iShengP 的 Z370F + i7–8700K + RX570 Hackintosh Build 黑蘋果建置 (Catalina ver. with OpenCore)

黑蘋果 / By iShengP / 2020 年 1 月 16 日 2020 年 2 月 23 日

這是一篇面向有點程度的新手以及高手參考用的 Hackintosh Build。

本文章更新於 2020/2/6
對應 OpenCore 0.5.5 版

若有什麼漏了或是錯了就抱歉了，
幫我回報到信箱或是我的Telegram(@iShengP)吧，有空慢慢修，感謝您。

前言

自從 OpenCore 0.5 開始公測以來我就一直在看風向，不過後來那些開發者紛紛跳槽 OC，不再對 Clover 進行相容性測試，然後仔細鑽研了一下內部的構造跟好處之後，我就一直不停地進行測試...結論就是過了好幾個月後，在這個已經有不少前輩在這塊耕耘的同時，我也終於裝好能用啦！以下就是針對我電腦的一些內容，如果你的晶片組跟我不一樣還是規格差太多了就建議別向我提問了，去討論去找解答比較快吧（笑）。

寫在前頭就是 OpenCore 就目前來看還是屬於開發中階段，寫文的當下版本是 0.5.4（其實是拖稿好幾個月終於發了哈哈），因為 config.plist 的內容都會因為版本更新而做更動，所以不建議直接抄所有的選項開機，建議還是用一下大腦以及看一下最新的資訊按照自己的需求做選項的調整喔。

還有就是我目前看網路上中文的教學內容是很少（一隻手數得出來），內容差異頗大，雖然有具有權威性內容充實的深入講解，這真的很棒，但還是有機翻超嚴重的中文教學存在，所以我的推薦還是去看英文教學找資料可以避免很多問題產生就是了。另外就是我的技能程度自認為就只有 3 out of 5，大概是可以讓電腦95%正常工作、可以當日常使用的程度，所以如果有些翻譯很怪，或是被我整段跳過沒解釋因為覺得不知道也沒差的內容就請多包涵，或是在回覆的地方標注一下讓我在下一版的內容補回去吧。

最後的最後，這篇文章雖然會帶過所有的安裝流程，但廢話應該很多，所以如果你想要的是一篇乾淨利落，所有步驟直接列出，Get the job done的話，請不要繼續閱讀下去了。老師說需要debug的東西才是真的有價值的東西，所以如果只是按照步驟跟選項的話那日後debug應該會吐血，畢竟沒有知識底子支撐。

這篇文章將會有很多來自這裡的內容，也同時是新手參考成功率最高的文章之一，所以在這邊很感謝編寫的作者群為世界貢獻了一篇超棒的參考內容。

Opencore Vanilla Desktop Guide
Last edited: Febuary 13, 2020

另外也推薦這兩篇文章，裡面有很仔細地用中文寫 Quirks 的意思，也有一些更進階的內容。

精解OpenCore
由于个人能力有限, 教程中难免会有些疏漏, 这里推荐大家在参阅本教程的同时也阅读以下资料:

OpenCore 官方文档 — OpenCore 最权威的资料, 没有之一!!!
xjn's Blog — xjn 大佬的博客, 对台式机非常友好的教程, 内存管理写的非常详细
OC-little — 宪武大佬的 OC ACPI 热补丁示例, 非常值得学习
Opencore Vanilla Desktop Guide
使用OpenCore引导黑苹果 — Xjn's Blog
目前为止，最新版的OpenCore(V0.5.1)已经证明了自己在10.14.6至10.15 beta 11上的稳定性。考虑到即将来临的10.15正式版，我想现在写一份关于z390等比较新的主板，如何使用OpenCore去引导新系统是一个不错的时间点。

最後，如果英文好的人，果然還是最推薦官方的說明書。我比較常在這邊翻裡面的 Quirks 參考，當然也是目前寫得最清楚的。

acidanthera/OpenCorePkg
OpenCore front end. Contribute to acidanthera/OpenCorePkg development by creating an account on GitHub.

為什麼 OpenCore？

大概就是

- 檔案超小，開機比 Clover 快
- 對於 FileVault 加密支援比較好（雖然我不覺得大部分的人需要）
- 支援 Mac 的開機快捷鍵，例如 option 開機選單、⌘+R 復原模式、⌘+⌥+P+R 清除NVRAM 等
- 讓電腦的整體運作與功能模式比 Clover 更像真的 Mac
- 大部分開發者在未來的元件開發已經轉向 OpenCore，對 OC 進行測試與相容
- 大部分的設定及選項已經濃縮集中在一個檔案當中，選項更為簡單

還有更多...

所以現在加入 OpenCore 大家庭其實還算是幫自己弄個未來吧，好處很多，但是仍然對使用者不友善，因此我推薦新手就文章發佈的時間點還是使用 Clover 裝機，等到日後 OpenCore 出了 1.0 版之類的，或是各種懶人使用者友善的工具紛紛出現再跳槽也不遲。

硬體

- 處理器：Intel Core i7—8700K
- 散熱器：Corsair H115i
- 主機板：ASUS ROG STRIX Z370-F GAMING
- 顯示卡：ASUS ROG STRIX-RX570-O4G-GAMING
- 記憶體：GSKILL Trident Z DDR4 3200MHz CL16 8GBx2
- SSD：Intel SSD 730 Series (240GB，2.5in SATA 6Gb/s，20nm，MLC)
- SSD：Micron Crucial MX500 500GB
- HDD：Toshiba 4TB (MD04ACA400)
- HDD：Seagate IronWolf 4TB (ST4000VN008)
- WiFi、BT：Fenvi FV-T919
- 機殼：Fractal Design Define R6
- 螢幕：BenQ EW3270U (32吋，3840×2160)
- 鍵盤：Apple Magic Keyboard with Numeric Keypad
- 觸控板：Apple Magic Trackpad 2
- 滑鼠：Logitech MX Master 2S

有看過上次那篇就會知道我的設備有弄精簡一些啦，就目前來看最適合並且懶人的裝機還是 Intel CPU+Z370+AMD GPU 的方案，Intel 其他晶片組（如 Z390、H370 或是 X299 之類的）CPU設定上步驟確實會多一些，不過硬體上的需求就捏上去絕對不會錯的。而AMD香到不行的CPU目前沒什麼問題，Ryzen 跟 Threadripper 都很適合裝機，據說還有各種一鍵工具比Intel還更好更快裝機的方式可以用，不過那邊我就比較沒涉略了，有需要的就去看看吧。顯卡的部分除了用內顯的使用者以外，Power User 我推 Vega 56、RX5700（Navi 架構 Catalina 限定）以上的硬體，舊Polaris非Vega的顯卡如RX570之類的除非你真的很窮再來考慮吧。總歸一句就是硬體跟白蘋果越像難度越低，這是從古至今不變的道理。

題外話就是雖然我現在用了蘋果原廠的周邊，但是我這邊測試是沒辦法靠藍牙無線進 BIOS（舊款的 Wireless Keyboard 沒這個問題），所以在那種場合或是要進 Linux 或 Windows 的話還是推薦接線使用。

三個不，一個要

這個我後來想到臨時加的，應該能解決一些問題

不

- 不要使用網路上任何人提供的 EFI，除非你知道你自己在幹嘛。
  在你還沒有了解裡頭的項目之前，都不應該使用裡面給的數值進行開機，就算你的硬體跟對方的硬體差不多。這樣不僅自己根本除不了錯，上網問浪費別人的時間，盡可能地從0將整個安裝碟建立起來。
  寫了這段話的我根本就是幫自己挖了坑，可能之後一陣子會陸續把我沒提到的Quirks給補上吧。

- 不要沒有備份就直接嘗試安裝，尤其是工作用機。
  畢竟永遠都不會知道下一秒可能就把什麼東西搞砸了，雖然理論上這種事情不太會發生。
- 遇到問題不要直接發篇文或是寫一封信伸手問為什麼會這樣
  提問前先Google，自己發生的問題有99%的機率別人也都發生過，也會有不錯的解決方案。（當然這時候連第一點都沒辦法辦到的時候就完全沒辦法除錯了。）
- 不要隨意更新 macOS，就算要更新請爬文。
  這算是常識了，基本上新的 macOS 出來「理論上」不會有什麼大問題可以直接升上去，但有些時候就是會發生衝突或是驅動有待修正的部分（例如說 macOS 10.15.1 修了顯卡驅動所以就把顯示功能給搞掛了）。所以升級前先上網看看有沒有別人也有相同的問題吧

要

- 多備份EFI
  在安裝隨身碟上面測試沒問題了再套用到系統碟的EFI，或是覆蓋先前的Clover EFI 或是舊版本的 OpenCore。 基本上那個 EFI 算是整個流程中最關鍵的角色，也決定了你安裝的成功與否，所以為了避免自己把先前的 EFI 搞砸了最好隨時有備份，就算在 Linux 或是 Windows 底下也能修正錯誤。

準備

準備 軟體

- OpenCorePkg
  OpenCore 本體。 目前寫文章最新版是 0.5.4（見 releases） （如果要自己編譯的話，clone 下來雙擊 macbuild.tool 按照步驟就會自行編譯好了）
- ProperTree
  有針對 OpenCore 優化的跨平台 plist 編輯器，編修 config.plist 的工具。 使用方式就是 clone 下來點擊 .command 檔就會打開了，其他平台就點其他的執行檔吧。 話說目前我推薦的就只有這個工具。XCode 曾經為官方推薦之一，但是我用就是會把表格弄壞，極度不推薦，另外 OpenCore Configurator 是大家公認絕對不推薦。反正除了 ProperTree 以外的工具就是不要使用就對了，不要浪費你的生命（笑）
- mountEFI
  一個好用的掛載EFI磁區小程式 clone下來給權限 chmod +x MountEFI.command，日後雙擊 .command 檔開，輸入數字跟密碼就可以掛載對應的磁區。

下載之後放在一起備用

準備 EFI 資料夾

將 OpenCore 解壓縮，將裡頭的 EFI 資料夾挪出來，我們接下來要將東西放進去。

最後整理完大概會長得像這樣

刪除 工具

將 EFI/OC/Tools 內的工具全刪除，包含

- CleanNvram.efi — OpenCore 已內建
- VerifyMsrE2 — 檢測 CfgLock 工具，安裝時不需要

準備 驅動

AppleSupportPkg — 些必備的UEFI驅動程式。包含：

- ApfsDriverLoader.efi（必需 APFS驅動）
- VBoxHfs.efi（必需 HFS驅動）。

將解壓縮後的 .efi 檔整理在一起，放入 EFI/OC/Driver 裡頭（與 FwRuntimeServices.efi 放在一起）

並且將下列驅動刪除

- AppleUsbKbDxe.efi — 適用於 Legacy BIOS 的鍵盤驅動，Ivy Bridge 更新的架構不適用
- NvmExpressDxe.efi — 適用於 Haswell 以前的架構韌體無 Nvme 驅動
- XhciDxe.efi — 適用於 Sandy Bridge 以前的架構韌體無 XHCI 驅動

提醒一點就是 OpenCore 的 driver 跟 Clover 不共用，不要搞混了。

準備 kexts

kexts 是什麼？就是 Kernel Extension，核心擴充的意思。 你可以想像成 macOS 的驅動程式就好。

這邊要補充說明的一點就是專為 Clover 開發的 kext 在 OpenCore 是不適用的，所以請不要交替使用。以下是我這次所需要的 kext：

- VirtualSMC.kext
  必需 跟 FakeSMC.kext 相同，為 SMC 晶片的模擬器，如果你沒有這兩個之一的話那你就沒有辦法開機。包含的 SMCProcessor.kext（CPU 溫度資訊讀取）、SMCSuperIO.kext（風扇溫度資訊讀取）這次會使用到。
- Lilu.kext
  必需 提供對任何 kext 以及 進程(process) 進行修補的能力的 kext
- AppleALC.kext
  必需 提供 AppleHDA.kext 支援非官方的解編碼器的能力，主要是支援非官方的音效晶片。需要搭配 Lilu.kext 運作。
- WhateverGreen.kext
  必需 曾經的 IntelGraphicsDVMTFixup, NvidiaGraphicsFixup, CoreDisplayFixup, 和 Shiki 組合在一起的 kext。提供支援以及修復 Intel、NVidia、AMD 顯示卡的能力。需要搭配 Lilu.kext 運作。
- IntelMausi.kext
  有線網路的 kext。基本上新的 Intel 有線網路晶片都適用。
- USBInjectAll.kext
  突破 USB 僅支援 15 孔限制的 kext。（如果你已經有做了 USBPorts.kext 或是有放對應 SSDT 修正之類的 USB Mapping 動作就不需要再放這個了）
- AirportBrcmFixup.kext
  提供支援以及修復 WiFi 網卡連接的能力。需要搭配 Lilu.kext 運作。
- BrcmPatchRAM3.kext
  提供非官方藍芽支援以及修復藍芽連接的能力。需要搭配 BrcmFirmwareData.kext 運作。
- BrcmFirmwareData.kext
  作為 Injector 使用，搭配 BrcmPatchRAM3.kext 使用的必要 kext。

要記得就是上面給的 kext 並不是「適合所有電腦」使用，官方所有的支援列表在這裡。

將解壓縮後的 .kext 檔 整理在一起，放入 EFI/OC/Kexts 裡頭

準備 SSDT

- SSDT-EC.aml
  偽裝EC控制器，可以使用 SSDTTime 來產生
- SSDT-USBX.aml
  強制啟用 USB 電源參數，這個檔案是已經預編好給Skylake後以及AMD使用者使用的，不需要另行準備 SSDT-EC-USBX.aml。
- SSDT-PLUG.aml
  修改 Plugin-Type，可以使用 SSDTTime 來產生

SSDTTime

Windows 使用者打開 SSDTTime 的 repo ，點下右方的 clone or download，下載到桌面或是某個地方並解壓縮，雙擊 SSDTTime.bat 開啟程式。

macOS使用者就直接 clone 下來吧（當然你也可以用上述方法，執行檔是 SSDTTime.command ），打開終端機一行一行輸入

```
git clone https://github.com/corpnewt/SSDTTime
cd SSDTTime
./SSDTTime.command
```

就會開啟程式。

如果他跳出如下的錯誤訊息，那估計你電腦裡面沒有 Python 3。

```
Extracting
An error occurred :(
[Errno 2] No such file or directory: &#39;/var/folders/2t/bf6qym613ws6rltfgz9106v00000gn/T/tmpsjw9okvn/tmphqtr463g/iasl.zip&#39;
Something went wrong :( - Aborting!
Could not locate or download iasl!
```

在 macOS 底下建議使用 homebrew 快速裝一下，指令：brew install python
Windows 底下上 Python 官網 安裝即可

如果是 Windows 或是 Linux，輸入 4 提取DSDT，
Mac 使用者就自己看是要用 Clover 按 F4 或是開機到其他系統取吧。

主畫面，輸入 D 繼續

在這個畫面將取出來的 DSDT 找到 DSDT.aml 拖入視窗內，按下 return 繼續。跳回主畫面就會看到已經載入

選擇 2 就會進到修正 EC 的頁面，修正完畢會自動開啟資料夾

回到主畫面之再輸入 3 進到設定 plugin-type 的頁面，修正完畢會自動開啟資料夾

修正完畢之後的資料夾長這樣

別忘了將 SSDT-USBX.aml 也一同下載下來，將編譯好的 .aml 檔 請整理在一起，放入 EFI/OC/ACPI 裡頭

做到這邊，再放一次完成品讓你檢查一下整理出來的結果是否長得像這樣吧？

準備 安裝隨身碟

安裝隨身碟的製作方式有三種

1. 如果你手邊有 Mac 的話，從 Mac App Store 下載 macOS Catalina，用 Create Install Media 的工具製作安裝隨身碟。
2. 利用 Recovery dmg 檔從網路進行安裝。
3. 不進行安裝，直接製作 OpenCore 隨身碟。

1、3 的話見 macOS 操作的章節，2 的話見 Windows 操作的章節

macOS 操作

Windows 使用者請跳到下一章節

隨身碟格式化

到「磁碟工具程式」，選擇你的隨身碟，點擊「清除」

- 名稱 隨意
- Mac OS 擴充格式（日誌式）
- GUID 分割區配置表

選擇結束後，點擊「清除」進行格式化，完成之後關閉磁碟工具程式即可。

下載 macOS 安裝程式

如果你不進行安裝，請直接跳過此步驟

macOS Catalina Mac App Store 連結

下載就好，跳出安裝 macOS Catalina 的視窗就將視窗關起來就可以了。

安裝程式寫入隨身碟

如果你不進行安裝，請直接跳過此步驟

那我們就對隨身碟進行寫入的動作，蘋果已經幫我們做好了功能。
如果你是 GUI 控的話，你可以直接下載 DiskMaker 之類的工具省下打以下指令的麻煩。

輸入指令：

```
sudo /Applications/Install macOS Catalina.app/Contents/Resources/createinstallmedia --volume /Volumes/USB
```

這樣子終端機就會幫我們把下載下來的 macOS 安裝檔灌進名為 USB 的隨身碟裡頭，非常的簡單。
完成之後就會看到隨身碟變成了名為 Install macOS Catalina 的隨身碟。

掛載 EFI 磁區

打開 mountEFI.command 之後，會看到下方畫面

在清單裡頭找到你的隨身碟，像我的 USB 在 5，那就輸入 5 後按下 return

輸入使用者密碼後就會看到 Finder 裡有 EFI 磁區掛載起來了。

將剛剛準備好的 EFI 資料夾複製到隨身碟的 EFI 磁區的根目錄底下。

別忘了，是把東西放在EFI磁區（又稱ESR）的EFI資料夾裡面喔，不要像我一樣除錯了n個鐘頭就是出在這個錯誤

Windows 操作

macOS 使用者請跳回上一章節，當然你要用這章節的也不是不行啦，不過我只有在 Windows 上測試而已，理論上應該是不會有問題才對

下載 macOS 復原

假設各位的 Windows 電腦沒有 git 能用好了。 打開 gibMacOS 的 repo ，點下右方的 clone or download，下載到桌面或是某個地方並解壓縮。

雙擊 gibMacOS.bat 開啟工具

tips 如果開不起來或是發生錯誤，回去檢查電腦有沒有 Python 3。

輸入 R Toggle Recovery Only 開啟純還原模式（不下載完整系統）

選擇系統，抓最新的 Full Install，我這邊輸入 3 繼續

軟體就會開始下載到 gibmacos–mastermacOS Downloadspublicreleasexxx–xxxxx – 10.x.x macOS xxx 裡面。

下載完畢，就可以將視窗關起來。

安裝程式寫入隨身碟

跟剛剛同藥的程式 makeinstall.bat 開啟工具

上面會列出你的隨身碟，我要灌在我的創見的16GB隨身碟（6），然後我想要請軟體一次幫我把 OpenCore 灌好（O），輸入 6O 繼續

格式化前確認，輸入 y 繼續

程式詢問剛剛下載的檔案在哪裏，我們打開檔案總管裡的程式資料夾 gibmacos–mastermacOS Downloadspublicrelease 裡面可以找到剛剛下載的檔案，點選 複製路徑 貼到剛剛的程式裡面。

灌好了，就可以將視窗關起來

你會發現多一個 BOOT 資料夾，這個可以當作 EFI 磁區來使用，點進去可以看到 OpenCore 已經放好了。（但是沒有放 config.plist，所以當然沒辦法開機啦）

config.plist 編輯

首先，我們要先複製一份 config.plist 來當作參考修改，從解壓後的 OpenCore 資料夾 /Docs 裡面複製一份 Sample.plist 複製到 EFI/EFI/OC 底下，重新命名為 config.plist

額...如果你是 Windows 使用者的話就自己調整放到相對位置吧，EFI目錄的結構都是相同的，只是 macOS 在 EFI 磁區，Windows 在 BOOT 磁區就對了。

ProperTree

打開 ProperTree
macOS 使用者點擊資料夾內的 ProperTree.command 開啟，
Windows 使用者點擊資料夾內的 ProperTree.bat 開啟。

tips 如果開不起來或是發生錯誤，回去檢查電腦有沒有 Python 3。

點擊 File > Open 開啟 EFI/EFI/OC 底下的 config.plist

接下來我們要編輯這些項目：

我這邊並沒有將所有的 Quirks 清楚說明，只有提到我需要的，建議還是可以上網找有解釋所有的 Quirks 的文章如 [這個](https://blog.daliansky.net/OpenCore–BootLoader.html)，一個一個對照自己是否有需要並修改數值。

ACPI–Add

這裡是設定SSDT的地方，請將自己手邊裡的ACPI資料夾內的SSDT，確認 Enabled == True，如幔有對應的 .aml 檔請自己複製一份手動新增，舉個例子

其他選項看你要不要刪，我自己是刪了，方便我除錯 複製的方法就是直接點擊數字012 command–c 再點擊 Add（或是父物件）command–v 貼上即可，不要搞錯了

- SSDT–PLUG 允許原生的 CPU 電源管理。剛剛 SSDTTime 產生的。
- SSDT–EC 透過尋找電腦裡 PNP0C09，修正 EC 裝置的位置，Catalina使用者必備，剛剛 SSDTTime 產生的。
- SSDT–USBX 強制啟用 USB 電源參數。

ACPI–Block

阻擋一些 ACPI 表的載入，無視吧。

ACPI–Patch

部分裝置的 ACPI 修正，不過我沒用到就無視吧。

- Comment
  修正的名稱
- Count
  修正的數量，0代表安裝至全部
- Enable
  啟用停用

- Find
  欲修改的ACPI
- Replace
  修改後的ACPI，長度需相同

ACPI-Quirks

- FadtEnableReset: FALSE
  啟用傳統硬體重新開機以及關機功能，不建議使用除非有必要
- NormalizeHeaders: FALSE
  清理 ACPI 檔頭，只有 High Sierra 使用者有需要
- RebaseRegions: FALSE
  觸發重新定位ACPI記憶體映射，只有自定義 DSDT 使用者有需要
- ResetHwSig: FALSE
  休眠相關，給重開機或是休眠有問題的人使用
- ResetLogoStatus: FALSE
  OEM Windows logo 修正用

Booter-Quirks

- AvoidRuntimeDefrag: TRUE
  修正 UEFI 一些功能如日期、時間、NVRAM、電源控制等
- DevirtualiseMmio: TRUE
  降低記憶體 Stolen 的容量，修正部分 Z390 主板記憶體問題
- DisableSingleUser: FALSE
  禁止 Cmd+S 以及 -s 使用，接近使用 T2 晶片的電腦
- DisableVariableWrite: FALSE
  給那些無法使用 NVRAM 的電腦（可用 SSDT-PMC 代替）
- DiscardHibernateMap: FALSE
  重新利用休眠記憶體，傳統架構電腦適用
- EnableSafeModeSlide: TRUE
  允許 Slide 值用於安全模式
- EnableWriteUnprotector: TRUE
  移除 CRO 執行保護
- ForceExitBootServices: FALSE
  確保記憶體映射更換時 ExitBootServices 呼叫成功，必要時使用
- ProtectCsmRegion: FALSE
  修正部分睡眠問題，必要時使用
- ProvideCustomSlide: TRUE
  Slide 值衝突相關
- SetupVirtualMap: TRUE
  修正 SetVirtualAddresses
- ShrinkMemoryMap: FALSE
  大型記憶體映射不適用時使用
- SignalAppleOs: FALSE
  讓電腦以為每次開機都是macOS，適用於部分 MacBook Pro 不允許 iGPU 於 Windows 使用。

DeviceProperties-Add

layout-id 是指音效卡的 Layout ID，詳情可以查看 AppleALC 的音效卡對應代號，可以預設的 00000000 就好，將 layout-id 填在開機參數那邊。

AAPL,ig-platform-id 則是指內顯的設定，如果你是要用內顯驅動螢幕，使用 07009B3E，如果你像我是使用獨顯，使用內顯作為純運算用，使用 0300923E。

```
tips:
AAPL,ig-platform-id 這邊使用的是 hex-swapped 的數值
數值轉換的範本為轉換前 0xAABBCCDD，轉換後為 DDCCBBAA
```

Kernel-Add

這裡將 kext 資料夾對應的 kext 給填上去。 至於 ExecutablePath 和 PlistPath 內的路徑可以直接右鍵顯示套件內容找找看 info.plist 在哪裏。

- BundlePath
  kext 的檔名，如 lilu.kext
- Enabled
  啟用停用
- ExecutablePath
  路徑，你可以對 kext 又見顯示套件內容看路徑在哪裡，如沒有空白即可，如 Contents/MacOS/Lilu

- PlistPath
  info.plist 路徑，如 Contents/Info.plist

Kernel–Quirks

- AppleCpuPmCfgLock: FALSE
  適用於 BIOS 無法解鎖 CFG 鎖的主機板
- AppleXcpmCfgLock: FALSE
  適用於 BIOS 無法解鎖 CFG 鎖的主機板
- AppleXcpmExtraMsrs: FALSE 修正部分 Pentium 以及 Xeon 處理器相關
- AppleXcpmForceBoost: FALSE
  強制最大倍頻，適合長時間跑運算的電腦，Xeon 處理器相關
- CustomSMBIOSGuid: FALSE GUID 相關，適用於部分 Dell 電腦
- DisableIOMapper: TRUE
  VT–D 選項相關，啟用選項能開啟 VT–D
- DummyPowerManagement: FALSE
  NullCPUPowerManagement 替代品，適用於 AMD 處理器
- ExternalDiskIcons: FALSE
  內接硬碟被當外接硬碟修正，適用於 Z87 平台用 Nvme 硬碟。
- IncreasePciBarSize: FALSE
  修正部分X99相關錯誤
- LapicKernelPanic: FALSE
  修正 HP 相關錯誤
- PanicNoKextDump: TRUE
  Kernal Panic 的時候會顯示時間與原因
- PowerTimeoutKernelPanic: TRUE
  修正部分驅動程式於電源轉換時造成的 Kernel Panic，大多時候是數位音效驅動
- ThirdPartyDrives: TRUE 啟用 TRIM，Nvme 硬碟使用者選 FALSE
- XhciPortLimit: TRUE 解除 USB 15孔限制

Misc–Boot

開機畫面相關設定

- HibernateMode: None
  黑蘋果建議遠離休眠
- HideSelf: TRUE
  開機選單隱藏 EFI 磁區
- PollAppleHotKeys: FALSE
  允許使用 Apple 內建的開機快捷鍵，部份硬體需要 AppleUsbKbDxe.efi 才能運作。如果你沒辦法操作開機選單的話建議停用。 快捷鍵列表
  Cmd+V: 文字模式
  Cmd+Opt+P+R: 清除 NVRAM
  Cmd+R: 復原模式
  Cmd+S: 單用戶模式
  Option/Alt: 內建開機選單
- Timeout: 5
  OpenCore 開機選單時間（秒）
- ShowPicker: TRUE
  顯示 OpenCore 開機選單的 UI
- TakeoffDelay: 0
  按鍵延遲時間
- UsePicker: TRUE
  使用 OpenCore 原生開機選單

Misc–Debug

- DisableWatchDog: TRUE
  能解決部分macOS開機卡住的問題
- Target: 67
  顯示更多除錯資訊，穩定版使用者請無視
- DisplayLevel: 2147483714
  顯示更多除錯資訊，穩定版使用者請無視

Misc–Security

- AllowNvramReset: TRUE
  允許使用開機選單或是快捷鍵（如果有開啟）重設nvram

- AllowSetDefault: TRUE
  允許使用 CTRL+Enter 或 CTRL+Index 設定預設開機硬碟
- AuthRestart: FALSE
  啟用重開機時略過 FileVault 加密密碼，可能會有安全漏洞所以看需求吧
- ExposeSensitiveData: 6
  顯示更多除錯資訊，穩定版使用者請無視
- RequireSignature: FALSE
  跟加密驗證有關的設定，我們用不到，打開的話開機會卡住
- RequireVault: FALSE
  跟加密驗證有關的設定，我們用不到，打開的話開機會卡住
- ScanPolicy: 0
  開機時顯示所有能用的硬碟

Misc-Tools

OpenCore 工具相關，不過剛剛已經把工具都刪掉了所以這邊可以無視

- Name
  名稱
- Enabled
  啟用停用
- Path
  路徑，如 Shell.efi

Misc-Entries

讓 OpenCore 抓取特殊的開機路徑，這點其實可以無視

- Name
  顯示在開機選單的名稱
- Enabled
  啟用停用
- Path
  開機碟的 PCI 路徑，可以從 OpenCoreShell 之類的工具找到，如 PciRoot(0x0)/Pci(0x1D,0x4)/Pci(0x0,0x0)/NVMe(0x1,09-63-E3-44-8B-44-1B-00)/HD(1,GPT,11F42760-7AB1-4DB5-924B-D12C52895FA9,0x28,0x64000)/EFIMicrosoftBootbootmgfw.efi

NVRAM-Add

- UIScale: 02
  選單螢幕比例縮放，一般人無縮放用 01 即可，像我使用 4K 螢幕有開縮放用 02
- boot-args: -v dart=0 debug=0x100 keepsyms=1 alcid=20 開機代碼：
  -v 文字模式開機，相同於 Cmd+V
  dart=0 修正開啟 vt-d 造成的錯誤
  debug=0x100 修正 Watchdog 可能造成的錯誤
  keepsyms=1 當 Kernel Panic 時印出錯誤
  alcid=[代號] AppleALC layout ID，像我的主機板Z370-F為20，所以填20，你可以在這裡查詢你自己的
- csr-active-config: 00000000
  SIP 設定，建議是開啟安全性較佳
  00000000 — 完全開啟
  30000000 — 允許未簽署的kext
  E7030000 — SIP 完全關閉
- nvda_drv:
  NVidia 驅動相關，Catalina無法使用無視。
- prev-lang:kbd: 0x7a682d48616e743a32
  若無設定此項安裝畫面將會變成俄文
  多國鍵盤支援，預設是俄文，繁體中文為(zh-Hant:2) 0x7a682d48616e743a32，英文(zh-en:0)為 0x7A682D656E3A30
  完整列表可查詢 AppleKeyboardLayouts.txt，並轉換為 16 進位。
  不過我還是不知道zh-Hant:2這選項打哪來的，按照上面列表應該是要zh-Hant:-16900才會對應到繁體中文、注音鍵盤才對啊，這個有待之後研究，反正現在就是會work就對了

NVRAM-Block

- LegacyEnable: FALSE
  允許 NVRAM 資訊寫入於 nvram.plist，無原生 NVRAM（Z370以外）必須
- LegacyOverwrite: FALSE
  允許韌體參數複寫於 nvram.plist，無原生 NVRAM（Z370以外）必須
- LegacySchema
  用於設定 NVRAM 參數，搭配 LegacyEnable: YES 使用

- WriteFlash: TRUE
  允許對記憶體寫入

PlatformInfo

這邊就是有關於 SMBIOS 的部分，要模擬電腦為一台 Mac。

- MLB
  主板編號，GenSMBIOS 產生
- ROM
  ROM編號，看是要用真正的 Mac 取出還是隨便填一個 MAC 位址囉
- SystemProductName
  機型，內顯使用者使用 iMac18,1，獨顯使用者如我使用 iMac18,3，目前我試是相容性最好。
  當然雖然說 iMac 19,1 19,2 的硬體比較接近我的設備理應是較好，但是這機型跟 USBInjectAll 的相容性不太好。
- SystemSerialNumber
  序號，GenSMBIOS 產生
- SystemUUID
  UUID，GenSMBIOS 產生

GenSMBIOS—Mac 序號產生

打開終端機，一行一行輸入

```
git clone https://github.com/corpnewt/GenSMBIOScd GenSMBIOSchmod +x GenSMBIOS.command
```

程式打開會抓取一些資訊

進到主畫面，會顯示 MacSerial 找不到，輸入 1 繼續

下載中

回到主畫面就可以看到 MacSerial 有抓取到版本 2.1.1，輸入 3 產生 SMBIOS 資訊吧

輸入機型與產生數量，輸入 iMac18,3 20 產生 20 個 27 寸 iMac (2017) 的序號

複製裡頭的序號(Serial)到蘋果官網驗證吧

我們打開蘋果的保固查詢頁面

[查看服務和支援保固狀態 – Apple 支援]

將序號填入搜尋框內，確認是否有人用過了

當網頁跳出您輸入的序號無效，那就代表序號沒人使用。

確定無人使用後，那就填進上面的 SystemProductName、SystemSerialNumber 跟 MLB 裡面

將對應資訊填入 config.plist 當中

- Automatic: TRUE
  自動建立 SMBIOS 訊息
- SpoofVendor: TRUE
  隱藏硬體品牌資訊
- AdviseWindows: FALSE
  修正部分老 Windows 電腦 EFI 磁區不在硬碟的前端
- UpdateDataHub: TRUE
  更新 Data Hub 欄位
- UpdateNVRAM: TRUE
  更新 NVRAM 欄位
- UpdateSMBIOS: TRUE
  更新 SMBIOS 欄位
- UpdateSMBIOSMode: Create
  覆寫位於 EfiReservedMemoryType 的欄位，Dell 筆電需要啟用 CustomSMBIOSGuid 選項

UEFI–Drivers

跟剛剛的 kext 一樣，將 Drivers 資料夾的 efi 對應的寫上來

UEFI–Input

一些 FileVault 與鍵盤相關的設定

- KeyForgetThreshold: 5
  按鍵延遲
- KeyMergeThreshold: 2
  觸發時間
- KeySupport: TRUE
  使用 OpenCore 內建鍵盤支援而非 AppleUsbKbDxe.efi
- KeySupportMode: Auto
  鍵盤多國轉換支援
- KeySwap: FALSE
  Option 跟 Command 交換
- PointerSupport: FALSE
  用於損壞的游標支援，常用於 ASUS Z87 主機板
- PointerSupportMode:
  特殊 OEM 通道，目前僅支援 ASUS Z87/Z97 主機板

UEFI-Protocols

這裡大多數的選項是面相白蘋果的，其實可以無視
（↑其實是想偷懶了）

- ConsoleColtrol: TRUE （剛開機時以文字顯示替換掉蘋果Logo）

UEFI-Quirks

- AvoidHighAlloc: FALSE
  修正部分主機板相關問題，必需時用（Gigabyte GA-Z77P-D3 (rev. 1.1)）
- ExitBootServicesDelay: 0
  修正部分主機板相關問題，必需時用（ASUS Z87-Pro 使用 FileVault 2）
- IgnoreInvalidFlexRatio: FALSE
  修正 MSR_FLEX_RATIO 無法停用，Skylake以前系統必需
- IgnoreTextInGraphics: FALSE
  修正開機UI相關錯誤
- ProvideConsoleGop: TRUE
  啟用 GOP(Graphics output Protcol)
- ReleaseUsbOwnership: FALSE
  從韌體驅動釋放 USB 控制器
- RequestBootVarFallback: TRUE
  修正開機相關錯誤
- RequestBootVarRouting: TRUE
  建議所有系統開啟以利系統更新運作
- ReplaceTabWithSpace: FALSE
  修正 UEFI 無法使用 Tab 鍵
- SanitiseClearScreen: TRUE
  修正高解析度顯示 OpenCore，建議所有 1080p+ 螢幕的使用者開啟
- ClearScreenOnModeSwitch: FALSE
  修正螢幕出現半邊畫面，文字模式切換相關錯誤
- UnblockFsConnect: FALSE
  修正HP電腦相關開機錯誤

做到這邊，config.plist 大致算完成了 這邊放上我自己的 config.plist 給大家參考，不過建議還是得要一行一行了解細節就是了。

安裝

建議在重開機之前，將剛剛的那些資料、驅動跟工具等等再丟進安裝隨身碟內（丟在 Install macOS Catalina 的根目錄即可），並且將目前隨身碟的 EFI 磁區也給複製一份放到安裝隨身碟磁區內，等等會用到。將電腦關機。

電腦開機，按下 Delete 進到 BIOS（不同品牌快捷鍵不太一樣），調整設定

- 離開 > 載入最佳化預設值
- Ai Tweaker Ai 智慧超頻：XMP
- 進階 中央處理器設定 > Intel Virtualization Technology：開啟
- 進階 系統代理設定 > VT-d：關閉
- 進階 系統代理設定 顯示設定 > 首選顯示卡：PCIE
- 進階 系統代理設定 顯示設定 > 混合多螢幕輸出模式：開啟
- 進階 內建裝置設定 > RGB 燈光 > When system is in sleep, hibernate or soft off states：關閉
- 進階 USB Configuration > Legacy USB 支援：自動
- 啟動 啟動設定 > 快速啟用：Disabled
- 啟動 安全啟動選單 > 作業系統類型：其他作業系統

F10 存檔，重新進入 BIOS

- 進階 系統代理設定 顯示設定 > DVMT Pre-Allocated：192M

F10 存檔，重新開機

接下來重開機後按 F8 以隨身碟開機

基本上沒問題的話會看到 OpenCore 陽春的開機選單，如有需要切換請輸入數字（不過理論上預設就是隨身碟 Install macOS Catalina 就是了）

然後也會沒問題一路安裝到 macOS 內，這點我就不再多做說明了。你只要記得到目前為止每次開機都要記得切換到隨身碟開機。

最後

搞定進桌面，將隨身碟的 EFI 資料夾的備份放到 Macintosh HD 的 EFI 磁區內，並且重開機以硬碟開跑跑看開機是不是正常的，正常的話就可以打完收工囉！

等穩定下來之後，如果不想要再看到文字的開機介面，那就可以把 boot-args 的 -v 給拿掉，就會看到美美的蘋果開機畫面囉！

後記

最後也沒有想說什麼了，不知道為什麼這篇寫起來比上一篇還坎坷。所以我覺得 OpenCore 是一個看似很複雜，但又沒有那麼複雜的東西啦，雖然偶爾還是會有點懷念 Clover 那種容易不少的安裝模式，不過再過三年五年，OpenCore 也能到達那個很友善的時候吧！

另外就是有人要伸手要 config.plist 的話那還是算了吧，反正你拿回去也是要修正才能用，那為什麼不一開始就自己弄呢？（笑）

更新紀錄

2020/1/17 — 文章推薦、三不一要
2020/2/5 — 大幅修正文章內容，新增 Windows 內容、GenSMBIOS 內容，對應 OpenCore 0.5.5

在您回應提問之前...

額額我很常漏掉 Medium 的訊息，所以建議你直接 Telegram @iShengP 或是直接 email 給我能夠獲得比較即時的回答就是了（苦笑）。

0

文章導覽

← Previous 文章
Next 文章 →

Leave a Comment

發佈留言必須填寫的電子郵件地址不會公開。 必填欄位標示為 *

Type here..

Name*

Email*

Website

在瀏覽器中儲存顯示名稱、電子郵件地址及個人網站網址，以供下次發佈留言時使用。

這個網站採用 Akismet 服務減少垃圾留言。進一步瞭解 Akismet 如何處理網站訪客的留言資料。

內容目錄