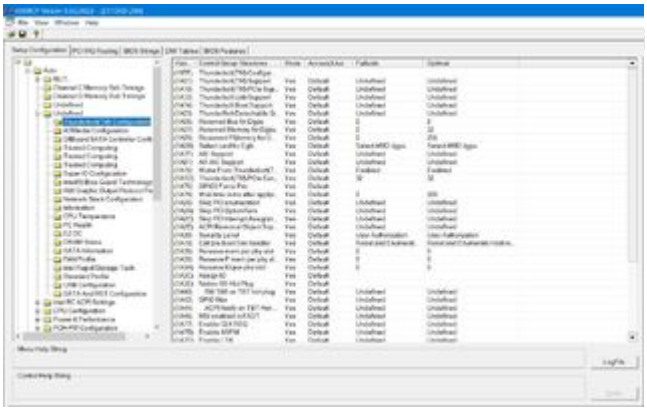
Mork vom Ork Was genau oder besser gesagt welche Version hast du den versucht zu patchen?

Edit: Ich bin eine Blindschleiche...

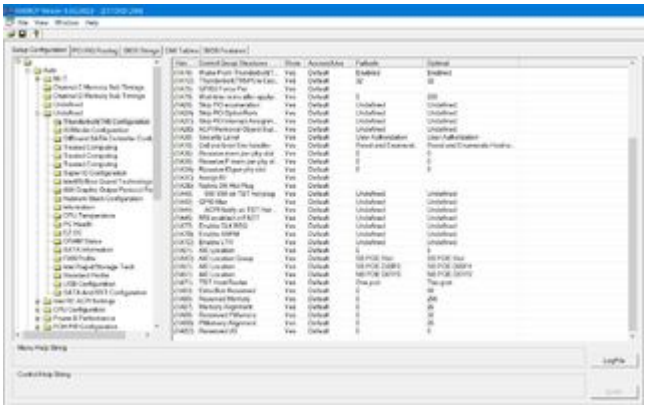
Aber auch bei mir verabschiedet sich AMIBCP



Beitrag von „Mork vom Ork“ vom 3. September 2019, 12:26


on F20k:

-



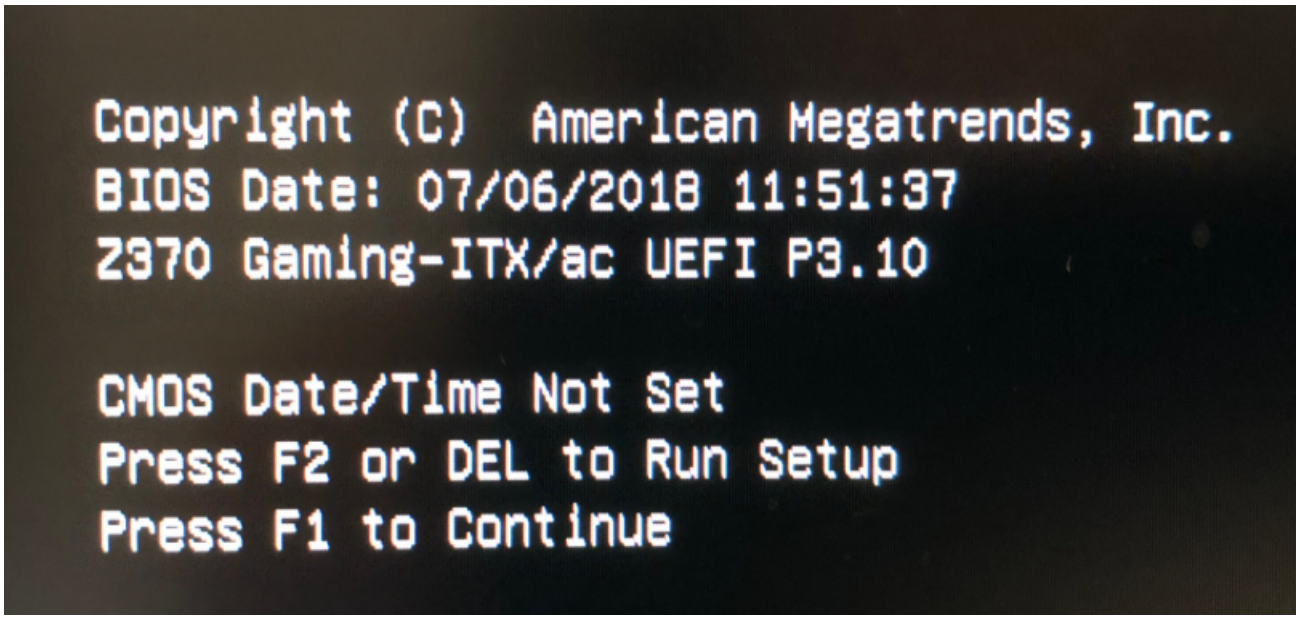
öffnen kann ich Sie mit der AMIBCP v.5.02.0023, nur eben leider nicht sichern.

Beitrag von „ResEdit“ vom 3. September 2019, 19:17

Das Board ist heute tatsächlich gekommen (hatte die Hoffnung schon fast aufgegeben, DHL hat sich da echt was geleistet – aber egal) und ich muss sagen, sooooo einfach war es mit dem

Hacky noch nie.

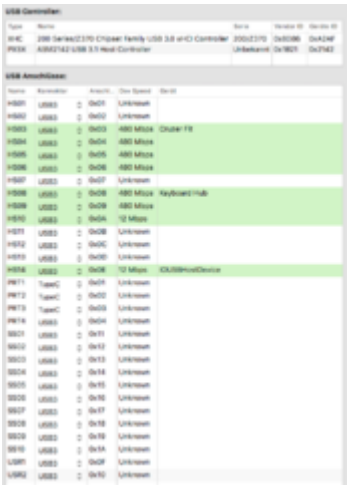
Beim ersten Einschalten kam diese Nachricht:



Dann habe ich im BIOS lediglich den Stick mit der EFI von meinem Gigabyte Z370N ausgewählt (nix an der EFI verändert, ich schwör!) und das Teil war dann blitzartig unter 10.12.6 einsatzbereit. Keine spezielle BIOS Einstellungen geladen, alles so gelassen wie das Board geliefert wurde. Hat mich sehr beeindruckt.

Natürlich zeigt er in den Systeminformationen unter Thunderbolt nichts an. Das wäre jetzt wirklich die Härte gewesen, wenn das auf Anhieb geklappt hätte.

Bin jetzt etwas unschlüssig, wie ich am sinnvollsten weitermache. Wahrscheinlich erst mal die USB Situation checken, da liegt einiges im Argen, nirgendwo USB3, der USB-C Port reagiert auch noch nicht.



Beitrag von „Mork vom Ork“ vom 3. September 2019, 19:49

Aber ich sagte es doch auch schon: die von mir hier besprochene Lösung zeigt nach wie vor **KEINE** Thunderbolt-Geräte in den Systeminformationen unter dem Punkt Thunderbolt!

Das Tutorial sorgt lediglich für funktionierenden **HotPlug/HotSwap** - nicht mehr, aber auch nicht weniger.

- - - - -

Habe soeben das Gigabyte Z170X SOC FORCE reaktiviert: aufgespielt war BIOS rev. 20e

Habe ich soeben aktualisiert auf rev. 20k

GC Alpine Ridge Karte steckt in PCIe Slot 7 - ist der von hinten gesehen ganz rechte x16 Slot. TB-Header Kabel ist gesteckt: kein Thunderbolt erkannt.

Werde jetzt die einzelnen Slots durchtesten.

- - - - -

Jetzt weiss ich wieder, warum ich das Gigabyte Board eingemottet habe:

das Ding hat zwar einen Thunderbolt-Header, erkennt auch die USB-C Anschlüsse der GC Alpine Ridge Karte - macht aber um's verrecken kein Thunderbolt.

Ich meine auch mich zu erinnern, schon damals den Gigabyte-Support angeschrieben zu haben, welcher mir dann mitgwtwilt hat, das dieses Board definitiv

KEIN Thunderbolt macht, wegen des PLX PEX8747 chips, welcher dafür sorgt, dass 32 Lanes für die GPU zur Verfügung stehen. Irgendetwas war da, weswegen

ich das Board wieder eingelagert habe.

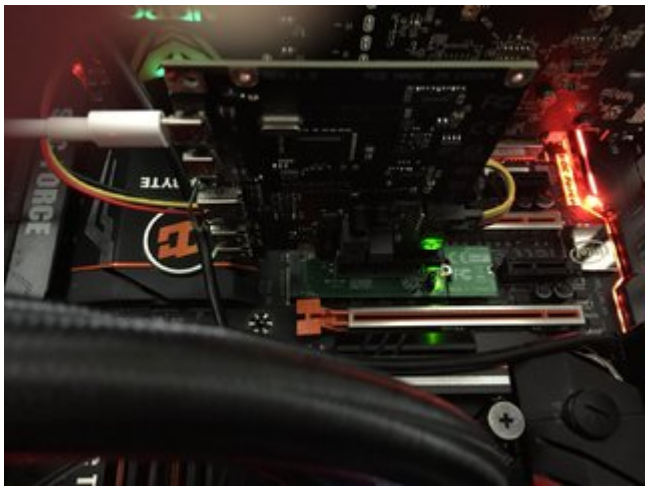
Und es gelingt mir derzeit auch nicht, Thunderbolt aktiviert zu bekommen. Es werden mir auch keine Thunderbolt-Settings im BIOS angeboten. Sch....!

- - - - -

Ok, rausgefunden, warum das Gigabyte Z170X SOC FORCE kein Thunderbolt nimmt: es hat **keinen** x4-Slot !

Thunderbolt läuft aber nur auf X4-Slots.

Aber: das Board hat ja 3 M.2 Slots. Also habe ich getrixt: im dritten M.2 Slot sitzt ein M.2-auf-PCIe-X4 Adapter: et voila: THUNDERBOLT am Z170X SOC FORCE

 - 

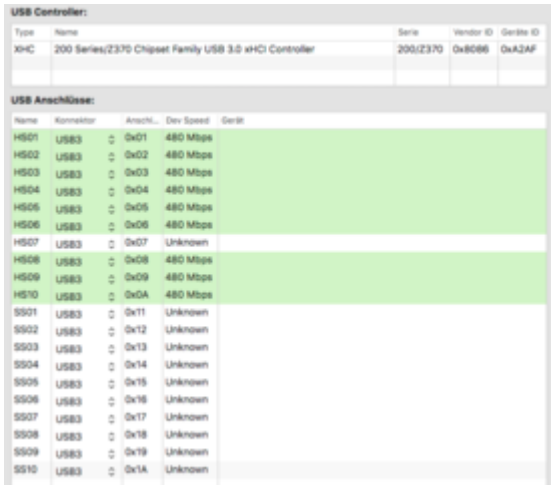
Zugegeben: noch nicht die eleganteste Art, aber machbar. Brauch ich nur ein passendes "Riser"-Kabel und schon kann ich die Karte sauber im Gehäuse verlegen.

Beitrag von „ResEdit“ vom 3. September 2019, 21:12

Oh – sorry, du hattest das womöglich als versteckte Kritik einsortiert. War keinesfalls so gemeint. Wenn ich TB3 nutzen will, ist die Anzeige in den Systeminformationen das letzte, was mir dabei wichtig erscheint. Alles ist gut.

Bin derzeit noch bei der Konfiguration der USB Ports. Der USB-C Port wird nicht erkannt (angesteckte Geräte werden nicht angezeigt), könnte das was mit dem aktivierten TB3 im BIOS zu tun haben? Meine BIOS Version ist oben im Bild, soll ich upgraden? Hier die Situation mit den Ports:

1. Geräte mit 480 Mbps:



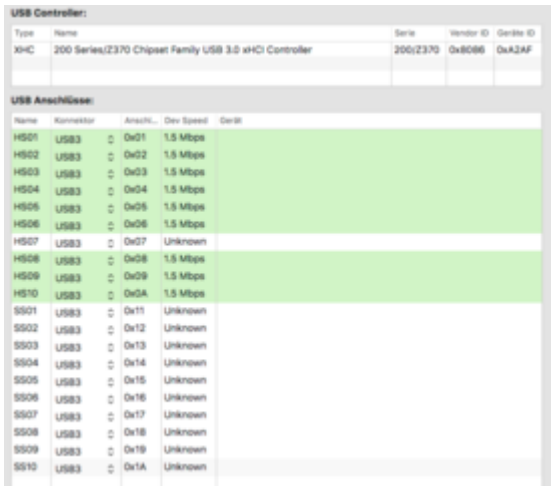
USB Controller:

Type	Name	Serie	Vendor ID	Geräte ID
XHC	200 Series/Z370 Chipset Family USB 3.0 xHCI Controller	200/Z370	0x8086	0xA2AF

USB Anschlüsse:

Name	Konnektor	Anschl...	Dev Speed	Gerät
HS01	USB3	0x01	480 Mbps	
HS02	USB3	0x02	480 Mbps	
HS03	USB3	0x03	480 Mbps	
HS04	USB3	0x04	480 Mbps	
HS05	USB3	0x05	480 Mbps	
HS06	USB3	0x06	480 Mbps	
HS07	USB3	0x07	Unknown	
HS08	USB3	0x08	480 Mbps	
HS09	USB3	0x09	480 Mbps	
HS10	USB3	0x0A	480 Mbps	
SS01	USB3	0x11	Unknown	
SS02	USB3	0x12	Unknown	
SS03	USB3	0x13	Unknown	
SS04	USB3	0x14	Unknown	
SS05	USB3	0x15	Unknown	
SS06	USB3	0x16	Unknown	
SS07	USB3	0x17	Unknown	
SS08	USB3	0x18	Unknown	
SS09	USB3	0x19	Unknown	
SS10	USB3	0x1A	Unknown	

2. Geräte mit 1.5 Mbps:



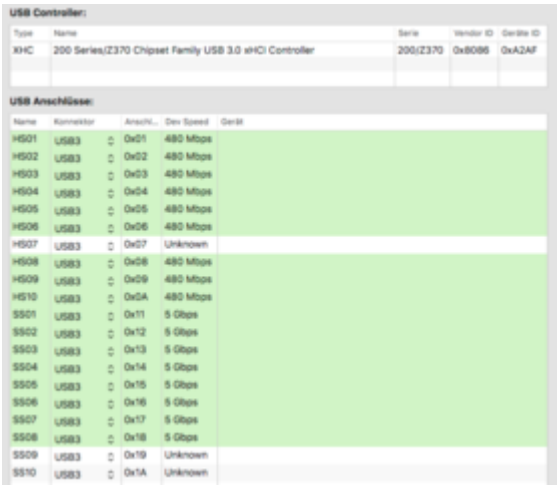
USB Controller:

Type	Name	Serie	Vendor ID	Geräte ID
XHC	200 Series/Z370 Chipset Family USB 3.0 xHCI Controller	200/Z370	0x8086	0xA2AF

USB Anschlüsse:

Name	Konnektor	Anschl...	Dev Speed	Gerät
HS01	USB3	0x01	1.5 Mbps	
HS02	USB3	0x02	1.5 Mbps	
HS03	USB3	0x03	1.5 Mbps	
HS04	USB3	0x04	1.5 Mbps	
HS05	USB3	0x05	1.5 Mbps	
HS06	USB3	0x06	1.5 Mbps	
HS07	USB3	0x07	Unknown	
HS08	USB3	0x08	1.5 Mbps	
HS09	USB3	0x09	1.5 Mbps	
HS10	USB3	0x0A	1.5 Mbps	
SS01	USB3	0x11	Unknown	
SS02	USB3	0x12	Unknown	
SS03	USB3	0x13	Unknown	
SS04	USB3	0x14	Unknown	
SS05	USB3	0x15	Unknown	
SS06	USB3	0x16	Unknown	
SS07	USB3	0x17	Unknown	
SS08	USB3	0x18	Unknown	
SS09	USB3	0x19	Unknown	
SS10	USB3	0x1A	Unknown	

3. Geräte mit 5 Gbps:



USB Controller:

Type	Name	Serie	Vendor ID	Geräte ID
XHC	200 Series/Z370 Chipset Family USB 3.0 xHCI Controller	200/Z370	0x8086	0xA2AF

USB Anschlüsse:

Name	Konnektor	Anschl...	Dev Speed	Gerät
HS01	USB3	0x01	480 Mbps	
HS02	USB3	0x02	480 Mbps	
HS03	USB3	0x03	480 Mbps	
HS04	USB3	0x04	480 Mbps	
HS05	USB3	0x05	480 Mbps	
HS06	USB3	0x06	480 Mbps	
HS07	USB3	0x07	Unknown	
HS08	USB3	0x08	480 Mbps	
HS09	USB3	0x09	480 Mbps	
HS10	USB3	0x0A	480 Mbps	
SS01	USB3	0x11	5 Gbps	
SS02	USB3	0x12	5 Gbps	
SS03	USB3	0x13	5 Gbps	
SS04	USB3	0x14	5 Gbps	
SS05	USB3	0x15	5 Gbps	
SS06	USB3	0x16	5 Gbps	
SS07	USB3	0x17	5 Gbps	
SS08	USB3	0x18	5 Gbps	
SS09	USB3	0x19	Unknown	
SS10	USB3	0x1A	Unknown	

Wie gesagt, wenn ich was bei USB-C einstöpsle, passiert nix. BIOS?

Beitrag von „MacGrummel“ vom 3. September 2019, 21:45

Aktivier mal USB über Thunderbolt, das ist ein Standart-Blocker..

Beitrag von „Mork vom Ork“ vom 3. September 2019, 22:03

ResEdit

welches System fährst Du?

Ich habe unter Mojave 10.14.6 angefangen mit meinen ganzen Tests - und dabei zunächst die "USB 15-Port Limit" patches gesetzt und in "kexts-Others" den "USBInjectAll.kext" aktiv.

Darüber habe ich bei meinem 370er Board rausgefunden, das bei mir Port HS10 und SS10 aktiv sein muessen, damit USB-C der GC Titan Ridge funktionieren.

Selbiges hat bei mir dann auch unter Catalina beta 7 funktioniert. USB Portlimit patch setzen und USBInjectAll.kext nutzen. Damit feststellen, welche Ports genutzt werden. Der HS-Port

wird nur Nutzung von USB-C-auf-USBG3.0-A Adapter benötigt, um auch USB2.0 devices an der GC Alpine/Titan Ridge nutzen zu können. nutzen zu können

---

Beitrag von „apfelnico“ vom 4. September 2019, 02:55

Mork vom Ork

Erst mal dicken Respekt, das zum Thema zu machen. Kann allerdings nicht so ganz nachvollziehen, was nun an der SSDT neu sein soll. HotPlug funktioniert schon lange, habe auch einigen Leuten hier geholfen, dort funktioniert es ebenso. In deiner (Basis-) SSDT ist einiges drin, was noch raus kann. Um mal bei deiner SSDT zu bleiben:

In Zeile 23-27 wird auf externe Quellen verwiesen. Der Verweis auf XHC_ kann ersatzlos gestrichen werden, es gibt keinerlei weiteren Bezug in der SSDT darauf. Im Original werden hier in der Folge USB2 vom XHC geroutet, ist hier völlig unsinnig. Auch heißt je nach System dieses Device auch mal XHCI oder anders. Egal, es kommt auch gar nicht weiter vor in deiner SSDT. Der Aufruf einer externen Methode DTGP funktioniert natürlich auch nur, wenn sie denn wo anders schon enthalten ist, darauf sollte noch hingewiesen werden (inject per Clover in die DSDT, oder enthalten in einer weiteren SSDT). Der letzte Eintrag PXSX kann ebenso raus, ist dieser doch schon exakt mit Pfad weiter oben beschrieben worden.

Die Methode NTFY (Zeile 34) kann ersatzlos gestrichen werden, völlige Nebelkerze. Denn mit "Name (STA, Zero" blendest du das vorhandene Device "PXSX" aus, um für ein eigenes Device "UPSB" auf Adresse 0x0 (Zero) Platz zu schaffen, "Notify" wird also niemals ausgeführt.

Die _DSM Methoden habe ich mal um die unsinnige "if"-Schleife reduziert.

Beschreibung Definitionsblock "TbtOnPCH" finde ich auch nicht so elegant, denn diese SSDT sollte doch allgemeingültig sein. Also nicht nur auf irgendwelche RPxx vom PCH, sondern auch direkte PCIe der CPU zugeordnet.

Es bleibt also, wie in vielen anderen SSDTs zum Thema, zum einen der Pfad zum Device "NHI0"

und einige wichtige Beschreibungen per _DSM-Methode eingefügt.

Deine Erkenntnis, dass der komplette Baum wie in der originalen Apple SSDT zu Problemen führt, ist richtig. Besser ist es allerdings, die Stamm-Devices DSBx noch aufzunehmen, denn hier können noch jeweils _DSM-Methoden hinzugefügt werden, die sich in der Kette vererben ("AAPL,slot-name"). Der Vorteil, nun werden auch die angeschlossenen Geräte in der PCI-Sektion (Systembericht) angezeigt! Weiterhin kann so auch der XHCI-Controller beschrieben werden. Habe ich mal integriert und "XHC5" genannt, damit dieser nicht namentlich mit eventuell weiteren XHCx-Controllern kollidiert.

Ich habe die überarbeitete SSDT mal im Anhang gesetzt (Pfade natürlich überarbeiten), teste es gern. Der Teil des Tutorials, welcher sich dem BIOS widmet, finde ich sehr spannend. Ich habe das Glück, ein BIOS zu haben welches alle relevanten Einstellungen zu Controller zulässt; dieses allen Boards zugänglich zu machen – noch mal dicken Respekt.

Beitrag von „DSM2“ vom 4. September 2019, 03:13

Die Sache an dem ganzen Vorgehen ist aber das aus welchen Gründen auch immer, dass ganze bei Thunderbolt Onboard funktioniert und mit der reinen alten Methode eben nicht.

Ich selbst habe das ganze nicht analysiert und habe persönlich eine andere Lösung für Onboard Chipsets die aber recht komplex ist.

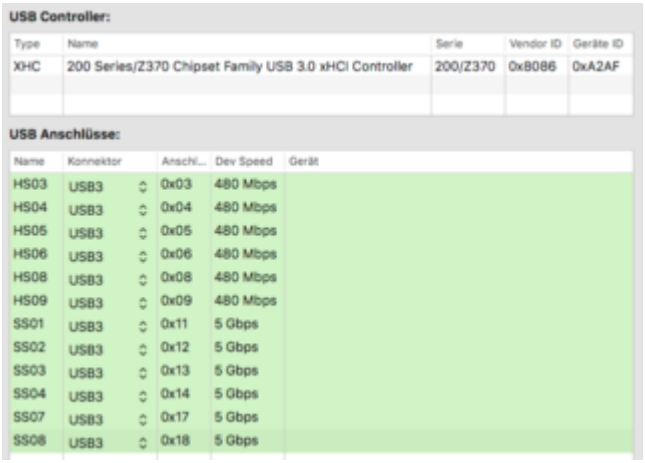
Fakt ist das es mit der oben beschriebenen Methode definitiv funktioniert und einfacher umsetzbar ist als mein Ansatz für Thunderbolt Onboard.

Beitrag von „ResEdit“ vom 4. September 2019, 08:54

> Zitat von Mork vom Ork

> ResEdit
>
> welches System fährst Du?

10.12.6, wie schon gesagt. Habe deinen Hinweis mit USB HS10 und SS10 zu spät gelesen. Den USB2 Header und den USB3 Port auf dem MoBo habe ich im BIOS deaktiviert, dann die nicht genutzten Ports mit uia_exclude=HS01;HS02;HS07;HS10;HS11;HS12;HS13;HS14;SS05;SS06;SS09;SS10;USR1;USR2 ausgeklammert. Das sieht dann jetzt so aus:



USB Controller:

Type	Name	Serie	Vendor ID	Geräte ID
XHC	200 Series/Z370 Chipset Family USB 3.0 xHCI Controller	200/Z370	0x8086	0xA2AF

USB Anschlüsse:

Name	Konnektor	Anschl...	Dev Speed	Gerät
HS03	USB3	0x03	480 Mbps	
HS04	USB3	0x04	480 Mbps	
HS05	USB3	0x05	480 Mbps	
HS06	USB3	0x06	480 Mbps	
HS08	USB3	0x08	480 Mbps	
HS09	USB3	0x09	480 Mbps	
SS01	USB3	0x11	5 Gbps	
SS02	USB3	0x12	5 Gbps	
SS03	USB3	0x13	5 Gbps	
SS04	USB3	0x14	5 Gbps	
SS07	USB3	0x17	5 Gbps	
SS08	USB3	0x18	5 Gbps	

Mehr USB brauche ich auch nicht. Habe dann eine Plugable Docking Station UD-CAM (ASIN B078WXTYTW) mit TB3 Kabel (!) am TB3 Port angeschlossen und siehe da: Die dort integrierte Soundkarte wird als USB-Tonausgabegerät erkannt, das darin befindliche USB-Netzwerk-Interface (10/100/1000 RJ45) ebenfalls. Auch die HDMI Schnittstelle wird erkannt, habe jedoch (bisher) kein Dual-Monitor Setup erfolgreich testen können. Immerhin wird das USB-Protokoll über TB3 weitergereicht, obwohl ich HS10 und SS10 "unsichtbar" konfiguriert habe.

Bin echt erstaunt, wie locker sich das bislang alles gemacht hat, im BIOS habe ich lediglich die nervigen LED-Lichtorgeln des MoBos deaktiviert und die iGPU als Standard gesetzt. Sonst wie gesagt NULL Konfiguration im BIOS vorgenommen.

Edit: Sorry, hatte ich vergessen: Onboard BT und WLAN habe ich ebenfalls deaktiviert im BIOS. Brauche ich beides nicht.

Beitrag von „Mork vom Ork“ vom 4. September 2019, 09:16

apfelnico

Danke für Dein Feedback. Und danke für den Hinweis mit dem XHC-Eintrag. Das war mir zwar bewusst, dass dieser eigentlich noch rausgelöscht gehört, aber ist im Eifer des Gefenchts dann doch noch übersehen worden.

Bezüglich der alten Methode kann ich nur sagen:

ich habe diverse TBT-SSDT Dateien bei mir ausprobiert und wirklich **keine** davon ermöglichte mir ein sauberes HotPlug auf meinen beiden ASRock Boards. Keine.

Daher war ich ja so versessen darauf, eine nicht nur für mich, sondern allgemein funktionierende Lösung zu finden, nachdem ich Alex' Video gesehen hatte, da ich mir dachte: "...es muss ja irgendwie gehen."

Letztlich habe ich nicht getestet, ob es nun nur Dank der neuen Einstellungen f. TB im BIOS funktioniert, oder ob auch die "NOTIFY"-Funktion ihren Teil dazu beiträgt. Ich habe für mich und bei meinen diversen Tests halt immer

nur festgestellt: nimmt man eins der Elemente wieder weg, funktioniert HotPlug bei mir wieder nicht mehr. So habe ich z.B getestet, was passiert, wenn ich nur die BIOS-Einstellungen belasse und gänzlich auf eine TB-

SSDT verzichte. Fazit: HotPlug funktioniert NULL, also gar nicht. Für USB-C devices nie ein Problem, aber wirkliche TB devices funktionieren eben nur dann, wenn sie zu Beginn des Rechnerstarts gesteckt sind und einmal

abgezogen, lassen sie sich nicht erneut reaktivieren.

Und mit dem ganzen UPSB/DSBx Wust hatte ich ja auch geschrieben, das ich es in der Tutorial-SSDT bewusst weggelassen habe, da ich dadurch in einem längeren, hintereinandergeschalteten Strang an TB-Devices das

Problem hatte, das ich die Kette dann nicht an jeder x-beliebigen Stelle unterbrechen konnte, ohne das mein Rechner umgehend einen Neustart gemacht hat. Habe ich die selbe Kette aber unter Nutzung der im Tutorial

besprochenen SSDT genutzt, konnte ich die Kette an jeder x-beliebigen Stelle trennen und ggf. neu stecken.

Das dadurch die bequeme Möglichkeit der Benennung der Devices mittels der _DSM-Methode verloren geht, habe ich zu Gunsten der Funktionalität ersteinmal in Kauf genommen. Gerne gehe ich dazu aber nachträglich

nochmal genauer darauf ein. Darum bat ich ja um Euer Feedback

Beitrag von „apfelnico“ vom 4. September 2019, 10:28

> Zitat von Mork vom Ork
>
> Letztlich habe ich nicht getestet, ob es nun nur Dank der neuen BIOS Einstellungen funktioniert, oder ob auch die "NOTIFY"-Funktion ihren Teil dazu beiträgt.

Im Original https://osy.gitbook.io/hac-min…details/thunderbolt-3-fix wurde beschrieben, welche "_Exx" Funktion (Edge Triggered GPE) ein "Notify" für genau den benutzten Slot beinhaltet. Bei mir ist es "_E4C". Eine Methode Namens "NTFY" wurde ermittelt, diese wird in der DSDT benutzt. Entweder man editiert diese direkt in der DSDT, oder man versucht diese über Clovers DSDT-Patches durch Rename unbrauchbar zu machen. So sieht diese bei mir zum Beispiel aus:

Code

```
1. Scope (_GPE)
2. {
3. Method (NTFY, 0, Serialized)
4. {
5. If ((NOHP == One))
6. {
7. Switch (ToInteger (TBSE))
8. {
9. Case (One)
```

```
{
Notify (\_SB.PCI0.RP01, Zero) // Bus Check
}
Case (0x02)
{
Notify (\_SB.PCI0.RP02, Zero) // Bus Check
}
Case (0x03)
{
Notify (\_SB.PCI0.RP03, Zero) // Bus Check
}
Case (0x04)
{
Notify (\_SB.PCI0.RP04, Zero) // Bus Check
}
Case (0x05)
{
Notify (\_SB.PCI0.RP05, Zero) // Bus Check
}
Case (0x06)
{
Notify (\_SB.PCI0.RP06, Zero) // Bus Check
}
Case (0x07)
{
Notify (\_SB.PCI0.RP07, Zero) // Bus Check
}
Case (0x08)
{
Notify (\_SB.PCI0.RP08, Zero) // Bus Check
}
Case (0x09)
{
Notify (\_SB.PCI0.RP09, Zero) // Bus Check
}
Case (0x0A)
{
Notify (\_SB.PCI0.RP10, Zero) // Bus Check
}
Case (0x0B)
{
```

```
51. Notify (\_SB.PCI0.RP11, Zero) // Bus Check
52. }
53. Case (0x0C)
54. {
55. Notify (\_SB.PCI0.RP12, Zero) // Bus Check
56. }
57. Case (0x0D)
58. {
59. Notify (\_SB.PCI0.RP13, Zero) // Bus Check
60. }
61. Case (0x0E)
62. {
63. Notify (\_SB.PCI0.RP14, Zero) // Bus Check
64. }
65. Case (0x0F)
66. {
67. Notify (\_SB.PCI0.RP15, Zero) // Bus Check
68. }
69. Case (0x10)
70. {
71. Notify (\_SB.PCI0.RP16, Zero) // Bus Check
72. }
73. Case (0x11)
74. {
75. Notify (\_SB.PCI0.RP17, Zero) // Bus Check
76. }
77. Case (0x12)
78. {
79. Notify (\_SB.PCI0.RP18, Zero) // Bus Check
80. }
81. Case (0x13)
82. {
83. Notify (\_SB.PCI0.RP19, Zero) // Bus Check
84. }
85. Case (0x14)
86. {
87. Notify (\_SB.PCI0.RP20, Zero) // Bus Check
88. }
89. Case (0x15)
90. {
91. Notify (\_SB.PCI0.RP21, Zero) // Bus Check
```

```
92. }
93. Case (0x16)
94. {
95. Notify (\_SB.PCI0.RP22, Zero) // Bus Check
96. }
97. Case (0x17)
98. {
99. Notify (\_SB.PCI0.RP23, Zero) // Bus Check
100. }
101. Case (0x18)
102. {
103. Notify (\_SB.PCI0.RP24, Zero) // Bus Check
104. }
105. Case (0x1A)
106. {
107. Notify (\_SB.PC01.BR1A, Zero) // Bus Check
108. }
109. Case (0x1B)
110. {
111. Notify (\_SB.PC01.BR1B, Zero) // Bus Check
112. }
113. Case (0x1C)
114. {
115. Notify (\_SB.PC01.BR1C, Zero) // Bus Check
116. }
117. Case (0x1D)
118. {
119. Notify (\_SB.PC01.BR1D, Zero) // Bus Check
120. }
121. Case (0x1E)
122. {
123. Notify (\_SB.PC02.BR2A, Zero) // Bus Check
124. }
125. Case (0x1F)
126. {
127. Notify (\_SB.PC02.BR2B, Zero) // Bus Check
128. }
129. Case (0x20)
130. {
131. Notify (\_SB.PC02.BR2C, Zero) // Bus Check
132. }
```

```
133.  Case (0x21)
134.  {
135.  Notify (\_SB.PC02.BR2D, Zero) // Bus Check
136.  }
137.  Case (0x22)
138.  {
139.  Notify (\_SB.PC03.BR3A, Zero) // Bus Check
140.  }
141.  Case (0x23)
142.  {
143.  Notify (\_SB.PC03.BR3B, Zero) // Bus Check
144.  }
145.  Case (0x24)
146.  {
147.  Notify (\_SB.PC03.BR3C, Zero) // Bus Check
148.  }
149.  Case (0x25)
150.  {
151.  Notify (\_SB.PC03.BR3D, Zero) // Bus Check
152.  }
153.
154.  }
155.  }
```

Alles anzeigen

Um dann in der Folge in einer eigenen SSDT die Funktion neu zu erschaffen, mit an geeigneter Stelle modifizierte Notify, die bis zum NHI0 reicht. Sinnvoll wäre es natürlich, auch die restlichen unbearbeiteten "Cases" wieder aufzunehmen. Das kann also je nach System ganz anders aussehen, hier war es ein NUC. Und wie gesagt, deine "NTFY" hängt an "PXSX", welches du gerade "unschädlich" gemacht hast. Wenn es wichtig wäre, wäre es besser im aktiven Strang "UPSB" aufgehoben. Noch besser jedoch das Original ausgetauscht unter "Scope (_GPE)".

> Zitat von Mork vom Ork
>
> Und mit dem ganzen UPSB/DSBx Wust hatte ich ja auch geschrieben, das ich es in der Tutorial-SSDT bewusst weggelassen habe, da ich dadurch in einem längeren, hintereinandergeschalteten Strang an TB-Devices das
>
> Problem hatte, das ich die Kette dann nicht an jeder x-beliebigen Stelle unterbrechen

> konnte, ohne das mein Rechner umgehend einen Neustart gemacht hat. Habe ich die selbe Kette aber unter Nutzung der im Tutorial
>
> besprochenen SSDT genutzt, konnte ich die Kette an jeder x-beliebigen Stelle trennen und ggf. neu stecken.
>
> Das dadurch die bequeme Möglichkeit der Benennung der Devices mittels der _DSM-Methode verloren geht, habe ich zu Gunsten der Funktionalität ersteinmal in Kauf genommen.

Das habe ich ja verstanden und auch als richtig geschlussfolgert kommentiert. Nur ist es sinnvoll, (nicht den kompletten Strang!), sondern die ursprünglichen "DSBx" (nicht nur "DSB0") ebenfalls mit aufzunehmen (schaue doch dazu in meine überarbeitete Version). Es ändert sich somit nichts mit dem "Kettenproblem"! Aber da hier bereits nun schon eine _DSM-Methode gesetzt wird (mit dem Parameter "AAPL,slot-name"), wird dieser in der sich später aufbauenden Kette "vererbt". Es geht also gar nicht darum, eine lange Kette zu bauen und ein Gerät direkt per _DSM-Methode zu benennen, sondern dass das Gerät, unabhängig davon wie es heißt und ob der Hersteller es auch benannt hat, dass dieses Gerät eben in der PCI-Sektion des Systemberichts/Systeminformationen auftaucht. Denn hier ist es für den Nutzer sehr einfach einzusehen, ob überhaupt ein Treiber für dieses Gerät geladen wurde. Ansonsten tappt man da im Dunkeln. Kannst du gern ausprobieren. _DSM-Methode von z.B. "DSB1" entfernen, schon sieht man an diesem Strang in der PCI-Sektion keine Geräte mehr (die natürlich dennoch funktionieren).

Hast du meine überarbeitete SSDT mal ausprobiert? Ich möchte ja nur helfen, einerseits Dinge zu verschlanken, andererseits wichtige Dinge zu integrieren.

Beitrag von „kuckkuck“ vom 4. September 2019, 12:08

Kurze Frage, wieso eigentlich die ganzen _STA Funktionen? Tauchen die Geräte in den original Tables nicht auf oder sorgt im Original irgendeine _OSI Konstruktion für die Deaktivierung der Devices? Inwiefern sollten die Device bereitgestellt werden, wenn ein anderes OS als macOS bootet? Hier sollte man evtl. darüber nachdenken vorhandene _OSIs zu verändern oder die gesetzten _STAs mit _OSIs zu versehen um cross-platform Support zu gewährleisten, Treiber wie OpenCore nutzen ACPI Tabellen auch bei nicht-macOS boot. Sorry, besitze selber keinerlei betroffene Hardware, deswegen kann ich nicht selber reinschauen.

> Zitat von DSM2
>
> persönlich eine andere Lösung für Onboard Chipsets die aber recht komplex ist.

Würdest du die vielleicht mal hier kurz beschreiben? Dann können wir mal drüber schauen, ob sich das irgendwie vereinfachen/verallgemeinern lässt. Ich weiß nicht, ob du dabei mit ACPI arbeitest, aber wäre definitiv interessant. 4 (oder eher 32) Augen sehen mehr als 2

Beitrag von „apfelnico“ vom 4. September 2019, 12:18

> Zitat von kuckkuck
>
> Kurze Frage, wieso eigentlich die ganzen _STA Funktionen? Tauchen die Geräte in den original Tables nicht auf

Nein, diese Devices gibt es nicht in der original ACPI. Siehst du schon allein daran, dass es als "Device" in der SSDT definiert ist, nicht "Scope" (also erweiterte Einträge eines vorhandenen Devices). Mit der Methode _STA wird der aktuelle Status abgefragt. Deaktivierung von z.B. PXSX erfolgt nicht direkt mit einer Methode, sondern ändern des Parameters "Name (_STA, Zero)". Somit wird der Weg frei für das neue Device "UPSB". Sonst würde alles an PXSX hängen, was grundsätzlich nun auch kein Problem wäre.

Beitrag von „kuckkuck“ vom 4. September 2019, 12:44

Jup, darauf hab ich nicht geachtet, mir war nicht bewusst inwiefern hier FW Settings die ACPI Tabellen beeinflussen und evtl. gesamte Devices streichen, wer weiß (obwohl, gibts sowas überhaupt?). Ich vermute mal die Devices stammen aus Apple-ACPI Dumps? Matcht der Treiber auf UPSB?

Mir ist bewusst wie _STA funktioniert, was ich sage ist, um es noch sauberer zu machen würde ich in diesem Fall noch _OSI Abfragen einbauen. Sprich:

Code

```
1. Scope (_SB.PCI0.RP01)
2. {
3. Scope (PXSX)
4. {
5. If (_OSI ("Darwin"))
6. {
7. Return (Zero)
8. }
9. Else
10. {
11. Return (0x0F)
12. }
13. }
14.
15. Device (UPSB)
16. {
17. Name (_ADR, Zero) // _ADR: Address
18. Method (_STA, 0, NotSerialized) // _STA: Status
19. {
20. If (_OSI ("Darwin"))
21. {
22. Return (0x0F)
23. }
24. Else
25. {
26. Return (Zero)
27. }
28.
29. }
30. }
31. }
```

Alles anzeigen

Zumindest bei UPSB oder Untergeräten wie DSBx sollte das so möglich sein.

Beitrag von „apfelnico“ vom 4. September 2019, 17:55

kuckkuck

Das würde doch reichen:

Code

```
1. Scope (_SB.PCI0.RP01)
2. {
3. Scope (PXSX)
4. {
5. If (_OSI ("Darwin"))
6. {
7. Name (_STA, Zero) // _STA: Status
8. }
9. Else
10. {
11. }
12. }
13.
14. Device (UPSB)
15. { ...
```

Alles anzeigen

Wenn "Darwin", dann PXSX ausblenden und normal fortfahren. Bei allem anderen wird es nicht ausgeblendet, und der TB-Controller wird wie gehabt an PXSX angehängt. Der Rest der SSDT ist somit hinfällig ...

Beitrag von „kuckkuck“ vom 4. September 2019, 18:04

Der Teil mit der offenen Else Funktion gefällt mir, dann kann die FW oder sonst ein OS hier noch komplett selbstständig walten solange Darwin nicht gecalled wird. Der wichtigste Teil ist damit getan.

Ob man den ganzen Rest mit aktivem Status anhängen will, wenn nicht macOS gebotet wird, ist ansichtssache. Da würde ich einfach noch für UPSB eine entsprechende if Konstruktion in die _STA Methode einbauen, macht ja keinen Sinn in diesem Fall weitere Devices einzubinden,

die absolut ins Leere laufen und unbedingt aktiv sein wollen.

Beitrag von „apfelnico“ vom 4. September 2019, 18:10

kuckkuck

Dann so

Code

```
1. DefinitionBlock ("", "SSDT", 2, "HACKI", "TBOLT3", 0x00000000)
2. {
3. External (_SB_.PCI0.RP01, DeviceObj)
4. External (_SB_.PCI0.RP01.PXSX, DeviceObj)
5. External (DTGP, MethodObj) // 5 Arguments
6.
7. Scope (_SB.PCI0.RP01)
8. {
9. If (_OSI ("Darwin"))
10. {
11. Scope (PXSX)
12. {
13. Name (_STA, Zero) // _STA: Status
14. }
15.
16. Device (UPSB)
17. {
18. Name (_ADR, Zero) // _ADR: Address
19. Method (_STA, 0, NotSerialized) // _STA: Status
20. {
21. Return (0x0F)
22. }
23.
24. Method (_DSM, 4, NotSerialized) // _DSM: Device-Specific Method
25. {
26. Local0 = Package (0x02)
27. {
28. "PCI-Thunderbolt",
```

```
One
}
DTGP (Arg0, Arg1, Arg2, Arg3, RefOf (Local0))
Return (Local0)
}

Device (DSB0)
{
Name (_ADR, Zero) // _ADR: Address
Method (_STA, 0, NotSerialized) // _STA: Status
{
Return (0x0F)
}

Method (_DSM, 4, NotSerialized) // _DSM: Device-Specific Method
{
Local0 = Package (0x04)
{
"AAPL,slot-name",
Buffer (0x09)
{
"Built In"
},

"PCIHotplugCapable",
One
}
DTGP (Arg0, Arg1, Arg2, Arg3, RefOf (Local0))
Return (Local0)
}

Device (NHI0)
{
Name (_ADR, Zero) // _ADR: Address
Name (_STR, Unicode ("Thunderbolt")) // _STR: Description String
Method (_STA, 0, NotSerialized) // _STA: Status
{
Return (0x0F)
}

Method (_DSM, 4, NotSerialized) // _DSM: Device-Specific Method
{
```

```
71. Local0 = Package (0x0B)
72. {
73. "AAPL,slot-name",
74. Buffer (0x07)
75. {
76. "Slot-3"
77. },
78.
79. "device_type",
80. Buffer (0x19)
81. {
82. "Thunderbolt 3-Controller"
83. },
84.
85. "model",
86. Buffer (0x2C)
87. {
88. "Intel JHL7540 Titan Ridge Thunderbolt 3 NHI"
89. },
90.
91. "name",
92. Buffer (0x23)
93. {
94. "Titan Ridge Thunderbolt Controller"
95. },
96.
97. "power-save",
98. One,
99. Buffer (One)
100. {
101. 0x00 // .
102. }
103. }
104. DTGP (Arg0, Arg1, Arg2, Arg3, RefOf (Local0))
105. Return (Local0)
106. }
107. }
108. }
109.
110. Device (DSB1)
111. {
112. Name (_ADR, 0x00010000) // _ADR: Address
```

```
Method (_STA, 0, NotSerialized) // _STA: Status
{
Return (0x0F)
}

Method (_DSM, 4, NotSerialized) // _DSM: Device-Specific Method
{
Local0 = Package (0x04)
{
"AAPL,slot-name",
Buffer (0x09)
{
"Built In"
},

"PCIHotplugCapable",
One
}
DTGP (Arg0, Arg1, Arg2, Arg3, RefOf (Local0))
Return (Local0)
}
}

Device (DSB2)
{
Name (_ADR, 0x00020000) // _ADR: Address
Method (_STA, 0, NotSerialized) // _STA: Status
{
Return (0x0F)
}

Method (_DSM, 4, NotSerialized) // _DSM: Device-Specific Method
{
Local0 = Package (0x04)
{
"AAPL,slot-name",
Buffer (0x09)
{
"Built In"
},

"PCIHotplugCapable",
```

```
One
}
DTGP (Arg0, Arg1, Arg2, Arg3, RefOf (Local0))
Return (Local0)
}

Device (XHC5)
{
Name (_ADR, Zero) // _ADR: Address
Method (_STA, 0, NotSerialized) // _STA: Status
{
Return (0x0F)
}

Method (_DSM, 4, NotSerialized) // _DSM: Device-Specific Method
{
Local0 = Package (0x10)
{
"AAPL,slot-name",
Buffer (0x07)
{
"Slot-3"
},

"model",
Buffer (0x22)
{
"Intel JHL7540 Titan Ridge USB 3.1"
},

"name",
Buffer (0x1F)
{
"Titan Ridge USB 3.1 Controller"
},

"USBBusNumber",
Zero,
"UsbCompanionControllerPresent",
One,
"AAPL,XHCI-clock-id",
One,
```

```
"IOPCIExpressCapabilites",
0x02,
"IOPCIHPType",
0x02
}
DTGP (Arg0, Arg1, Arg2, Arg3, RefOf (Local0))
Return (Local0)
}
}
}

Device (DSB4)
{
Name (_ADR, 0x00040000) // _ADR: Address
Method (_STA, 0, NotSerialized) // _STA: Status
{
Return (0x0F)
}

Method (_DSM, 4, NotSerialized) // _DSM: Device-Specific Method
{
Local0 = Package (0x04)
{
"AAPL,slot-name",
Buffer (0x09)
{
"Built In"
},

"PCIHotplugCapable",
One
}
DTGP (Arg0, Arg1, Arg2, Arg3, RefOf (Local0))
Return (Local0)
}
}
}
}
Else
{
}
}
```

239. }

Alles anzeigen

Beitrag von „Mork vom Ork“ vom 4. September 2019, 18:13

apfelnico

Ok, das mit der ganzen NOTIFY Routine habe ich soeben bei mir getestet. Diese scheint echt irrelevant zu sein. Hier scheinen also eher die korrekten Thunderbolt-Einstellungen im BIOS echten Einfluss auf "**HotPlug/HotSwap on the fly**" zu nehmen.

~~Den Teil kann ich also schonmal wieder aus dem Tutorial streichen~~. Tutorial entsprechend angepasst. Den Rest werde ich gerne ebenfalls auf beiden Boards testen.

Beitrag von „kuckkuck“ vom 4. September 2019, 18:13

apfelnico Das klingt nach einer Idee!

Beitrag von „apfelnico“ vom 4. September 2019, 18:35

Mork vom Ork

Prima, hatte ich es mir doch gedacht. Habe bei mir jetzt auch aufbauend auf der unten angehängten Version (lediglich Pfade geändert) die SSDT getestet, die If-Schleife funktioniert ebenfalls:

Edit: Eine gute Idee von kuckkuck

Denn mit OpenCore wird die SSDT immer in die ACPI geladen, egal was für ein System. Es gibt

keine Unterscheidung wie bei Clover zwischen macOS und Windows. Somit wird diese SSDT nur aktiv, wenn auch "Darwin" am Start ist.

Edit2: USB werde ich mir noch anschauen. Und wenn das alles ein Ende hat, dann könnte vielleicht Sascha_77 einen Generator schreiben, der den vorhandenen TB-Controller erkennt und an welchem Gerät der hängt und eine fertige SSDT ausspuckt. Wäre echt fein ...

Beitrag von „locojens“ vom 5. September 2019, 17:56

In dem Zusammenhang finde ich diesen Artikel ja spannend... wird da Apple mitschwimmen?

https://www.techadvisor.co.uk/...h-industry/usb-4-3693906/

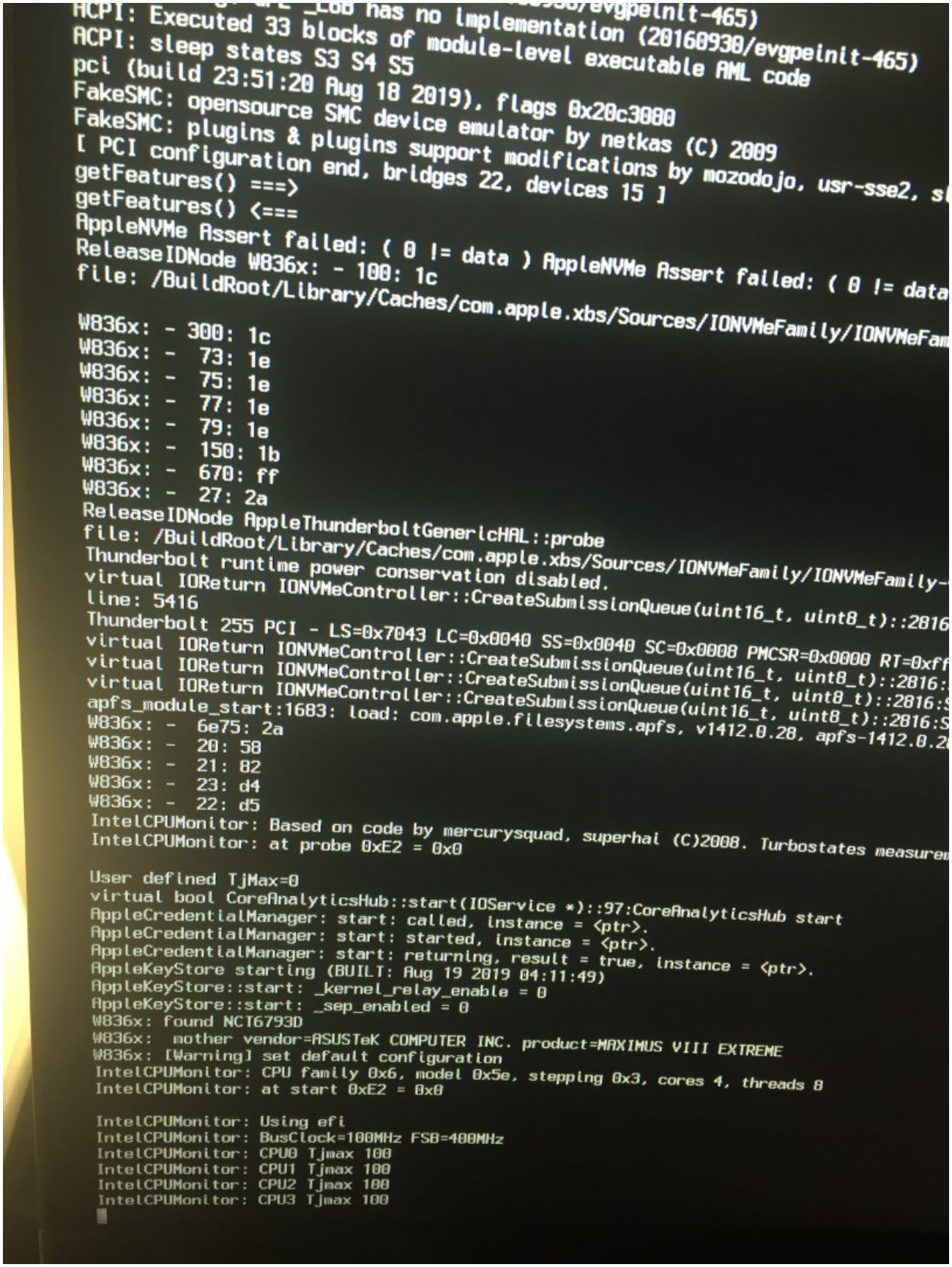
Quelle: https://www.techadvisor.co.uk

Beitrag von „Mork vom Ork“ vom 9. September 2019, 21:21

ich kapere nur ungern fremde Threads, aber ich benötige hier mal die Profis der ASUS-Fraktion:

ich habe gestern mein ***ASUS Z170 Maximus VIII Extreme*** aus der Mottenkiste geholt und reaktiviert.

Derzeit bleibt der Bootvorgang mit aktuellem CLOVER (EFI findet Ihr im Anhang) immer an dieser Stelle hängen:


ACPI: Executed 33 blocks of module-level executable AML code
ACPI: sleep states S3 S4 S5
pci (build 23:51:20 Aug 18 2019), flags 0x20c3080
FakeSMC: opensource SMC device emulator by netkas (C) 2009
FakeSMC: plugins & plugins support modifications by mozodojo, usr-sse2,
[PCI configuration end, bridges 22, devices 15]
getFeatures() ===>
getFeatures() <===
AppleNVMe Assert failed: (0 != data) AppleNVMe Assert failed: (0 != data
ReleaseIDNode W836x: - 100: 1c
file: /BuildRoot/Library/Caches/com.apple.xbs/Sources/IONVMeFamily/IONVMeFam
W836x: - 300: 1c
W836x: - 73: 1e
W836x: - 75: 1e
W836x: - 77: 1e
W836x: - 79: 1e
W836x: - 150: 1b
W836x: - 670: ff
W836x: - 27: 2a
ReleaseIDNode AppleThunderboltGenericHAL::probe
file: /BuildRoot/Library/Caches/com.apple.xbs/Sources/IONVMeFamily/IONVMeFamily-
Thunderbolt runtime power conservation disabled.
virtual IOReturn IONVMeController::CreateSubmissionQueue(uint16_t, uint8_t)::2816
line: 5416
Thunderbolt 255 PCI - LS=0x7043 LC=0x0040 SS=0x0040 SC=0x0008 PMCSR=0x0000 RT=0xff
virtual IOReturn IONVMeController::CreateSubmissionQueue(uint16_t, uint8_t)::2816:
virtual IOReturn IONVMeController::CreateSubmissionQueue(uint16_t, uint8_t)::2816:
apfs_module_start:1683: load: com.apple.filesystems.apfs, v1412.0.28, apfs-1412.0.2
W836x: - 6e75: 2a
W836x: - 20: 58
W836x: - 21: 82
W836x: - 23: d4
W836x: - 22: d5
IntelCPUMonitor: Based on code by mercurysquad, superhai (C)2008. Turbostates measure
IntelCPUMonitor: at probe 0xE2 = 0x0
User defined TjMax=0
virtual bool CoreAnalyticsHub::start(IOService *)::97:CoreAnalyticsHub start
AppleCredentialManager: start: called, instance = <ptr>.
AppleCredentialManager: start: started, instance = <ptr>.
AppleCredentialManager: start: returning, result = true, instance = <ptr>.
AppleKeyStore starting (BUILT: Aug 19 2019 04:11:49)
AppleKeyStore::start: _kernel_relay_enable = 0
AppleKeyStore::start: _sep_enabled = 0
W836x: found NCT6793D
W836x: mother vendor=ASUSTeK COMPUTER INC. product=MAXIMUS VIII EXTREME
W836x: [Warning] set default configuration
IntelCPUMonitor: CPU family 0x6, model 0x5e, stepping 0x3, cores 4, threads 8
IntelCPUMonitor: at start 0xE2 = 0x0
IntelCPUMonitor: Using efi
IntelCPUMonitor: BusClock=100MHz FSB=400MHz
IntelCPUMonitor: CPU0 Tjmax 100
IntelCPUMonitor: CPU1 Tjmax 100
IntelCPUMonitor: CPU2 Tjmax 100
IntelCPUMonitor: CPU3 Tjmax 100

und ich komme partout nicht weiter. Im letzten PCIe Port hängt eine GC Titan Ridge, im ersten PCIe Port eine RX460.

Ich habe schon sämtliche BIOS-Einstellungen und Werteveränderung in der "config.plist" durch... ohne Erfolg. Er bleibt immer wieder genau hier hängen.

So i need your HELP, please 

PS: Booten würde er hier ein macOS Catalina ß7

Beitrag von „CMMChris“ vom 9. September 2019, 23:45

DSDT RTC Patch getestet? Bin mir zwar nicht sicher ob es daran liegt weil nach den dafür typischen Meldungen noch anderer Kram kommt, aber man weiß ja nie.

Beitrag von „Mork vom Ork“ vom 10. September 2019, 00:59

Habe es zumindest ersteinmal unter MOJAVE zum Laufen bekommen. Bastele derzeit noch am Feintuning... melde mich, wenn es brauchbare Fortschritte gibt

- - - - -

Habe ich mir doch gedacht: ohne die in meinem Tutorial beschrieben Funktion "AIC Location: NB PCIE D01F2" ist auf meinem ASUS Z170 Maximus VIII Extreme **kein** echtes TB3-HotPlug/HotSwap möglich!

Egal welche Einstellungen ich auch durchspiele: er macht mir **kein** HotPlug/HotSwap im laufenden Betrieb. **Ja**, er setzt mir das abgesteckte TB-Device auf rot, aber **nein**, er plugt es mir nicht an einem

anderen Port wieder sauber an. **Keine Chance**. Ich habe es auch mal mit der SSDT-Datei (am Anfang dieser Seite) von apfelnico versucht - mit dem selben Ergebnis: abgestöpselt werden wird erkannt,

an anderem Port wieder anstecken bleibt leider ergebnislos.

- - - - -

Entgegen der Auskunft von GIGABYTE ist es mir nun aber doch gelungen, auf dem Z170X SOC FORCE eine GC Titan Ridge zum Laufen zu bringen.

Ich musste dazu nur mit AMIBCP die Thunderbolteinstellungen im BIOS "aufbohren" und ein wenig rum experimentieren. Zwar ist das verhalten noch

typisch "Hackintosh"-mässig - sprich kein natives HotPlug/HotSwap - aber das wird noch. Somit ist bewiesen, das Thunderbolt nicht nur auf x4-Slots

läuft, sondern auch in x8 oder x16 Slots, man muss eben nur die korrekten Settings dafür finden und setzen.

- - - - -

Schade...

Frage:	Gibt es ein Firmwareupdate für Thunderbolt3 für die GC Titan Ridge rev1.0 Thunderbolt 3 Karte? Eines, bei der die NVMe Version groesser als 0x23 (35) ist? Wenn ja, wo kann man dieses FirmwareUpdate herunterladen? Is there a firmware-updatetool availllable to update the thunderbolt 3 NVMe firmware to something newer than rev. 23 for the Gigabyte GC Titan Ridge rev1.0 thunderbolt3 PCIe card? If so, whare can it be downloaded?	23.08.2019 18:14

Sehr geehrter GIGABYTE Kunde,

Update von unserem Team:

Antwort: Zur Zeit steht keine aktuellere Firmware Für die GC-Titan Ridgezur Verfügung.

vielen Dank für Ihre Anfrage.

Beitrag von „maxman“ vom 18. September 2019, 11:21

hat jemand das patched bios für ein asrock fatal1ty z370 gaming m-itx und kann es mir schicken? sowie die ssdts?

wenn ich mein gepatches bios laden will, wird gemeckert mit "secure flash bios check failed" oder so

update: es gibt wohl möglichkeiten den secure flash zu deaktivieren.. scheint mir aber viel try'N'error und ich wollt mir nicht gleich mein neues board zerschiessen

bin auf der neusten bios version 4.10

danke & gruß max

Beitrag von „Mork vom Ork“ vom 18. September 2019, 14:10

Du kannst ein "gemodded" Bios nicht via Secure Flash auf das Board spielen.

Dazu musst du FPTW64 nutzen. Hänge ich heute Abend mal hier an.

Das gemoddete BIOS wird mit folgendem Befehl geflasht: fptw64.exe -bios -f BIOSNAME.xyz (XYZ steht dabei für die Endung)

Ich melde mich heute Abend nochmal ausführlich dazu, da ich im Moment @work bin.

Beitrag von „maxman“ vom 18. September 2019, 14:29

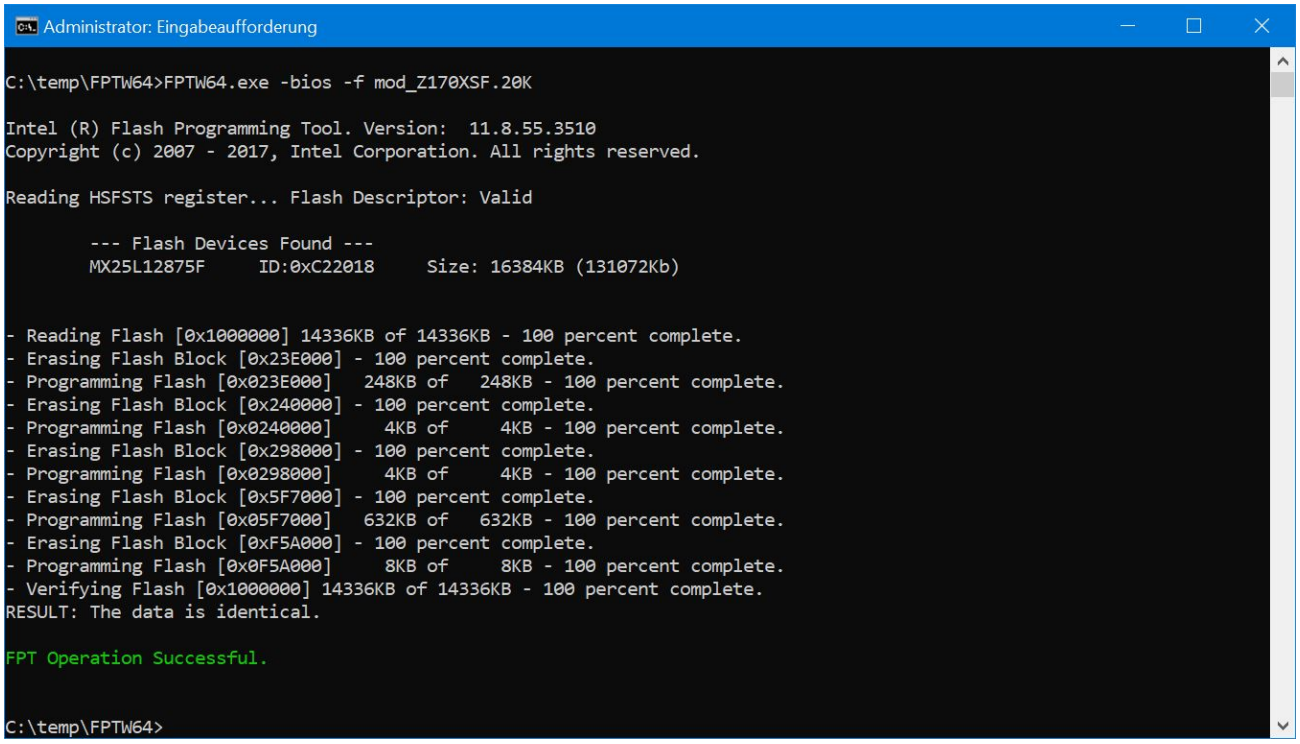
wenn ich das gemodded image speicher bleibt es bei "Z37GIX_4.10"

dann einfach mit fptw64 rein? das programm hab ich

danke dir aufjedenfall für deine arbeit!

Beitrag von „Mork vom Ork“ vom 18. September 2019, 14:43

wie gesagt, ich flashe ein gemodded BIOS immer mit "fptw64.exe -BIOS -F Z37GIX_4.10"



Beitrag von „maxman“ vom 18. September 2019, 15:09

hat wunderbar geklappt, brauchte nur die 11.8 version von fp[illegible] bios drin..und deutlich mehr einstellungen im bios. weiter gehts

update: alle einstellungen im bios sind nun so wie deinem screenshot.

leider finde ich im ioexplorer gar keine thunderbolt hardware...und es geht auch nicht, im clover screen das thunderbolt uad interface einzuschalten..und dann zu verwenden..

mac18,1 habe ich bei mir stehen im "über mac" muss das zwingend ein anderer sein, damit macos das überhaupt aufm schirm hat mit thunderbolt?

ansonsten habe ich ein vanilla macos drauf.....ohne multibeast...

danke!

Beitrag von „MacGrummel“ vom 19. September 2019, 11:02

maxman Der kleine iMac 18,1 hat zwar nur einen Schirm mit HDMI-Auflösung, aber trotzdem zwei Thunderbolt-3-Ports mit jeweils 4k-Auflösung über die Intel 640. Und mit TB läuft die Definition auch in dem kleinen Z-Box Mi553 meiner Freundin..

Läuft das Tool auch ohne WindDOS? Mein kleines AsRock-z390n-Board ist noch im Test-Stadium. Aber mit TB-hotplug wäre ich ja einen guten Schritt weiter!

P.S.: Genau lesen schadet nie, sorry. Ich hab mich schon gefreut, mich hier anhängen zu können. Aber wenn man über Tage nur mit dem iPhone ins Forum geht: Mein Board ist ein AsRock z390 Phantom Gaming itx/ac. Und ich hab auch zZt. kein Windows für's BIOS-Patchen oder andere Tools zur Verfügung..

Beitrag von „DSM2“ vom 22. September 2019, 11:04

Das Bios des ASRock Z390 Phantom Gaming ITX/ac benötigt keinerlei Patchereien oder Windows Tools.

MacGrummel

Beitrag von „maxman“ vom 24. September 2019, 11:18

Leider bekomme ich mein Apollo Twin Thunderbolt Interface so gar nicht zu laufen. (unabhängig vom hotplug)

Unter Windows 10 läufts sofort. Dazwischen ist ein Apple T3 auf T2 Adapter.

Hat jemand vlt noch Tipps?

Beitrag von „DSM2“ vom 24. September 2019, 11:52

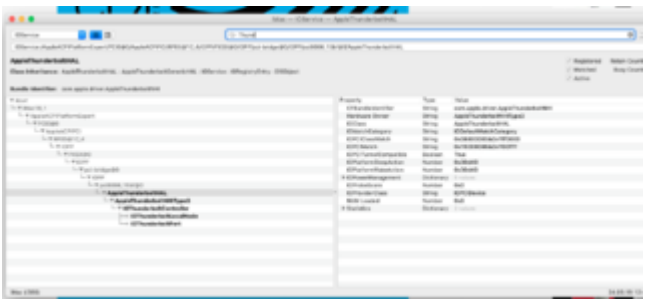
Ich würde vom Bauchgefühl ganz stark auf Bios Settings tippen.

Was hast du im Bios in Bezug auf Thunderbolt eingestellt?

Beitrag von „maxman“ vom 24. September 2019, 12:03

Okay habe das Interface jetzt endlich zum laufen bekommen. Aktuell wieder das normale asrock bios.

Thunderbolt Adapter ganz raus gezogen...reboot.. dann im clover boot screen wieder dran. tada interface wird erkannt:



bedeutet jetzt wieder bios modden und die richtigen einstellungen finden? und dann eine passende ssdt erstellen?

hot plug ist mir gar nicht 100% so wichtig...nur das aus und einstecken nervt.... kann man da nicht mit den wait times für thunderbolt was regeln?

Gruß & Danke

Beitrag von „DSM2“ vom 24. September 2019, 12:05

Wofür den jetzt genau den Bios mod?

Läuft doch, einfach jetzt nur die ssdt an dein System anpassen und gut.

Beitrag von „maxman“ vom 24. September 2019, 12:05

> Zitat von DSM2
>
> Wofür den jetzt genau den Bios mod?
>
> Läuft doch, einfach jetzt nur die ssdt an dein System anpassen und gut.

kabel raus/rein vor nem boot ist mega unpraktikabel...

was konkret bringt mir die ssdt anpassung überhaupt? nur hotplug?

Beitrag von „DSM2“ vom 24. September 2019, 12:08

Wieso Kabel rein raus?

Konfigurieren deine Kiste korrekt und dann funktioniert alles wie es soll.

Welche Thunderbolt Lösung Verwendest du?

Onboard Chip? Thunderbolt Karte und falls ja welche?

Beitrag von „maxman“ vom 24. September 2019, 12:10

ich nutze ein asrock z370 fatal1ty gaming itx... also onboard intel thunderbolt

wenn ich die kiste einfach so mit angestecktem adapter boote finde macos / universal audio keine karte..und kein thunderbolt.

nur wenn ich den adapter im clover boot anstecke gehts...

ssdt anpassung würde das problem lösen?

danke für deine hilfe

Beitrag von „DSM2“ vom 24. September 2019, 12:20

Einfach testen würde ich sagen!

Alle Schritte sind beschrieben.

Am ASRock Z390 itx greift diese Lösung nicht aber vielleicht hast du ja Glück.

Beitrag von „maxman“ vom 24. September 2019, 12:30

wie editiere ich die ssdt? textfile ist es ja nicht..

und welche könnte ich als start nehmen?

der link in der anleitung ist ein 404:

"Beispiel SSDT für Thunderbolt: hackintosh-forum.de/attachment/111617/"

Beitrag von „DSM2“ vom 24. September 2019, 13:08

Nimm die SSDT am Ende des ersten Posts und nicht aus der Beschreibung, weitere Option die SSDT von apfelnico.

Anhand von diesem Beispiel dann laut How To vorgehen und den Path überprüfen an dem dein Thunderbolt Controller hängt,

diesen dann entsprechend in der SSDT anpassen, falls dieser von dem in der SSDT hinterlegten Path abweicht.

Um den Path zu überprüfen benötigst du IORegistryExplorer und für das editieren der SSDT ist meiner Meinung nach die beste Lösung MaciASL

Beitrag von „maxman“ vom 27. September 2019, 21:53

leider klappt die verwendung mit meinem apollo twin thunderbolt interface noch nicht...

bei meiner online recherche habe ich das gefunden:

https://github.com/osy86/HaC-M…ails/thunderbolt-3-fix.md

vlt tut sich ja im bereich thunderbolt und dem z370 m-itx noch mehr in zukunft

---

Beitrag von „DSM2“ vom 27. September 2019, 22:29

Einen Interessen Ansatz beinhaltet es, bei dem der Verfasser zwar ein kleines Problem hat, für welches ich aber womöglich die passende Lösung kenne...

Beitrag von „maxman“ vom 27. September 2019, 22:56

Meine aktuellen bios einstellungen:

Beitrag von „kolutshan“ vom 11. Oktober 2019, 20:53

Hi, ich habe jetzt mehrfach versucht die verschiedenen SSDT files (von Mork vom Ork und apfelnico) anzupassen. Leider komme ich nicht sehr weit damit, da mir MaciASL direkt compiler error zurückmeldet und ich die Änderungen nicht speichern kann.

Leider ist mir auch nicht wirklich klar, was genau ich falsch mache... Die Adresse meines Titan Ridge Controllers lautet: _SB_.PCI0.RP21.PXSX

Falls jemand eine Idee hat, würde ich mich sehr freuen.

Vielen Dank!

Edit: ok, jetzt hat es funktioniert. Die Lösung war eine ältere Version von MaciASL. Die Version mit der ich es zu erst versucht habe, hat beim öffnen von Datein den Inhalt verändert und vollkommen anders dargestellt als im Eingangsposting zu sehen. Dadurch hat das natürlich alles keinen Sinn mehr ergeben und der Compiler hat nur noch Unsinn angezeigt. Mit einer älteren Version hat es nun geklappt.

Beitrag von „hitman20“ vom 18. Oktober 2019, 19:03

Ich habe meinen Dell XPS 15 auf Catalina geupdated und wollte mal die Thunderbolt SSD testen, allerdings funktioniert der Thunderboltanschluss nach einem Ruhezustand nicht mehr. In der IOReg sehe ich dann nur noch RP15 ohne weitere Informationen. Mein TB Anschluss hängt auf RP15. Gibt es hier noch eine Möglichkeit, dass dieser nach dem Sleep wieder funktioniert? Die SSDT habe ich nur auf RP15 angepasst. Wenn der Rechner gestartet wurde, geht es ohne Probleme, bis auf nach dem Ruhezustand.

Beitrag von „nori_1000“ vom 26. Januar 2020, 19:20

Hallo ich komme nicht weiter! da ich nur im moment USB-C habe, aber kein hot plug!

wenn ich die HHD vor dem Booten einstecke funktioniert sie, wenn ich sie wider ausstecke und dann wieder ein kommt sie nicht wider!

anbei meine EFI

Board ist Asrock Z390 Phantom ITX

Beitrag von „apfelnico“ vom 26. Januar 2020, 19:31

Ich denke, du redest von Thunderbolt. Hot Plug für USB-C funktioniert wie an jedem anderen USB-Port selbstverständlich.

Beitrag von „nori_1000“ vom 26. Januar 2020, 19:49

Thunderbolt kann ich noch nicht Test, ich meine USB-C wenn ich die Platte nach dem Boote anhänge wird sie nicht erkannt!

Hilfe hat jemand ein Firmware für Thunderbolt für mein Board,

Asrock Z390 Phantom Gaming ITX

Habe es aus Blödheit mit einem von Gigabyte überschriebenen! Danke

---

Beitrag von „DSM2“ vom 9. Februar 2020, 13:26

Das war keine gute Entscheidung, zumal man wenn man sowas vorhat immer ein Backup ziehen sollte!

Da wirst du recht aufgeschmissen sein, ich wage zu bezweifeln das einer der User mit einem Z390 Phantom Gaming ITX einen EEPROM Reader hat.

Kannst ein Board ordern und sowohl das neue als auch dein totgeflashtes zu mir schicken, dann kann ich versuchen das ganze zu fixen, falls du den Chip nicht schon gegrillt hast.

Im Notfall hab ich aber Ersatz da.

Anschließend retournierst du das neue Board wieder.

Beitrag von „nori_1000“ vom 9. Februar 2020, 14:01

DSM2. Der Chip ist noch in Ordnung! Nur unter Windows bekomme ich Blue Screen sobald ich ein Thunderbolt Gerät anstecke! Unter MacOS funktioniert es aber ohne Hot Plug, das war auch

der Grund des Flashes?

Komme gerne auf dein Angebot zurück, wenn ich keine andere Lösung finde! Mit was liest du den Chip aus und flasht ihn? Grüße

---

Beitrag von „nori_1000“ vom 12. Februar 2020, 17:07

DSM2. Juhu Dank dem Ausgezeichnetem Support von Asrock sie haben mir die Aktuelle Firmware V20 zugesendet, ist für z370 Gaming_ITX und z390 Phantom Itx,

Nun geht auch Hot Plug!! wer sie benötigt soll sich melden!

Danke noch für deine Unterstützung! 

Beitrag von „DSM2“ vom 12. Februar 2020, 17:39

Werf Sie doch einfach hier rein zum Download.

Hotplug ging an beiden auch so schon aber vielleicht möchte jemand mit dieser ja basteln.

Beitrag von „nori_1000“ vom 12. Februar 2020, 18:02

Hier die Firmware V20 für Asrock Z370 itx und Z390 itx Thunderbolt

Habe aber noch das Problem das USB-C an diesem Port nicht unter Catalina Funktioniert

---

Beitrag von „apfelnico“ vom 12. Februar 2020, 19:16

nori_1000

Wie sieht denn deine SSDT dazu aus, wird die beim Systemstart geladen und eingebunden? Check IORegistryExporer bzw Systembericht.

Kannst du die mal hochladen?

Beitrag von „nori_1000“ vom 12. Februar 2020, 20:53

@apfelnico

Hier mal was du gewünscht hast wenn du mehr brauchst bitte melden!

Beitrag von „apfelnico“ vom 12. Februar 2020, 21:09

nori_1000

Im Ioregistryexplorer werden keine weitere DSBx angezeigt, an DSB2 hängt jedenfalls der XHCI-Controller.

Hast du im BIOS bei den Thunderbolteinstellungen auch USB Support aktiv?

Probiere mal diese SSDT im Austausch:

Beitrag von „nori_1000“ vom 12. Februar 2020, 22:16

@apfelnico

also habe folgendes getestet: mit deiner SSDT

Thunderbolt Festplatte vor dem Booten angeschlossen wird erkannt, Hot Plug geht, wenn ich dann eine USB-C Festplatte anschließe wird die USB Platte nicht erkannt, danach geht Thunderbolt auch nicht mehr!

Wenn ich nach dem Systemstart die TB HDD anschliesse wird sie auch nicht mehr erkannt!

Usb-c an dem TB Board funktioniert unter Macos nicht, under windows10 kein Problem!

im Bios kann ich den Usb support nicht aktivieren wenn Pre-Boot ACL aktiv ist

aber auch mit Aktivierung geht usb nicht!

Beitrag von „apfelnico“ vom 12. Februar 2020, 22:39

nori_1000

Kannst du mir bitte zwei IORegistryExplorer-Files (File-Menü, Save) schicken? Ein Screenshot ist nicht so aussagekräftig.

Und zwar eins mit SSDT-TBOLT3 und eins ohne (SSDT aus "patched" entfernen, neu starten, nun IORegistryExplorer erneut).

Ich hab da so einen Verdacht ...

Danke.