



# OpenCore

Reference Manual (0.6.~~1~~.2)

[2020.09.11]

- GSTT — GoptStop
- HDA — AudioDxe
- KKT — KeyTester
- MMDD — MmapDump
- OCPAVP — PavpProvision
- OCRST — ResetSystem
- OCUI — OpenCanopy
- OC — OpenCore main
- VMOPT — VerifyMemOpt

#### Libraries:

- AAPL — OcDebugLogLib, Apple EfiBoot logging
- OCABC — OcAfterBootCompatLib
- OCAE — OcAppleEventLib
- OCAK — OcAppleKernelLib
- OCAU — OcAudioLib
- OCAV — OcAppleImageVerificationLib
- OCA — OcAcpiLib
- OCBP — OcAppleBootPolicyLib
- OCB — OcBootManagementLib
- OCCL — OcAppleChunkListLib
- OCCPU — OcCpuLib
- OCC — OcConsoleLib
- [OCDC — OcDriverConnectionLib](#)
- OCDH — OcDataHubLib
- OCDI — OcAppleDiskImageLib
- ~~OCFSQ — OcFileLib, UnblockFs quirk~~
- OCFS — OcFileLib
- OCFV — OcFirmwareVolumeLib
- OCHS — OcHashServicesLib
- OCIA4 — OcAppleImg4Lib
- OCIC — OcImageConversionLib
- OCII — OcInputLib
- OCJS — OcApfsLib
- OCKM — OcAppleKeyMapLib
- OCL — OcDebugLogLib
- OCMCO — OcMachoLib
- OCME — OcHeciLib
- OCMM — OcMemoryLib
- OCPI — OcFileLib, partition info
- OCPNG — OcPngLib
- OCRAM — OcAppleRamDiskLib
- OCRTC — OcRtcLib
- OCSB — OcAppleSecureBootLib
- OCSMB — OcSmbiosLib
- OCSMC — OcSmcLib
- OCST — OcStorageLib
- OCS — OcSerializedLib
- OCTPL — OcTemplateLib
- OCUC — OcUnicodeCollationLib
- OCUT — OcAppleUserInterfaceThemeLib
- OCXML — OcXmlLib

## 8.5 Security Properties

1. AllowNvramReset  
Type: plist boolean  
Failsafe: false

- `OC_SCAN_ALLOW_DEVICE_SATA`
- `OC_SCAN_ALLOW_DEVICE_SASEX`
- `OC_SCAN_ALLOW_DEVICE_SCSI`
- `OC_SCAN_ALLOW_DEVICE_NVME`

#### 14. `SecureBootModel`

**Type:** plist string

**Failsafe:** `Default`

**Description:** Apple Secure Boot hardware model.

Sets Apple Secure Boot hardware model and policy. Specifying this value defines which operating systems will be bootable. Operating systems shipped before the specified model was released will not boot. Valid values:

- **Default** — Recent available model, currently set to `j137`.
- **Disabled** — No model, Secure Boot will be disabled.
- `j137` — `iMacPro1,1` (December 2017) minimum macOS 10.13.2 (17C2111)
- `j680` — `MacBookPro15,1` (July 2018) minimum macOS 10.13.6 (17G2112)
- `j132` — `MacBookPro15,2` (July 2018) minimum macOS 10.13.6 (17G2112)
- `j174` — `Macmini8,1` (October 2018) minimum macOS 10.14 (18A2063)
- `j140k` — `MacBookAir8,1` (October 2018) minimum macOS 10.14.1 (18B2084)
- `j780` — `MacBookPro15,3` (May 2019) minimum macOS 10.14.5 (18F132)
- `j213` — `MacBookPro15,4` (July 2019) minimum macOS 10.14.5 (18F2058)
- `j140a` — `MacBookAir8,2` (July 2019) minimum macOS 10.14.5 (18F2058)
- `j152f` — `MacBookPro16,1` (November 2019) minimum macOS 10.15.1 (19B2093)
- `j160` — `MacPro7,1` (December 2019) minimum macOS 10.15.1 (19B88)
- `j230k` — `MacBookAir9,1` (March 2020) minimum macOS 10.15.3 (19D2064)
- `j214k` — `MacBookPro16,2` (May 2020) minimum macOS 10.15.4 (19E2269)
- `j223` — `MacBookPro16,3` (May 2020) minimum macOS 10.15.4 (19E2265)
- `j215` — `MacBookPro16,4` (June 2020) minimum macOS 10.15.5 (19F96)
- `j185` — `iMac20,1` (August 2020) minimum macOS 10.15.6 (19G2005)
- `j185f` — `iMac20,2` (August 2020) minimum macOS 10.15.6 (19G2005)

`PlatformInfo` and `SecureBootModel` are independent, allowing to enabling Apple Secure Boot with any SMBIOS. Setting `SecureBootModel` to any valid value but `Disabled` is equivalent to **Medium Security** of Apple Secure Boot. To achieve **Full Security** one will need to also specify `ApECID` value.

Enabling Apple Secure Boot is more demanding to incorrect configurations, buggy macOS installations, and unsupported setups. Things to keep in mind:

- Just like on T2 Macs you will not be able to install any unsigned kernel drivers and several signed kernel drivers including NVIDIA Web Drivers.
- The list of cached drivers may be different, resulting in the need to change the list of **Added** or **Forced** kernel drivers. For example, `I080211Family` cannot be injected in this case.
- System volume alterations on operating systems with sealing, like macOS 11, may result in the operating system being unbootable. Do not try to disable system volume encryption unless you disable Apple Secure Boot.
- If your platform requires certain settings, but they were not enabled, because the obvious issues did not trigger before, you may get boot failure. Be extra careful with `IgnoreInvalidFlexRatio` or `HashServices`.
- Operating systems released before Apple Secure Boot landed (e.g. macOS 10.12 or earlier) will still boot until UEFI Secure Boot is enabled. This is so, because from Apple Secure Boot point they are treated as incompatible and are assumed to be handled by the firmware just like Microsoft Windows is.
- On older CPUs (e.g. before Sandy Bridge) enabling Apple Secure Boot might cause slightly slower loading by up to 1 second.
- Since `Default` value will increase with time to support the latest major release operating system, it is not recommended to use `ApECID` and `Default` value together.

Sometimes the already installed operating system may have outdated Apple Secure Boot manifests on the **Preboot** partition causing boot failure. If you see the “OCB: Apple Secure Boot prohibits this boot entry, enforcing!” message, it is likely the case. When this happens you can either reinstall the operating system or copy the manifests (files with `.im4m` extension, like `boot.efi.j137.im4m`) from `/usr/standalone/i386` to `/Volumes/Preboot/<UUID>/System/Library/CoreServices`. Here `<UUID>` is your system volume identifier. [On](#)

- `4D1EDE05-38C7-4A6A-9CC6-4BCCA8B38C14:HW_BID`  
Hardware BoardProduct (e.g. `Mac-35C1E88140C3E6CF`). Not present on real Macs, but used to avoid extra parsing of SMBIOS tables, especially in `boot.efi`.
- `4D1EDE05-38C7-4A6A-9CC6-4BCCA8B38C14:HW_MLB`  
Hardware BoardSerialNumber. Override for MLB. Present on newer Macs (2013+ at least).
- `4D1EDE05-38C7-4A6A-9CC6-4BCCA8B38C14:HW_ROM`  
Hardware ROM. Override for ROM. Present on newer Macs (2013+ at least).
- `7C436110-AB2A-4BBB-A880-FE41995C9F82:prev-lang:kbd`  
ASCII string defining default keyboard layout. Format is `lang-COUNTRY:keyboard`, e.g. `ru-RU:252` for Russian locale and ABC keyboard. Also accepts short forms: `ru:252` or `ru:0` (U.S. keyboard, compatible with 10.9). Full decoded keyboard list from `AppleKeyboardLayouts-L.dat` can be found [here](#). Using non-latin keyboard on 10.14 will not enable ABC keyboard, unlike previous and subsequent macOS versions, and is thus not recommended in case you need 10.14.
- `7C436110-AB2A-4BBB-A880-FE41995C9F82:security-mode`  
ASCII string defining FireWire security mode. Legacy, can be found in `IOFireWireFamily` source code in `IOFireWireController.cpp`. It is recommended not to set this variable, which may speedup system startup. Setting to `full` is equivalent to not setting the variable and `none` disables FireWire security.
- `4D1EDE05-38C7-4A6A-9CC6-4BCCA8B38C14:UIScale`  
One-byte data defining `boot.efi` user interface scaling. Should be `01` for normal screens and `02` for HiDPI screens.
- `4D1EDE05-38C7-4A6A-9CC6-4BCCA8B38C14:DefaultBackgroundColor`  
Four-byte BGRA data defining `boot.efi` user interface background colour. Standard colours include `BF BF BF 00` (Light Gray) and `00 00 00 00` (Syrah Black). Other colours may be set at user's preference.

## 9.5 Other Variables

The following variables may be useful for certain configurations or troubleshooting:

- `7C436110-AB2A-4BBB-A880-FE41995C9F82:boot-args`  
Kernel arguments, used to pass configuration to Apple kernel and drivers. There are many arguments, which may be found by looking for the use of `PE_parse_boot_argn` function in the kernel or driver code. Some of the known boot arguments include:
  - `acpi_layer=0xFFFFFFFF`
  - `acpi_level=0xFFFF5F` (implies `ACPI_ALL_COMPONENTS`)
  - `arch=i386` (force kernel architecture to `i386`, see [KernelArch](#))
  - `batman=VALUE` (`AppleSmartBatteryManager` debug mask)
  - `batman-nosmc=1` (disable `AppleSmartBatteryManager` SMC interface)
  - `cpus=VALUE` (maximum number of CPUs used)
  - `debug=VALUE` (debug mask)
  - `io=VALUE` (IOKit debug mask)
  - `keepsyms=1` (show panic log debug symbols)
  - `kextlog=VALUE` (kernel extension loading debug mask)
  - `nv_disable=1` (disables NVIDIA GPU acceleration)
  - `nvda_drv=1` (legacy way to enable NVIDIA web driver, removed in 10.12)
  - `npci=0x2000` (legacy, disables `kIOPCIConfiguratorPFM64`)
  - `lapic_dont_panic=1`
  - `slide=VALUE` (manually set KASLR slide)
  - `smcdebug=VALUE` (`AppleSMC` debug mask)
  - `-amd_no_dgpu_accel` (alternative to WhateverGreen's `-radvesa` for new GPUs)
  - `-nehalem_error_disable`
  - `-no_compat_check` (disable model checking [on 10.7+](#))
  - `-s` (single mode)
  - `-v` (verbose mode)
  - `-x` (safe mode)

There are multiple external places summarising macOS argument lists: [example 1](#), [example 2](#).

- `7C436110-AB2A-4BBB-A880-FE41995C9F82:bootercfg`  
Booter arguments, similar to `boot-args` but for `boot.efi`. Accepts a set of arguments, which are hexadecimal 64-bit values with or without `0x`. At different stages `boot.efi` will request different debugging (logging) modes

- **TryOverwrite** — **Overwrite** if new size is  $\leq$  than the page-aligned original and there are no issues with legacy region unlock. **Create** otherwise. Has issues with some firmwares.
- **Create** — Replace the tables with newly allocated `EfiReservedMemoryType` at `AllocateMaxAddress` without any fallbacks.
- **Overwrite** — **Overwrite** existing `gEfiSmbiosTableGuid` and `gEfiSmbiosTable3Guid` data if it fits new size. Abort with unspecified state otherwise.
- **Custom** — Write SMBIOS tables (`gEfiSmbios(3)TableGuid`) to `gOcCustomSmbios(3)TableGuid` to workaround firmwares overwriting SMBIOS contents at `ExitBootServices`. Otherwise equivalent to **Create**. Requires patching `AppleSmbios.kext` and `AppleACPIPlatform.kext` to read from another GUID: "EB9D2D31" - "EB9D2D35" (in ASCII), done automatically by `CustomSMBIOSGuid` quirk.

*Note:* A side effect of using **Custom** approach is making SMBIOS updates exclusive to macOS, avoiding a collision with existing Windows activation and custom OEM software but potentially breaking Apple-specific tools.

## 6. Generic

**Type:** plist dictionary

**Description:** Update all fields. This section is read only when **Automatic** is active.

## 7. DataHub

**Type:** plist dictionary

**Optional:** When **Automatic** is true

**Description:** Update Data Hub fields. This section is read only when **Automatic** is not active.

## 8. PlatformNVRAM

**Type:** plist dictionary

**Optional:** When **Automatic** is true

**Description:** Update platform NVRAM fields. This section is read only when **Automatic** is not active.

## 9. SMBIOS

**Type:** plist dictionary

**Optional:** When **Automatic** is true

**Description:** Update SMBIOS fields. This section is read only when **Automatic** is not active.

# 10.2 Generic Properties

## 1. SpoofVendor

**Type:** plist boolean

**Failsafe:** false

**Description:** Sets SMBIOS vendor fields to `Acidanthera`.

It is dangerous to use Apple in SMBIOS vendor fields for reasons given in `SystemManufacturer` description. However, certain firmwares may not provide valid values otherwise, which could break some software.

## 2. AdviseWindows

**Type:** plist boolean

**Failsafe:** false

**Description:** Forces Windows support in `FirmwareFeatures`.

Added bits to `FirmwareFeatures`:

- `FW_FEATURE_SUPPORTS_CSM_LEGACY_MODE` (0x1) - Without this bit it is not possible to reboot to Windows installed on a drive with EFI partition being not the first partition on the disk.
- `FW_FEATURE_SUPPORTS_UEFI_WINDOWS_BOOT` (0x20000000) - Without this bit it is not possible to reboot to Windows installed on a drive with EFI partition being the first partition on the disk.

## 3. ProcessorType **Type:** plist integer

**Failsafe:** Automatic

**Description:** Refer to SMBIOS ProcessorType.

## 4. SystemProductName

**Type:** plist string

**Failsafe:** MacPro6,1

**Description:** Refer to SMBIOS `SystemProductName`.

**SMBIOS:** System Enclosure or Chassis (Type 3) — Type

**Description:** Chassis type, refer to Table 17 — System Enclosure or Chassis Types for more details.

20. ChassisVersion

**Type:** plist string

**Failsafe:** OEM specified

**SMBIOS:** System Enclosure or Chassis (Type 3) — Version

**Description:** Should match BoardProduct.

21. ChassisSerialNumber

**Type:** plist string

**Failsafe:** OEM specified

**SMBIOS:** System Enclosure or Chassis (Type 3) — Version

**Description:** Should match SystemSerialNumber.

22. ChassisAssetTag

**Type:** plist string

**Failsafe:** OEM specified

**SMBIOS:** System Enclosure or Chassis (Type 3) — Asset Tag Number

**Description:** Chassis type name. Varies, could be empty or MacBook-Aluminum.

23. PlatformFeature

**Type:** plist integer, 32-bit

**Failsafe:** 0xFFFFFFFF

**SMBIOS:** APPLE\_SMBIOS\_TABLE\_TYPE133 - PlatformFeature

**Description:** Platform features bitmask. Refer to AppleFeatures.h for more details. Use 0xFFFFFFFF value to not provide this table.

24. SmcVersion

**Type:** plist data, 16 bytes

**Failsafe:** All zero

**SMBIOS:** APPLE\_SMBIOS\_TABLE\_TYPE134 - Version

**Description:** ASCII string containing SMC version in upper case. Missing on T2 based Macs. Ignored when zero.

25. FirmwareFeatures

**Type:** plist data, 8 bytes

**Failsafe:** 0

**SMBIOS:** APPLE\_SMBIOS\_TABLE\_TYPE128 - FirmwareFeatures and ExtendedFirmwareFeatures

**Description:** 64-bit firmware features bitmask. Refer to AppleFeatures.h for more details. Lower 32 bits match FirmwareFeatures. Upper 64 bits match ExtendedFirmwareFeatures.

26. FirmwareFeaturesMask

**Type:** plist data, 8 bytes

**Failsafe:** 0

**SMBIOS:** APPLE\_SMBIOS\_TABLE\_TYPE128 - FirmwareFeaturesMask and ExtendedFirmwareFeaturesMask

**Description:** Supported bits of extended firmware features bitmask. Refer to AppleFeatures.h for more details. Lower 32 bits match FirmwareFeaturesMask. Upper 64 bits match ExtendedFirmwareFeaturesMask.

27. ProcessorType

**Type:** plist integer, 16-bit

**Failsafe:** Automatic

**SMBIOS:** APPLE\_SMBIOS\_TABLE\_TYPE131 - ProcessorType

**Description:** Combined of Processor Major and Minor types.

Automatic value generation tries to provide most accurate value for the currently installed CPU. When this fails please make sure to create an issue and provide sysctl machdep.cpu and dmidecode output. For a full list of available values and their limitations (the value will only apply if the CPU core count matches) refer to Apple SMBIOS definitions header here.

28. MemoryFormFactor

**Type:** plist integer, 8-bit