Erledigt RTCMemoryFixup.kext

Beitrag von „roqueeee“ vom 11. Februar 2019, 17:24

RTCMemoryFixup - github.com

Habe gerade entdeckt, dass es mittlerweile eine Kext gibt, die RTC Fehler/Resets beheben soll. Steige noch nicht ganz durch, wie man die Kext nutzen kann. Scheinbar kann man per Boot-arg bestimmt kritische Abschnitte der RTC ausschließen.

Hat hier jemand schon Erfahrung mit der Kext gesammelt?

Mein Board (P55) benötigt bisher einen DSDT-Patch oder Clover-on-the-fly-Patching um Cmos resets nach einem Neustart zu verhindern. Wegen der RTC-Patches funktionierte leider Hibernation nie bei mir. Wäre natürlich super, wenn sich mit der Kext da was zurechtbiegen ließe. Daher mein Interesse.

Grüße

Beitrag von „derHackfan“ vom 12. Februar 2019, 18:35

Einfach mal das Kext in der EFI im Ordner Other ablegen hast du probiert?

Beitrag von „roqueeee“ vom 14. Februar 2019, 09:24

Wenn ich die Kext einfach nur injiziere habe ich nach einem Neustart einen Biosreset (Habe vorher in Clover „FixRTC“ deaktiviert).

Ich werde evtl. am Wochenende, wenn etwas Zeit ist, mal mit den boot args rumspielen und gucken ob sich was ergibt.

Von 00 bis FF wird aber eine lange Testreihe werden (256) und wenn man ganze Bereiche abwählt wird es eine sehr lange Prozedur werden

RTCMemoryfixup - insanelymac.com

Daran werde ich mich auch noch orientieren.

Beitrag von „CMMChris“ vom 14. Februar 2019, 09:48

Wozu die Kext nutzen wenn's ohne auch läuft?

Beitrag von „roqueeee“ vom 15. Februar 2019, 10:34

Naja, AppleRTC und FixRTC beheben eigtl. nur die Symptome und lösen das Problem nicht. Nämlich, dass die RTC nicht einwandfrei implementiert ist. Laut Entwickler: "..with RTCMemoryFixup it is possible to use all banks of cmos memory (256 bytes), and simulate writing into some critical bytes (you need to find which ones)?". Theoretisch ziehe ich solche Lösungen vor. Da die Kext aber scheinbar relativ ungebräuchlich ist und es auch nur wenig Dokumentation gibt, hatte ich gehofft, dass hier jemand schon mal Erfahrungen damit gemacht hat.

Die Devise "never change a running system" ist ja auch langweilig, da lernt man nichts neues

---

Beitrag von „shark“ vom 15. Februar 2019, 10:40

Hat das auch was mit rtc wakes beim ruhezustand zu tun, oder ist es eine andere baustelle?

Beitrag von „roqueeee“ vom 15. Februar 2019, 10:57

Also meine Problematik sind Cmos-Resets nach einem Neustart. AppleRTC und FixRTC sind auch darauf ausgelegt.

Grundsätzlich sind wakes aus dem Ruhezustand sehr häufig auf falsch konfiguriertes USB in Kombination mit dem Port Limit von macOS zurückzuführen. Hast du dich in die Richtung schon schlaugemacht?

Grüße

Beitrag von „CMMChris“ vom 15. Februar 2019, 11:01

RTC Wakes haben nichts mit USB zu tun. Power Nap nutzt RTC Wakes. Entweder ist also Power Nap nicht abgeschaltet in den Einstellungen oder durch ein Darkwake Flag aktiviert. Darkwake = 10 aktiviert Power Nap z.B. auch.

Beitrag von „roqueeee“ vom 15. Februar 2019, 11:03

Okay, danke Chris für die Klarstellung! Nicht meine Baustelle 

Beitrag von „shark“ vom 15. Februar 2019, 12:52

ich habe bereits darkwake=no und darkwake=0 getestet.

Powernap ist ausgeschaltet.

hier das log:

2019-02-12 11:41:55.592440+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-12 11:41:55.592442+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-12 14:01:46.814471+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-12 14:01:46.814473+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-13 16:57:49.372978+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: RTC (Alarm)

2019-02-13 16:57:49.372980+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: RTC (Alarm)

2019-02-13 18:58:38.129452+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-13 18:58:38.129454+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-13 21:39:07.194298+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-13 21:39:07.194300+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-13 22:03:25.201876+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-13 22:03:25.201877+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-13 22:30:56.951761+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: RTC (Alarm)

2019-02-13 22:30:56.951763+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: RTC (Alarm)

2019-02-14 00:31:41.226027+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-14 00:31:41.226029+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-14 09:57:44.434633+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-14 09:57:44.434635+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-14 10:46:42.939146+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-14 10:46:42.939148+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-14 12:19:51.446566+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-14 12:19:51.446568+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-14 14:28:40.473922+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: RTC (Alarm)

2019-02-14 14:28:40.473924+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: RTC (Alarm)

2019-02-14 16:29:28.212518+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

2019-02-14 16:29:28.212520+0100 localhost kernel[0]: (AppleACPIPlatform) AppleACPIPlatformPower Wake reason: PBTN (User)

Beitrag von „kuckkuck“ vom 15. Februar 2019, 13:30

Grundsätzlich existieren für uns unter CMOS Offsets bis Byte 256. Die bekannten Patches limitieren das ganze auf 128 Byte, was im Normalfall das Problem bereits behebt.

Sollte keiner der bekannten Patches (wie DSDT oder config Fix) eingebaut sein und man benutzt RTCMemoryFixUp, wird wieder die volle Bandbreite benutzt (256 Byte) und dementsprechend kann es wieder zu CMOS resets etc kommen (Inkompatibilität von AppleRTC und Firmware).

Es gilt jetzt also die problematischen Offsets zu finden und zu definieren. Daraufhin werden die entsprechenden Offsets blockiert und können keinen reset mehr erzeugen, RTCMemoryFixUp simuliert jedoch weiterhin Read/Write auf diese Offsets.

Mir ist kein Weg bekannt die problematischen Offsets aus einem Log o.ä zu lesen, das heißt es gilt testen. Ist RTCMemoryFixUp installiert und der Hacky resettet sich nach Sleep/Reboot, kann es losgehen.

Die Offsets 00-0D (0-**13**) sind unkritisch, die brauchen wir nicht testen. Die Idee ist jetzt erst einmal die problematische Memory Bank zu finden (die Erste geht von 00-7F/0-127 und die Zweite von 80-FF/128-256) (im Worst Case haben beide ein Problem).

Dafür excluden wir per Bootarg erstmal 13-127, also **rtcfx_exclude=0D-7F** und testen durch Sleep, Wake und Reboot. Sollte das Problem behoben sein, müssen wir zwischen 0D-7F genauer die problematischen Offsets finden.

Sollte das Problem nicht behoben sein, testen wir die zweite Bank, also **rtcfx_exclude=80-FF**.

Sagen wir mal, das Problem ist jetzt erst behoben, nun können wir im Bereich 80-FF (128-256) weiter testen.

Eine Taktik um nicht jeden Bereich einzeln testen zu müssen, ist den Bereich immer zu halbieren, um zu erfahren in welcher Hälfte das Problem liegt. Hierfür bietet sich jetzt also zB folgendes an: **rtcfx_exclude=80-C0**. Existiert das Problem mit diesen Excludes ebenfalls nicht, können wir im Bereich 80-C0 (128-192) weiter testen (zB wieder die Hälfte). Existiert das Problem hingegen wieder, können wir uns den Bereich C0-FF weiter anschauen um hier den problematischen Offset(-Bereich) zu finden. Und so weiter bis man keinen Bock mehr hat...

Schon an diesem Punkt ist die Methode cleaner als das limitieren auf 128 Bytes only (RehabMan Methode), da nun bereits über 128 Bytes verfügbar sind – eine Verbesserung.

Beitrag von „roqueeee“ vom 15. Februar 2019, 15:19

Schöne Erklärung kuckkuck! Wollte gerade Anfangen, einen kleinen Guide zu schreiben, aber das muss ich nun nicht mehr. Wenn du früher geschrieben hättest, hätte ich mir ein paar graue Haare erspart.

Bei meinem P55 USB3 Board gibt es 2 kritische Bereiche: 0x58-0x59 und sowie 0xB4-0xB7.

Wenn das schreiben in 58 und 59 erlaubt ist, habe ich nach einem Neustart einen kompletten Bios-Reset.

Wenn das schreiben in B4-B7 erlaubt ist, werden meine OC-Einstellungen auf Auto zurückgesetzt, was, wie man sich vorstellen kann, für die Stabilität des Systems nicht gerade förderlich ist.

Dabei ist B4=CPU PLL, B5=System Memory Multiplier, B6=CPU Vcore, B7=VTT Voltage.

Kann natürlich sein, dass noch weitere Bereiche meiner RTC beschrieben werden, die eigentlich nicht verändert werden sollten. Das kann ich aber nicht weiter nachvollziehen. In der Hinsicht sind AppleRTC oder FixRTC wahrscheinlich die sichereren Lösungen.

Mit boot-arg "rtcfx_exclude=B4-B7,58-59" gibt es bei mir keine Probleme mehr. Dies sollte für viele Gigabyte H55- und P55-Boards gelten, da die sich in der Regel sehr ähneln. Aber man muss es natürlich selber testen!

Mein ursprünglicher Plan war ja, doch noch einmal zu versuchen, ob ich Hibernate zum Laufen kriege.

Kurzer Hintergrund: IOHibernateRTCVariable speichert in der RTC in den Bereichen 0x80 bis 0xAB Infos zu Hibernate, am wichtigsten den Encryption key für das Hibernate-Image.

Wenn ich im BIOS S3 aktiviere und in macOS "pmset -a hibernatemode 25" aktiviere startet der Rechner nach dem Versuch in den Tiefschlaf zu gehen neu und ich habe einen kompletten Biosreset. Wenn ich nun mit RTCMemoryfixup die Bereiche 80-AB schreibschütze klappt Hibernate immer noch nicht, aber ich habe keinen Biosreset mehr. Der Rechner startet einfach ganz normal neu.

Also dachte ich, ich benutze die HibernationFixup.kext. Die versucht den Key in den NVRam anstatt in die RTC zu schreiben.

Leider gibt es einen Haken: Seit High Sierra kriege ich auf meinem Board keinen nativen NVRam mehr zum laufen, und HibernationFixup braucht nativen NVRam.

Lange Rede kurzer Sinn: Hibernate läuft immer noch nicht, aber zumindest zerschießt mir macOS nicht mehr das BIOS wenn es versucht zu schlafen.

Nächste Baustelle wäre, den nativen NVRAM wieder zum Laufen zu kriegen. Aber das verschiebe ich erstmal.

Beitrag von „kuckkuck“ vom 15. Februar 2019, 17:55

Sehr interessanter Beitrag, danke dafür!

Ich würde dich sehr gerne dazu anregen vielleicht doch einen kleinen Guide mit deinen eigenen Worten zu schreiben und auf die von dir erwähnten Ziele noch etwas näher einzugehen, ich finde das sehr interessant! Vielleicht willst du ja einen Thread eröffnen und dort deine Schritte in Richtung Hibernate dokumentieren (oder hier), ich denke das könnte für viele nützlich und interessant sein, Hibernate ist schon ein wenig ein Sahnehäubchen auf der Torte  Die parallele zu S3 ist ja auch ein schöner Fund...

Beitrag von „roqueeee“ vom 15. Februar 2019, 19:00

Den Guide für RTCMemoryfix könnte ich nicht besser schreiben. Entscheidend ist es, die Methode mit dem Halbieren der Offsets zu nutzen, sonst sitzt man noch ein Jahr später dran. Zum Glück bin ich da auch drauf gekommen.

Ein kleiner Zusatz unabhängig von RTCMemoryFix:

Wenn ich den AppleRTC-Patch (Kext Patch) nutze, habe ich keine Bios-Resets mehr, aber die OC Einstellungen werden auf Auto gestellt (0x58 und 0x59 werden nicht überschrieben, aber 0xB4-0xB7). Das spricht meiner Meinung dafür, dass die Schreibvorgänge in B4-B7 durch einen anderen Teil des Systems verursacht werden, nicht durch AppleRTC.

Man sollte also eher den FixRTC-Patch oder einen DSDT Patch nutzen, als den AppleRTC Kext Patch. Oder eben RTCMemoryFix. Zumindest bei mir löst AppleRTC das Problem nicht vollständig.

Wenn ich das Thema nativer NVRAM weiterverfolge, kann ich gerne einen neuen Beitrag schreiben. Allerdings habe ich noch keinen Ansatz, wie ich da weiter vorgehen kann.

Dieser Thread kann ja gerne RTCMemoryFix gewidmet bleiben! 

Beitrag von „iPhoneTruth“ vom 21. Juli 2019, 16:05

Zur Klärung und damit RTCMemoryFixup.kext richtig funktioniert hätte ich hierzu doch noch ein paar Fragen:

1. Muß in der config.plist sowohl 'Apple RTC' (unter 'Kernel an Kext Patches') als auch 'FixRTC' (unter 'Acpi / Fixes') deaktiviert sein?

2. Wenn ich wie in Post 11 von kuckkuck beschrieben vorgehe und dann nach einigem Arbeiten am Hackintosh im Terminal den Befehl

"log show --last boot --style syslog | fgrep "Wake reason" eingebe, darf dann bei dem, was mir das Terminal ausspuckt, kein 'Wake reason: RTC (Alarm)' mehr erscheinen?

3. Oder muß unter 'Systemeinstellungen / Energie sparen / Die Option "Power Nap" bei angeschlossenen Netzteil (Batteriebetrieb) aktivieren' deaktiviert sein, damit dann kein 'Wake reason: RTC (Alarm)' mehr erscheint?

4. Sprich, hilft der Terminalbefehl "log show --last boot --style syslog | fgrep "Wake reason" um die Funktionsweise von RTCMemoryFixup.kext zu kontrollieren oder kann die Ausgabe dabei irreführend sein?

Beitrag von „kuckkuck“ vom 22. Juli 2019, 11:17

1. mit einem richtig konfigurierten RTCMemoryFixUp brauchst du AppleRTC und FixRTC nicht mehr. RTCMemoryFixUp ist hier, wie in den Posts weiter oben beschrieben, die cleanere und nachhaltigere Lösung.

2. RTC Wakes haben wie von CMMChris oben beschrieben nichts mit RTCMemoryFixUp zu tun, sondern sind Wakes die durch PowerNap zustande kommen. RTCMemoryFixUp behebt eine Reset-Problematik die auf der Inkompatibilität vom AppleRTC Treiber und unserer Firmware beruht.

3. RTC Wakes kannst du wenn dann durch PowerNap Settings oder Darkwake Modi beeinflussen.

4. Nein, funktioniert nicht und die von Apple gewählten Namen sind hier definitiv irreführend...

Beitrag von „kuckkuck“ vom 18. August 2019, 10:00

roqueeee Wie hast du eigentlich deine BIOS Settings bei deinen Tests überprüft? Ich finde keine komfortable Methode... Die Settings einzeln im BIOS zu überprüfen finde ich langweilig, die Größe von gespeicherten BIOS Profilen bleibt immer gleich und die SHA Summe der Profile verändert sich schon bei reboots

---

Beitrag von „roqueeee“ vom 20. August 2019, 14:10

Leider kenne ich auch keine gute Methode. Habe die Veränderungen nach dem Neustart im Bios nachgeguckt. War dementsprechend auch ziemlich zeitaufwändig.

Wegen der mangelnden Überprüfbarkeit (evtl. Änderungen in einer Bank der RTC die nicht visuell im Bios repräsentiert sind) bin ich auch wieder zu FixRTC zurückgekehrt.

Habe eben mal ein bisschen gegoogelt wie ein Cmos aufgebaut ist. Typischerweise gibt es wohl Bänke die als Prüfsumme für einzelne Abschnitte der Rtc dienen. Wäre natürlich super wenn man die auslesen und überprüfen könnte. Die gesamte Rtc per Prüfsumme zu checken stell ich mir sinnlos vor, da zumindest die Echtzeituhr ja ständig weiter zählt, dementsprechend käme da immer was anderes raus.

Beitrag von „kuckkuck“ vom 20. August 2019, 15:15

Hmm... Umständlich...

Wieso FixRTC, damit blockierst du doch 128 Bytes komplett, dann doch lieber mit RTCMemoryFixUp die zweite MemoryBank blockieren oder großzügig ausmisten, dann werden writes wenigstens noch simuliert

Den NVRam haste nicht hinbiegen können? Mal mit OpenCore + FWRuntimeServices probiert, da lässt sich ja so ziemlich alles konfigurieren oder partiell deaktivieren was im Weg stehen könnte...

Bei mir auf einer Ozmosis Testmaschine kriege ich Hibernate nicht auf die Reihe, egal ob RTCMemory (vermutlich) komplett gepatcht ist oder nicht, bei Hibernatemode 25 resultiert Ruhezustand immer in einem perfekten Neustart. Inwiefern mein altes Ozmosis Pferd nötige Services für funktionellen Hibernate bereitstellt habe ich aber auch nicht ausreichend nachgeforscht, ebenfalls nur Tests mit dem ancient OsxAptioFix2 gemacht, wahrscheinlich liegt hier der Hund bereits begraben. Oder was meinst du mhaeuser?

Weiteres Testpotenzial gibt es auf jeden Fall, auch habe ich nur Mode 25 probiert, irgendwo mal was von 21 und 27 gelesen, aber ob da was dran ist oder sich jemand verrechnet hat weiß ich nicht Sicherlich eine Bitmaske, aber aus was die sich zusammensetzt

---

Beitrag von „roqueeee“ vom 22. August 2019, 22:50

Mit dem Nvram habe ich mich ehrlich gesagt nicht weiter beschäftigt!

Eigentlich läuft bis auf ein paar Kleinigkeiten (Nur S1, kein nativer Nvram, kein USB3 on board, Rotes Bild bei Netflix in Safari) alles Rund.Bei mir steht voraussichtlich eh bald ein Upgrade der Hardware an, deswegen hatte ich in letzter Zeit nicht mehr so viel Motivation, mit dem aktuellen Setup herumzuexperimentieren.

Mein Uralt-Rechner hat noch nicht einmal Uefi, dementsprechend konnte ich auch noch nicht Opencore testen.

Habe nach deinem Kommentar doch wieder Rtcmemoryfixup reingenommen, und mich für das komplette Sperren der 2ten Bank entschieden. Werde dann also mit rtcfx_exclude=58-59,80-FF fahren. Wenns da Langzeitschwierigkeiten gibt, werde ich mich hier melden.

Ich wollte nochmal die Gründe auflisten, die meiner Meinung nach gegen RTCMemoryfixup sprechen:
-Es ist nicht transparent nachvollziehbar, ob man wirklich alle Abschnitte der RTC ausschließt, die ausgeschlossen werden sollten.

-Sollte es in Zukunft Probleme mit Lilu oder RTCMemoryfixup geben, könnte es passieren, dass die RTC nicht ordnungsgemäß schreibgeschützt wird. Dies würde im schlimmsten Fall dazu führen, dass der Rechner nicht hochfährt.

-Das Finden der kritischen Bereiche ist zeitintensiv und mühselig.

Beitrag von „EdD1024“ vom 9. Dezember 2019, 18:27

Hallo zusammen,

Ich habe auf der Suche nach Sleep/Wake-Problemlösungen diesen Thread aufgegabelt - ich kämpfe sein einiger Zeit mit Sleep/Wake auf meinem älteren KIRA 102 Ultrabook, welches ansonsten astrein (mittlerweile mit Catalina) funktioniert.

Folgendes Problem:

- Sleep funktioniert, aufwachen lässt sich der Laptop nur per Power-Button, dann crasht es direkt.
- Wenn eine Tastatur oder BT Mouse-Receiver in einem USB Port steckt, wacht das Gerät darüber tadellos auf. Eingebaute Tastatur zum Aufwachen funktioniert leider nicht.

Meine Frage wäre eigentlich, wie man systematisch das Problem eingrenzen könnte.

Danke in Voraus für jeden Hiweis!

Beitrag von „kuckkuck“ vom 9. Dezember 2019, 20:15

Dafür solltest du damit anfangen die verschiedenen Sleep Logs zu studieren. Am besten in einem eigenen Thread zu deinem Problem, denn das geht ein wenig über das Thema RTCMemoryFixUp hinaus. Um RTC Probleme anzugehen, könntest du RTC erstmal mittels ACPI oder KextPatch patchen, den Patch per Bootlog über prüfen (RTC only x/265 memory banks available) und schauen ob es einen Einfluss hat. Wie du deinen Laptop wecken kannst hängt stark vom ACPI ab. Hier müsste man beispielsweise LID0 patchen, damit wake durch Klappe öffnen geht etc.

Beitrag von „hackopti2“ vom 4. Mai 2021, 19:05

> Zitat von roqueeee
>
> Dabei ist B4=CPU PLL, B5=System Memory Multiplier, B6=CPU Vcore, B7=VTT Voltage.

spannend, wie hast du diese Werte denn zuordnen können?

Beitrag von „roqueeee“ vom 5. Mai 2021, 23:59

Nach einem Neustart konnte man die veränderten Werte in der Biosoberfläche nachvollziehen.

Hatte damals per Hand einen Overclock gemacht und wusste deswegen wie die Werte eigentlich aussehen sollten.

Beitrag von „hackopti2“ vom 6. Mai 2021, 06:58

ahhh, ok, dachte sowas wird in UEFI-Variablen gespeichert.