Laptop Ivy Bridge

Support	Version
Initial macOS Support	OS X 10.7, Lion
Last Supported OS	macOS 11, Big Sur

Starting Point

So making a config.plist may seem hard, it's not. It just takes some time but this guide will tell you how to configure everything, you won't be left in the cold. This also means if you have issues, review your config settings to make sure they're correct. Main things to note with OpenCore:

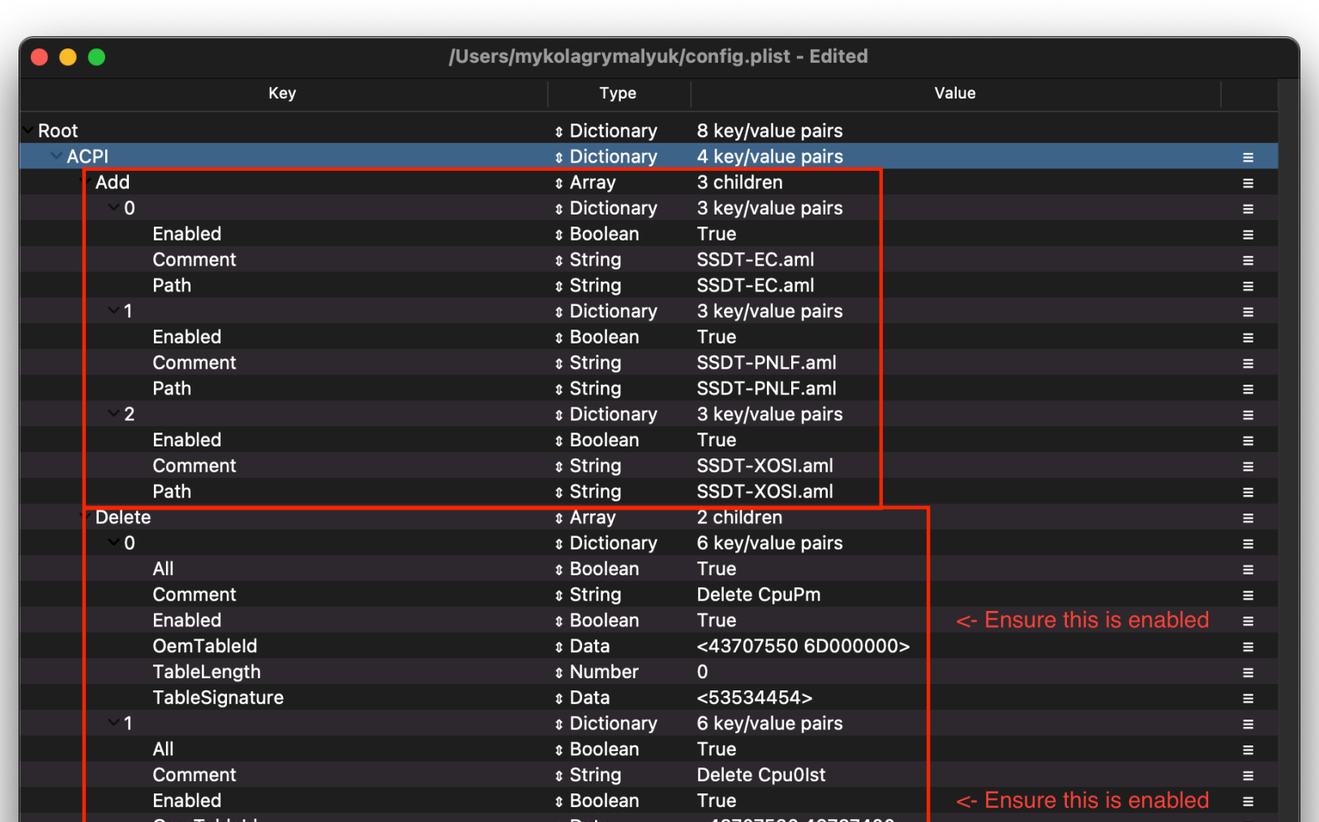
- **All properties must be defined**, there are no default OpenCore will fall back on so **do not delete sections unless told explicitly so**. If the guide doesn't mention the option, leave it at default.
- **The Sample.plist cannot be used As-Is**, you must configure it to your system
- **DO NOT USE CONFIGURATORS**, these rarely respect OpenCore's configuration and even some like Mackie's will add Clover properties and corrupt plists!

Now with all that, a quick reminder of the tools we need

- ProperTree
  - Universal plist editor
- GenSMBIOS
  - For generating our SMBIOS data
- Sample/config.plist
  - See previous section on how to obtain: config.plist Setup

> **WARNING**
>
> Read this guide more than once before setting up OpenCore and make sure you have it set up correctly. Do note that images will not always be the most up-to-date so please read the text below them, if nothing's mentioned then leave as default.

ACPI



Add

This is where you'll add SSDTs for your system, these are very important to **booting macOS** and have many uses like USB maps , disabling unsupported GPUs and such. And with our system, **it's even required to boot**. Guide on making them found here: **Getting started with ACPI**

For us we'll need a couple of SSDTs to bring back functionality that Clover provided:

Required SSDTs	Description
SSDT-PM	Needed for proper CPU power management, you will need to run Pike's ssdtPRGen.sh script to generate this file. This will be run in post install .
SSDT-EC	Fixes the embedded controller, see Getting Started With ACPI Guide for more details.
SSDT-XOSI	Makes all _OSI calls specific to Windows work for macOS (Darwin) Identifier. This may help enabling some features like XHCI and others.
SSDT-PNLF	Fixes brightness control, see Getting Started With ACPI Guide for more details. Note that Intel NUCs do not need this
SSDT-IMEI	Needed to add a missing IMEI device on Ivy Bridge CPU with 6 series motherboards, **not needed for 7 series motherboards**

Note that you **should not** add your generated `DSDT.aml` here, it is already in your firmware. So if present, remove the entry for it in your `config.plist` and under EFI/OC/ACPI.

For those wanting a deeper dive into dumping your DSDT, how to make these SSDTs, and compiling them, please see the **Getting started with ACPI page.** Compiled SSDTs have a **.aml** extension(Assembled) and will go into the `EFI/OC/ACPI` folder and **must** be specified in your config under `ACPI -> Add` as well.

Delete

Info

This blocks certain ACPI tables from loading, for us we really care about this. Main reason is that Apple's XCPM does not support IvyBridge all too well and can cause AppleIntelCPUPowerManagement panics on boot. To avoid this we make our own PM SSDT in Post-Install and drop the old tables(Note that this is only temporary until we've made our SSDT-PM, we'll re-enable these tables later):

Removing CpuPm:

Key	Type	Value
All	Boolean	YES
Comment	String	Delete CpuPm
Enabled	Boolean	YES
OemTableId	Data	`437075506d000000`
TableLength	Number	0
TableSignature	Data	`53534454`

Removing Cpu0Ist:

Key	Type	Value
All	Boolean	YES
Comment	String	Delete Cpu0Ist
Enabled	Boolean	YES
OemTableId	Data	`4370753049737400`
TableLength	Number	0
TableSignature	Data	`53534454`

Patch

Info

This section allows us to dynamically modify parts of the ACPI (DSDT, SSDT, etc.) via OpenCore. For us, we'll need the following:

- OSI rename
  - This is required when using SSDT-XOSI as we redirect all OSI calls to this SSDT

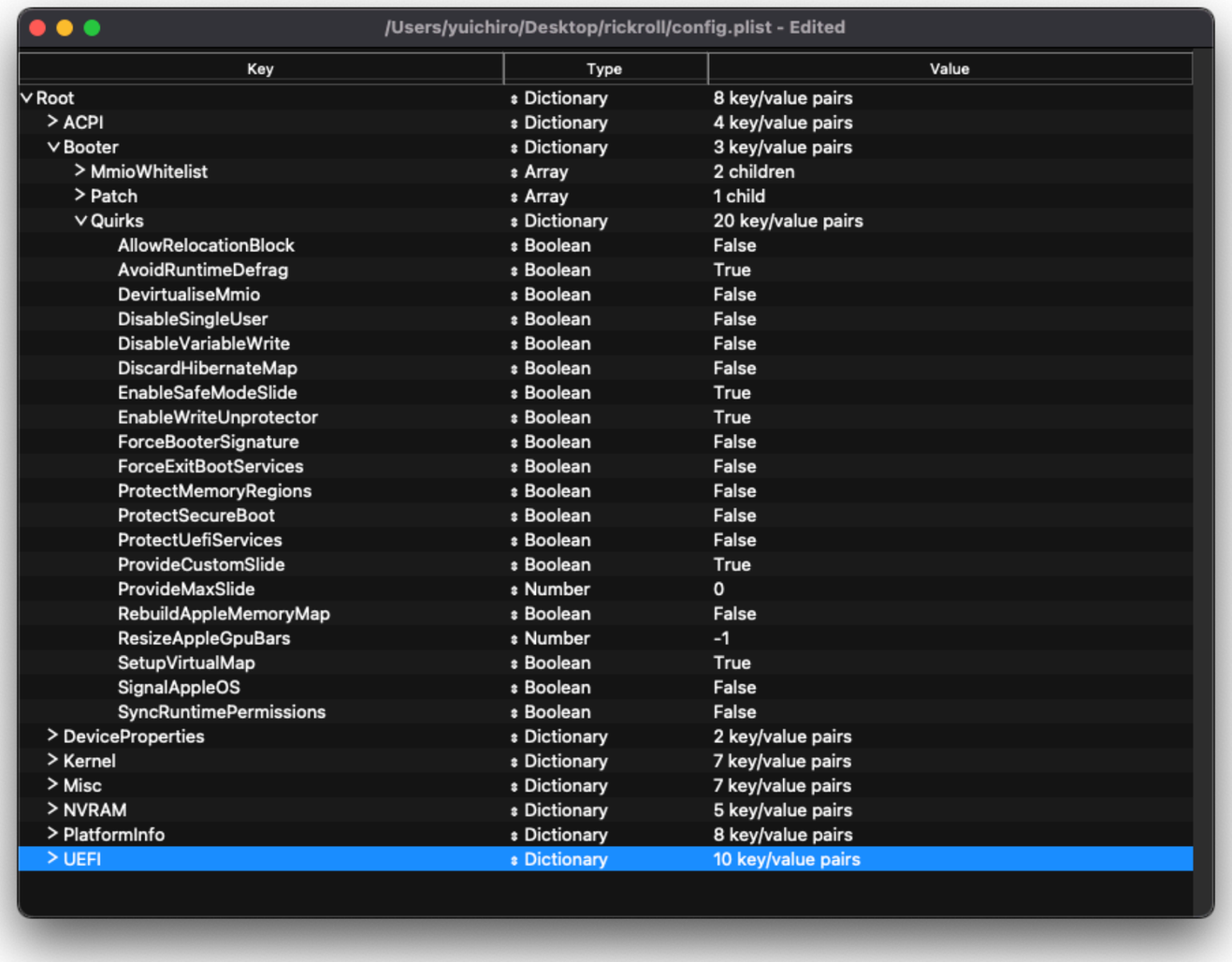
Comment	String	Change _OSI to XOSI
Enabled	Boolean	YES

Count	Number	0
Limit	Number	0
Find	Data	`5f4f5349`
Replace	Data	`584f5349`

Quirks

Settings relating to ACPI, leave everything here as default as we have no use for these quirks.

Booter



This section is dedicated to quirks relating to boot.efi patching with OpenRuntime, the replacement for AptioMemoryFix.efi

MmioWhitelist

This section is allowing spaces to be pass-through to macOS that are generally ignored, useful when paired with `DevirtualiseMmio`

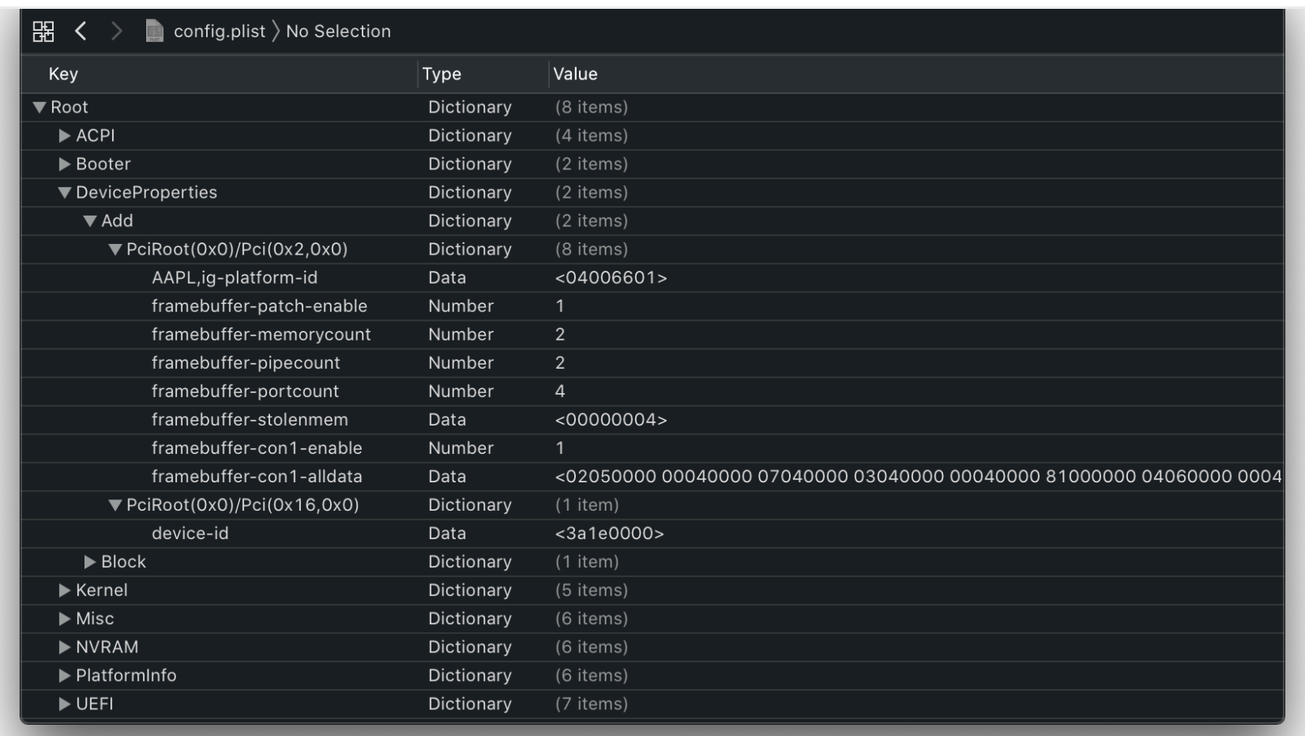
Quirks

Info

Settings relating to boot.efi patching and firmware fixes, for us, we leave it as default

▶ More in-depth Info

DeviceProperties



Add

Sets device properties from a map.

PciRoot(0x0)/Pci(0x2,0x0)

This section is set up via WhateverGreen's Framebuffer Patching Guide and is used for setting important iGPU properties.

The config.plist doesn't already have a section for this so you will have to create it manually.

When setting up your iGPU, the table below should help with finding the right values to set. Here is an explanation of some values:

- **AAPL,ig-platform-id**
  - This is used internally for setting up the iGPU
- **Type**
  - Whether the entry is recommended for laptops(ie. with built-in displays) or for Intel NUCs(ie. stand alone boxes)

Generally follow these steps when setting up your iGPU properties. Follow the configuration notes below the table if they say anything different:

1. When initially setting up your config.plist, only set AAPL,ig-platform-id - this is normally enough
2. If you boot and you get no graphics acceleration (7MB VRAM and solid background for dock), then you likely need to try different `AAPL,ig-platform-id` values, add stolenmem patches, or even add a `device-id` property.

AAPL,ig-platform-id	Type	Comment
`03006601`	Laptop	To be used with **1366 by 768** displays or lower
`04006601`	Laptop	To be used with **1600 by 900** displays or higher, see below for addition patches
`09006601`	Laptop	To be used with some devices that have `eDP` connected monitor (contrary to classical LVDS), must be tested with **03006601** and **04006601** first before trying this.
`0B006601`	NUC	To be used with Intel NUCs

Configuration Notes

- VGA is *not* supported (unless it's running through a DP to VGA internal adapter, which apparently only rare devices will see it as DP and not VGA, it's all about luck.)
- If you're using `04006601` as your ig-platform-id, you may need to add the following parameters to fix external outputs as otherwise you will only have one output. (Credit to Rehabman)

Key	Type	Value	Explanation
`framebuffer-patch-enable`	Number	`1`	*enabling the semantic patches in principle* (from the WhateverGreen manual)
`framebuffer-memorycount`	Number	`2`	Matching FBMemoryCount to the one on `03006601` (1 on `04` vs 2 on `03`)
`framebuffer-pipecount`	Number	`2`	Matching PipeCount to the one on `03006601` (3 on `04` vs 2 on `03`)
`framebuffer-portcount`	Number	`4`	Matching PortCount to the one on `03006601` (1 on `04` vs 4 on `03`)

`framebuffer-stolenmem`	Data	`00000004`	Matching STOLEN memory to 64MB (0x04000000 from hex to base 10 in Bytes) to the one on `03006601` Check here for more information.
`framebuffer-con1-enable`	Number	`1`	This will enable patching on *connector 1* of the driver. (Which is the second connector after con0, which is the eDP/LVDS one)
`framebuffer-con1-alldata`	Data	`02050000 00040000 07040000 03040000 00040000 81000000 04060000 00040000 81000000`	When using `all data` with a connector, either you give all information of that connector (port-bused-type-flag) or that port and the ones following it, like in this case. In this case, the ports in `04` are limited to `1`: `05030000 02000000 30020000` (which corresponds to port 5, which is LVDS) However on `03` there are 3 extra ports: `05030000 02000000 30000000` (LVDS, con0, like `04`) `02050000 00040000 07040000` (DP, con1) `03040000 00040000 81000000` (DP, con2) `04060000 00040000 81000000` (DP, con3) Since we changed the number of PortCount to `4` in a platform that has only 1, that means we need to define the 3 others (and we that starting with con1 to the end).

PciRoot(0x0)/Pci(0x16,0x0)

Sandy/IvyBridge Hybrids:

Some laptops from this era came with a mixed chipset setup, using Ivy Bridge CPUs with Sandy Bridge chipsets which creates issues with macOS since it expects a certain IMEI ID that it doesn't find and would get stuck at boot(As Apple's iGPU drivers require an IMEI device), to fix this we need to fake the IMEI's IDs in these models

- To know if you're affected check if your CPU is an Intel Core ix-3xxx and your chipset is Hx6x (for example a laptop with HM65 or HM67 with a Core i3-3110M) through tools like AIDA64.
- In your config add a new PciRoot device named `PciRoot(0x0)/Pci(0x16,0x0)`

Key	Type	Value
device-id	Data	`3A1E0000`

PciRoot(0x0)/Pci(0x1b,0x0)

`layout-id`

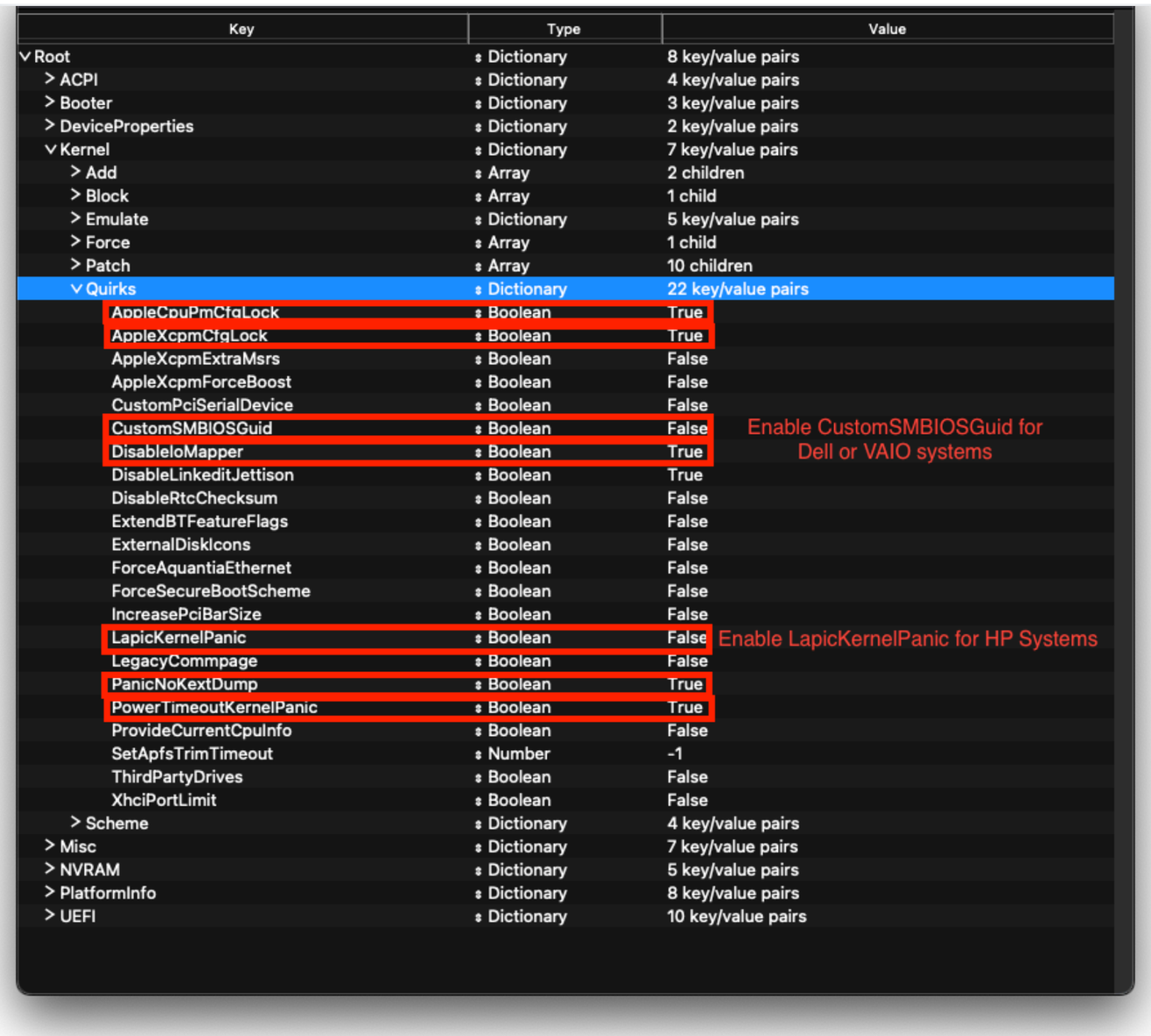
- Applies AppleALC audio injection, you'll need to do your own research on which codec your motherboard has and match it with AppleALC's layout. AppleALC Supported Codecs .
- You can delete this property outright as it's unused for us at this time

For us, we'll be using the boot argument `alcid=xxx` instead to accomplish this. `alcid` will override all other layout-IDs present. More info on this is covered in the Post-Install Page

Delete

Removes device properties from the map, for us we can ignore this

Kernel



Add

Here's where we specify which kexts to load, in what specific order to load, and what architectures each kext is meant for. By default we recommend leaving what ProperTree has done, however for 32-bit CPUs please see below:

▶ More in-depth Info

Emulate

Needed for spoofing unsupported CPUs like Pentiums and Celerons

- **Cpuid1Mask**: Leave this blank
- **Cpuid1Data**: Leave this blank

Force

Used for loading kexts off system volume, only relevant for older operating systems where certain kexts are not present in the cache(ie. IONetworkingFamily in 10.6).

For us, we can ignore.

Block

Blocks certain kexts from loading. Not relevant for us.

Patch

Patches both the kernel and kexts.

Quirks

Info

Settings relating to the kernel, for us we'll be enabling the following:

Quirk	Enabled	Comment
AppleCpuPmCfgLock	YES	Not needed if `CFG-Lock` is disabled in the BIOS
DisableIoMapper	YES	Not needed if `VT-D` is disabled in the BIOS
LapicKernelPanic	NO	HP Machines will require this quirk
PanicNoKextDump	YES	
PowerTimeoutKernelPanic	YES	
XhciPortLimit	YES	Disable if running macOS 11.3+

▶ More in-depth Info

Scheme

Settings related to legacy booting(ie. 10.4-10.6), for majority you can skip however for those planning to boot legacy OSes you can see below:

▶ More in-depth Info

Misc

Boot

Info

Quirk	Enabled	Comment
HideAuxiliary	YES	Press space to show macOS recovery and other auxiliary entries

▶ More in-depth Info

Debug

Info

Helpful for debugging OpenCore boot issues(We'll be changing everything *but* `DisplayDelay`):

Quirk	Enabled
AppleDebug	YES
ApplePanic	YES
DisableWatchDog	YES
Target	67

▶ More in-depth Info

Security

Info

Security is pretty self-explanatory, **do not skip**. We'll be changing the following:

Quirk	Enabled	Comment
AllowSetDefault	YES	
BlacklistAppleUpdate	YES	
ScanPolicy	0	
SecureBootModel	Default	Leave this as `Default` for OpenCore to automatically set the correct value corresponding to your SMBIOS. The next page goes into more detail about this setting.
Vault	Optional	This is a word, it is not optional to omit this setting. You will regret it if you don't set it to Optional, note that it is case-sensitive

▶ More in-depth Info

Serial

Used for serial debugging (Leave everything as default).

Tools

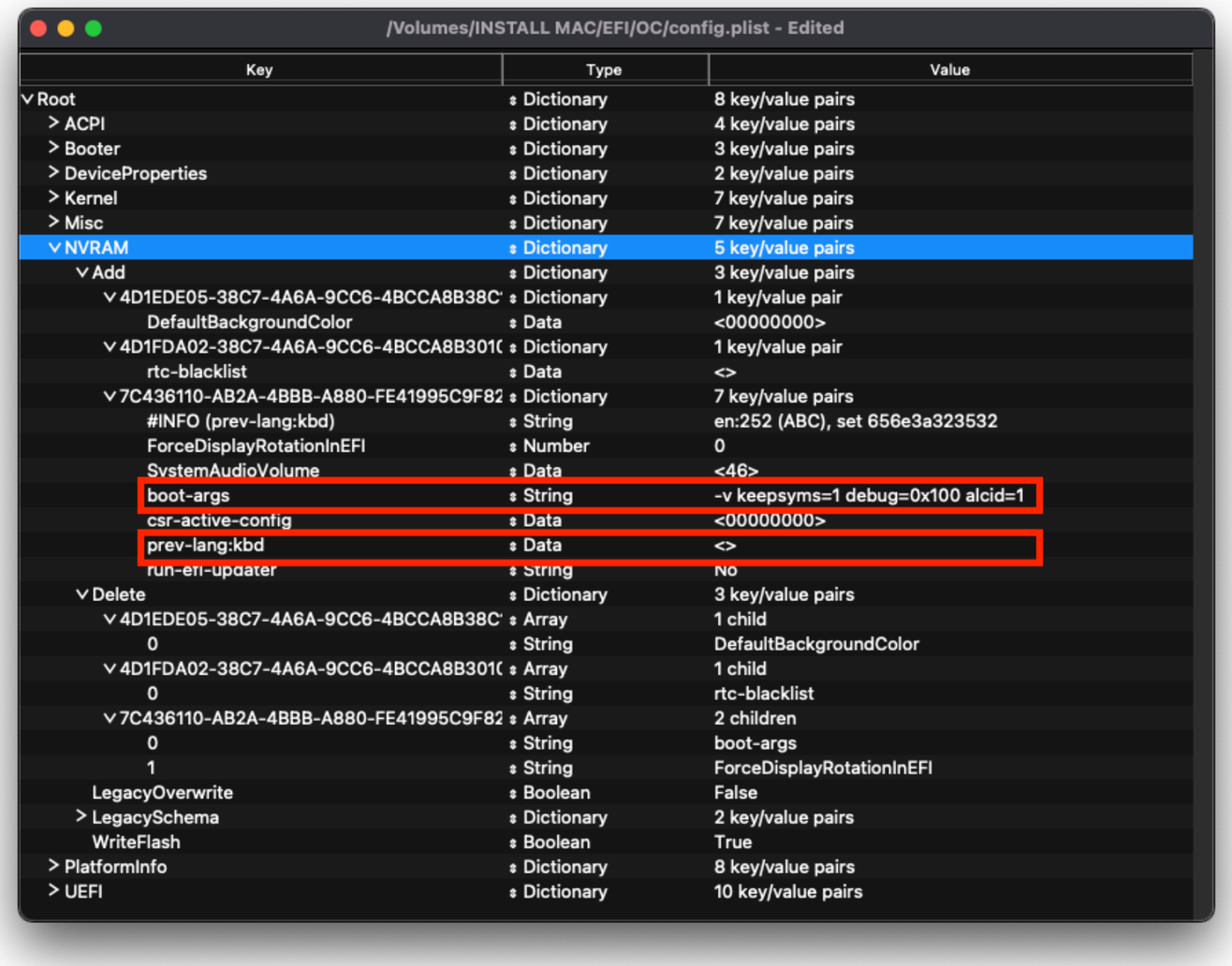
Used for running OC debugging tools like the shell, ProperTree's snapshot function will add these for you.

Entries

Used for specifying irregular boot paths that can't be found naturally with OpenCore.

Won't be covered here, see 8.6 of Configuration.pdf for more info

NVRAM



Add

4D1EDE05-38C7-4A6A-9CC6-4BCCA8B38C14

Used for OpenCore's UI scaling, default will work for us. See in-depth section for more info

▶ More in-depth Info

4D1FDA02-38C7-4A6A-9CC6-4BCCA8B30102

OpenCore's NVRAM GUID, mainly relevant for RTCMemoryFixup users

7C436110-AB2A-4BBB-A880-FE41995C9F82

System Integrity Protection bitmask

- **General Purpose boot-args**:

boot-args	Description
-v	This enables verbose mode, which shows all the behind-the-scenes text that scrolls by as you're booting instead of the Apple logo and progress bar. It's invaluable to any Hackintosher, as it gives you an inside look at the boot process, and can help you identify issues, problem kexts, etc.
debug=0x100	This disables macOS's watchdog which helps prevents a reboot on a kernel panic. That way you can *hopefully* glean some useful info and follow the breadcrumbs to get past the issues.
keepsyms=1	This is a companion setting to debug=0x100 that tells the OS to also print the symbols on a kernel panic. That can give some more helpful insight as to what's causing the panic itself.
alcid=1	Used for setting layout-id for AppleALC, see supported codecs to figure out which layout to use for your specific system. More info on this is covered in the Post-Install Page

- **GPU-Specific boot-args**:

boot-args	Description
-wegnoegpu	Used for disabling all other GPUs than the integrated Intel iGPU, useful for those wanting to run newer versions of macOS where their dGPU isn't supported

- **csr-active-config**: `00000000`
  - Settings for 'System Integrity Protection' (SIP). It is generally recommended to change this with `csrutil` via the recovery partition.
  - csr-active-config by default is set to `00000000` which enables System Integrity Protection. You can choose a number of different values but overall we recommend keeping this enabled for best security practices. More info can be found in our troubleshooting page: Disabling SIP
- **run-efi-updater**: `No`
  - This is used to prevent Apple's firmware update packages from installing and breaking boot order; this is important as these firmware updates (meant for Macs) will not work.
- **prev-lang:kbd**: <>
  - Needed for non-latin keyboards in the format of `lang-COUNTRY:keyboard`, recommended to keep blank though you can specify it(**Default in Sample config is Russian**):
  - American: `en-US:0` (`656e2d55533a30` in HEX)
  - Full list can be found in AppleKeyboardLayouts.txt
  - Hint: `prev-lang:kbd` can be changed into a String so you can input `en-US:0` directly instead of converting to HEX
  - Hint 2: `prev-lang:kbd` can be set to a blank variable (eg. `<>`) which will force the Language Picker to appear instead at first boot up.

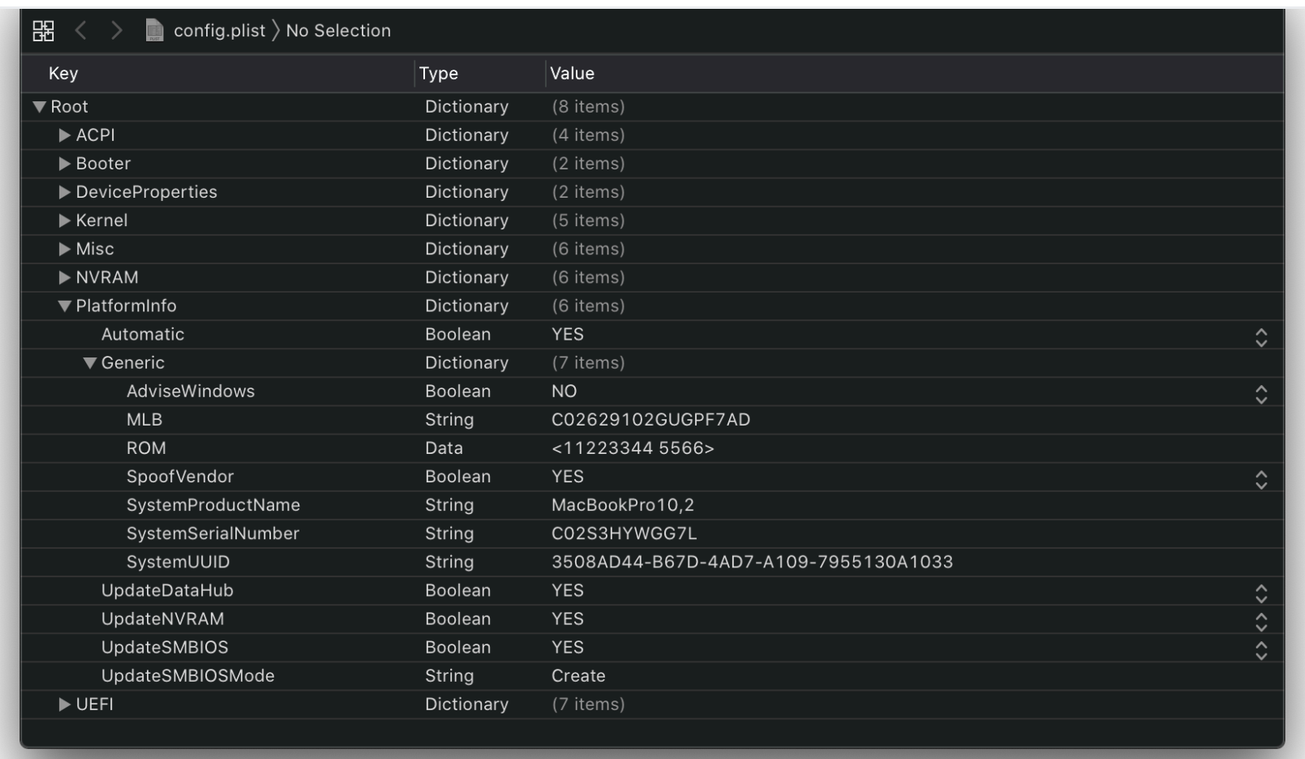
Key	Type	Value
prev-lang:kbd	String	en-US:0

Delete

Forcibly rewrites NVRAM variables, do note that `Add` **will not overwrite** values already present in NVRAM so values like `boot-args` should be left alone.

- **LegacySchema**
  - Used for assigning NVRAM variables, used with `OpenVariableRuntimeDxe.efi`. Only needed for systems without native NVRAM
- **WriteFlash**: YES
  - Enables writing to flash memory for all added variables.

PlatformInfo



Info

For setting up the SMBIOS info, we'll use CorpNewt's GenSMBIOS application.

For this Ivy Bridge example, we'll chose the iMac13,2 SMBIOS - this is done intentionally for compatibility's sake. The typical breakdown is as follows:

SMBIOS	CPU Type	GPU Type	Display Size
MacBookAir5,1	Dual Core 17W	iGPU: HD 4000	11"
MacBookAir5,2	Dual Core 17W	iGPU: HD 4000	13"
MacBookPro10,1	Quad Core 45W	iGPU: HD 4000 + dGPU: GT 650M	15"
MacBookPro10,2	Dual Core 35W(High End)	iGPU: HD 4000	13"
Macmini6,1	Dual Core NUC	iGPU: HD 4000	N/A
Macmini6,2	Quad Core NUC	iGPU: HD 4000	N/A

Note: The following SMBIOS are only supported up-to and including macOS 10.15, Catalina. For cases where you must boot Big Sur, see below:

▶ Big Sur SMBIOS table

Run GenSMBIOS, pick option 1 for downloading MacSerial and Option 3 for selecting out SMBIOS. This will give us an output similar to the following:

```sh
  ##########################################################
 #              MacBookPro10,2 SMBIOS Info                #
##########################################################

Type:         MacBookPro10,2
Serial:       C02KCYZLDNCW
Board Serial: C02309301QXF2FRJC
SmUUID:       A154B586-874B-4E57-A1FF-9D6E503E4580
```

The `Type` part gets copied to Generic -> SystemProductName.

The `Serial` part gets copied to Generic -> SystemSerialNumber.

The `Board Serial` part gets copied to Generic -> MLB.

The `SmUUID` part gets copied to Generic -> SystemUUID.

We set Generic -> ROM to either an Apple ROM (dumped from a real Mac), your NIC MAC address, or any random MAC address (could be just 6 random bytes, for this guide we'll use `11223300 0000`. After install follow the Fixing iServices page on how to find your real MAC Address)

Reminder that you need an invalid serial! When inputting your serial number in Apple's Check Coverage Page, you should get a message such as "Unable to check coverage for this serial number."

Automatic: YES

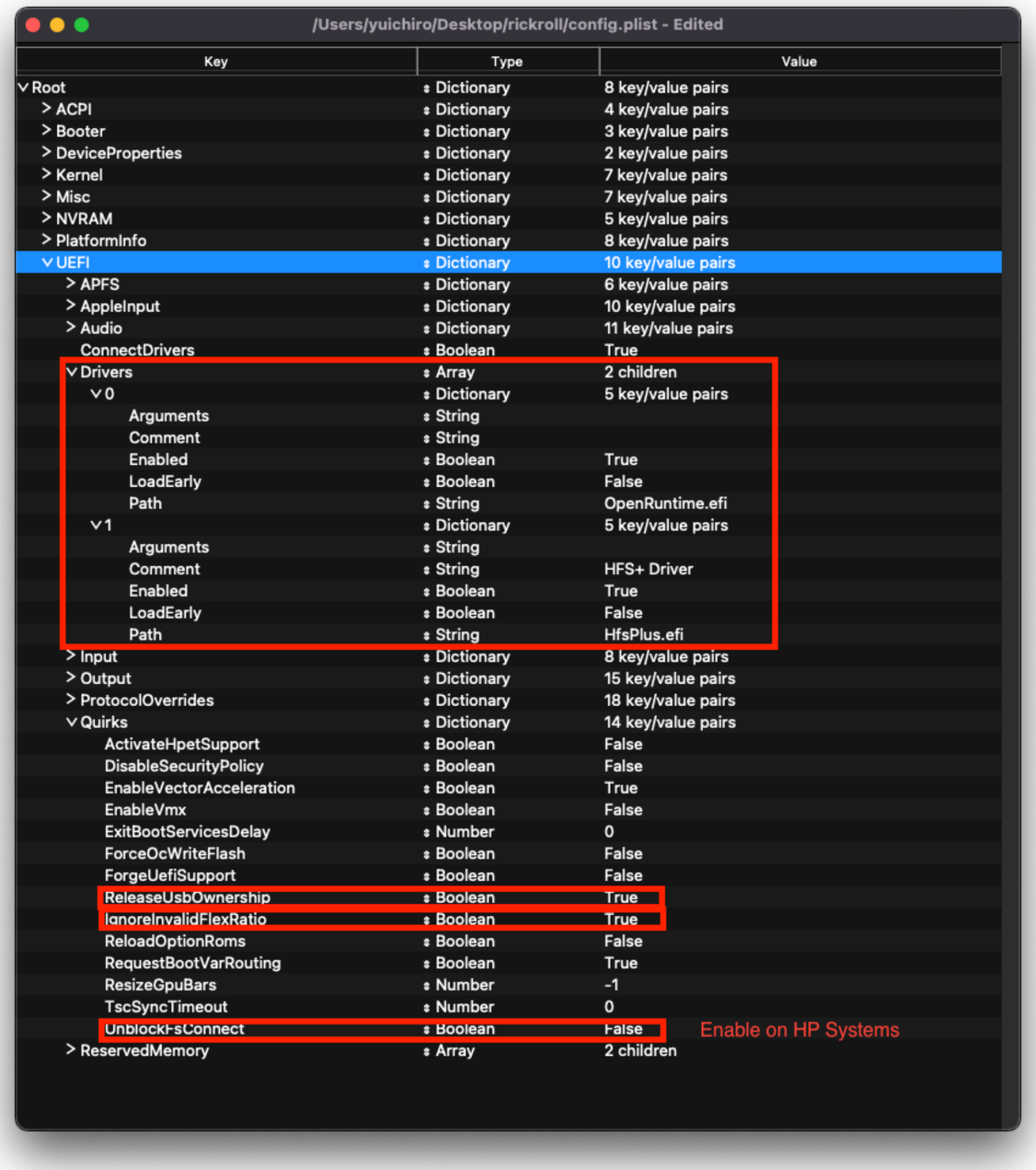
- Generates PlatformInfo based on Generic section instead of DataHub, NVRAM, and SMBIOS sections

:::

Generic

▶ More in-depth Info

UEFI



ConnectDrivers: YES

- Forces .efi drivers, change to NO will automatically connect added UEFI drivers. This can make booting slightly faster, but not all drivers connect themselves. E.g. certain file system drivers may not load.

Drivers

Add your .efi drivers here.

Only drivers present here should be:

- HfsPlus.efi
- OpenRuntime.efi

▶ More in-depth Info

APFS

By default, OpenCore only loads APFS drivers from macOS Big Sur and newer. If you are booting macOS Catalina or earlier, you may need to set a new minimum version/date. Not setting this can result in OpenCore not finding your macOS partition!

macOS Sierra and earlier use HFS instead of APFS. You can skip this section if booting older versions of macOS.

APFS Versions

Both MinVersion and MinDate need to be set if changing the minimum version.

macOS Version	Min Version	Min Date
High Sierra (`10.13.6`)	`748077008000000`	`20180621`
Mojave (`10.14.6`)	`945275007000000`	`20190820`
Catalina (`10.15.4`)	`1412101001000000`	`20200306`
No restriction	`-1`	`-1`

Audio

Related to AudioDxe settings, for us we'll be ignoring(leave as default). This is unrelated to audio support in macOS.

Input

Related to boot.efi keyboard passthrough used for FileVault and Hotkey support, leave everything here as default as we have no use for these quirks. See here for more details: Security and FileVault

Output

Relating to OpenCore's visual output, leave everything here as default as we have no use for these quirks.

▶ More in-depth Info

ProtocolOverrides

Mainly relevant for Virtual machines, legacy macs and FileVault users. See here for more details: Security and FileVault

Quirks

Info

Relating to quirks with the UEFI environment, for us we'll be changing the following:

Quirk	Enabled	Comment
IgnoreInvalidFlexRatio	YES	
ReleaseUsbOwnership	YES	
UnblockFsConnect	NO	Needed mainly by HP motherboards

▶ More in-depth Info

ReservedMemory

Used for exempting certain memory regions from OSes to use, mainly relevant for Sandy Bridge iGPUs or systems with faulty memory. Use of this quirk is not covered in this guide

Cleaning up

And now you're ready to save and place it into your EFI under EFI/OC.

For those having booting issues, please make sure to read the Troubleshooting section first and if your questions are still unanswered we have plenty of resources at your disposal:

- r/Hackintosh Subreddit
- r/Hackintosh Discord

Config reminders

HP Users:

- Kernel -> Quirks -> LapicKernelPanic -> True
  - You will get a kernel panic on LAPIC otherwise
- UEFI -> Quirks -> UnblockFsConnect -> True

Intel BIOS settings

- Note: Most of these options may not be present in your firmware, we recommend matching up as closely as possible but don't be too concerned if many of these options are not available in your BIOS

These are the main options to check for, if you can't find it or an equivalent for it, just skip it.

Disable

- Fast Boot
- Secure Boot
- Serial/COM Port
- Parallel Port
- VT-d (can be enabled if you set `DisableIoMapper` to YES)
- Compatibility Support Module (CSM) (**Must be off in most cases, GPU errors/stalls like `gIO` are common when this option is enabled**) (or Legacy Support, or Hybrid Boot)
- Thunderbolt (For initial install, as Thunderbolt can cause issues if not setup correctly, if available)
- Intel SGX
- Intel Platform Trust
- CFG Lock (MSR 0xE2 write protection)(**This must be off, if you can't find the option then enable `AppleCpuPmCfgLock` under Kernel -> Quirks. Your hack will not boot with CFG-Lock enabled**)

Enable

- Hyper-Threading
- Execute Disable Bit
- EHCI/XHCI Hand-off
- OS type: Windows 8.1/10 UEFI Mode (some motherboards may require "Other OS" instead)
- DVMT Pre-Allocated(iGPU Memory): 32MB or higher
- SATA Mode: AHCI

Once done here, we need to edit a couple extra values. Head to the Apple Secure Boot Page