

OpenCore

Reference Manual (0.0~~2~~.3)

[2019.06.06]

Default value: false

Description: Provide reset register and flag in FADT table to enable reboot and shutdown on legacy hardware. Not recommended unless required.

2. ~~IgnoreForWindows~~**Type:** ~~plist boolean~~**Default value:** ~~false~~**Description:** ~~Disable all sorts of ACPI modifications when booting Windows operating system.~~

~~This flag implements a quick workaround for those, who made their ACPI tables incompatible with Windows, but need it right now. Not recommended, as ACPI tables must be compatible with any operating system regardless of the changes.~~

~~Note: This option may be removed in the future.~~

3. NormalizeHeaders

Type: plist boolean

Default value: false

Description: Cleanup ACPI header fields to workaround macOS ACPI implementation bug causing boot crashes. Reference: Debugging AppleACPIPlatform on 10.13 by Alex James aka theracermaster. The issue is fixed in macOS Mojave (10.14).

4. RebaseRegions

Type: plist boolean

Default value: false

Description: Attempt to heuristically relocate ACPI memory regions. Not recommended.

ACPI tables are often generated dynamically by underlying firmware implementation. Among the position-independent code, ACPI tables may contain physical addresses of MMIO areas used for device configuration, usually grouped in regions (e.g. `OperationRegion`). Changing firmware settings or hardware configuration, upgrading or patching the firmware inevitably leads to changes in dynamically generated ACPI code, which sometimes lead to the shift of the addresses in aforementioned `OperationRegion` constructions.

For this reason it is very dangerous to apply any kind of modifications to ACPI tables. The most reasonable approach is to make as few as possible changes to ACPI and try to not replace any tables, especially DSDT. When this is not possible, then at least attempt to ensure that custom DSDT is based on the most recent DSDT or remove writes and reads for the affected areas.

When nothing else helps this option could be tried to avoid stalls at `PCI Configuration Begin` phase of macOS booting by attempting to fix the ACPI addresses. It does not do magic, and only works with most common cases. Do not use unless absolutely required.

5. ResetLogoStatus

Type: plist boolean

Default value: false

Description: Reset BGRT table `Displayed` status field to false.

This works around firmwares that provide BGRT table but fail to handle screen updates afterwards.

6. Identifier
Type: plist string
Default value: Empty string
Description: Kext bundle identifier (e.g. `com.apple.driver.AppleHDA`) or `kernel` for kernel patch.
7. Limit
Type: plist integer
Default value: 0
Description: Maximum number of bytes to search for. Can be set to 0 to look through the whole kext or kernel.
8. Mask
Type: plist data
Default value: Empty data
Description: Data bitwise mask used during find comparison. Allows fuzzy search by ignoring not masked (set to zero) bits. Can be set to empty data to be ignored. Must equal to **Replace** in size otherwise.
9. MatchKernel
Type: plist string
Default value: Empty string
Description: Adds kernel driver to selected macOS version only. The selection happens based on prefix match with the kernel version, i.e. `16.7.0` will match macOS 10.12.6 and `16.` will match any macOS 10.12.x version.
10. Replace
Type: plist data
Default value: Empty data
Description: Replacement data of one or more bytes.
11. ReplaceMask
Type: plist data
Default value: Empty data
Description: Data bitwise mask used during replacement. Allows fuzzy replacement by updating masked (set to non-zero) bits. Can be set to empty data to be ignored. Must equal to **Replace** in size otherwise.
12. Skip
Type: plist integer
Default value: 0
Description: Number of found occurrences to be skipped before replacement is done.

6.6 Quirks Properties

1. AppleCpuPmCfgLock
Type: plist boolean
Default value: false
Description: Disables `PKG_CST_CONFIG_CONTROL (0xE2)` MSR modification in AppleIntelCPUPowerManagement.kext, commonly causing early kernel panic, when it is locked from writing.
Note: This option should avoided whenever possible. Modern firmwares provide **CFG Lock** setting, disabling which is much cleaner. More details about the issue can be found in [VerifyMsrE2](#) notes.
2. AppleXcpmCfgLock
Type: plist boolean
Default value: false
Description: Disables `PKG_CST_CONFIG_CONTROL (0xE2)` MSR modification in XNU kernel, commonly causing early kernel panic, when it is locked from writing (XCPM power management).
Note: This option should avoided whenever possible. Modern firmwares provide **CFG Lock** setting, disabling which is much cleaner. More details about the issue can be found in [VerifyMsrE2](#) notes.
3. [DisableIoMapper](#)
Type: [plist boolean](#)
Default value: [false](#)
Description: [Disables IOMapper support in XNU \(VT-d\), which may conflict with the firmware implementation.](#)

Note: This option is a preferred alternative to dropping DMAR ACPI table and disabling VT-d in firmware preferences, which does not break VT-d support in other systems in case they need it.

4. ExternalDiskIcons

Type: plist boolean

Default value: false

Description: Apply icon type patches to IOAHCIPort.kext to force internal disk icons for all AHCI disks.

Note: This option should avoided whenever possible. Modern firmwares usually have compatible AHCI controllers.

5. ThirdPartyTrim

Type: plist boolean

Default value: false

Description: Patch IOAHCIFamily.kext to force TRIM command support on AHCI SSDs.

Note: This option should avoided whenever possible. NVMe SSDs are compatible without the change. For AHCI SSDs on modern macOS version there is a dedicated built-in utility called **trimforce**.

6. XhciPortLimit

Type: plist boolean

Default value: false

Description: Patch various kexts (AppleUSBXHCI.kext, AppleUSBXHCIPCI.kext, IOUSBHostFamily.kext) to remove USB port count limit of 15 ports.

Note: This option should avoided whenever possible. USB port limit is imposed by the amount of used bits in locationID format and there is no possible way to workaround this without heavy OS modification. The only valid solution is to limit the amount of used ports to 15 (discarding some). More details can be found on AppleLife.ru.

7 Misc

7.1 Introduction

This section contains miscellaneous configuration entries for OpenCore behaviour that does not go to any other sections

7.2 Properties

1. Boot
Type: plist dict
Description: Apply boot configuration described in Boot Properties section below.
2. Debug
Type: plist dict
Description: Apply debug configuration described in Debug Properties section below.
3. Security
Type: plist dict
Description: Apply security configuration described in Security Properties section below.
4. [Tools](#)
Type: [plist array](#)
Description: [Add new entries to boot picker.](#)
[Designed to be filled with plist dict values, describing each block entry. See Tools Properties section below.](#)

7.3 Boot Properties

1. ConsoleMode
Type: plist string
Default value: Empty string
Description: Sets console output mode as specified with the WxH (e.g. 80x24) formatted string. Set to empty string not to change console mode. Set to **Max** to try to use largest available console mode.
2. ConsoleBehaviourOs
Type: plist string
Default value: Empty string
Description: Set console control behaviour upon operating system load.

Console control is a legacy protocol used for switching between text and graphics screen output. Some firmwares do not provide it, yet select operating systems require its presence, which is what **ConsoleControl** UEFI protocol is for.

When console control is available, OpenCore can be made console control aware, and and set different modes for the operating system booter (**ConsoleBehaviourOs**), which normally runs in graphics mode, and its own user interface (**ConsoleBehaviourUi**), which normally runs in text mode. Possible behaviours, set as values of these options, include:

- Empty string — Do not modify console control mode.
- **Text** — Switch to text mode.
- **Graphics** — Switch to graphics mode.
- **ForceText** — Switch to text mode and preserve it (requires **ConsoleControl**).
- **ForceGraphics** — Switch to graphics mode and preserve it (require **ConsoleControl**).

Hints:

- Unless empty works, firstly try to set **ConsoleBehaviourOs** to **Graphics** and **ConsoleBehaviourUi** to **Text**.
- On APTIO IV (Haswell and earlier) it is usually enough to have **ConsoleBehaviourOs** set to **Graphics** and **ConsoleBehaviourUi** set to **ForceText** to avoid visual glitches.
- On APTIO V (Broadwell and newer) **ConsoleBehaviourOs** set to **ForceGraphics** and **ConsoleBehaviourUi** set to **TextForceText** usually works best.
- [On Apple firmwares ConsoleBehaviourOs set to Graphics and ConsoleBehaviourUi set to Text is supposed to work best.](#)

Note: While it may appear obvious, but you have to use an external method to verify `OpenCore.efi` and `BOOTx64.efi` for secure boot path. For this you are recommended to at least enable UEFI SecureBoot with a custom certificate, and sign `OpenCore.efi` and `BOOTx64.efi` with your custom key. More details on customising secure boot on modern firmwares can be found in Taming UEFI SecureBoot paper (in Russian).

5. ScanPolicy

Type: plist integer, 32 bit

Default value: 0xF0103

Description: Define operating system detection policy.

This value allows to prevent scanning (and booting) from untrusted source based on a bitmask (sum) of select flags. As it is not possible to reliably detect every file system or device type, this feature cannot be fully relied upon in open environments, and the additional measures are to be applied.

Third party drivers may introduce additional security (and performance) measures following the provided scan policy. Scan policy is exposed in `scan-policy` variable of 4D1FDA02-38C7-4A6A-9CC6-4BCCA8B30102 GUID for UEFI Boot Services only.

- 0x00000001 (bit 0) — `OC_SCAN_FILE_SYSTEM_LOCK`, restricts scanning to only known file systems defined as a part of this policy. File system drivers may not be aware of this policy, and to avoid mounting of undesired file systems it is best not to load its driver. This bit does not affect dmg mounting, which may have any file system. Known file systems are prefixed with `OC_SCAN_ALLOW_FS_`.
- 0x00000002 (bit 1) — `OC_SCAN_DEVICE_LOCK`, restricts scanning to only known device types defined as a part of this policy. This is not always possible to detect protocol tunneling, so be aware that on some systems it may be possible for e.g. USB HDDs to be recognised as SATA. Cases like this must be reported. Known device types are prefixed with `OC_SCAN_ALLOW_DEVICE_`.
- 0x00000100 (bit 8) — `OC_SCAN_ALLOW_FS_APFS`, allows scanning of APFS file system.
- 0x00000200 (bit 9) — `OC_SCAN_ALLOW_FS_HFS`, allows scanning of HFS file system (must be blessed).
- 0x00010000 (bit 16) — `OC_SCAN_ALLOW_DEVICE_SATA`, allow scanning SATA devices.
- 0x00020000 (bit 17) — `OC_SCAN_ALLOW_DEVICE_SASEX`, allow scanning SAS and Mac NVMe devices.
- 0x00040000 (bit 18) — `OC_SCAN_ALLOW_DEVICE_SCSI`, allow scanning SCSI devices.
- 0x00080000 (bit 19) — `OC_SCAN_ALLOW_DEVICE_NVME`, allow scanning NVMe devices.
- 0x00100000 (bit 20) — `OC_SCAN_ALLOW_DEVICE_ATAPI`, allow scanning CD/DVD devices.
- 0x00200000 (bit 21) — `OC_SCAN_ALLOW_DEVICE_USB`, allow scanning USB devices.
- 0x00400000 (bit 22) — `OC_SCAN_ALLOW_DEVICE_FIREWIRE`, allow scanning FireWire devices.
- 0x00800000 (bit 23) — `OC_SCAN_ALLOW_DEVICE_SDCARD`, allow scanning card reader devices.

Note: Given the above description, 0xF0103 value is expected to allow scanning of SATA, SAS, SCSI, and NVMe devices with APFS file system, and prevent scanning of any devices with HFS or FAT32 file systems in addition to not scanning APFS file systems on USB, CD, USB, and FireWire drives. The combination reads as:

- `OC_SCAN_FILE_SYSTEM_LOCK`
- `OC_SCAN_DEVICE_LOCK`
- `OC_SCAN_ALLOW_FS_APFS`
- `OC_SCAN_ALLOW_DEVICE_SATA`
- `OC_SCAN_ALLOW_DEVICE_SASEX`
- `OC_SCAN_ALLOW_DEVICE_SCSI`
- `OC_SCAN_ALLOW_DEVICE_NVME`

7.6 Tools Properties

1. Comment

Type: plist string

Default value: Empty string

Description: Arbitrary ASCII string used to provide human readable reference for the entry. It is implementation defined whether this value is used.

2. Name

Type: plist string

Default value: Empty string

Description: Human readable tool name displayed in boot picker.