Khaki clover

Author Slice

Khaki clover. Version 5.0 revision 5120-GitHub
Moscow, 2020

Page 2

Table of contents

Khaki clover. Version 5.0, revision 5120 3
Moscow, 2020

Page 4

Page 5

Page 6

Page 7

Page 8

Foreword

What is it about? Yes, of course not about a flower growing in a meadow to the delight of cows. We are talking about software, a new type of bootloader that allows an ordinary computer run an unusual operating system - Mac OSX. Apple of this does not allow, first of all, motivating that “we cannot provide performance on non-Apple computers ”. Well, let's put system at your own peril and risk. Well, you shouldn't get any commercial benefit from this, in order to avoid other legal complications. Non-Apple computer with the installed Mac OSX is called Hackintosh, the origin of the word is clear.

To load MacOS on Hackintosh, you need a special bootloader, there are many different ones, but basically it can be divided into two classes: FakeEFI and RealEFI.

FakeEFI was invented by David Elliot many years ago, and operates on a simple principle: let's do the view that EFI has already worked for us, let's leave in our memory traces of its activities (boot-args and the entire tree of tables), leave in memory EfiRuntime in a simplified form "Not supported", and let's start the core mach_kernel. This is how the Chameleon works, and it works successfully, but for small with exceptions like the Boot Disk panel. It is possible that over time, Apple will give us and other problems associated with the lack of Runtime Services. January 2013: this It happened! iMessage stopped working because it definitely needs SetVariable (), which in Chameleon is "Not Supported". They somehow overcame, but the Chameleon again Problems. Legacy bootloader options: Chameleon, enoch, Chimera, PC-EFI, revoboot.

RealEFI should have been flashed instead of BIOS, but for those who have a motherboard on BIOS based bootable EFI coined. This EFI boot system on a BIOS machine invented by Intel, an open source project at tianocore.org. (Original the project has sunk into oblivion! It survived only in the Clover project). This bootloader itself called DUET. But the trouble is, it loads EFI, but it loads the operating system Mac OSX is not available there. The next step is required, adapt DUET to Mac OSX requirements. New motherboards already have EFI, but also unusable to download Hackintosh. EFI boot options fall into two categories: for PC BIOS - bareboot, XPC, and for UEFI BIOS - Ozmosis, OpenCore. Clover serves both categories.

This loader got the name Clover from one of the project founders kabyl, who saw the similarity of the “Command” key, which exists only on Macs, with four-leaf clover.

> ***Four-leaf clover*** *- single plantclover , has at at least one four-plate sheet , unlike conventional three-plate. In the Western tradition, there is a belief that it is the plant brings luck to the finder, especially if it is found by accident.[1]... According to legend, each of the plates of a four-plate sheet represents something . is hope , the second is faith , the third islove , and the fourth -*
>
> *By the way, the original green logo looks more like a rabbit cabbage than a clover. Namely Oxalis, and it is in flower shops both three-leafed and four-leafed :)*

In Russian we call the bootloader "Clover".

The project is being developed on the forums http://www.projectosx.com/forum/index.php?showtopic=2562&st=0 RIP

https://applelife.ru/threads/clover.42089/
1 I wonder how you can find clover by chance? Regularly pinching the grass in the meadow ?!

Page 10

http://sourceforge.net/projects/cloverefiboot/
https://www.insanelymac.com/forum/327-clover/
https://github.com/CloverHackyColor/CloverBootloader

According to the text of the book:
What you need to pay attention to is highlighted in red.
In green, what is deprecated and no longer supported is saved in the text for
help.
Headings are highlighted in blue.
Key words are highlighted in black.

Development chronology

The need for a new bootloader arose from Chameleon's inability to boot
the then system 10.7 (Lion).

The project started on March 4, 2011 at the initiative of Kabyl, who, however, having told everything,
which he managed to understand by that time, he avoided development, and soon disappeared altogether. I have
there are serious suspicions that he is no longer in this world.

The first launch of the MacOSX system with a modified DUET took place on April 6
2011 year. http://www.projectosx.com/forum/index.php?
showtopic = 2008 & view = findpost & p = 13810

On May 4, serious problems of the new bootloader were formulated, without their solution in the new
the project made no sense. The pause dragged on until August, for alone to cope with these
problems seemed unreal to me.

In the meantime, the Chameleon came to life, having coped with the Lyon load, and I for a while
worked on my brunch. However, the Chameleon admins ignored me, so I am
threw. Then Ninja showed up with his iBoot, and I joined him in trying to do
EFI-bootloader and fix hanging problems. This project started in August 2011, and along the way
I was tweaking DUET (CloverEFI) using CloverEFI + iBoot sum. However, dirty
the origin of this aybut did not allow to develop properly.

August 09, 2011 with the participation of dmdimon a Russian font for the loader was made. I am the
time I am doing SMBIOS and ACPI at a much higher level than it was in
Chameleon.

On October 19, 2011, the problem of launching Duet on a laptop is finally solved. Before that was
just a reboot.

November 14, 2011 the appearance of cats in the Clover interface. That is, for 10.4 we draw a Tiger,
for 10.5 Leopard and so on. Nice innovation! Appearance plays a role.

December 14, 2011 memory panic problem on low-end OSX systems, on Lyon resolved
and older, for some reason there was no such problem.

On January 05, 2012 the sleep problem was solved. It was from this moment that the project could be considered
viable. By this time Ninja had already left the stage, and I decided to start
own project of the graphical menu of the bootloader based on the already known rEFIt. is he
licensed clean, and now it was possible to fight for the international recognition of the project.
This is how Clover-v2 was born.

It took two months to create a new shell, and the first publication took place on February 29 2012. Actually rEFIt already existed, it just was not suitable for compilation in the environment

Page 11

EDK2, and all its libraries had to be rewritten and replaced with our own. Come on and add everything hackintosh-specific to the project. Along the way, together with jadran, tools were developed compiling the project. Now with gcc-4.4, and now 64 bits.

March 09, 2012 Dmazar, whom I have known since August 2011, joined the project with his the idea to make a Clover-based UEFI bootloader.

March 31, 2012 Gyk made interactivity for entering parameters in the bootloader shell - Options Menu.

April 12, 2012 crazybirdy made the Clover installer.

April 21, 2012 Dmazar defeated UEFI download, but continued to work on the project - improve and fix. That is, OsxAptioFixDrv was created.

On June 05, 2012 pcj appeared and offered its sources with new technologies: DSDT patch, Kexts Inject, Kernel patch, which raised our bootloader to a completely new one a level unattainable by competitors.

18 September 2012. Pené called Dmazar and me to a round table to think about iCloud problems. Resolved by September 21st.

September 30, 2012 the appearance of the mouse in the bootloader interface.

October 19, 2012 animation in the bootloader shell is made.

October - December 2012 step by step made native resolution in the bootloader for Nvidia, ATI, Intel, for CloverEFI and for UEFI.
These advances have raised the bar on what a good GUI should be.
bootloader. I would also like to acknowledge Blackosx's contribution to improving theme support.

On January 09, 2013 the problem of paid iMessage was solved, which could not be done in Chameleon. It was a fundamental victory for the very idea of an EFI bootloader. Nothing is impossible, and in The chameleon repeated this method a month later, but users have now gone to Clover.

Spring 2013. Through the efforts of JrCs, Clover has acquired additional utilities and internationalization. 20 languages in installer, control panel and service automatic updates. Scripts significantly increased in functionality compilation, start and end. The bootloader has become a hackintosh service complex systems.

On July 27, 2013, the UEFI boot sleep problem, which has been hanging since autumn 2012, is finally resolved. It seems with the OS update it appeared.

September 29, 2013 also corrected sleep, shutdown and restart during UEFI boot. FROM at this point, you can put UEFI boot as the main method on those computers where it is possible.

On January 20, 2014, made the possibility of deep sleep - Hibernation. Not in all cases, but Clover is currently the only bootloader that can.

February 2014. Project of the Month award at sorceforge.net.

05 April 2014 Completion of development on revision 2652 is announced. This, of course, does not cancel possible improvements in the future, just nothing will change dramatically the main thing has already been done.

June 2014. Apple Launches 10.10DP1 Yosemite System, First Success Begins

installations and new fixes in Clover for the new system. And then it turned out that
The chameleon is unable to boot this system. There are single successful reports from

Page 12

bareBoot and Ozmosis bootloaders, which are also EFI bootloaders using
part of the Clover codes. Clover became the main Hackintosh downloader. Chameleon again
corrected, he loads Yoshu, but so far without iMessage.

August 21, 2014. Dmazar fixed NVRAM performance on UEFI boot for
some who have not worked before.

In the meantime, version 2k has stepped over revision 3000. New compiler with
LTO optimization, new code size, fixes for old bugs, improvements in
algorithms. Almost nothing new, but Clover just got better.

January 2015. There is still no Clover-3 announced by Apianti, Clover-2
acquired version 2.3k, which reflects revision> 3000.

All developers fled, projectosx.com ceased to exist. I stayed
alone, but I continue to work.

June 2015. The release of the 10.11 El Capitan system. Clover uploaded it without question.
The chameleon is stuck again. We managed, but, as I see, ordinary users are interested in the Chameleon
no longer show. Interest is shown only by all the same 6 people who
support. Chameleon / Chimera is still used by newbies who started life with
Multibeast, and have not yet heard of Clover, or AMD CPU users who are not
developers and they just don't know what to do with Clover. One new problem, injection
kekstov stopped working. The solution was provided by a user who accidentally looked at the light
solstice (revision 3258). Ozmosis users were left without a new system for a long time, before
than its developers provided a new version with this patch.

Spring 2016. Zenith432 made compilation with Xcode, we no longer need
gcc.

Summer 2016. With new coders, new major changes in the Clover interface. Scrolling
wherever needed, checkboxes and radio buttons, new design styles.

Fall 2016. Together with vit9696, the long-awaited support for FileVault2 technology has been made.

Winter 2017. Revision 3999 came out. Then there will be 4000 and version 2.4k. Clover po
still needs development, because support for new systems is needed, for Sierra
10.12.4, for example, needed a new patch.

Summer 2017. The High Sierra system has appeared with the new apfs file system. For
There is no chameleon driver yet. For EFI loaders, there is a native one from Apple. By the way multibeast
long ago switched to Clover, like other commercial projects.

Autumn 2017. Victory over the black screen Radeonov. This was done by vit9696 in its kekst
WhateverGreen, Mieze told how it works, and I added as a tick in the config
Clover's RadeonDeInit = true. The solution is incomplete, in the sense of a graphics factory, but enough
good for sleep Radeonov.
And again vit9696 found a solution to how to fix AptioFix so that the native works
NVRAM, now almost everyone!

Spring 2018. vit9696 figured out how RTC is used in macOS, and thus solved
the problem of saving the hibernation key, and other RTC problems.

June-December 2018. SVG vector graphics support, and, accordingly,
scalable themes.

January 2019. The appearance of sound in the Clover interface, thanks to Goldfish64.

Khaki clover. Version 5.0, revision 5120
Moscow, 2020

Page 13

July 2019. A series of new patches for download 10.15. And revision 5000! Now Clover got version 2.5k.

May 2019. vit9696 created a new OpenCore bootloader based on new ideas with its own ideology, he is more promising, but Clover continues to live and develop. We will coexist and cooperate. Clover would be worth cleaning up from unnecessary, from crooked, to take the best of OpenCore, but this is unrealistically large work, we are improving slowly.

I also want to note that Vector-sigma joined the Clover team, which took over the entire user-space. Controls compilation scripts, execution scripts, installer, and created special application Clover.app. That is, it controls everything that was previously under ward of the JrCs. So the bootloader binding is also alive and well.

September 2019. We carried out a major transformation of the entire project, moved to GitHub, which, after purchasing them by Microsoft, became a stable and fast site, and along with this, the following changes have occurred:

- now Clover's sources are controlled not by the SVN protocol, but by the GIT protocol. In that there are downsides, namely the problem with revision numbering, but there are also pluses, for example the ability to bisect to find a recent mistake.

- tired of lagging behind strange "improvements" of EDK2. Before that, we used a stable version UDK2018, but it is clearly outdated, and with the new EDK2 Clover was just not compatible. The solution is to include all the required libraries from the new EDK2 just to the Clover project already with our patches, and add those libraries that are already excluded from it, taking them from the old UDK2018. It's about supporting legacy computers on Core2Duo and the like. Now Clover does not need external libraries, all the necessary is contained in its repository, and all this will be updated by the developers Clover himself, based on changes in EDK2. By the way, bisection has become possible precisely with the rejection of any external sources, everything you need is contained in one folder.

- now the main compiler of the project is gcc-9.2 (already gcc-10), in contrast to the clang it is capable of performing LinkTimeOptimisation, that is, discarding unused libraries at the linking stage. The size of codes has decreased from 960kb to 870kb! Moreover, we we get additional diagnostics of possible errors. One of them is that EDK2 allows the use of pointers to an empty array, and this was contrary to some Clover's algorithms. After a small amendment, the design started playing differently! And other bugs could go away.

- changed the structure of the drivers folder, since we completely abandoned the idea of compilation in 32 bits, and specifying 64 bits is redundant.

- changed the structure of the kexts folders, now you can connect and disconnect in the interface different versions of kexts to check which one works better.

January 2020. Clover's next major makeover. There is a new developer Jief Machak who suggested translating the project to C ++. This is non-trivial action because it requires changing some libraries and compilation methods. But he did not come empty-handed, he brought the necessary files dated 1997, obviously already then solved such a problem with C ++. Now it adapts to new conditions.

I accepted the challenge and now we are in the team began transformations in Clover. Also Pene us helps. Of course, C ++ means compiling files written in C, so

Page 14

at the first stage of the transition to C ++, nothing happened. But there is no point in talking about new language, and program in the old one. To take advantage of the benefits of C ++, we must use them. Let's leave Linus Torvalds alone with his curses against C ++, its motives are clear, Linux is already written in C, and it cannot be altered, it remains just look for an excuse for this. And we can rewrite, at least partially, still Clover much smaller in size, and not everything can be rewritten, keeping in mind compatibility language. And yet, what are we doing? Introducing new programming paradigms and we rewrite codes for new requirements. And this elementary leads to his rethinking, development of new algorithms, and elimination of old bugs, untangling pasta. GUI written on new algorithms, on classes, and therefore old topics have suffered, but they can cure, and in Clover everything is still not fixed, work is underway. And the result of the transformations this is faster work and the elimination of bugs that previously could not unravel.

Summer 2020. The appearance of the 11.0 Big Sur system showed that the myth of non-flying character patches are invalid. Flew, both in Clover and in OpenCore. Well, we work further, the loader is alive while it develops.

Tactical and technical characteristics

EFI - Extensible Firmware Interface - extensible interface for accessing hardware dependent functions. Unlike BIOS, which takes 64kb and is written in 16-bit codes, EFI takes from 4MB, is written in 32 or 64-bit codes, and is positioned as hardware-independent, although ... of course, miracles do not happen, and 100% compatibility with any platform is impossible to achieve. The real UEFI BIOS is, of course, hardware-dependent.

Clover is an EFI bootloader for operating systems for computers that already have UEFI BIOS (Unified EFI ...), and for computers without one. At the same time operating systems can support EFI booting (OSX, Windows 7-64EFI, Linux), or not (Windows XP), in the latter case legacy-boot is provided - return to the old BIOS boot scheme through rubble sectors. Strictly speaking, Clover is not a bootloader, but rather Boot Manager, which prepares and runs native boot loaders for different OS (boot.efi, grub.efi, bootmgfw.efi).

EFI is not only the initial stage of OS boot, it also creates tables and services, which are available for use in the OS, and its performance depends on correctness this stage. You can't boot OSX on embedded UEFI, just like you can't boot OSX from pure Duet. CloverEFI and CloverGUI do a lot of tweaking embedded tables for OSX launch capability:

- SMBIOS table (DMI) is filled with data that emulates real computers Apple Macintosh - OSX startup condition. Serial numbers are fictitious, but suitable. However, it is desirable that the user substitutes unique numbers. Why doesn't GRUB load HakOS, for example? Because they have no right include the serial numbers of the poppies ...
- ACPI tables contained in the computer's ROM usually contain errors and disadvantages, most often due to the laziness of manufacturers: in the APIC table, the number of CPU cores, no NMI data, no Reset register in the FACP table, wrong power profile, data for EIST is missing in SSDT tables, and about DSDT in general a long conversation. Clover tries to fix it all;
- OSX also seeks to obtain information about additional devices from the bootloader, such as video card, network, sound, etc. through the Device Properties string mechanism.

Clover generates such information;

Page 15

- BIOS-based computers tend to use USB initially
  loading in Legacy mode, which becomes unacceptable when transferring control to
  OS. The bootloader switches the USB operating mode;
- OSX also exchanges information with EFI via special NVRAM,
  which is accessed through RuntimeSevices not present in legacy
  bootloaders. Clover provides such an exchange of information, and two-way,
  which makes Firewire work properly, the ability to download the Nvidia Web driver,
  the ability to use the Boot Disk control panel to
  automatic reboot to another system. NVRAM is needed for the possibility
  registration in iCloud and iMessage services, for hibernation, for FileVault2 and
  others;
- the ConsoleControl protocol is required, which is absent in Duet, other similar
  little things;
- it is necessary to fill in some data in the EFI / Platform via the DataHub protocol,
  absent in Duet, and not always present in UEFI. Most serious
  the FSBFrequency value, the definition of which is the task of the bootloader, given in
  DMI is inaccurate or absent at all;
- before starting work, the CPU must be correctly initialized, but,
  since motherboards are made universal, for a whole range of different
  CPU, there is no processor data in the internal tables, or
  some universal, incorrect in a particular case. Clover implements
  complete detection of the installed CPU and makes the necessary corrections in the tables, and in
  the CPU itself. As one of the results - Turbo mode is turned on;
- BIOS somehow uses computer memory, breaks it into regions of a certain
  assignment, but often it is not done the way you need to start macOS;
  The driver is engaged in correcting the memory card, and other related trifles
  OsxAptioFixDrv, and its descendants, which, for a number of reasons, was isolated from Clover in
  separate module;
- one more little thing. The sources of DUET, and indeed the whole EDK2, are written universal under
  different iron ...., but the dependence on iron is made through constants. I.e
  it is assumed that the user compiles Duet for one specific
  configuration. Clover's goal is to be universal, with platform autodetect.

All this is done automatically, and a beginner can even use Clover
not understanding anything about these problems. Well, for an advanced user Clover
provides the ability to manually change many parameters. Their consideration
and is the purpose of this book. By the way, some begin their acquaintance with
Hackintosh from studying a config. This is the wrong approach. You will fix the config
after trying to download something. More on that later.

What is what?

In a nutshell, Clover performs four main functions:

1. It intercepts control so that the BIOS does not load the operating system, so that it
   was done through Clover. BIOS calls Clover, and he loads the operating system, by the way also
   and asks the user which of several operating systems to load.
2. Modifies the data that the BIOS transfers to the operating system, this is the main

requirement for MacOS download. For example the serial number.
3. Modifies the MacOS system itself to work on this hardware.

Page 16

4. In some cases, Clover corrects the state of the devices, which at one time was not made a BIOS. Examples: ResetHDA, RadeonDeInit, HaltEnable.

Windows and Linux do not need point 3, and the second is rare. But on the first point clarification is needed, since newbies start with the question "Why not boot".
When you turn on or when you restart the computer, the operating system boots from Clover's help takes the following path.
Option A . BIOS based computer (legacy boot)
BIOS → MBR → PBR → boot → CLOVERX64.efi → OSloader (boot.efi in case of MacOSX, bootmgfw.efi for Windows, grub.efi for Linux).
Unclear? In words:
1. When you turn on the computer, the BIOS starts. According to his settings, he chooses from which disk start if legacy boot.
2. BIOS reads the zero sector from this disk. The sector is called MBR. Codes are written into memory, and the BIOS transfers control to them.
3. The program in the MBR looks for a boot partition on this disk, reads from there the first sector in memory, and transfers control to the codes. This is called the PBR sector.
4. The PBR program looks for a file called boot in its file system, loads it into memory, and transfers control to it. In our case, in the boot contains DUET, or CloverEFI.
5. CloverEFI looks for the CloverX64.efi file, loads it, and transfers control to it. Further, as in option B.
Option B . Computer based on UEFI BIOS (new circuit, UEFI boot)
UEFI_BIOS → CLOVERX64.efi → OSloader (or BootX64.efi)
1-5. If the BIOS is UEFI, then it itself is able to find the starting EFI module in the EFI section. By By default, such a file should be named EFI / BOOT / BOOTX64.EFI. In some BIOSes the name EFI / microsoft / boot / bootmgfw.efi is stitched. And we can teach BIOS to load immediately EFI / CLOVER / CLOVERX64.EFI file. Only this must still be learned by yourself. For those, who does not know how, the file EFI / BOOT / BOOTX64.EFI is also replaced by Clover.
6. Finally we loaded CLOVERX64.EFI, we see the choice of operating systems, and we can download some of them.

For all this to be carried out, the following files must be registered in the following locations:

MBR sector

the zero block of external media from which the boot occurs (HDD, SSD, USB Stick, USB HDD, DVD). The first 440 bytes should be written to this block, one of the options:
boot0af (Active First) - its role in finding the active partition in the MBR partitioning of the disk, and transfer management to its PBR sector. A hybrid MBR / GPT partition scheme is possible. If the markup is pure GPT, that is, there are no active partitions, then the transfer occurs control on the EFI partition, which is recognized by GUID = C12A7328-F81F-11D2-BA4B-00A0C93EC93B. If you suddenly reformat this section, check which one it has now GUID. There is an option when the first partition is as expected FAT32, but the boot is not goes because its signature is 0C00, and it should be EF00. Corrected by the gdisk utility.
boot0ss (Scan Signature) - search for the first partition with signature 0xAF, i.e. HFS + partition with installed OSX, and transfer control to its PBR. Thus, you can load a system with an HFS + partition on a GPT partitioned disk, but only from the first such partition. If a such a partition is not found, then the search continues for FAT32, or ExFAT partitions, in this order

checks the signature, and transfers control to it.

Page 17

boot0ab - search for a partition with the signature 0xAB - Apple Boot Partition. That is, Recovery. For perverts. In reality, Recovery has never served for legacy downloads.
boot0md is a combo option that looks for an HFS + partition across multiple disks rather than only on the main point. Strange option, not included in any installation options.

PBR sector

the first blocks of each section on the selected medium. Loader phase is written here two. This loader knows the file system of its partition, and is able to find a file there with named boot to load it and transfer control to it. Respectively, there are options for different file systems.
boot1h2 - for HFS + file system and support for boot file length up to 472kb. Old boot1h variant distributed with the Chameleon bootloader only supports 440kb file (472 needed). **For those who first installed the Chameleon, I remind you once again: the sector PBR must be updated with the boot1h file from the Clover package, otherwise it will not will start.** The boot1h2 option has a 2 second pause to switch the bootloader. IN The chameleon seems to have finally fixed that ....
boot1h - also, but without a pause.
boot1f32alt - for the FAT32 file system. This file system has write support, therefore it is very convenient to install a bootloader on it. You can use an EFI partition, you can use the flash drive as it is, since flash drives are already on sale formatted in FAT32[2] . Has a pause of 2 seconds.
boot1f32 - no pause option.
Attention! The file system must be exactly FAT32, not MSDOS, for FAT16 unacceptable. The bootloader does not work on it!
boot1x - Zenith432 made a boot sector which is for the exFAT file system. Unfortunately, the driver inside Clover is only ReadOnly, however, this file system very interesting for external media, because it supports files larger than 4GB, and at the same time supported by recording by both Mac and Windows.
These sectors (more precisely, phase 2 loaders) have another useful function. They have an initial start delay of two seconds while waiting for keyboard input. The entered digit will be appended to the file name, i.e. pressing key 1 on black screen at the very beginning of the boot (after the BIOS message "booting from ...") we load the boot1 file, by pressing key 3 we will load the file boot3, by pressing key 6 we will load the file boot6. Meaning in keeping in one place different versions of loaders, or even different bootloaders by simply giving them different numbers. For instance:
boot - Clover, current version, or test version
boot1 - Chameleon
boot3 - Clover-32bit, tested, working version (displays number 3)
boot4 - Clover-32bit, with BiosBlockIO driver
boot5 - Clover-64bit, shortened so as not to overlap EBDA (see below)
boot6 - Clover-64bit, tested, working version (displays the number 6)
boot7 - Clover-64bit with BiosBlockIO driver, working with any controllers, which are supported by BIOS. (displays the letter B or L or 7)
In addition to these options, the PBR sector may contain a Windows boot manager that knows file system NTFS; GRUB knowing the EXT4 filesystem; and others without relationship to Clover. At least at this stage.

2: for historical reasons, it is recommended to reformat each flash drive, and even be sure to

Windows system. Today this recommendation is outdated and has no basis. Factory format
suitable for Clover installation. Although ... it's worth checking that out.

Boot or CloverEFI

In the case of the Chameleon, the "boot" file is the entire bootloader. In Clover's case, this
the file contains the entire EFI system, as well as the boot service for transferring control to the next
stage. All this, in option B, should be considered already present in the computer's ROM.
In reality, it turns out that not everything is there, and some details (so-called drivers) should be loaded
additionally. The details already collected in the boot file for option A.
Unlike the previous stages, the boot file already differs in bitness, i.e. separate
options for 32 and 64 bit download. In most cases, you should choose 64bit
option if the processor supports this instruction set (attention! Pentium 4 and Yonah
- only 32 bits).
However, if for some reason it is supposed to work only in a 32-bit OS, then
it makes sense to load EFI32. It is smaller in size by 20%, and accordingly
faster. Unfortunately, not compatible with Windows 7 EFI, which is always 64bit. **November 2016** :
goodbye Clover 32! I am no longer interested in supporting him.

At its core, the boot file is a modified DUET. Moreover
corrections in essence are unlikely to reach 1%, but this percentage provides
a fundamental difference from Duet - Clover works for the purpose for which it is intended.
If anyone thinks that I have been doing Nny for three years, editing DUET, and that it is enough
take vanilla, and add AppleSim to it, then bon voyage! Far from it, but
I have no purpose to describe all the subtleties of this development. It's already done. Further
we call this program collectively CloverEFI. With each revision, its binaries
differ, because EDK2 changes, and sometimes its sources, for example, the latest changes
were like StrCpy => StrCpyS, kind of a safer version. However, the functionality
CloverEFI hasn't changed for a very long time, so if you have a problem with a new
version, put the version that worked for you. So included that
was officially distributed on SF.NET, the old boot7-MCP79 file was included, because for some
the reason the new boot7 stopped working with this chipset.

Another minor problem. PBR is looking for boot. But the Windows installer has a folder with this
name, in the same place, at the root. You have to rename your boot file to boot5, and
use an alternative download. Somewhere I made a version of boot1f32, which is
By default it loads boot5 at once, fix in one byte. But this version is not present in
standard installer.

CLOVERIA32.e2 and CLOVERX64.e2 or CloverGUI

This file, in two versions of different bitness, is a graphical shell
boot loader for choosing an operating system, for installing additional options, for
loading additional drivers and actually for starting the OS. Graphics and menus
are originally based on the rEFIt [3] project , which is reflected in the directory name and in the About menu.
At the moment, the original part is hardly 5% of the entire program, and even then
in corrected form. Currently, both graphics and menus are rewritten to C ++ classes, so
that there are memories of the original codes.
In what follows, we will collectively call this program CloverGUI.

Folder structure

In addition, the loader needs additional files, the folder structure will be approximately
next.

EFI:

```
BOOT:
    BOOTX64.efi
```

3 People often ask "Why not rEFInd?" I answer "Because he appeared later than Clover"

Page 19

```
CLOVER:
    ACPI:
        WINDOWS:
            SLIC.aml
        origin:
        patched:
            DSDT.aml
            SSDT-1.aml
    CLOVERX64.efi
    themes:
        black_green:
            BoG_LucidaConsole_10W_NA.png
            icons:
                func_about.png
                os_clover.icns
            banner.png
            background.png
            Selection_big.png
            theme.plist
        Clovy:
        cesium:
            theme.svg
    config.plist
    drivers:
        BIOS:
            FSInject.efi
            SMCHelper.efi
        UEFI:
            CsmVideoDxe.efi
            DataHubDxe.efi
            FSInject.efi
            VBoxHFS.efi
            OsxAptioFix3Drv.efi
            SMCHelper.efi
    kexts:
        10.7:
        10.11:
        Off:
        Other:
    misc:
    OEM:
         Inspiron 1525:
        ACPI:
            origin:
            patched:
                DSDT.aml
        config.plist
        kexts:
            10.5:
            10.6:
                Injector.kext
                VoodooSDHC.kext
            10.7:
                VoodooTSC.kext
            Other:
        UEFI:
            ACPI:
```

origin:
patched:
DSDT.aml

Page 20

config.plist
kexts:
ROM:
tools:
Shell64U.efi

That is, the CLOVERX64.efi file must be located at / EFI / CLOVER /, and the font BoG_LucidaConsole_10W_NA.png in the / EFI / CLOVER / themes / black_green / folder. Really these, but also other folders are more full of content. In the course of the story we will describe in more detail what it serves for what.

A few words about the / EFI / CLOVER / OEM / folder

The folder is designed to store download options for different configurations. Typical the situation when we create a bootable USB flash drive, and on it, in addition to the general config /EFI/CLOVER/config.plist, there are also well-calibrated / EFI / CLOVER / OEM / Inspiron 1525 / config.plist and /EFI/CLOVER/OEM/H61M-S1/UEFI/config.plist, as well as their worked out DSDT.aml, different for different computers, and different sets of kexts.

The folder name is determined from SMBIOS and you can look at the boot.log exactly how your computer is called:

0: 100 0: 000 Running on: 'Z170X-UD5 TH' with board 'Z170X-UD5 TH-CF'

The first line contains the name of the entire system, typical for laptops, but on desktops there is something abstract. The second line contains the model of the motherboard, convenient for desktops, but not for laptops. Both names are suitable for the name of your folder, choose more understandable.

Also, your folder may contain a UEFI folder in order to have different configs for UEFI (option B) and for legacy boot (option A) on the same computer (although I personally I doubt that someone needs it).

EFI drivers

I will separately mention the drivers32, drivers64, drivers64UEFI folders, respectively for 32, for 64-bit boot option A - BIOS boot, and for UEFI boot option B. Composition these folders will be different for different BIOS revisions, as well as for different configurations sections. **Attention! The folder structure in revision 5000 has changed, now it is one drivers folder, with BIOS and UEFI subfolders. 32Bit is gone forever.**

It should be noted that these drivers are valid only for the duration of the bootloader operation. On the they do not affect the loaded operating system, except indirectly, by how devices will be initialized (fourth Clover function).

What should be put in these folders? User's choice.

- **NTFS.efi** - NTFS file system driver, for the ability to load Windows EFI, however ... it seems that it is not really needed, because the EFI Windows loader is also in ESP, to FAT32.
- **HFSPlus.efi** - HFS + file system driver, required to run MacOSX. It is required for option B, but in A it is already present in the boot.
- **VboxHFS.efi** - legal alternative for HFSPlus.efi, slightly less speed. The new version supports links, and even more than the native one Apple's HFSPlus.efi. HardLink, SymLink are supported! Whereas HFSPlus.efi only hard links. In the modern version of Clover, the downloaded file displaces built-in. That is, if the boot file has an embedded VboxHFS.efi, and in the folder drivers / BIOS / there is HFSPlus.efi then the last one will work.

- **VBoxExt2.efi** - EXT2 / 3 file system driver, required to run Linux EFI.
  Similar to **VboxExt4.efi.** Again, if Grub is not in the EFI section.

Page 21

- **Fat.efi -** FAT32 file system driver. Of course, he is already there. However,
  AMI UEFI BIOS contains a very crooked driver, so I preferred to load my
  option.
- **FSInject.efi** - a driver that intercepts the file system, for the possibility
  force the injection of some cakes into the system, if it does not calculate it itself
  necessary. Difficult to understand? We will return to this issue later when
  we will consider the ForceKextsToLoad key
- **PartitionDxe.efi** - actually, there is such a driver in CloverEFI, and in UEFI it is, but
  only that one is not designed for either an Apple partition or an MBR / GPT hybrid. Conclusion: in
  Option B sometimes needs a driver. If you have normal GPT, you can get by.
- **OsxFatBinaryDrv.efi** - the required driver for option B, provides launch
  fat (Fat) modules such as boot.efi on systems prior to 10.9. In newer
  systems can do without it.
- **OsxAptioFixDrv.efi** - a special driver designed to correct the memory card,
  which is created by the AptioEFI AMI curve, otherwise the OS cannot start.
- **OsxAptioFix2Drv.efi** - slightly modified version. It was possible with him
  Hibernate in system 10.9.1 at UEFI boot! But unfortunately this option is not
  ships 10.7.5.
- **OsxAptioFix3Drv.efi** - new version corrected by vit9696 to support
  native NVRAM, now you can do without its emulation!
- **AptioMemoryFix.efi** is a completely redesigned driver from this series,
  by vit9696, with optional automatic search for the best value
  slide = xxx. At the moment it is recommended to use it first.
- **OsxLowMemFix.efi** - a simplified version of AptioFix, suitable for some
  strange UEFI BIOS variants (Insyde H2O).
- These five Aptio variants will not be used at the same time, even if all
  are present, Clover has a choice of one of them by priority:
  AptioMemoryFix, OsxAptioFix3Drv.efi, OsxAptioFix2Drv.efi, OsxAptioFixDrv.efi and
  the last is **OsxLowMemFix.efi** .
- For the new revision of Clover 5120, it is recommended to forget all these *** Fix *** drivers , and
  use the amount of **OcQuirks** + **OpenRuntime** , which are now included in the package
  Clover;
- **NvmExpressDxe** - **NvmExpress** controller driver, which is positioned as
  replacing SATA for SSD drives, it should be borne in mind that if the controller is
  motherboard, such a driver is most likely already in the UEFI BIOS. Needed for
  additional controller.
- **Usb * .efi, UHCI.efi, EHCI.efi, XHCI.efi, OHCI.efi** - a set of USB drivers for those
  cases of option B, when the built-in drivers for some reason do not work well. Why would
  all of a sudden? Perhaps there is some kind of tie to other functions that had to
  disable.
- **PS2Mouse…, PS2MouseAbsolute…, UsbMouse…** - a set of drivers to support
  mouse pointer / trackpad / touchpad in the CloverGUI interface. To the operating room
  the system is not affected by these drivers. The UEFI BIOS also has such a driver, you should not
  interfere with him.
- **AptioInputFix** - also for mouse and keyboard, if the option from the UEFI BIOS works
  crooked. By vit9696.
- **DataHubDxe.efi** - this driver is already present in option A, and it is quite possible

is also in UEFI. But in case it isn't there, it's worth downloading the external one. No conflict will arise, but there will be confidence that it is. More and more often I see complaints

Page 22

users that something does not load or does not work for them. And they don't have DataHubDxe! Where are you from so cunning? On tonimake, or what, you were convinced that this driver excess? Then don't come to me either.

- **CsmVideoDxe.efi** - video driver that provides more range screen sizes than the built-in UEFI is needed for option B if the video card does not has UEFI VideoBIOS. There are options when this driver is incompatible with graphics card, such as integrated Intel, be careful. Driver requires enabling the CSM Enable checkbox in the BIOS.
- **GrubNTFS, GrubEXFAT, GrubHFSPLUS, GrubUDF** ... - a set of drivers for the most different filesystems adapted from the GRUB source to work in composition of Clover. It is very gratifying that we now have support for all these file systems, and licensed purely. Thanks to AnV who got hold of and adapted source codes. They have some advantages over our drivers, for example GrubHFS supports compressed volumes, but the status of the entire set is more likely to be "beta". They slow and buggy. And the set of features is not impressive. HFS - no support links, UDF - not all headers read, EXFAT - no record (I would like to!).
- **AppleImageCodec.efi** - PNG and BMP file decoder, needed for FileVault2.
- **AppleKeyAggregator.efi** - creates a special protocol for entering a password into the FileVault2 interface.
- **AppleUITheme.efi** - creates the protocol by which FileVault2 draws the background screen image (in Sierra, for example, these are blurry mountains).
- **AppleKeyFeeder.efi** - required to enter a password in the FileVault2 interface when availability of PS2 keyboard. There is an alternative to **AptioInputFix.efi** from vit9696.
- **FirmwareVolume.efi** - creates the FirmwareVolume protocol, which, however, does not used by nowhere other than FileVault2 which expects to find an image there cursor. Yes. It is there.
- **SMCHelper.efi** - creates the AppleSMCProtocol, which should interact with the SMC chip, and receive information about the state of the peripherals from there. We made an emulator that returns the values that are expected of it. Clover himself and FakeSMC are responsible for filling in real content. saves keys to NVRAM if there is a FakeSMC with this function. All this necessary primarily for FileVault2, but also has an impact on the system and without him. The return of keys to the system is not provided, since there is doubts about the appropriateness of this. There is an external VirtualSMC alternative from vit9696, with its own laws.
- **HashServiceFix.efi** - Creates protocols of the HashService group if they are not in UEFI. FileVault2 interacts with them, as can be seen from the logs. But the visual effect I'm not had seen.
- **APFS.efi** - the driver for the new file system introduced in 10.13. This driver so far does not have an open analogue, so we use what is in the installed system from Apple along the path /usr/standalone/i386/apfs.efi, or →
- **ApfsDriverLoader** - An open source alternative to the APFS driver. Thanks to savvas and others members of the acidantera group. Its essence is to load the correct APFS driver just from the container, thus always having the correct version.
- **AudioDxe** - audio driver developed by Goldfish64. Required for sound when start up the computer. However, this is decoration, if such a sound is not needed, then without driver can live. A bug was noticed that after that in the system itself the sound

disappears, then just uninstall this driver.

Page 23

Loading cakes

In the folder structure, we see the shared folder / EFI / CLOVER / kexts / with folders 10.7 /, 10.11 / and so on, as well as folders named Off / and Others /. Let's consider the question in more detail.
Kexts are "Kernel Extensions" for the MacOSX operating system, for the most part are the drivers of some devices. Or a pseudo driver of a nonexistent devices like FakeSMC. We download them because Apple's clean system just isn't supports our devices, or we want to introduce new functionality. Speech here it is about MacOSX, for Linux and Windows no kexts are loaded.

Starting with revision 3281, all kexts from the Others / folder are loaded first, and then from the folder with a specific operating system number. This makes sense since most kexts are not tied to the system version, and work the same with any of them, FakeSMC in particular. Specific folders may be needed if the driver is loaded only for one system. AppleHDA, for example, each system has its own (I don't know, however, is it loaded with Clover). For me, only FakeSMC makes sense, let the rest will be in the system itself, and even then, only at the time of installation, because in the installer everything cakes are packed in a cache, and it is impossible to slip something of your own there (although ... it seems OpenCore it can). But since people want to load all the cakes with Clover, let there be such an opportunity.
Options:

Firstly, if you write the menu manually (Custom → Entries), then you for each item can be described by InjectKexts = Yes / No / Detect .

Secondly, there is a common installation for all

```
<key> SystemParameters </key>
<dict>
    <key> InjectKexts </key>
    <string> Detect </string>
```

Detect means that kexts are not injected if FakeSMC is detected in the system folder. That is, why inject it again if it already exists? However, if you put a new system as an update to the old one, then Detect cannot be used. There is a Fake in the old system, and in the cache of the new system does not have it, which means that in the Clover config used for installation the value of this key must be <true />.

Another moment. An additional check for unsigned kexts has appeared in the Captain, to overcome it, you must turn off SIP (System Integrity Protection), this is done in config in the CsrActiveConfig key

```
<key> RtVariables </key>
<dict>
    <key> CsrActiveConfig </key>
    <string> 0x3F7 </string>
```

Starting with revision 4233 in the Clover menu, it became possible to block loading separate kexts from the / EFI / CLOVER / kexts / * folders. So you can put there new, questionable kekst, and try to boot. If not, then when you try again launch, you can simply block this kext. And starting from revision 5052 you can put dubious kext in the / EFI / CLOVER / kexts / Off / folder, for example VoodooPS2 from Vasi Pupkina, who promises that his cake is better, so you have. And your kekst is in the folder Other. Having loaded into the Clover menu, click on the Space bar, and there you see a menu on the ban injecting kexts. Kexts in the Off folder are prohibited by default, and in the Other folder are allowed default. For trial you enable this new VoodooPS2, and disable the old one. If a nothing came of it, then when you reboot the old one will work again. If all is well and Vasya

is right, you are already changing manually where which kext should be.

Page 24

Development of

The project has no commercial significance for licensing reasons, and also very large in volume to do it alone, so the most reasonable decision to do it with open source[4] , and have everyone contribute. Until 2019 the project was based on the sf.net server in the repository
http://cloverefiboot.sourceforge.net/
With version 2.5k, we moved to a new repository at
https://github.com/CloverHackyColor/CloverBootloader
there are also compiled versions, ready for use
https://github.com/CloverHackyColor/CloverBootloader/releases

Lyrical digression. In creating a project, and even such a large one, labor is required the following points:

- Collection of documentation, datasheets, specifications, sample programs for the supplied task, and more information about hacker finds. I had a good start dot - Chameleon, in which "everything" works (in quotes, however!). True time goes, and new processors, new video cards, new requirements for newer versions of OSX, and you need to search for descriptions again, or do new tests. The starting point was systematized by Kabyl, bow, first of all, to him.
- When there is a problem statement, input data and what it is desirable to get at the output, it is required to write an algorithm, preferably compact and fast, preferably unmistakable and secure. Programmers love such tasks, and most bows in this project were given to just such assistants. I especially want highlight Dmazar and Gyk, they did really difficult things in the early days becoming a bootloader.
- The time comes for a tedious and difficult, but uncomplicated (?) Job of writing thousands lines of code, where you don't need to think too much, copy-paste with corrections. And here it is, sorry, practically no one helped me all three years. Your project is you and work in. Unless Apianti kissed, special thanks to him. Such rhetorical question: if I adapted other people's sources to my project, how big is my contribution? In any case, we write according to samples ...
- Next comes the work on testing and identifying errors. I have plenty of assistants, you can't even remember all of them. 20,000 posts on applelife.ru alone. Testing can be different, from simple whining that nothing works, to specific instructions on what to change in the code. All this is useful, even nagging, for makes you think about how to make it not there (it doesn't matter if it solved the problem or no, we must stop whining!).
- The worst kind of error, it's a fundamental problem. The first three problems I Solved alone for more than six months: 1. CP in junior systems, 2. Lack of sleep, 3. No start on laptop. The following problems were solved together with Dmazar and Pené: 4. iCloud, 5.iMessage. And the problem is still hanging legacy-boot. I don't see anyone to work, although there are those who want to see the result. 2018: about legacy boot you can forget it, you don't need it.
- Additional things worth mentioning: compilation scripts, installer, system scripts and applications that are directly related to the project, although and are not part of the bootloader. All this huge work practically lies outside

4 Open source projects, however, have some limitations in use, so to speak
"Licensing". I got the GPL license, Intel does not recommend using it. Clover goes BSD.

Page 25

my competence, even though I started. The main contributors here are Jadran, Crasybirdy, JrCs, yes and apianti kissed. This is now the responsibility of vector-sigma.

- Well, the program must be documented. Again, painstaking and not too interesting work. Here special thanks to xsmile for translating this book into English, and in this form she entered WIKI, where already different people from time do their part.

A separate word about Dmazar. This is not just a contributor, this is a second author. Clover in in its current form, this is not my author's work, without Dmazar it would only be legacy loader. Dmazar made UEFI boot. He owns the following technologies:

- OsxAptioFixDrv with all options, otherwise UEFI will not load macOS. Note that the work of this driver is still incomprehensible to any programmer, only the author understood him; Vit9696 came, figured it out, and made a new version of AptioMemoryFix, with NVRAM support.
- DmpUefiCalls, without which we would not have been able to find out what MacOS needs;
- EmuVariableUefi - for those who do not have an iron NVRAM, and there were most. Without this driver on some configurations and boot impossible.
- as well as work with NVRAM inside Clover;
- patches of kexts and kernels. Initial sources from pcj, who threw them up and ran away, and Dmazar brought to mind without even putting his copyright;
- FSInject needed for patching kexts. Again, no one knows how it works anymore;
- OsxFatBinaryDxe, required for systems prior to 10.8, because those systems had boot.efi FatBinary, which UEFI doesn't understand;
- hibernate. We worked together, but it was Dmazar who pointed out how this in general can be achieved;
- and in general helped me to correct semantic errors in my codes. Right, there is safety in numbers.

The rest of the developers and contributors are somehow mentioned in the source code and in installer.

I am writing this chapter with the expectation that there will be more people who want to work on the project, and for it is necessary to learn how to compile at least.

Requisites

In order to compile the project, you also need a compiler, and libraries - an elementary truth. What's in this case? The libraries are the original EDK2 modules. What is the correct name? Framework? Environment? I must say in Russian **ENVIRONMENT** . Downloaded from the same server.http://sourceforge.net/projects/edk2/

Since the whole project was created for Hackintosh and for the sake of Hackintosh, first of all compilation of the project in the MacOSX environment is considered. This, however, is not the only one possibility. EDK2 itself provides compilation also on Windows, Linux, and some other systems and environments. For Windows you need to have Visual Studio 20xx, for MacOSX must have Xcode Command Line Tools installed, and on Linux gcc compiler included by default. MacOSX built-in tools are still not sufficient for compiling the project, so it is proposed, as for Linux, to download and build a new version gcc using the builgcc.sh script in the Clover folder. Why not Clang? Because Klang is still

so far does not issue working codes. We are waiting. Wait! Thanks Zenith432, he finally made a workable compilation with Klang, in connection with which this text is made

Page 26

changes. More changes due to the fact that I propose to use not constantly a changing EDK2, and a stable branch UDK2018. Yes, it's enough. However, the option with EDK2 also exists.

Starting from the modern version 2.5k and this has undergone a radical change, the previous paragraph should be considered obsolete. All the necessary environment libraries now included with Clover, and its repository is complete for compilation, no external links are no longer needed. And also the gcc compiler is again recognized as the best option, now version 10. Nevertheless, EDK2 sources still exist and are being improved, we should sometimes to look at them, to compare with their own.

Writing codes

If you are interested, you can join the team of programmers and offer your a piece. We always meet halfway, although we have certain requirements for codes. Initially, the project was made in the C language, formatted according to special rules Tiano Code Style. https://github.com/tianocore/tianocore.github.io/wiki/Code-Style-C However, with the transition to C ++, we do not want to use so many macros, and we prefer to use standard programming style. You must understand that Tiano's requirement is the ability to compile sources on any platform, with the weakest C compilers. And in general EDK2 is not a working project, it is just a set of sources for building your project. But we are doing a real project and we we set our own rules for powerful compilers and modern hardware.

About C ++.
1. By collecting structures and functions into logically combined classes, we simplify the code, and make it more visual. Moreover, it turns out that many techniques are used repetitively, so this is a direct route to code shortening. Code Reusing.

2. You do not need to declare variables in advance, as required by the C language. The variable must advertised where it is used. First, it is clearer, you do not need to scroll the text back to see the announcement, secondly, the compiler can arrange an additional optimization, thirdly, programmer errors on repeated declarations are reduced, which are not prohibited by the language. An example is a for loop (int i = 0; i <0; ++ i). In old C it was impossible, and therefore forbidden in Tiano, but we'd better declare the loop index like this.

3. Nightmares with Allocate / Free can be eliminated using the capabilities compiler for automatic memory management constructor / destructor.

4. Refuse to use arrays in favor of vectors. Arrays do not control going out of bounds, whereas vectors are easy.

5. Refusing to copy pointers is the biggest source of bugs in a project. But allows you to like something to save. Use links or copy entire structures. It seems expensive, but it pays off with sinless work.

6. Avoid overtyping. Record (INTN) N means logical error. If the variable N is used in a signed expression, that is, which can become less than zero, then why has it been declared unsigned? And similarly with pointer types. C ++ does not encourage this overriding, and we require that overriding be used as little as possible.

7. Use more constant variables and functions, that is, we indicate

to the compiler that their meaning will not change, the compiler and the code accordingly

Page 27

the other will apply. And for the programmer violating the prohibition there will be a warning. it called security, and it's supported by the compiler.

8. We do not have an STL library, but we create our own classes, and translate all the sources to their using.

9. Tiano Code Style implies the use of CONST, VOID, STATIC written in capital letters. What for??? To make the language more like Fortran, or what ?! we leaning towards more standard spellings **const, void, static** , and so on.

Compilation

Now to the point. The reader who has looked into this chapter cannot be simple by definition. a user, and he does not need to tell how to use the terminal.

1. Download the sources.

```
cd ~
mkdir src
git clone https://github.com/CloverHackyColor/CloverBootloader.git
cd CloverBootloader
```

2. Collecting compilation tools

```
./buildgettext.sh
./build_gcc10.sh
./buildnasm.sh
```

However, you don't have to build NASM yourself, but just download the latest release from official site http://www.nasm.us/pub/nasm/releasebuilds/, currently version 2.14.02, and place in the ~ / src / opt / local / bin / folder

If you forgot any of these, then the Clover compilation script will do it for you.

3. We collect Clover, either partially or completely, the script

```
./buildme
```

will ask you what exactly you want. Vector-sigma made a menu there with a choice from the test build before full release.

OBSOLETE!

1. Download the sources and prepare the environment:

```
cd ~
mkdir src
cd src
git clone https://github.com/tianocore/edk2 -b UDK2018 --depth 1 UDK2018
cd UDK2018
svn co svn: //svn.code.sf.net/p/cloverefiboot/code/ Clover
make -C BaseTools / Source / C
cd Clover
```

Or use traditional EDK2. It continues to develop its own way, while Clover uses those functions that have long been made and debugged, therefore it is enough to use UDK2018, but if you want, then you can, with a specific revision, or with the latter, if it is known that it is workable with Clover:

svn co -r 18198 https://svn.code.sf.net/p/edk2/code/trunk/edk2 edk2
svn co https://svn.code.sf.net/p/edk2/code/trunk/edk2 edk2

Page 28

2. Collecting compilation tools
./buildgettext.sh
./buildgcc-4.9.sh
./buildnasm.sh
Now gcc is no longer really needed, Clover compiles perfectly with the built-in clang, and
Xcode should be installed anyway, so the compilation switch -t XCODE8
is now the default key.
NASM needs version 2.13.03 at least, better than the newest one, it seems to already exist in the system,
but the old version, so it needs to be either compiled by the above script, or just
download the finished release here: http://www.nasm.us/pub/nasm/releasebuilds/
The correct version should be here
~ / src / opt / local / bin / nasm
or just / opt / local / bin /
mkdir -p / opt / local / bin /
sudo cp ~ / Downloads / nasm-2.13.03 / nasm / opt / local / bin

To compile with clang, you need the MTOC utility, you can compile it yourself from the latest
source codes,
./buildmtoc.sh
or just take the finished file that comes with the Clover source in the folder
Clover / BuildTools / usr / local / bin / mtoc.NEW.zip
unpack, and put in /usr/local/bin/mtoc.NEW
Now it is no longer supplied, if you still do not have it, build the version with a script, or
borrow the finished version on the Internet. It doesn't need to be updated. Nobody ever
watched one version work differently than the other.

3. We fix the EDK2 environment according to our needs, choosing the appropriate patches
cd ..
./edksetup.sh
cd Clover
./buildgettext.sh
cp -R / Patches_for_UDK2018 / * ../

4. Now you can collect CloverEFI itself. For example like this:

./ebuild.sh
./ebuild.sh -mc --no-usb
./ebuild.sh -ia32
The default is now -x64 and --xcode8, if you want to compile with gcc, then
./ebuild.sh -gcc49
or even -gcc53, or -t XCODE5
Other compilation scripts have been created, as a rule, self-documented. See,
choose, use.
5. One subtlety. The repository does not contain HFSPlus.efi files for 32 and 64 bits, due to their
privacy. There are two options: get these files from other sources, or use
free VboxHfs. This driver is functional, but slightly slower.
It was
foreign file system support
#INF Clover / VBoxFsDxe / VBoxHfs.inf
INF RuleOverride = BINARY Clover / HFSPlus / HFSPlus.inf

Do (now this is done by default)

```
# foreign file system support
```


Page 29

```
INF Clover / VBoxFsDxe / VBoxHfs.inf
#INF RuleOverride = BINARY Clover / HFSPlus / HFSPlus.inf
```

6. Similarly, apfs.efi is missing, but there is no our analogue, you need to take from the system 10.13+ from folders usr / stanadlone / i386 / apfs.efi, or use ApfsDriverLoader.efi
7. The project does not stand still, so this instruction may eventually turn out to be inoperative due to some minor change. This project is for those who think who will be able to figure out what the matter is, and how to be.
8. Well, now we collect the Installer

```
cd ~ / src / edk2 / Clover / CloverPackage /
./makepkg
./makeiso
```

Done! Some steps could be omitted if you are doing for yourself and not in the first time. For example, instead of downloading the entire project, just update svn up, or better ./update.sh (including EDK2 update, but not needed for UDK2018), exclude 32-bit compilation, and do not build the package and disk image.
For dummies prepared fully automated scripts CloverGrower and its Pro version. It's funny. A script to help the teapot compile the package himself. Teapot it is better to download a ready-made installer.
NB! In revisions 3000+, a new compilation method LinkTimeOptimization was introduced, significantly reducing code size. Compiler gcc-4.8 without it, 4.9 with it.
The size without it is 675520, with optimization 633536, and it works the same way, only faster.
This is not necessary for the klang.
Starting with revision 3059, it was possible to reduce the size of the boot file (CloverEFI) by shifting Translation Page Table from address 0x90000 to address 0x88000, File size was 483Kb, became 450Kb, and at the same time it works more securely on some chipsets. Actually it is at Zenith432 had a problem that Clover does not work on his beech, he found an overlap addresses. And since LTO has significantly reduced the size of the codes, it became possible for TPT move. Now Clover works for him too!
Compiling with a key

```
./ebuild.sh --low-ebda
```

Now this switch is the default, to cancel, you need the –std-ebda switch .
It is also interesting for developers to know where the letter "B" or "6" is formed, which displayed on the screen when CloverEFI starts. Looking for an offset with a hex editor 0x02a9, there we see byte 0x42 or 0x36, respectively. By the way, undo this letter it is impossible, since this is a trick for Clover to start at all. BIOS produces initialization of the video adapter at this moment. Can be changed to 0x20 space.

Making a DEBUG version of Clover

When people complain that Clover won't start and therefore can't provide no reports, I can't help but a couple of standard tips. And here is the man who will be able to slightly edit the codes and compile, will be able to reach point X, and then we are already together we will determine what is wrong.
Option 1 . At legacy loading hangs on the message 6_. It means hanging inside Duet itself. Making a log is impossible. Perhaps, perhaps, display messages on screen. The sequence of work Duet is as follows:
1. The starting sector is st32_64.s. It displays this very digit 6, reading

BIOS memory cards, switching A20, and switching the processor from 16 bits to 32 and 64.
My sector variant differs from vanilla one in that it works on a laptop. It is possible

Page 30

that for someone there will be problems here. You need to insert the output of other numbers in other places,
and watch.
2. Start codes C. File efi32.s or efi64.s. There is hardly anything to slow down here.
3. File Efiloader.c. Here you can insert output to the screen using the procedure
PrintHeader ('A'); which is currently commented out.
4. File DxeIpl / DxeInit.c. It is also possible to embed PrintHeader, but still include Debug.c in
compiling this module.
5. DxeCore. Here execution is already spreading, and it is more difficult to track where and what is already more difficult. Output on the screen you can do the same.
6. CLOVERX64.EFI itself is loaded in the BdsBoot.c / BdsLibBootViaBootOption () procedure.
At this point in the program for displaying on the screen, you can already use the standard
the AsciiPrint procedure ("Horror! \ n");
Option 2. The Duet itself works, which can be checked by pressing the space bar immediately after output 6 on
screen. Either we have a UEFI download, and there is no Duet, we should run
CloverGUI, but it is not, or is, but hangs.
The standard method Boot-> Debug = true does not suit you, because you need to track the location
in more detail.
In this case, in the Platform.h file, remove the comment on the 11th line
// # define DEBUG_ALL 2
or in the files of interest, in the upper lines, put DEBUG_xxx 2. In this case, all the output
from the DBG team ("Nightmare # 3 \ n"); will go to the screen. And in this way it will be possible
interactively observe where the execution of the program will reach before it
will hang.
Shl. Don't use Russian letters as I illustrated here! This does not work. FROM
switching to C ++ Russian letters can be used in debug
Option 3. Compile the debug version with breakpoints, run under control
QEMU with gdb special version installed. Dmazar once tried this path. By-
to me, these efforts are not worth the set goal. A simple trace is always sufficient.
Option 4 . When compiling, specify ./ebuild.sh -D DEBUG_ON_SERIAL_PORT
and connect another computer to the serial port that can receive letters on
serial port (terminal in Windows). The variant works on QEMU, if you specify at startup
flag "-serial stdio"

Installation

Using the installer

What is the installer for? To install the program! Why do this
manually, the installer will do everything more accurately than you yourself! The only condition that you have on this computer already has MacOSX. One of the options that you launched the installation
DVD with a different bootloader, and from the MacOSX installation interface launched the installer. IN
depending on the OS language, the installer will work in Russian, in English, or even
Chinese. Here are the instructions for the English version, because in Russian and so
you can figure it out, but I don't know Chinese either. There are 20 languages in the current version, including
Indonesian, maybe anyone needs it.

Page 31

So,
Follow the Continue and OK keys, read and agree to the license agreements
(hmm, are they there?), and we come to a choice, what we install, where and why.

Change Install Location - choose where to install the bootloader. If you intend to put
to the EFI partition, then just select the partition with the current system. MacHDD in this sample. AND
check the "Install Clover in the ESP" checkbox. The installer will find the EFI partition on the same disk.

Page 32

Customize or Standard Install - select download options
If you put the cursor on one of the lines, then in the lower field there will be a short description of this option.
Install Clover for UEFI Boot only - this option cancels the installation of files
boot. In the English version, **Install for UEFI motherboard** is written , and users, of course,
check this box. As a result, we get, as usual, that Clover is not loaded. I am very
I regret that the installer developers came up with this item for their own purposes.
People, for heaven's sake, don't tick this box, even if you have a computer with UEFI BIOS. The UEFI boot you are interested in will still work!
Install Clover in the ESP (Install Clover on an EFI ESP partition) is the best option when
there is such a partition (GPT partition scheme). The installer cannot see this section,
therefore, in the disk selection menu, point to the partition that lies on the same disk, on the ESP
which we want to install the bootloader. Assuming this section has MacOSX,
where scripts, control panel and updater will be installed (in Russian it is too long
"Automatic update program").
Before installation, you should unmount the ESP partition. And for Catalina, maybe
you need to give the command before starting the installer

```
sudo mount -uw /
```

which will allow scripts to write files to system folders.
If the system writes that the installer cannot be opened because its author is
by an unidentified developer, you can go to System Preferences, Security, and
press the button "Open anyway". Or you can run the command in the terminal

```
sudo spctl --master-disable
```

However, it's not my business to teach you the tricks of using macOS.
Bootloader (Bootloader) is a variant with BIOS (variant A), which uses
CloverEFI, or with UEFI (option B).

- **Don't update MBR and PBR sectors** - don't update sectors because they are already

is, or just for option B;

Page 33

- **Install boot0af in MBR** - boot using boot0af, i.e. search for active section. The installer will make the selected section active. Exception is installation to the EFI partition, it is not made active, and boot0af will not find an active partition load the boot file from the EFI partition, which is what we need to implement legacy boot from GPT disk, from ESP partition.
- **Install boot0ss in MBR** - boot using boot0ss, i.e. search for HFS + partition, even if it is inactive. The installer does not change the currently active partition. it made for configuration with an active Windows partition - he needs it.

Use Alternative Booting PBR - as stated in chapter "What is what", the PBR sector can be with a pause of 2 seconds for pressing keys 1-9, or without it. With this option we will set the sector with a pause. This item is no longer in installer;)
CloverEFI is, as you can see from the list, the choice of the bootloader bitness. Either 32 bits or 64 bit. Also here is a special **BiosBlockIO** variant . This is a variant of CloverEFI-64, which has the special name boot7, and is intended for computers with non-standard SATA controller. This driver works through BIOS, and, as a rule, works with any controller (BIOS should work with them!). But there are also misfires eg Dell Inspiron 1525. Another special **boot7-MCP79** . It's like the only working option for the MCP79 chipset, however there is nothing specific there for this chipset, may work in other cases as well. This option was found by Oscar09, also who offered the YUSB patch for this chipset.
Drivers . The choice of drivers is explained above in the "What is what" chapter.
Install RC system scripts to main partition - These are rc.local scripts and rc.shutdown.local, which are executed by OSX at login and logout - required part of the entire Legacy Clover concept. For those with a working NVRAM for UEFI download, these scripts are not needed.
Install scripts on all other sections - if there is more than one on the computer partition with MacOSX. The installer is smart enough not to install them on Windows partitions or Linux.

Install Clover Control Panel - This control panel helps update Clover, choosing a theme and setting NVRAM variables.

Khaki clover. Version 5.0, revision 5120
Moscow, 2020

Page 34

Installing the bootloader manually

It is needed in two cases: when catching fleas and for diarrhea. First, when a person is good knows what he is doing and wants to control every step, not trusting the installer (but in vain!), and, secondly, when installing from under another OS, where the installer cannot be launched.

OSX

It is highly discouraged to do this for someone who does not know what a terminal is.
Installation on HFS + partition in MBR or hybrid partitioning . Why MBR? This is very a standard situation when a computer already exists, and already with information, nothing to lose it is impossible, you can only install a new bootloader.

Installing the MBR sector

```
cd BootSectors
sudo fdisk440 -f boot0 -u -y / dev / rdisk0
```

What's in this team?
fdisk440 is a special version of the fdisk utility, **revised** to only use 440 bytes of the zero sector, there is information that this is necessary for compatibility with Windows (waking up problem) which Apple hasn't taken care of.
boot0 - the file described above in the chapter "What is what"
rdisk0 is the physical device you are going to install the bootloader on. Make sure that it really has number 0.
These files are supplied with Clover. In the new version fdisk440 is excluded, in the first queue due to the inability to compile it in a new environment. Now use dd instead, as shown for Linux. And you can make the section active as standard fdisk. But you don't need this for GPT partitioning.

Setting the PBR sector

```
sudo dd if = boot1h2 of = / dev / rdisk0s9
```

boot1h2 - PBR sector file for HFS + file system, differs from similar support for large boot files, and the ability to select boot1,3,6 by hotkey. Details in the chapter "What is what".
rdisk0s9 is the ninth partition on the selected device ... Why the ninth? And so that fools They did not spoil anything, stupidly repeating the written commands, there is probably no such section. AND you need to put a real number, for example, the first section.
Well, after the MBR and PBR sectors are successfully written to the selected device / selected section, this section should be made active

```
fdisk440 -e / dev / rdisk0
> f 9
> w
> q
```

The nine in the second line is again the section number (there are four of them!) - draw a conclusion.
Now you can copy the boot file and the EFI folder to the root of the partition on this partition.

Installation on a FAT32 partition.
Unlike the previous method, there is one subtlety here. The PBR sector should contain section geometry. This information is entered there during the partitioning process, so the loss of such information is fraught with consequences. The very method of setting the sector gets complicated

```
dd if = / dev / rdisk1s9 count = 1 bs = 512 of = origbs
```

cp boot1f32alt newbs
dd if = origbs of = newbs skip = 3 seek = 3 bs = 1 count = 87 conv = notrunc

Page 35

dd if = newbs of = / dev / rdisk1s9 count = 1 bs = 512

boot1f32alt - already mentioned in the chapter "What is what" - sector for installation on a FAT32 partition. But not FAT16! Be careful!
rdisk1s9 is again the ninth partition on the first device. Plug in your numbers. Rest the letters and numbers in this recipe are not subject to discussion or revision.
The rest of the steps are the same as installing on HFS +.
For owners of hard drives with a sector size of 4k. Attention!
In the first and fourth commands, instead of bs = 512, you should write bs = 4096.
Shl. However ... this is not a fact! The installation with 512 worked fine for me with a 4k disk.

Installation on an exFAT partition
Starting with revision 3040, it became possible to install Clover on a format section exFAT. For example, I have an external HDD with this format, I no longer repartition it I can, but in the old way there is nowhere to install Clover. Installing a PBR sector requires a special boot1-install utility written by Zenith432. The command is simple
./boot1-install -u -y / dev / disk2s1
Basically, Zenith made both HFS + and FAT32 installed by this utility.
It is important that the corresponding sectors (boot1h, boot1f32, boot1x) are in the same folder.
For EXFAT, the sector is named boot1x. Option with pause boot1xalt.

There is, however, one more point when setting these sectors. **The volume where you want to install sector, you must first unmount!** And this is possible only if the disk is not systemic. Otherwise, boot from other media to perform this procedure.

Linux

There is also a terminal under Linux, and almost the same commands, but installation is possible only on FAT32. The differences are as follows.

- instead of rdisk1 there will be sdb - see your version of Linux for more details.
- instead of fdisk440 to write MBR, you need to use the same dd

```
dd if = / dev / sdb count = 1 bs = 512 of = origMBR
cp origMBR newMBR
dd if = boot0 of = newMBR bs = 1 count = 440 conv = notrunc
dd if = newMBR of = / dev / sdb count = 1 bs = 512
```

Windows

From under Windows it also makes sense to install the bootloader only on a FAT32 flash drive, for this just run the script
makeusb.bat E:
where E: is the letter of your flash drive. The presence of several sections on it is not assumed. It's Windows!
All files required for the script are included with Clover. However, the installer must first be unpacked, or these files must be obtained directly from svn. They are located in the edk2 / Clover / CloverPackage / CloverV2 / BootSectors folder
After executing the script, the USB flash drive must be removed and reinserted, then copied to its a boot file and an EFI folder.
Better yet, use BootDiskUtility.exe by Cvad which will help you generate a USB flash drive from under Windows.
http://www.applelife.ru/threads/delaem-zagruzochnuju-clover-fleshku-s-macosx-iz-windows.37189/

Page 36

Recommended installation options

I would personally recommend such options for installation:
1. The main hard drive is divided into GPT (GUID Partition Table). This is a modern circuit, the most correct for Mac OS, and supported by Windows 7 and older. Such a scheme has an invisible ESP (EFI System Partition) 200MB. By default it is already formatted in FAT32 and no need to reformat it, even harmful. Here and you should install Clover, both for legacy and for UEFI download.
For UEFI download, you just need to copy (or install by the installer) folder EFI with all the necessary content. But in order for BIOS to see this boot option need to copy file /EFI/CLOVER/CLOVERX64.EFI to file /EFI/BOOT/BOOTX64.EFI, if not. To whom it is not clear, to do in two steps: copy to the / EFI / BOOT / folder and then rename to BOOTX64.EFI.
To boot legacy, you need to write the boot0af file to the MBR sector of this disk, and to the PBR of this EFI partition write boot1f32 (or boot1f32alt with pause).
Do not make any section active under any circumstances!
Dell's laptop has the American Megatrend EFI BIOS, and it does not allow legacy booting to GPT disk. Alas! For the rest there is such an opportunity, and in the installer we do this: (the screenshot is outdated, but the meaning remains)

2. Winchester is already broken as FDISK Partition Scheme, commonly called ICBM. Old scheme, always used for Windows XP. On this disk, we have allocated a partition where we want install OSX for trial. About UEFI-loading here we are no longer talking, but here's legacy Clover, we may well use.
To do this, write the boot0ss file to the MBR sector. This sector will look for the HFS + partition, which we have made for Mac, and will transfer control to him. The active partition can stay Windows, he needs it more. The file boot1h must be written to the PBR sector of this section from the Clover kit. **I remind you once again that if you have used**

Chameleon, then the same file from the Chameleon kit is registered there, and it will not

Page 37

work with Clover. Must be replaced!
In the installer the following checkboxes

3. You have decided to provide a separate hard drive for your Mac. There is an option just BIOS
choose what to load, and prepare the disk according to the first scenario. Or bet on the main
hard drive to the MBR sector file boot0md. Its property is that it will seek
HFS + partition across all hard drives, in which case it will work as the second option.
Now take another look at the manual installation chapter.
And finally. Isn't it time to completely abandon Windows XP, and install Windows
UEFI, for example seven? MBR disk can be converted to GPT without data loss
the gdisk utility. MacOS will only benefit from this, and Windows one way or another once every six months
reinstalling. If anyone misunderstood, then Windows UEFI is perfectly placed by Clover on
a machine that does not have a UEFI BIOS.

Registration

Theme selection

Now we choose a theme. What is the topic? These are design elements: banner, background
image, drawings of icons and buttons, font, united by a single artistic
by design.

Page 38

There are three themes in the installer, my cesium, a theme from Clovy, these are vector themes, and raster from Blackosx, for some reason he didn't offer his SVG theme to the installer. Besides them there are "Easter eggs", who saw, he knows, for the rest it will be a surprise. And more in Clover a built-in theme from Clovy called embedded. If you don't put any topic, then this one is all the same will be available. And there is an application that will download any theme out of 40 for you online. available on the gita. I will not go into more detail, everything is transparent there.
http://www.insanelymac.com/forum/topic/302674-clover-theme-manager/
In the modern version, you can specify the theme "random" and on every next boot observe different design.
Let's review the topics briefly. In reality, the set of topics is much wider, look at the forums for who what offers.
http://www.applelife.ru/threads/themes-temy-dlja-zagruzchika-clover.36074/
http://www.insanelymac.com/forum/topic/288685-clover-themes/
http://clover-wiki.zetam.org/Theme-database
https://sourceforge.net/p/cloverefiboot/themes/

Now vector-sigma has made its own Clover Theme Manager inside Clover.app
The button looks like a Movie.

Khaki clover. Version 5.0, revision 5120
Moscow, 2020

Page 39

Choosing a theme

Khaki clover. Version 5.0, revision 5120
Moscow, 2020

Page 40

Clover bootloader themes

Metal . By Slice. Topic # 1

Black-green . Author blackosx. The theme that comes with this installer by default.

Page 41

steampunk . Author medik.

And dozens of others ... Creating your own theme is a special story with its own rules.

Configuring the interface in con2g.plist

Themes also include a number of parameters specified in the config.plist file. For old versions see old versions of instructions.

Interface settings are done in EFI / CLOVER / config.plist in the GUI section

<key> GUI </key>
<dict>

The interface can be graphical or textual (starting from revision 1764). For this it is written

<key> TextOnly </key>
<true />

Probably only in Russia there are lovers of the text interface, Total Commander, Volkov Commander, DOS, etc. etc. At your service!

Page 42

If false, then Clover works in graphical mode.
Text screen resolution, here, as well as in Shell and on boot.efi screen, can be set through

<key> ConsoleMode </key>
<string> 0 / Min / Max / some number </string>

The number can be found in your boot.log, or put Max - it will be maximum possible resolution.

The design of the graphical shell depends on the selected theme. Default theme selectable in variable

<key> Theme </key>
<string> metal </string>

However, a theme can also be selected in the Control Panel, and that choice will be defining. If the wrong theme is specified there (there is no such theme.plist file by specified path), the theme from the plist will be selected. If a nonexistent theme, the screen is built-in theme (**embedded is**), the latest version of Clovy. Theme can be changed in the bootloader menu, there will be a list of installed themes (roar 1955), and you can ask which one you need. The interface will be repainted after exiting Main menu. (revision 1936)
random - the theme will be randomly selected from the list of installed ones on each download.
One more point of choosing a topic. If your monitor has a resolution, say 2560x1600, then icons and font on the screen will be too small. You have to choose a big topic. Beginning with revision 4438 Clover will do this automatically. On a small monitor uses a theme let's say metal, but on a large monitor it will try to find the metal @ 2x theme, if any. The criterion for a "large" monitor is the number of lines is more than 1100, because I am 1080 lines quite and a small topic will do. A question for designers when they will make large analogs of their themes. Scalable themes have been made since 4862. Separate chapter.

As for this config item, it looks the same for raster and vector
themes, the only difference is in the name. Clover himself distinguishes one from the other in content.
folders. If there is theme.plist, then it is a raster theme. If theme.svg is there, then it is a vector theme.

Page 43

<key> EmbeddedThemeType </key>
<string> DayTime </string>
The options are Dark and Light. Introduced from revision 4644. Variants were invented before
the invention of the clock, it was possible to choose dark or light built-in themes. And now
by the hour, light during the day, dark at night. Introduced from revision 4773.

<key> Timezone </key>
<integer> 3 </integer>
Clover's modern revision, 4773+, introduces the concept of "time of day." When you start Clover
reads the current time from BIOS, it usually costs us by Greenwich Mean Time, and adds to it
value from this parameter, zone. You can write with a +3 or, say, -5. In total
local time is obtained. Then Clover determines if it is in the interval from 8:00 to
20:00, then it is considered a day, otherwise night. This affects the above built-in theme,
if DayTime is specified it certainly affects any vector theme, it itself should
to determine what it affects, it affects the sound in any topic. Sound from the file plays during the day
sound.wav, at night from the sound_night.wav file. If no night sound is detected, plays
day.

<key> PlayAsync </key>
<false />
Starting with revision 4833, sound appeared in Clover with the AudioDxe.efi driver, author
Goldfish64, who doesn't want to deal with Clover, well, okay, he doesn't mind using it
solutions. PlayAsync determines whether the sound plays in sync with Clover's work, or
asynchronously. That is, during synchronous playback, nothing works until the sound is
will end. With asynchronous, sound flows regardless of what is happening on the screen. I.e,
sound starts after message on screen ... scan entries ... then interface appears
Clover, the sound continues, then the boot.efi lines appear, the sound continues,
then the kernel is launched, and the kexts are launched in turn. Starting with one of them, the sound
breaks off. Well, that is, of course, if the sound.wav file is long.
Clover 4862 had a bug, asynchronous sound hung up the computer. Since 4870 no problem,
you can set PlayAsync = true.

<key> CustomIcons </key>
<false />
If set to <true /> , then for each partition with the operating system there will be
search for an icon. VolumeIcon.icns at the root of the section and used instead of icons,
given by the theme. It is very convenient to create such an icon using MacOSX tools. Select the icon
disk and make a copy-paste

Page 44

<key> ScreenResolu_on </key>
<string> 1024x768 </string>

you can set the desired screen resolution to be larger than the standard 1024x768, if there is such a mode in the parameters of the video card and the screen itself. Clover tries set the highest possible resolution, however, he may be wrong. Check list of available modes by boot-log.
If the graphics section contains PatchVBios = Yes, then you will have the maximum resolution, available for this monitor. In this case, the ScreenResolution parameter may be superfluous. With some configurations, the PatchVBios parameter can be fatal - black screen with no signs of life.

There is one subtlety here. Clover must know the EDID of the monitor. Legacy Clover tries to get it through INT10 BIOS calls, often successfully, sometimes not. UEFI Clover asks for data from UEFI BIOS, which probably knows EDID for integrated video card, and probably does not know for the inserted one. See preboot.log and if there is no EDID, then enter it manually. See instructions below.

<key> Language </key>
<string> ru: 0 </string>

At the moment, setting the language makes sense only for the "Help" menu called by the F1 key. However, this value is transmitted to the system, and may affect the language of default.

<key> Mouse </key>

```
<dict>
    <key> Enabled </key>
    <true />
    <key> Speed </key>
    <integer> 2 </integer>
    <key> Mirror </key>
    <false />
    <key> DoubleClick </key>
    <integer> 500 </integer>
</dict>
```

Enabled - there are configurations when the mouse does not work, or even hangs, well, then it can be prohibited.
Speed 2 - cursor movement speed, reasonable values 2 - 8. For some mice Negative speed required, reverse travel. Value 0 means the mouse is disabled.
Mirror - and also make the opposite direction only along one coordinate.
DoubleClick 500 - pause in milliseconds to detect a double click. Value 500 has suited everyone so far. Removed in new revisions.

In the Clover interface, you can see legacy and ufi bootloaders for the installed operating systems. At the same time, there can be several bootloaders on one partition. Maybe you don't need everything that Clover found, you just need an indication of the real a couple of systems. You can hide both individual sections and entire classes from the interface bootloaders. The following sections in the config:
Hide - hide volumes by name or by their UUID.

<key> Hide </key>

```
<array>
    <string> WindowsHDD </string>
```

```
<string> B6Q3TX64.EFI </string>
<string> E223FF7F-F2DA-4DBB-B765-756F2D95B0FE </string>
</array>
```


Page 45

This is an array of strings that are included in the fully qualified partition name, see boot.log. So this way you can remove unnecessary partitions from the menu, for example Recovery or Preboot.

Or vice versa, specify what to scan

```
<key> Scan </key>
<dict>
    <key> Legacy </key>
    <string> First </string>
    <key> Entries </key>
    <true />
    <key> Tool </key>
    <true />
    <key> Linux </key>
    <false />
</dict>
```

For **Legacy** (i.e. bootloaders launched from PBR), there are options for values
No, First, Last - do not show at all, place at the beginning of the list, or at the end.
Linux - don't look for Linux bootloaders on every partition, it takes a lot of time.

The interface can be customized and more subtle, if you understand how and what to do. For take you to the next section (also in the GUI section)

```
<key> Custom </key>
<dict>
```

It contains arrays

```
<key> Entries </key>
<array>

<key> Legacy </key>
<array>

<key> Tool </key>
<array>
```

One element of the array contains a description of the selected item in the form of a dictionary

```
<dict>
    <key> Volume </key>
    <string> 454794AC-760D-46E8 -..... 2 </string>
    <key> Type </key>
    <string> OSX </string>
    <key> Title </key>
    <string> OS X 10.8.5 (12F36) Mountain Lion </string>
    <key> InjectKexts </key>
    <true />
    <key> NoCaches </key>
    <false />
        <key> BootBgColor </key>
        <string> 0x2C001EFF </string>
    <key> Hidden </key>
    <false />
    <key> SubEntries </key>
    <array>
        <dict>
            <key> Title </key>
            <string> Boot OS X 10.8.5 </string>
```

```
            <key> AddArguments </key>
            <string> -v </string>
        </dict>
```


Page 46

```
        </array>
    </dict>
```

And each menu item can contain more sub-items (SubEntries) that represent
are different options for calling the main member.
For InjectKexts, there may be a Detect option.
And on some Entry you can put

<key> Ignore </key>

```
<true />
```

<key> Scan </key>

```
<dict>
    <key> Kernel </key>
    <string> All / Newest / Oldest / First / Last / MostRecent / Earliest / None </string>
</dict>
```

This is for Linux.
All - any core
Newest - the newest
Oldest - the oldest
First - the first found
Last - last found
MostRecent - the newest version
Earliest - Oldest version
None - do not search for kernels.
Well Apianti, screw it up!
And the same for Custom entries

```
<dict>
    <key> Type </key>
    <string> LinuxKernel </string>
    <key> Kernel </key>
    <string> All / Newest / Oldest / First / Last / MostRecent / Earliest </string>
</dict>
```

<key> CustomLogo </key>

```
<true /> OR <false /> OR <string> Apple / Alternate / Theme / None / Path </string> OR
<data> PNG / BMP / ICNS base64 data </data>
```

true - the default style
false - disable the logo
Apple - apple gray on gray
Alternate - an alternative apple white on black
Theme - set by theme
None - there is no logo, but there is a background
Path - path to the logo file
<data> - the image is encoded as base64 and contains PNG data.

<key> ShowOp_mus </key>

```
<true />
```

The task was simple. My BIOS spontaneously turns off Optimus on my laptop, and I
I want to see when Clover is loading whether it is turned on or not, so that I can click on
key, and correct the situation.
The criterion is the number of video cards in the system, or the first Intel.

Intel + Discrete = Optimus. On screen word Intel
Discrete only. The word Discrete appears on the screen. Very comfortably!

Page 47

Design: theme.plist

Now the actual design in accordance with the selected theme. Theme.plist file is loaded
from the theme folder, and is unique to each one. The path for the metal theme is:
/EFI/CLOVER/themes/metal/theme.plist
The first theme parameters are copyright, like this

```
<key> Author </key>
<string> Slice </string>
<key> Year </key>
<string> 2012 </string>
<key> Description </key>
<string> Main metallic looking theme </string>
```

Next is the section with design parameters.

```
<key> Theme </key>
<dict>
```

The format of all the mentioned images is PNG, and it is necessary with the correct title. TO
For example, the Viewer saves files in the correct format, but not always.
Sometimes you need to resave through Photoshop.
Some of the interface elements can be excluded by the following set:

<key> Components </key>

```
<dict>
    <key> Banner </key>
    <true />
    <key> Functions </key>
    <true />
    <key> Label </key>
    <true />
    <key> Tools </key>
    <true />
    <key> Revision </key>
    <true />
    <key> MenuTitle </key>
    <true />
    <key> MenuTitleImage </key>
    <true />
    <key> Help </key>
    <false />
</dict>
```

If <true>, then the element is present, otherwise not.

Theme Style (since 3586, by Needy)

<key> BootCampStyle </key>

```
<true />
```

By default, the refit text is written in a separate line, and can be very long
"Boot Recovery from RecoveryHDD". But in BootCamp, like in Chameleon, it is customary to write
inscriptions right under the icons, but only briefly: macOS, Linux, Windows, because otherwise
fit. But this is true only if there are few sections, and each section has its own,
unique bootloader. If, for example, we have two systems, and one of them is closed for
FileVault2, we will have the following items
"Boot Recovery from RecoveryHDD"
"Boot macOS from RecoveryHDD"
"Boot macOS from SierraHDD"
The boot camp style is somehow uncomfortable here. So, you have a choice.

Screen background:

Page 48

<key> Background </key>

```
<dict>
    <key> Type </key>
    <string> Crop </string>
    <key> Path </key>
    <string> MetalBack.png </string>
    <key> Sharp </key>
    <string> 0x80 </string>
    <key> Dark </key>
    <true />
</dict>
```

The **Path** parameter sets the file name (or rather the path!) In which the background image lies on the whole screen. At the same time, the screen may be smaller or larger than the image, and what about this do is determined by the parameter

Type

Crop - crop a large image to fit the screen, or fill it with a background.
Tile - tiled with mosaic tiles.
Scale - stretch proportionally so that the image occupies the entire screen and more, under pruning.
Normal stretching produces square pixels, so usually some anti-aliasing is applied, however, this anti-aliasing spoils the edges.

Edge detection is made in Clover, its value is determined by the parameter

Sharp

If 0 - no detection, the edges are blurred. The maximum value is 0xFF = 255 - no blur.
0x80 - Creates some smart blur with sharp edge lines. Also paired the parameter works with it

Dark

If <true /> implies that you have a dark image with white lines, <false /> - light image with dark lines. This affects edge detection.

<key> Banner </key>

```
<string> logo-trans.png </string>
```

The banner is the central picture, there are size restrictions on it, depending on screen sizes. For example, in the dawn theme, the image is 672 × 190 pixels. This figure can be considered as the maximum. The logo should either be done opaque unless we are going to use a background image. Then the first the logo pixel defines the background color. Or the logo has an opaque element on transparent background, and the entire screen is covered with a background image. Trick from Eps: Top Left make the pixel opaque by 1%.
In new revisions "Banner" has parameters:

```
<key> Banner </key>
<dict>
    <key> Path </key>
    <string> logo_trans.png </string>
    <key> ScreenEdgeX </key>
    <string> left </string>
    <key> ScreenEdgeY </key>
    <string> top </string>
    <key> DistanceFromScreenEdgeX% </key>
    <integer> 10 </integer>
    <key> DistanceFromScreenEdgeY% </key>
    <integer> 10 </integer>
    <key> NudgeX </key>
    <integer> 8 </integer>
```

```
        <key> NudgeY </key>
        <integer> 5 </integer>
    </dict>
```


Page 49

Path - the path to the file, including the folder, for example VariantA \ Logo.png
ScreenEdgeX - horizontal report point (left / right / center)
DistanceFromScreenEdgeX - banner position, relative to the report point, as a percentage of screen size. This ensures correct positioning when changing permissions.
NudgeX - 1% is a lot, for a 1920 screen there will already be 19 pixels, so in this parameter do the refinement in pixel units.
Similarly vertically.

<key> Selec_on </key>

```
<dict>
    <key> Color </key>
    <string> 0xF3F3F380 </string>
    <key> Small </key>
    <string> Select_trans_small.png </string>
    <key> Big </key>
    <string> Select_trans_big.png </string>
    <key> OnTop </key>
    <true />
</dict>
```

Color - the color of the line selection in the menu. The artist sets the color in accordance with the general tone of the topic. The value 0x11223380 means the color red = 0x11, green = 0x22, blue = 0x33, alfa = 0x80. The last number is the degree of opacity, 0x80 corresponds to 50%. 0x00 would mean lack of selection. 0xFF will close the background image (letters on the opaque bar).
Big and **Small** are pictures that highlight the icons in the main menu in the top row - large, and at the bottom - small.
OnTop - location of the selection drawing (rev 1983). False - selection under the icon disk (traditionally for Refit), True - above the icon (traditionally for Chameleon).

<key> Font </key>

```
<dict>
    <key> Type </key>
    <string> Load </string>
    <key> Path </key>
    <string> BoG_LucidaConsole_10W_NA.png </string>
    <key> CharWidth </key>
    <integer> 10 </integer>
    <key> Proportional </key>
    <true />
</dict>
```

Type - the type of font. There are two built-in fonts **Black** and **White** (rev. 3706+), and a dozen loaded - **Load** . In this case, the file name is specified in the following parameter
Path - BoG_LucidaConsole_10W_NA.png
For each theme, its author has chosen the font that best suits his idea, should look in the attached file.
The following conventions are adopted for font names (blackosx)
BoG - Black On Gray - black on a gray background.
LucidaConsole is the name of the original font.
10W - letter width
NA - No Antialiasing. Thought out too.
The size of one character in the file is 16 pixels, however, the characters themselves take up less space, therefore the next parameter specifies the optimal width, and this, again, depends on the author's intention.

Proportional - starting with revision 3217 it became possible to use proportional fonts. For those who are not in the know, this is when the letter i takes much

Page 50

less space than m. A monospaced font is, for example, Courier, proportional eg Times. However, Clover is capable of compacting monospaced fonts, but still, the result will be better with specially trained ones. Here, for example, is a picture

compare the widths of letters i and m.
CharWidth 10 - you can use the width recommended by the font author, or you can change in your own way. 9 - tighter, 11 - less often.
Starting with revision 3537, this parameter affects the width of the text in the Options Menu, since in it is always Proportional = false, and the width of the letters is entirely determined by this parameter, and in line of information, the text is proportional, and is compressed in fact for each letter, even if in the matrix the width is much larger.

<key> Badges </key>

```
<dict>
    <key> Show </key>
    <true />
    <key> Inline </key>
    <true />
    <key> Swap </key>
    <false />
        <key> OffsetX </key>
            <integer> 32 </integer>
            <key> OffsetY </key>
            <integer> 32 </integer>
        <key> Scale </key>
            <integer> 7 </integer>
</dict>
```

Badzhik is a small drawing in the lower right corner of the main picture. Originally it is conceived that the main icon represents the disk (as in the bootcamp), and the badge tells which there is an operating system.
Show - whether to show the badge.
Swap - change the meaning of the icon and badge. Now the icon represents the OS, and the badge - device (in this case it is not interesting to show it).

Inline - show badge in a line with information about the selected icon. There is always an OS, regardless of the Swap parameter. See screenshot for iClover theme.

Page 51

OffsetX and **OffsetY** - offset of the badge from the upper left corner. If no offsets are specified, then the badge is located in the lower right corner.
Scale - the size of the badge in X / 16 units of the original size (in the example 7/16).
That is, in a standard theme, the size is 48 pixels, which corresponds to 6/16 of the standard icons.

```
<key> Scroll </key>
<dict>
        <key> Width </key>
        <integer> N </integer>
        <key> Height </key>
        <integer> N </integer>
        <key> BarHeight </key>
        <integer> N </integer>
        <key> ScrollHeight </key>
        <integer> N </integer>
</dict>
```

Since the settings menu may be longer than the vertical size of the screen,
menu, a scroll bar (Scroll) appears, its parameters are set by the theme, and there are parameters by default, for images included in the theme.

```
<key> Anime </key>
<array>
    <dict>
        <key> ID </key>
        <integer> 1 </integer>
        <key> Path </key>
        <string> logo_3D </string>
        <key> Frames </key>
        <integer> 15 </integer>
        <key> FrameTime </key>
        <integer> 200 </integer>
        <key> Once </key>
        <false />
        <key> ScreenEdgeX </key>
        <string> left </string>
        <key> ScreenEdgeY </key>
        <string> top </string>
        <key> DistanceFromScreenEdgeX% </key>
        <integer> 20 </integer>
        <key> DistanceFromScreenEdgeY% </key>
        <integer> 20 </integer>
        <key> NudgeX </key>
        <integer> 1 </integer>
        <key> NudgeY </key>
        <integer> 1 </integer>
        <key> RelativeXPos </key>
        <string> 50% </string>
        <key> RelativeYPos </key>
        <string> 10% </string>
    </dict>
</array>
```

The theme can contain animated images (clips). Series supported
PNG images with sequential numbers.

ID - defines the use of this clip.

#Logo	(1)
#About	(2)
#Help	(3)
#Options	(4)

#Graphics (five)
#CPU (6)
#Binaries (7)

Page 52

#DSDT fixes (8)
#BOOT Sequence (nine)
#SMBIOS (ten)
#Drop ACPI Tables (eleven)
#RC Scripts (12)
#USB (13)
#Themes (fourteen)
#Apple (21)
#WinXP (22)
#Clover (23)
#Linux (24)
#LinuxEFI (25)
BootX64.efi (26)
#Windows UEFI (27)
#Recovery (thirty)

The header images in each submenu are animated, as well as this animation is played at the selected item in the main menu.

1-10 - list of existing settings submenus.
21-27, 30-39 is the Boot Options menu of details, invoked by a space on the icon in main menu, or by right-clicking.

Those. on this screen the Lion will be animated if the ID 37 is set

Path - ML_Anim **-** The name of the animation, defines the name of the folder in which the individual frames with names
ML_Anim_000.png
ML_Anim_001.png
ML_Anim_008.png
ML_Anim_014.png

Page 53

In case of dropped frames, the last valid one will be used, i.e. as
frames 002-007, frame 001 will be used, and frame 008 will be used as 009-013.
if the plot does not change during this period of time.
Frames - 15 - The total number of frames in the animation. Missing ones will be filled in by
the above algorithm.
FrameTime - 100 - time interval between frames in ms. Variable interval
implemented using dropped frames.
Once - if <true /> **is** specified, the animation will be played only once, before exiting
main menu (right-click into milk on the main screen, or the key
Escape). If <false /> is specified, the animation is played in an infinite loop, for
the last frame goes to zero after the same interval, without an additional pause.
ScreenEdgeX - horizontal report point (left / right / center)
DistanceFromScreenEdgeX - movie position relative to the report point, in percent
screen size. This ensures correct positioning when changing
permissions.
NudgeX - 1% is a lot, for a 1920 screen there will already be 19 pixels, so in this parameter
do the refinement in pixel units.
In the latest revisions, they began to experiment with changing the location itself
theme elements:

<key> Origina_on </key>
<dict>

<key> DesignWidth </key>
<integer> 1920 </integer>

<key> DesignHeight </key>
<integer> 1080 </integer>

With these parameters, we indicate what screen resolution the theme was originally designed for,
in order to correctly recalculate the arrangement of elements at a different resolution.
A large section on the location itself.

<key> Layout </key>
<dict>

<key> Ver cal </key>
<true />

Page 54

<key> BannerO # set </key>

<integer> 80 </integer>

This is the distance from the banner to the main menu, sometimes it needs to be replaced to make the animation the banner did not overlap with the main menu icons.
Similarly

<key> Bu $ onO # set </key>

<integer> 20 </integer>

<key> TextO # set </key>

<integer> 30 </integer>

<key> AnimAdjustForMenuX </key>

<integer> 30 </integer>

You can also scale the main menu icons

<key> MainEntriesSize </key>

<integer> 200 </integer>

The default is 128, as it was before.
With the change of icons, you can also change the distance between them

<key> TileXSpace </key>

<integer> 20 </integer>

<key> TileYSpace </key>

<integer> 20 </integer>

You can also resize the selection.

Page 55

<key> Selec onBigWidth </key>

<integer> 288 </integer>

The default is 144. It matters if the selection is in the background.

Vector themes

Why is it needed

The decision to create GUI Clover's design based on vector graphics arose from one user complaint that you are starting from the same flash drive on different computers, on one small screen, and then a large topic simply does not fit, but on the other huge screen, and then the usual topic looks too small, and the text in general unreadable. How to combine this? The variant was created quickly, a theme with the @ 2x prefix. Then yes, in Clover's config the theme metal is specified, and depending on the monitor resolution either metal is loaded or metal @ 2x . The disadvantage of this approach is that no one rushed create doubled themes. And in general it turned out not too pretty.

Then I got the idea to make really scalable themes based on vector graphics. For those who do not at all understand what the matter is, the circle can be represented as PNG a file that has a very specific size in pixels, but you can write instruction <circle cx = ”100” cy = ”200” r = ”50”> and instruct the drawing program to make circle on the screen with such a position and such a radius in abstract units that will be converted to the required number of pixels depending on the screen size. In original it could be a plotter working in millimeters, for example.

You can't make a good design on some circles and squares, so what is needed here some vector graphics standard with a large number of possibilities. For open source project like Clover of course needs an open standard, well documented, and maintained by paint programs. The choice is unambiguous - SVG graphics. https://en.wikipedia.org/wiki/Scalable_Vector_Graphics

Now the next question is, how can I support it? This is necessary interpret these commands, and draw something appropriate, to put it mildly, not easy a task. And then I find the nanosvg project,https://github.com/memononen/nanosvg It is open source, allows you to copy to yourself, it contains only two files in the C language, then it can be used almost unchanged. The author stopped development despite numerous suggestions for improvement, like done like this, use it as it is. But this is not enough for me, I need more. In addition, it will not go almost unchanged, in Clover simply cannot use the standard C functions sin (), malloc (), sscanf (), qsort () and so on, I still had to make their implementation within the framework capabilities of EDK2. I did it, and also looked at the pull requests, with very good suggestions for improvement, and made some additions. So, vector pictures i can rasterize and display. Now they need to be used to create scalable theme. It is desirable not to spoil the support of existing PNG themes.

But theory without practice is dead, I can't do support for a topic if I don't have a sample such. The theory is as follows, at the first stage. Let the whole theme be a single file with named theme.svg, which by any viewer like GoogleChrome, Safari or just by space, QuickLook will look like Clover's interface. And already Clover will deal with this file, how to parse it into icons, and display them as it does raster theme. Thanks Clovy, he accepted the challenge and started drawing such a file, and as work we have already jointly found what needs to be done in Clover, and how it should be

formed such a file so that ends meet. About this on the forum How to do vector theme in Adobe Illustrator.

Page 56

M asshtabirovanii threads means that the on-screen interface will be of any size look the same, icons will occupy the same percentage of the screen size, in sense of the height of the screen. The theme does not scale in width, in favor of saving squares. That is, on screens with different aspect ratios, the picture will be different, strictly speaking, but a circle remains a circle and a square a square.

Vector themes also have the advantage of using fonts. In a raster theme there was only one font, somehow proportional or not, while in vector it can be several different fonts, truly proportional, and very smooth. In the future, more improvement is possible, for example, with animation, generally the perspectives unfold.

Although yes, I understand, for those who set Timeout = 0, or TextOnly, or embedded, this chapter not at all interesting.

How to make a vector theme

First, you need to put a drawing program in which it will be convenient for you to draw, and which can export this picture to SVG version 1.1 format. Although with some improvements over this standard. There are options, you can look at Google, what is offered on the market. My overview

1. **Adobe Illustrator** . The higher the version, the better. This is what I was looking for compatibility with. Firstly. It is comfortable and functional.
2. **Inkscape** . Advantage # 1 is its free, and, accordingly, legality. SVG it native format, but it has a number of its SVG extensions that are not supported no one else. Hooray! Version 1.0 now works in macOS from 10.11 to 10.15!
3. **LibreOffice Draw** . Also free and legal. But some kind of clumsy and the resulting SVGs are not very good. Doesn't support embedded SVG fonts.
4. **CorelDraw** . I just don’t know, I didn’t use it.
5. **BoxySVG** . Nice, simple, cheap program that allows you to create and edit individual SVG images, and immediately control the generated code, in contrast from older brothers who will generate the code only during the final export.

The lack of the program in the absence of means of solving the problem (I want to see something like this do?). It is clear in the chandelier, you draw, and you see what happens, as you draw, according to while editing, but here you need to have a good imagination to guess the result.

I will not consider other programs, at your discretion, just follow the standard version 1.1, and recommendations below.

Secondly, the theme.svg drawing should consist of well-defined components with quite definite names, since Clover selects drawings from there by name.

Full list:

Background, Banner, selection_big, selection_small, selection_indicator, pointer, scrollbar_background, scrollbar_holder, checkbox, checkbox_checked, radio_button, radio_button_selected,

by some mistake the scrollbar was defined with a minus instead of an underscore, because so in raster themes, in vector ones fixed.

Disk icons

vol_internal - internal hard drive;
vol_external - external media (usually USB);
vol_optical - CD / DVD drive;
vol_clover - bootloader disk (not used?)

vol_internal_hfs - disk partition with HFS + file system;
vol_internal_apfs - disk partition with APFS file system;
vol_internal_ntfs - disk partition with NTFS file system;
Khaki clover. Version 5.0, revision 5120 56
Moscow, 2020

Page 57

vol_internal_ext3 - disk partition with EXT file system;
vol_recovery - partition for Recovery, RecoveryHD and others
Operating system icons
os_clover
os_legacy
os_unknown
os_tiger Mac OS X 10.4 Tiger
os_leo Mac OS X 10.5 Leopard
os_snow Mac OS X 10.6 Snow Leopard
os_lion Mac OS X 10.7 Lion
os_cougar OS X 10.8 Mountain Lion
os_mav OS X 10.9 Mavericks
os_yos OS X 10.10 Yosemite
os_cap OS X 10.11 El Capitan
os_sierra macOS 10.12 Sierra
os_hsierra macOS 10.13 High Sierra
os_moja macOS 10.14 Mojave
os_cata macOS 10.15 Catalina
os_bigsur macOS 10.16 / 11.0 Big Sur

os_win Mandatory, used when there is no icon for the current OS Windows (on volume with NTFS file system)
os_vista Windows Vista, Windows 7, Windows 8, Windows 10
os_freedos
os_freebsd

os_linux Mandatory, used when there is no icon for the current OS Linux (on the with EXT file system)
os_ubuntu
os_suse
the rest of the Linuxes may someday be listed.
Second row service icons
tool_shell, func_clover, func_options, func_about, func_reset and **func_shutdown** as well **func_help** , which is displayed when calling F1 help.

Thirdly, the location of these icons inside the theme.svg file plays a role for the preview, but not for Clover himself. Except for the Banner. This is the central picture and it is located not just in a fixed place in the center of the screen, as it was in bitmap themes, the banner is now positioned on the screen as intended in the theme. Example can be seen in topic cesium. Neither the flower nor the dragon is centered.

Another point, all sorts of buttons and scrolls, in general, are not needed in the preview, as well as numerous icons of axes and disks, so they should be hidden from the preview, but left available to Clover. The visibility = ”hidden” attribute helps, which, however, Clover will ignore when he needs to draw this icon on the screen.

Vector themes support the new "night look" feature. Built-in the clock will tell Clover what time it is, recalculate it to local time, and the time from 8:00 to 20:00 is considered day and the rest is night. More on this later. In vector themes, everything icons can be duplicated with _night appended to the name.
For example, in addition to the main banner named "Banner", you can draw a second picture with

named "Banner_night". And so with all icons, for example "pointer" - "pointer_night", "os_mac" is "os_mac_night". They may differ simply in terms of illumination, or may be

Page 58

completely different. So we actually have two different topics in one, one displayed during the day, second at night.

Each icon must also contain an invisible rectangle defining its boundaries. The reason is simple, the icon may contain less content than its size, and The clover needs to know its full size to scale. The name of such the rectangle necessarily contains "BoundingRect_", but since the names must be unique, some other symbols, just numbers, or the full name of the icon itself.

Example.

```
<g id = "tool_shell" transform = "translate (300, 600)">
    <rect visibility = "hidden" id = "BoundingRect_ts" y = "0" width = "64" height = "64" />
    <g transform = "matrix (0.55 0 0 0.55 15 10)">
        <use xlink: href = "# knopka" width = "85.5" height = "85.5" />
    </g>
    <g transform = "matrix (0.55 0 0 0.55 25 43)">
        <text class = "st35 st36 st38 st39"> $ _ </text>
    </g>
</g>
```

We decipher word by word.

<g ...> - a group of images that make up one total. Its name is "tool_shell", which Clover uses the Shell icons in the second row to draw.

transform = "translate (300, 600)" - sets the shift of the icon on the preview. Clover ignores.

The second line <rect ..> is the same service invisible rectangle that defines the size of the icon, in our case 64x64. Unit of measurement pixels, compared to the size of the theme itself, which is 1600x900. Accordingly, on a screen with a height of 1800 the icon will be 128x128, for example.

In the third line, the images are merged again, and the overall transformation: size by in relation to the above boundaries, and the offset within them. The point is that in in a graphics editor, it is convenient for you to draw on a different scale, but then you need scale individual icons to scale the entire theme.

<use xlink: href = "# knopka" width = "85.5" height = "85.5" /> - this is in the original project nanosvg was not, and in general it is probably from SVG2.0 - support for symbols. The point is so as not to draw the same object multiple times, but just use the same picture. I have all the second row buttons using the same image named "knopka". And the next image is superimposed on it, already with its transformation, and as its two-letter text, for example "$ _". The classes of this text define font, size, fill color, and stroke color. Classes are defined in the file itself theme.svg, somewhere in the beginning.

There is a problem with the transformation in the Chandelier, probably as in other vector editors. It seeks to recalculate the coordinates of all internal objects in the image, instead of to leave the transform attribute outside the group. I haven't learned to fight it so I drew all objects at the origin, and then wrote the scale and offset just in a text editor. Maybe someone will have better luck with this exercise. Why it is important? Yes, because, either in Clover, or on the preview, the parts begin to creep into different sides, and catching them by means of Illustrator spoils the whole picture even more.

SVG support in Clover

Far from ideal. But we can see that a lot of things done by Illustrator works as expected.

Shapes : <rect>, <circle>, <ellipse>, <polyline>, <polygon> and just <path>

Shapes can be combined into <g> groups and <symbol> symbols. Moreover, each
union can contain name and transformation.

Page 59

Group is a static union as defined and displayed. For Clover
the names of the groups are essential, because he divides the picture into icons, which he then uses
for drawing GUI.
So <g id = "vol_internal"> defines a group of shapes that together create an image
internal drive. Within this group, a required <rect id = "BoundingRect_001" element
class = "st0" width = "128" height = "128" />. It is needed to determine the size of the group, for
in order for Clover to correctly position this image, for the internal drawings
are likely to be smaller than the overall size. The name of each element is unique in one
design, so the name of such a rectangle must contain a BoundingRect, and
any additional characters for uniqueness. This is not against the SVG standard, but
needed by Clover to manipulate individual elements.
The symbol is a dynamic union. Browsers will not show characters defined
in the theme, this set of objects is a template for use by other objects.
It will be used by the <use> operator
Example:

```
<symbol id = "HDIconBase" viewBox = "- 64 -64 128 128">
        <rect id = "BoundingRect_3_" x = "- 64" y = "- 64" class = "st0" width = "128" height = "128" />
        <path id = "bottom_3_" class = "st1" d = "M26,49h-52c-3.9,0-7-3.1-7-7v-66c0-3.9,3.1-7,7-
7h52c3.9,0,7,3.1,7,7v66C33,45.9,29.9,49,26,49
                z "/>
        <path id = "top_3_" class = "st2" d = "M26,47h-52c-2.8,0-5-2.2-5-5v-55c0-2.8,2.2-5,5-
5h52c2.8,0,5,2.2,5,5v55C31,44.8,28.8,47,26,47z "
                />
</symbol>
```

A symbol is defined here consisting of two closed curves (path) and one service
square. Styles / classes are defined in the file, I omit here for brevity.
Draw this shape inside the icon

```
<g id = "osx_moj">
        <rect id = "BoundingRect-2" class = "st0" width = "128" height = "128" />

        <use xlink: href = "# HDIconBase" width = "128" height = "128" x = "- 64" y = "- 64"
transform = "matrix (1 0 0 -1 64.0876 64)" style = "overflow: visible;" />
```

xlink: href is a link and points to a symbol called HDIconBase.
Unlike the SVG standard, Clover does not understand group references, only characters. Alas
and oh, but this must be taken into account when creating your design. That is, transfer the group to a symbol, and
link to this symbol.
Fill (fill) and **strokes** (stroke).
I can't say anything specific about strokes. It was made by the founder of the project
nanosvg Mikko Memononen seems to fully comply with the standard, and even supports
dash-line (dash-line). But let's talk about fillings.
A simple option is a uniform color fill. The color is set by 16-digit RGB constants:
fill: "# ffcc00" is orange. There are both real numbers and percentages.
Above the standard in Clover color names: fill: "white" and many more names, I will not
copy and paste a hundred words. Use the SVG standard, and if anything, see the source
Clover.
Another type of fill is gradients.
linearGradient - 100% standard compliance.
radialGradient - unfortunately the focus point is not supported yet, so
the radial gradient in Clover may not look exactly the way it is designed.
conicGradient - this is not in the standard (yet?). Discussion on the Internet is going on,

very interesting drawings made with this method, but the W3C has not yet
is considering including this in the standard, and accordingly browsers do not support it.
How can I not support if my Metal theme is created with conical gradients ?! AND

Page 60

now the cesium theme! Another question is, how did I draw it then? In Photoshop, he can do it.
Adobe didn't include this feature in Illustrator, probably in anticipation
standardization. Syntactically the same as radialGradient, only the color does not change according to
radius, and angle, assuming angle = 0 is stop = "0.0", and angle 360 is stop = "1.0". Well
intermediate values as you want. If you need to rotate the starting angle, there is
gradientTransform = "rotate (45)" and other transformations as in the standard.
For design, you can make a radial gradient, and then use a text editor to turn
its into a conical. The preview, unfortunately, is only in Clover. If anyone finds a second browser with
support cones, please let me know.
Dithering . And this is also not in the standard, although, in my opinion, it suggests itself. Brief theory,
for those who have never heard. If the gradient contains a row of points with a brightness of 150, and behind them with
brightness 151, then a step will be visible, but there are no intermediate values (noticeably on the topic
Clovy background with circles in early versions of Clover, no dithering support). Method
is to display either 150 or
151 is random, with a probability of 0.3. With a large number of points, it will be summed up in
150.3!
The standard does not have a sample syntax for this case, so we developed our own. And he
need namespace clover.
Code:

```
	<radialGradient clover: ditherCoarse = "16" id = "GrayRadialBackground_5_" cx = "441.2867"
cy = "0.7502" r = "1.0023" gradientTransform = "matrix (5.400000e-14 768 -874.24 4.700000e-14
1338.8547 -338690.875) "gradientUnits =" userSpaceOnUse ">
		<stop offset = "0" style = "stop-color: # 7C7C7C" />
		<stop offset = "1" style = "stop-color: # 5E5E5E" />
	</radialGradient>
```

The digital parameter coarse - coarseness, determines the length of the points view, between what and what
to choose. For large pictures, as a background, we chose the value 16.
For icons, 1 is enough, and it looks very smooth and almost imperceptible. Default
value = 0, which means the method is canceled.
I will show the illustrations later, however, at different times they were laid out on the sleigh.

Fill with a pattern
Here we are very far from complete implementation. In the standard, the image will be repeated
as much as needed to fill the filled shape. Only one is made in Clover
option:
The template is a PNG image, and it fills the desired outline in scale,
to completely fill it. In this case, the bitmap will be scaled with
dithering smoothing.
Why is it needed? The fact is that many of Apple's logos are photographs,
that is, a raster, and we have options: vectorize the raster (not very pretty), draw
something completely different, or include the raster in the vector. I did so.
Illustrator has such a function
1. Include the PNG image in the design. (embed).
2. We translate the figure into a pattern (pattern)
3. In the properties of the filled shape, specify the fill with a pattern with such and such ID.
The only problem is that the drawing does not reproduce as expected according to the specification, but
filled all the space with scaling to the size of the canvas.
Oh yeah, another limitation
<pattern id = "pattern_1234" ....>
According to Clover's rules, the pattern identifier must include the word pattern, otherwise

Clover has no way of understanding how we fill the shape. However, using inline PNGs
doesn't make much sense, in this case it was better to do just a PNG theme.

Page 61

Clips (clipPath)
Unlike the original nanosvg project, I have clips supported, but the algorithm
not mine, but taken from the pull request. I don't know how it works, and how adequately. Here in
The cesium Clover theme is drawn using them and it came out almost right. there is
some difference in the rendered image compared to the preview, but I don't know who
fault, clip support, or rendering itself. In general, you can use it, but carefully.

Texts and fonts

The original nanosvg project lacks this capability, but Clover has no text
unthinkable, so I did it. Normal viewing of images with embedded
The image file in fonts is in Safari, but not in Chrome and Explorer. They
show the text with their fonts, of course with distorted design. Embedding SVG
fonts are available in Illustrator, but absent, for example, in BoxySVG, which can
embed only TTF fonts. I don't even remember about other vector editors,
for all one way or another problem.
Clover's vector theme has its own set of rules.
1. You can use multiple fonts, while in bitmap themes only one. One
of these, the main one should be included in the theme.svg file itself. You can turn everything on, just
the file will swell in size. The rest can be put in the theme folder, next to the file
theme.svg. However, in this case, in the preview we will lose the differences in fonts. That is, Clover
will see all fonts, Safari will only be built-in, and Chrome will not see any.
2. At the moment it is possible to define three text styles (font, size and color): for
Options menu, for the Help and About screens, and for messages at the bottom of the screen, the same style
header. Maybe in the future we will expand the use of styles. This is done
special trick.

```
<g id = "MenuRows" class = "st0"> <text class = "st1"> Menu </text> </g>
<g id = "HelpRows" class = "st0"> <text class = "st2"> Help </text> </g>
<g id = "MessageRow" class = "st0"> <text class = "st3"> Boot macOS from HDD </text> </g>
```

Here the st1, st2, st3 classes have to define the text styles, and the st0 class just hide
unnecessary text from the preview (opacity: 0;).
As you can see, everything is according to the standard, just the purpose is different. Here we write the text only for
to indicate styles. And Clover himself will find what to write with these styles, by id
groups, that is, for the "MenuRows" menu, and for help on F1 "HelpRows".

3. But in general, text can be used in icons as an element of decoration. Can
use any font, but make it inline, and maybe embed this
font with the "only used glyph" option - that is, not the entire font, which weighs 500kb on average, but
Chinese and all 30MB, but only the letters used in the icon.
Moreover, you can apply transformations to the text: rotation, skew, etc. Can be done
stroke without filling, the letters will be "white" - I don't remember what it's called, in Windows this
there is no style, in Mac there is, and from the furry years.
Unlike the standard, there is no textPath - the arrangement of text along the line, for example along
perimeter of the circle. Maybe one day...
The text is sized in pixels. That is .st8 {font-size: 16px;}, but that's in design size
Topics. In real work, the theme will be scaled to the size of the monitor, along with
the size of the letters. Thus, for a theme in the 1366x768 design, the text with
size 12px, on a monitor with vertical 2144 it will be almost 36px.

Page 62

4. For my design, I need a font that contains special unicode characters. For instance . ⌘
I found such a font, it takes 30MB, where do I need it? I only take from there some
symbols, and embed them in my theme. The illustrator, seeing an unfamiliar font, replaces it with
my own, and kills everything I did in my theme ....
The first rule the font used must be installed on the system, not just on the theme.
But, I do not have those fonts that are installed on the system in the form of SVG. So so ... I call
my built-in font ArialMT, Helvetica and the like, I insert glyphs there from
the font I need, the illustrator shows it incorrectly, because in a different font,
but at least it doesn't kill. Quicklook shows correctly, and I can move the letters there with the pens,
where they should be.
I also have labels right on the icons. I did it just as an illustrator, his
font, and specified to export the font "used only glyphs". At the same time, I risk
updating the theme to receive less of a letter that you have not used before. So we
should in the original design of the theme embed text containing all English letters,
numbers and punctuation marks. Like this
<text class = "st1"> ABCDEFGHIJKLMNOPQRSTUVWXYZ ...
and so on. Then Illustrator won't forget about all these symbols when importing a theme. Well
put this text in an invisible group.
5. After using Illustrator and asking it to embed an SVG font,
only the fonts used, or anything at all, then you should check that it is there
fucked up.
For example
Code:
<font horiz-adv-x = "2048">
<! - Helvetica is a registered trademark of Linotype AG ->
<! - Copyright: Copyright 2018 Adobe System Incorporated. All rights reserved. ->
<font-face font-family = "MyriadPro-Regular" units-per-em = "2048" underline-position = "- 155"
underline-thickness = "101" />
<missing-glyph horiz-adv-x = "1298" />
<glyph unicode = "" horiz-adv-x = "569" />
<glyph unicode = "E" horiz-adv-x = "1366" d = "M175,1469l1071,0l0, -180l-877,0l0, -446l811,0l0, -
170l-811,0l0, -498l892,0l0, -175l-1086,0z "/>
<glyph unicode = "H" d = "M161,1469l201,0l0, -607l764,0l0,607l201,0l0, -1469l-201,0l0,687l-
764,0l0, -687l-201,0z "/>
First, bbox = "" is missing from the font-face element, but it is strictly necessary, look for what is there
must be inscribing with your hands.
Second, scroll through the list so that each glyph has a horiz-adv-x = "***" attribute,
it determines the width of the letter, that is, where the next one can be drawn.
There is a default value, but it does not always correspond to the real width of the letter, in
as a result, such a letter will run into a neighbor.
In this example, the letter H does not have such an attibut, and its real value is 1479, which is
not obvious. You can take from a letter of about the same width, and you can by a series
experiments to do a little more, or a little less.

Theme attributes

Clover needs his own additions compared to the standard, and this is done in his
namespace. To do this, at the beginning of the file, add
xmlns: clover = "https://sourceforge.net/projects/cloverefiboot"
This is a legal operation. Then we can add illegal things, but with the prefix
clover: as you did with Dithering above.
Another example is conical gradients, this is a deviation from the standard, so I had to introduce

optional attribute, allowed by the standard, but indicating to Clover what to do
act outside the box.

Page 63

```
    <radialGradient clover: conic = "1" id = "knopkaUp" cx = "14142.7324" cy = "- 40300.8711"
r = "37.5003" gradientTransform = "matrix (-0.706 -0.7082 0.7082 -0.706 38525.8438 -
18436.5312) "gradientUnits =" userSpaceOnUse ">
        <stop offset = "0" style = "stop-color: # FFF8D4" />
        <stop offset = "0.1" style = "stop-color: # EEE0B4" />
        <stop offset = "0.5" style = "stop-color: # 161616" />
        <stop offset = "0.9" style = "stop-color: # EEE0B4" />
        <stop offset = "1" style = "stop-color: # FFF8D4" />
    </radialGradient>
```

The radial gradient is described here, and this is how it will be understood by other people's programs.
But Clover will interpret it as a conical gradient.
And now the general theme settings

```
<clover: theme

BootCampStyle = "0"
SelectionOnTop = "0"
SelectionColor = "0x80808080"
NonSelectedGrey = "0"
VerticalLayout = "0"

BackgroundScale = "crop"
BackgroundDark = "1"
BackgroundSharp = "0x80"

Badges = "show"
BadgeOffsetX = "0x0"
BadgeOffsetY = "0xA"
BadgeScale = "0x10"
LayoutBannerOffset = "10"
LayoutButtonOffset = "0"
CharWidth = "16"

AnimeFrames = "39"
FrameTime = "2000"

Version = "100500"
Year = "2018"
Author = "Me"
Description = "My cool vector theme for Clover" />
```

All of these attributes are copied from the bitmap theme settings and serve the same purpose.
However, not all of those attributes are needed and used.

Conclusion

Vector themes are more promising, and I no longer have the slightest desire to have
dealing with raster. There is night mode, beautiful fonts, and scaling. maybe
make dynamic pictures like animation, only on a different level. This requires
reintroduce deviations from the SVG standard. But creating a vector theme is much more
laborious task, and so far only three have coped with it: Clovy with the Clovy theme, me with the theme
cesium, and blackosx with BGM_SVG theme, although he has not yet decided on the final
option. We checked the animation on his theme, but now for some reason he took it off,
probably waiting for vector animation. The fourth designer coped with the task: pkdesign
with the theme "Purple Swirl". Our regiment has arrived!
Cesium theme, daytime look.

Khaki clover. Version 5.0, revision 5120
Moscow, 2020

Page 64

Night view

Page 65

Configuring hardware

Creating the con2g.plist file

Actually, Clover does the configuration automatically. But the automaton never is perfect, so the user has the ability to change different parameters through the config.plist file, or simply in the Options menu when working in the graphical interface. Do you think you can tweak your settings better than a vending machine? Well, try it!

This is an xml file, however, at the moment it is convenient to consider it as a text file file. You can edit this file with a text editor or specialized a program like PlistEditor or Xcode, if available. Now the Clover package includes the Clover.app program that has a plist editing function, no worse than the ones mentioned programs. Use it! Together in Clover, a version of this config with commented out settings so you can see which settings are basically exist, but when used as such, they will not be taken into account.

Examples:

```
<key> #SuspendOverride </key>
<false />
```

The hash # means that the given key is generally excluded from the config. Clover uses the default setting.

```
<key> GUI </key>
<dict>
    <key> #Custom </key>
    <dict>
```

And here the whole Custom section is excluded.

As a general rule, if you don't know what value should be given to some parameter, exclude this parameter from the file altogether, or use #. Don't leave the parameter without values! And even more so, do not put a value that you do not understand!

The following option for making such a config for your computer is offered:

- install the sample file supplied by default, it contains only safe parameters;
- boot into Clover's graphical shell and go to the Options menu (there is such button in the bottom row, or simply by pressing the "O" key);
- use the up / down / enter / escape keys to walk around the menu, and try to understand, what they write there, and why;
- that we correct it clearly, we leave the incomprehensible as it is.
- we boot into the system. If it failed, we repeat the operation, but already changing parameters until complete success.

After logging into the system, go to the terminal, and type the command
cd ~ / App / clover-genconfig> config.plist
Assuming you put genconfig in your ~ / App folder first.
In this way, you get an almost complete config.plist with your most successful parameters with which you managed to boot.
Attention! The clover-genconfig utility depends on the Clover revision!
Since revision 5100 we have a Clover.app program that has the same function genconfig, and this utility is now deprecated.
A little more manual work for complete perfectionism. Below is a description config parameters.

All parameters are grouped into groups: ACPI, Boot, CPU, Devices, DisableDrivers, GUI, Graphics, KernelAndKextPatches, RtVariables, SystemParameters, SMBIOS, BootGraphics.

Page 66

Boot

<key> Timeout </key>

<integer> 5 </integer>

the bootloader entered the GUI and paused for 5 seconds before starting system by default. If during this time the user presses any key, time counting stops. Options: if 0 sec - GUI is not called, the system immediately starts, however, if you press the space bar before, we will go to the GUI. -1 (minus one) - the bootloader enters the menu, does not attempt to start. A pause for 25 seconds in the config is made by default so that the user can admire the animation. Time counting and autostart occurs only if the correct DefaultVolume is specified, and not the one in the default config. It happens that due to heavy animation, the timer is ticking slower than 1Hz, don't be alarmed, it depends on the topic. The timeout does not work if the default system is not defined in NVRAM. Go to system, into the system panel "Boot Disk", and reboot into it. IN next time the timeout will work! Option with Timeout = 0 can be replaced with option

<key> Fast </key>

<true />

In this case, additional saving of loading time is made by not load the interface and its elements. Those. already without a chance to enter the GUI. And without a chance something correct in case of error. Saving a whole second will probably enrich you ?! The system will immediately start booting from the partition specified in the next parameter

<key> DefaultVolume </key>

<string> MacHDD </string>

the name of the section, as you named it, as you see it in the bootloader log. However, the name can also be set in NVRAM after reboot from the control panel "Startup Disk". The name set in NVRAM takes precedence. there is option " LastBootedVolume ". That is, we will boot from the volume from which we loaded into last time. If the parameter is not specified at all, then only by the control panel.

You can also define a default bootloader

<key> DefaultLoader </key>

<string> bootmgfw.efi </string>

That is, if there are several bootloaders on one partition, then in this way we select required to boot by default. In this example, we assume to load UEFI Windows by default. If not set, boot.efi. Again, the "Boot Disk" will change this setting for the duration of the reboot, thus providing autorun Windows through the Mac panel. Unfortunately, there is no such service in Windows, return to Mac can only be done manually.

<key> Legacy </key>

<string> PBR </string>

Legacy Boot, required to run older versions of Windows and Linux, is highly dependent

from the hardware part, from building the BIOS, therefore several algorithms have been developed, and the choice of the algorithm is made in this way. Options:
LegacyBiosDefault - for those UEFI BIOS with LegacyBios protocol.
Khaki clover. Version 5.0, revision 5120 66
Moscow, 2020

Page 67

PBRtest , **PBR, PBRsata** - variants of the PBR boot algorithm, who is lucky with which.
In general, the legacy boot has not been achieved unconditionally. Easier and better already forget about legacy systems, and put UEFI system options. The oldest of them is Windows 7-64 and I personally see no reason to stick with WindowsXP. Someone has a 32 processor is the bit still working? Well, good luck then!

<key> Arguments </key>

<string> -v arch = i386 </string>

These are arguments that are passed to boot.efi, and it, in turn, passes some of them to the kernel systems. The specific list of kernel arguments should be found in the Apple documentation. List the arguments needed by boot.efi itself can be found in the com.apple.Boot.plist manual.
The most famous are the following
Kernel = mach_kernel.amd
slide = 0
darkwake = 0
nvda_drv = 1
For UEFI booting to a 10.8 or 10.9 system, slide = 0 is required. Since revision 1887 it is added automatically when needed. Since revision 4369 there is a driver AptioMemoryFix, and you don't have to write slide with it, it means automatic calculation.
Starting with revision 3712 (actually later due to bugs), in the Details menu, called by space, you have the opportunity to select the desired arguments from the list with the mouse:

```
L "arch = i386" , // 0
L "arch = x86_64" , // 1
L "-v" ,                          // 2
L "-s" ,                          // 3
L "-x" ,                          //4
L "nv_disable = 1" , // 5
L "slide = 0" ,                   // 6
L "darkwake = 0" , // 7
L "-xcpm" ,                       //8
L "-gux_no_idle" , // 9
L "-gux_nosleep" , // 10
L "-gux_nomsi" , // 11
L "-gux_defer_usb2" , // 12
L "keepsyms = 1" , // 13
L "debug = 0x100" , // 14
L "kextlog = 0xffff" , // 15
L "-alcoff" ,                     //sixteen
L "-shikioff" , // 17
L "nvda_drv = 1"                  //18
```

Shl. In Clover 4200 and above, the number of arguments is reduced to purely nuclear, otherwise the menu too long, and many of these arguments are not particularly needed by anyone.

<key> Debug </key>

<false />

Previously, this key was called Log, which caused confusion as to why and how.
Setting the value to <true /> will seriously slow down the work, but it makes it possible after reboot find out what the problem was, because each step will be followed by writing the debug.log file to disk. And if you started with a flash drive, then on it. But with a flash drive will run even slower. The real figure is 10 minutes just to enter the GUI. But, if everything hangs for you, you can press Reset, and then look for the file

/EFI/CLOVER/misc/debug.log, in which all logs for all downloads while this parameter is set. Starting with revision 3063, you still won't

Page 68

sit at the black screen - you will see the loading process on the screen! But if you don't like extra inscriptions on the screen, you can insert

<key> NoEarlyProgress </key>
<true />

These extra inscriptions will disappear

<key> XMPDetec_on </key>
<string> -1 </string>

The parameter specifies whether to detect XMP at boot. It depends on the BIOS, and mainly affects the correct detection of the installed memory. Also possible numeric values 1 or 2 - which XMP profile to use. Perhaps in this profile will be used for other purposes in the future.

<key> Secure </key>
<true />

"Safe Boot". This Microsoft invention has received a strong response from the computer world, they say, on new computers only Windows 8 will work, and in the world Hackintosh cried "end to hacking!" But everything turned out to be not so sad. Of course BIOS manufacturers have provided disabling this function. And also provided uploading certificates. For me, for example, these BIOS settings do not affect successful download.

Page 69

Apianti decided to do a little more. Let's, they say, sign Clover, using
some utility "Signing Tool", download the certificate, and let the BIOS work in the mode
SecureBoot. I do not understand anything about this, so I just quote what has already been done in
Clover, no comment. Hopefully comments will be added in the future.

<key> Policy </key>
<string> Deny / Allow / Query / Insert / WhiteList / BlackList. / User </string>

Deny - download only signed files.
Allow - upload any
Query - ask the owner
Insert - insert a signature into the database
WhiteList - admit by list
BlackList - exclude by list
User - check the lists first, and then ask the user.
The syntax is

<key> WhiteList </key>
<array>
<string> SOMEPATH.efi </string>
</array>

<key> BlackList </key>
<array>
<string> USB (0x1) / HD (0x0,0x1038833 ...) \ EFI \ BOOT \ BOOTX64.efi </string>
</array>

You may also need the key to ignore the hibernate, for the simple reason that
the image is good, but the technology itself does not work on this computer.

<key> NeverHibernate </key>
<false />
Or maybe we are very happy with Hibernate, and we don't want to hang around for five seconds waiting for
this will happen. Then we write

<key> SkipHibernateTimeout </key>
<true />

Second hibernate method:

<key> StrictHibernate </key>
<true />
This only works if there is a hardware NVRAM, but it is compatible with the technology
FileVault2, where the old method doesn't work. Any news. Lilu + HibernateFixup kexts
allow you to save nvram.plist when you go to Hibernate, that is, partially emulate the work
hardware NVRAM, and thus help to use StrictHibernate with mod 25 on
computers with emulated NVRAM.

<key> RtcHibernateAware </key>
<true />
Key for safe operation of the RTC during hibernation. By vit9696, questions to

him. He claimed that for 10.13.4 systems this is the only way, but I have the key in 10.13.6
still in NVRAM. However, you must put <true> for other reasons not related to
RTC, but with a different wake-up algorithm in 10.13.6.

Page 70

<key> HibernationFixup </key>

<true />

Author: lvs1974, explained in his thread how it works and when. Something like
the above situation when there is no hardware NVRAM.

<key> SignatureFixup </key>

<true />

When going into hibernation, the system leaves a signature in the image, which is then checked
boot.efi. With this key, we wanted to correct it. Probably in vain. It is more correct to leave on
the default is just zero and it works. In my opinion, this key is not needed.

And sometimes the system tries to fall into the recovery fashion, but at the same time it does not load from that
disk, we have many systems, this is Hackintosh. To get out of the enchanted ring into
we write to the config

<key> NeverDoRecovery </key>

<true />

With FileVault2 technology, it became possible to use hotkey, but this requires
cancel appointments already made in Clover.

<key> DisableCloverHotkeys </key>

<false />

SystemParameters

<key> CustomUUID </key>

<string> 511CE200-1000-4000-9999-010203040506 </string>

Unique identification number for your computer. If you do not supply this
key, some of the hardware information will be generated, if you want full control
above what is happening, write your 16-digit numbers.
But, for heaven's sake, don't copy my model numbers! They are no longer unique, fools
full of who copied them!

<key> InjectSystemID </key>

<false />

The same number will be injected in a different way, and the system properties will contain
transformed into something else. The point in this operation is to exactly repeat the UUID,
generated by the Chameleon. To do this, set <true />, and as CustomUUID we use
the value that is present with the Chameleon in the registry
IODeviceTree: / e / platform => system-id . Then in the profiler we will see something else
meaning, but the same as before with the Chameleon.
New users no longer know what "as before ..." is. They are already with Clover. Nearly
it is always enough to put <true /> in this parameter and no custom. If, however,
a non-unique number is generated, then it will not be easy to understand where the legs of the problem grow from.
So you better put it false.

<key> BacklightLevel </key>
<string> 0x0101 </string>

Page 71

This property is injected into the system and the system is aware of its existence. However the effect is noticeable only on very rare configurations. What is it? Monitor brightness ... how follows from the name. This property is also read from NVRAM, and, by default, the value set by the system is used. The value specified in the config is or set in the menu will override the default value.

Starting with revision 1865, additional Clover keys were introduced:

<key> InjectKexts </key>

<string> Detect </string>

Loading cakes always occurs if there is no FakeSMC in the cache. Otherwise it is assumed that all kexts are in the cache.
Whenever possible, the system starts with a cache, and it is up to the system to identify whether to use the cache, or need to recreate it. If the value is **Yes** , then Clover will forcefully inject kexts, even if they are in the cache.
Do not! Better put
<true />
This can be done in the Details menu, invoked by the spacebar on the icon systems. These keys are analyzed by the FSInject.efi driver, its presence is required, although in general, kexts are loaded from Clover's folders and without it. But loss is possible interdependencies.

<key> NoCaches </key>

<true />

This option worked on systems prior to 10.7. This is loading only caches, without caches. At all a strange desire to write this parameter to the config. Probably someone else did not load. But today the parameter is useless, the system is loaded from the cache anyway.

<key> NvidiaWeb </key>

<true />

Sets the flag for loading the Nvidia Web driver, the default value that you can change in the Clover interface in the Details menu to the called a space on the system icon. The old boot-arg method only worked before the Captain. In Sierra and above, this method does not work, therefore it is moved to a separate section of the config.

SMBIOS

This group of parameters is needed to mimic your PC under Mac. Clover will do it automatically, based on the detected CPU model, video card, and attribute mobility. However, you may want a different choice. Take MacTracker and pick up the Mac model that you like best, and then search on the Internet, or on familiar with all numbers and serials from this model. There is nothing special to comment on. These options are not for dummies. If you know them, change them, it won't work at random. Calculate them too it is impossible.

<key> ProductName </key>

<string> MacBook1,1 </string>

SMBIOS.table1-> ProductName
You can specify only the name of the product, and Clover will calculate everything from its own tables.

other parameters corresponding to this model. The rest of the parameters can be omitted.
enter, however, if you want other parameters than the default, enter those too.

Page 72

New parameters will be given priority. However, the list of names known to Clover is limited.
only 83 models, I will not list.

For other options, fill in all fields manually. In case you used
model unknown for Clover, then all fields will be initially filled with data from
MacPro3,1, iMac13,1 starting from revision 3900, and then overwritten with your data, if
you gave them. Except for BoardVersion, which will automatically repeat the model.
If the model is not specified, then Clover will substitute something from this list, see the menu,
how much you are satisfied with this choice. Change as you see fit.
Serial numbers are desirable to enter your own. You can take an exemplary one and change one
letter in the middle. This usually goes away. The first three and the last four cannot be changed.

<key> SmUUID </key>

<string> 00000000-0000-1000-8000-010203040506 </string>

SMBIOS.table1-> Uuid
It looks like it makes sense to write the mac-address of your network card here (the last six pairs
characters). This GUID will also be used if CustomUUID is not specified. Better not
set this parameter.

<key> FirmwareFeatures </key>

<string> 0xC0001403 </string>

SMBIOS.table128-> FirmwareFeatures
These numbers are outside the scope of the SMBIOS standard, this is something specific to Apple. IN
different real Macs you can find different numbers, there is no description anywhere,
except that in the bless source you can find

```
&& (featureFlags & 0x00000001)) {
contextprintf (context, kBLLogLevelVerbose, "Legacy mode suppported \ n");
```

Therefore, we also need to have an odd number here. New revisions of Clover Sherlock
did an automatic calculation of the "best" value. I don't know what it affects.
Vit9696 has done some work to determine these bits, see
https://github.com/acidanthera/EfiPkg/blob/master/Include/IndustryStandard/AppleFeatures.h

<key> PlajormFeature </key>

<integer> 3 </integer>

SMBIOS.table133-> PlatformFeature
This parameter is found in real Macs and is used by the Captain, however, which affects,
not found yet.
If the value is not specified, then table 133 will not be created.

<key> BoardSerialNumber </key>

<string> C02032101R5DC771H </string>

SMBIOS.table2-> SerialNumber
This parameter Clover supplies one specific one. You must substitute your
numbers. It is needed for iCloud and iMessage to work. Length is required 17 letters,
capital Latin and numbers. **The number laid down in Clover, most likely for a long time
banned.**

<key> BoardType </key>
<integer> 10 </integer>

Page 73

SMBIOS.table2-> BoardType
This parameter was introduced for MacPro, which has not 10 - Motherboard, but 11 - ProcessorBoard, apparently for historical reasons. The meaning is not obvious, but on Systems The profiler can see it.

<key> BoardVersion </key>
<string> MacBook1,1 </string>
SMBIOS.table2-> BoardVersion
Yes, the model should be written here too.

<key> Mobile </key>
<true />
Actually, Clover always correctly calculates whether a given platform is mobile (i.e. battery powered, requiring energy savings) or not. And the parameter needed if for some reason we want to cheat the system, indicate that there is no battery we do not, or vice versa.

<key> ChassisType </key>
<string> 0x10 </string>
SMBIOS.table3-> Type
This parameter serves as an indirect indication of whether our platform is mobile. Here is a table on SMBIOS standard

```
MiscChassisTypeOther                    = 0x01 ,
MiscChassisTypeUnknown                  = 0x02 ,
MiscChassisTypeDeskTop                  = 0x03 ,
MiscChassisTypeLowProfileDesktop = 0x04 ,
MiscChassisTypePizzaBox                 = 0x05 ,
MiscChassisTypeMiniTower                = 0x06 ,
MiscChassisTypeTower                    = 0x07 ,
MiscChassisTypePortable                 = 0x08 ,
MiscChassisTypeLapTop                   = 0x09 ,
MiscChassisTypeNotebook                 = 0x0A ,
MiscChassisTypeHandHeld                 = 0x0B ,
MiscChassisTypeDockingStation = 0x0C ,
MiscChassisTypeAllInOne                 = 0x0D ,
MiscChassisTypeSubNotebook              = 0x0E ,
MiscChassisTypeSpaceSaving              = 0x0F ,
MiscChassisTypeLunchBox                 = 0x10 ,
```

Clover selects the value as set in real Macs, in accordance with the selected your model. What does this affect, besides mobility - I do not know.

<key> ChassisAssetTag </key>
<string> LatitudeD420 </string>
SMBIOS.table3-> AssetTag
This field in real Macs is never filled, so we can use to for our needs, for example, we link to the HWSensors3 project.

<key> SmbiosVersion </key>
<string> 0x0300 </string>
Apple has its own SMBIOS standard, which does not coincide with others, previously it was most similar to standard 2.4, and Apple inserted that number. But in meaning it was more similar to 2.6, therefore Clover had previously set the figure to 2.6. Now on real Macs you can see the number

3.0, but this is a lie, the standard is still 2.x. However, we can put our
the figure, it seems, does not affect anything.

Page 74

<key> BiosVersion </key>

<string> IM131.88Z.F000.B00.1907241303 </string>

This set of numbers and letters determines whether your hackintosh will ask for a BIOS update, like real. But you don't need that, so you want to put the last values here if you know. And by default, Clover will give out its own numbers, which with each revision updated. You don't have to enter your numbers, but simply update Clover.

<key> E2Version </key>

<string> 288.0.0.0.0 </string>

Likewise, but I'm not sure if this figure affects the need for an update.
Displayed as Boot ROM Version. Different models use either BiosVersion, or EfiVersion.

<key> NoRomInfo </key>

<false />

Clover can generate SMBIOS.table11 with its own firmware numbers. Very convenient in About Mac see these values (AppleROM information).

Oddly enough, on real Macs this information is not, although there is such a table. If a if you want to look real, put this key in <true />.

<key> Trust </key>

<true />

The parameter is used to resolve the dispute between SMBIOS and SPD, whose memory parameters to be recognized as more accurate, in addition to the fact that there are also internal checks. By the default is true, which means SMBIOS (DMI) values are more accurate.
If, with neither true nor false, you cannot get the "correct" memory mapping in system, you have the opportunity to register everything manually (starting with revision 1896)

<key> Memory </key>

<dict>
<key> Channels </key>
<integer> 1/2/3 </integer>
<key> SlotCount </key>
<integer> 24 </integer>

```
<key> Modules </key>
<array>
    <dict>
```


Page 75

```
            <key> Slot </key>
            <integer> 0 </integer>
            <key> Size </key>
            <integer> 2048 </integer>
            <key> Frequency </key>
            <integer> 1600 </integer>
            <key> Vendor </key>
            <string> Some Company </string>
            <key> Part </key>
            <string> 123456ABCDEF </string>
            <key> Serial </key>
            <string> ABCDEF123456 </string>
            <key> Type </key>
            <string> DDR / DDR2 / DDR3 </string>
        </dict>
        ...
        <dict>
            <key> Slot </key>
            <integer> N </integer>
            <key> Size </key>
            <integer> 2048 </integer>
            <key> Frequency </key>
            <integer> 1600 </integer>
            <key> Vendor </key>
            <string> Some Company </string>
            <key> Part </key>
            <string> 123456ABCDEF </string>
            <key> Serial </key>
            <string> ABCDEF123456 </string>
            <key> Type </key>
            <string> DDR3 </string>
        </dict>
    </array>
</dict>
```

Some clarifications:
Channels - the number of memory channels. Very old computers had one channel. On the modern two. There are separate configurations (Clarkdale, for example) where three channel, that is, three-channel memory.
SlotCount - the total number of slots where memory **sticks** can be inserted. Displayed in About window. Now we draw an array of modules, describing only occupied slots. Empty even do not mention. In the Slot key, write its number from 0.

We write the size in megabytes and the speed in megahertz. We do not leave empty fields.

Page 76

In the serial number (Serial) and in the inventory number (Part), only capital letters are allowed letters, numbers, minus signs and periods.
On this, let me close the question with the correctness of memory mapping in the system.
(and still there was a poodle that said it was not displayed as he ordered!
In fact, he prescribed it incorrectly)

```
<key> Slots </key>
<array>
        <dict>
                <key> Device </key>
                <string> Nvidia </string>
                <key> ID </key>
                <integer> 2 </integer>
                <key> Type </key>
                <integer> 16 </integer>
                <key> Name </key>
                <string> PCIe Slot 0 </string>
        </dict>
```

This registers PCI devices in the System Profiler. This is how it looks:

To fill in these properties, write in the config

```
<key> SMBIOS </key>
<dict>
    <key> Slots </key>
    <array>
        <dict>
            <key> Device </key>
            <string> Nvidia </string>
            <key> ID </key>
            <integer> 2 </integer>
            <key> Type </key>
            <integer> 16 </integer>
            <key> Name </key>
            <string> PCIe Slot 0 </string>
        </dict>
        <dict>
```

```
<key> Device </key>
<string> LAN </string>
<key> ID </key>
```


Page 77

```
<integer> 3 </integer>
<key> Type </key>
<integer> 1 </integer>
<key> Name </key>
<string> Ethernet </string>
</dict>
</array>
```

And Clover will generate such tables. In order for the appropriate
_SUN properties, if they are not already there, you need to set the patch mask for these devices.
For this example, this is

```
<key> ACPI </key>
<dict>
<key> DSDT </key>
<dict>
<key> Fixes </key>
<dict>
<key> FixDisplay_0100 </key>
<true />
<key> FixLAN_2000 </key>
<true />
<key> NewWay_80000000 </key>
<true />
</dict>
```

If you write these properties manually, then they must correspond to the ID

```
Device (GFX0)
{
Name (_ADR, Zero) // _ADR: Address
Name (_SUN, 0x02)
… ..
Device (GIGE)
{
Name (_ADR, Zero) // _ADR: Address
Name (_SUN, 0x03)
```

Avoid ID = 0x00 and 0x01 due to optimizations in Zero and One. Clover may not be able to handle
such a patch.
At the moment, this trick is only possible with ATI, NVidia, LAN, WIFI devices ,
Firewire
These are predefined names, Clover will find a device that matches that name.
If you want a more accurate match, then first look in your OEM SBMIOS.
which you can get from the DarwinDumper report, which tables you have # 9, which SUN
tied to what devices. Then fix the DSDT to match these
appointments, and add your own.
Example:

```
Handle 0x0905, DMI type 9, 17 bytes
0000: 09 11 05 09 01 a6 08 03 03 02 00 04 02 00 00 00
0010: fe

System Slot Information
Designation: Ethernet
Type: x1 PCI Express x1
Current Usage: Available
Length: Short
ID: 2
Characteristics:
3.3 V is provided
Hot-plug devices are supported
```

Bus Address: 0000: 00: 1f.6

Page 78

That is, I have an Ethernet controller at address 0000: 00: 1f.6, and it is supposed to have _SUN = 2 (highlighted in green).
This device is in DSDT, but it doesn't have the _SUN property!

```
Device (GLAN)
{
    Name (_ADR, 0x001F0006) // _ADR: Address
```

We must edit! Maybe in the future, Clover will learn to do this automatically.
Slot-> Type is the type of slot from the PCI, PCIe x1, PCIe x2, ... PCIe x16 list, which are encrypted for short numbers 0, 1, 2, ... 16
In this example PCIe x1 is encrypted as Slot-> Type = 1. But since it is for this device table number 9 is already there, you do not need to write it to the config, it is enough to register in DSDT

```
Device (GLAN)
{
    Name (_ADR, 0x001F0006) // _ADR: Address
Name (_SUN, 0x02)
```

CPU

This group of parameters helps with CPU detection when internal algorithms are not cope.

<key> FrequencyMHz </key>

<string> 3200 </string>

Processor base frequency in MHz. Clover usually gets this value by calculations based on the ACPI timer, but if it turns out incorrectly, you can substitute this key.
This key only affects the digit in the system profiler. Cosmetics!
For example, for Hazvels the nominal value is 1800, and the initial speed is 2400. We will work at 2400, and for the profiler we will write 1800.

<key> BusSpeedkHz </key>

<string> 133330 </string>

This parameter is the base frequency of the bus, is critical for system operation, and passed from bootloader to kernel. If the frequency is wrong, the kernel will not start at all, if the frequency does not match slightly, there may be problems with the watch, and very strange system behavior.
The value in DMI is stored in MHz, and this is inaccurate, more correctly calculated from frequency CPU, well, you can choose your value more accurately, and write it in this key in kilohertz. For example, in my DMI it says 100 MHz, but for the clock it became better when I prescribed 99790kHz.
One moment. Some manufacturers have a different concept of what BusSpeed is and what is FSBSpeed, and enter into BIOS a value four times larger. Understand the correctness possible by range: it must be from 100 to 400 MHz, or by the formula
CPU Frequency = Bus Frequency * CPU Multiplier.
It is clear that if the ACS writes the bus frequency 1600 MHz, and the processor multiplier is 8, then the formula does not fit, there are no 12.8 GHz processors. Really it should be divided by 4.
Since revision 1060 there is an automatic frequency detection based on an ADC timer, and these he calculates values better than written in DMI.

Page 79

<key> UseARTFrequency </key>

<false />

SkyLake processors have a new base frequency parameter, which changes with more a small step than the bus frequency, the so-called ARTFrequency, its value is usually equal to 24MHz. Clover can calculate it and transmit it to the core, and the Captain will understand and use it. However, in practice, the calculated frequency leads to inaccurate operation, so it can just deny, in which case the system kernel will act in its own way. In newer versions of Clover rounding of this figure is made, as vit9696 believes there can be only three values, and they are round, up to 1MHz.

<key> QPI </key>

<string> 4800 </string>

In the system profiler, this value is called Processor Bus Speed or simply Bus Speed. Chameleon has an algorithm for calculating it for processors of the Nehalem family (and even that wrong!). In Clover, a corrected algorithm was made using Intel datasheets. IN the source code of the AppleSmbios kernel considers two options: either the value is already written in SMBIOS, as the manufacturer prescribed there, or BusSpeed * 4 is simply calculated. After long disputes, this value is included in the config - write what you like (MHz). To work it does not affect in any way - pure cosmetics. The latest QPI makes sense only for Nehalems, for everyone else here you need to have BusSpeed * 4. Or even nothing. If you forcibly write 0, then DMI table 132 will not be generated at all. Someone argues that this should be done on modern Macs. (Got it!)

<key> Type </key>

<string> 0x0201 </string>

This parameter was invented by Apple and is used in the About this Mac window by internal means that translate such a constant into a processor designation. Otherwise it will show "Unknown processor". Why couldn't the CPUID be called? (because there was still PowerPC). Well, or see table 4 in SMBIOS? No, Apple has its own worldview, but we have to adapt which processor is encrypted. Mostly Clover knows all ciphers, but, since progress does not stand still, the possibility of manually change this setting. The correctness of the setting of this parameter is controlled in the box "About this Mac". Again, cosmetics are pure water.
There is information from vit9696
https://github.com/acidanthera/EfiPkg/blob/master/Include/IndustryStandard/AppleSmBios.h

The group of parameters related to C-state has been moved back to the ACPI section, ACPI-> SSDT.
The following keys are defined here. Excluded today!

<key> C2 </key>
<true />

For modern computers set to false.

<key> C4 </key>
<true />

According to the specification, either C3 or C4. We choose C4. For Evie, set it to false.

<key> C6 </key>
<true />

C6 is known only on mobile computers, however, you can try on

turn on the desktop. Set to true on Evie and Hazwell.

Page 80

Note that with these C-states people often complain about poor sound / graphics / sleep. Be carefully, or exclude them altogether.

<key> Latency </key>

<integer> 250 </integer>

This is the delay for turning on the C3 state. Critical value 0x3E8 = 1000. Smaller - speedstep turns on, more - does not turn on. On native speakers, it is always 0x03E9, that is speedstep doesn't work. On the Hacks, we have to choose what we want, to be like native, or turn on power management. A reasonable value in the second case is 0x00FA as found on some laptops.
MacPro5,1 = 17
MacPro6,1 = 67
aiMak13.2 = 250

<key> SavingMode </key>

<integer> 7 </integer>

Another interesting parameter for controlling speedstep. It affects the MSR register 0x1B0 and defines the behavior of the processor:
0 - maximum performance
15 - maximum energy saving.
With the iMac12 model, I have intermediate states with the last two keys.
However, I have no hard evidence of what influences what.

<key> QEMU </key>

<true />

When testing Clover in a QEMU virtual machine, I found that it was wrong emulates Intel processor. As a temporary measure, this key was made, however, it does not fix everything. The miracle hasn't happened yet. But in the single fashion I can load the system.

<key> TurboDisable </key>

<true />

Useful for laptops so they don't overheat.

<key> HWPEnable </key>

<true />

Intel Speed Shift technology for Skylake processors has been introduced since revision 3879. By goodwin_c. If true, then 1 is written to the MSR 0x770 register. One problem, it is not clear yet, what to do with it. If you put the computer to sleep and then wake it up, the MSR value is 0x770 will reset to 0. And Clover is no longer able to push him back. There is only one option for now, always we put 0, that is, at this point <false />, and already somewhere in the system we try to put 1 in this register. There is an assumption that the system itself will set this bit, if other conditions are met, such as the correct model in SMBIOS.
It is more correct to manage this with the help of a kekst, which will expose unit
https://github.com/headkaze/HWPEnable

<key> HWPValue </key>

<string> 0x30002a01 </string>

This value turned out to be the most appropriate. This value will be written to the MSR register 0x774. Only if MSR 0x770 is 1. Otherwise, this register is not available.

<key> TDP </key>
<integer> 95 </integer>

Page 81

This is Thermal Design Power, taken into account in p-states when generating Control tables
Powered by the processor.

Graphics

This group of parameters is used to inject the properties of the video card, like this
does, for example, Natit.kext. There are a lot of parameters that are actually injected, but this is
mostly constants, some calculated, some given in an internal table, and
only completely separate parameters are entered via the config.

<key> GraphicsInjector </key>
<true />

Actually enabling this injection function. By the way, it is enabled by default, because
injection should work with a clean config - a condition for starting the system. Turn off
injection is worth it if you know the best way.
For some modern cards like Nvidia 6xx or Radeon 6xxx, injection is the default
disabled, because the native factory is working. Incomplete, but on the desktop
you can enter.
In revision 1921+ this parameter is deprecated, but it is supported , now video cards
are injected separately, by vendor, because on modern computers almost
there is always built-in Intel, and sometimes there is no need to enable its injection.

<key> Inject </key>
<dict>
<key> Intel </key>
<false />
<key> ATI </key>
<true />
<key> NVidia </key>
<false />
</dict>

<key> VRAM </key>
<integer> 1024 </integer>

The amount of video memory in MB. In fact, it is automatically detected, but if you write
correct meaning - no one gets hurt. Realistically, however, I don't remember a single incident
so that this parameter helps someone in something. If you see 7MB, don't try to change
this parameter is useless. You need to start a video card. For example, for mobile Radeon
there is a trick to use LoadVBios = true - and the memory will become correct.

<key> LoadVBios </key>
<true />

Loading a video BIOS from a file that should be in the EFI / CLOVER / OEM / xxx / ROM folder
or EFI / CLOVER / ROM and have a file name vendor_device.rom like 1002_68d8.rom. it
sometimes it makes sense if you use a patched video bios. At the request of Ermak, starting with
revision 3222 you can use a longer file name including sub-vendor and
sub-revision 10de_0f00_1458_3544.rom. He needs it to test on one
computer different video cards.
There are also problems that the video card does not show its video BIOS to the system, although
system and requires, for example in the case of mobile radios. In this case, you can put
this parameter to Yes, but do not slip any file. Clover will take VideoBIOS from

legacy memory at 0xc0000, oddly enough, it is almost always there, and now
Clover injects it into the system, and the mobile radeon turns on!

Page 82

Another clarification. It turns out that the BIOS stitched into the ROM of the card does not coincide with the fact that formed at the address 0xc0000 - the shadow of the rum. So, we need him, the shadow one, and not the BIOS that we burn with the programmer.
In short. For mobile radios, set Yes, although there is no file, for the rest cards No. History has not recorded any other options.
And now new times have come. For computers with UEFI-only BIOS, on the legacy address there is no VideoBios. We put it in the file and wait for new solutions. Or ... and so works?

<key> DualLink </key>

<integer> 0 </integer>

By default, the value is injected, but for some older configurations this parameter = 1 leads to quadrupling of the screen. Setting to 0 helps, as in the above example.

<key> BootDisplay </key>

<integer> 1 </integer>

Indicates which display is the primary display. It is he who will light up at the start and wake up after sleep. Usually this is number 0, but sometimes the outputs are numbered in the wrong ok, look in iorege like yours. Revision 3399.

<key> PatchVBios </key>

<true />

Clover is patching the VideoBios shadow at 0xC0000 to support the video mode that is the highest for the connected monitor. For example, in The EDID of the monitor has a 1920x1080 mod, but there is no such mod in VideoBios. Clover will register her in as the first mod and will launch into use. If the monitor does not generate EDID by itself, it can be injected as shown below.
There were cases when enabling this patch led to panic, black screen when trying boot up. Disable this option for the first attempt.
Or the patch uses the value from the config.plist file

```
<key> GUI </key>
<dict>
    <key> ScreenResolution </key>
    <string> 1440x900 </string>
```

If the automation is wrong, you can register the VideoBios patch manually, according to the standard Find / Replace algorithm.

<key> PatchVBiosBytes </key>

```
<array>
    <dict>
        <key> Find </key>
        <data> gAeoAqAF </data>
        <key> Replace </key>
        <data> gAeoAjgE </data>
    </dict>
</array>
```

You can make several patches 0,1,2 in one BIOS ... For example, success with Nvidia came with four patches.
This example from the VideoBios ATIRadeon HD6670, replacing the 1920x1440 fashion with more acceptable 1920x1080. With this method, you should choose a fashion with the same horizontal line.
To successfully set the full screen resolution in the Clover interface, and then in

system, if using safe mode (without video drivers), you must have
EDID. For this, a group of parameters has been introduced

Page 83

<key> EDID </key>
<dict>

In this dictionary, such items

<key> Inject </key>
<true />

1. There are monitors without DDC, such as laptop panels.
2. There are options when there is DDC, but the Apple drivers do not see it.
On the second option, we simply put Inject = yes, and Clover will extract the EDID himself, and slip it
drivers. The need for such an action is noted intopic about mobile radeons... For
UEFI-only computers, we consider the first option, since Clover will not extract VideoBios.
For the first option, we still need to register a new EDID with pens. We write like this:

<key> Custom </key>

<data> AP /////// wAyDADfAAAAAAASAQOAIRV4CunVmVlTjigm
UFQAAAABAQEBAQEBAQEBAQEBAQEB3iGgcFCEHzAgIFYAS88QAAAY3iGgcFCEHzAgIFYAS88QAAAAAAAA /
gBXNjU3RwAxNTRXUDEKAAAA / gAjMz1IZYSq / wIBCiAgAJo = </data>

By the way, the artificially slipped EDID will also be used for automatic patching.
video bios. See above.
Where to get EDID? Well, since it cannot be extracted from this computer in any way, then we take someone else's. the main thing
the condition is for the correct maximum resolution.
I put EDID from Della Inspiron into my sample config-sheet. Matrix 1440x900.
The letters in this example are standard XML encryption, if you look through
PlistEditor, then we see a more human picture

Another option for making EDID is to use ViewSonic EDID Editor
(version 3.1.5) , which, if desired, can be easily ported to OSX. There are also OSX editors,
for example AW_EDID_Editor. But this no longer concerns Clover himself. Study theory.
Clover gives you the ability to inject your EDID, good quality.

There is also information that the Apple drivers check the manufacturer, so there was
a patch was invented so that the automatically extracted EDID would get better on Apple.

Page 84

<key> ProductID </key>

<string> 0x9221 </string>

<key> VendorID </key>

<string> 0x1006 </string>

I cannot say that it helped someone in any way. Quite the opposite, substituted values, and
brightness is no longer adjustable.
New Keys

<key> HorizontalSyncPulseWidth </key>

<string> 0x11 </string>

There is such a parameter in the EDID specification, spelled out in bytes 63 and 65 in the Detailed section
Timing. Some hackers discovered that the parameter affects the known problem of eight
apples. Look for explanations on the forums who managed to achieve what with the change of this
parameter. Yes, it does, and even very much!

<key> VideoInputSignal </key>

<string> 0x80 </string>

The second parameter is from the same hackers. This is byte 0x14 in EDID that defines the properties of the connector.
bit 7 - analog = 0 or digital = 1.
bits 6 - 1 are only defined for analog signals.
bit 0 for digital means the signal is VESA DFP 1.x compliant

<key> VideoPorts </key>

<integer> 2 </integer>

The number of video outputs on the card, including TVO and / or HDMI. Selected frame from
The Apple list may not match our actual map. Affects the quantity
injected connectors. May help combat imaginary monitors.

<key> FBName </key>

<string> Makaka </string>

This parameter is specific to ATI Radeon, for which there are three dozen different
framebuffers without any pattern to anyone. Clover automatically selects from
tables on most of the well-known cards are the most appropriate name. However, others
users of the exact same card dispute, they need a different name. So write in
this parameter is what you think is the most correct. General rule: don't know what
write, delete the parameter altogether.
But don't write to this monkey! Specially prescribed for the absurd - no, it is still copied to
your config!
There is an idea that the whole difference in these frames lies in the set of connectors, and
since you are going to patch them anyway, it makes no difference which one you take for
basis. Unless, the frame must match the video card family. For example Wormy does not
will work with Radeon 6670.

<key> RadeonDeInit </key>

<true />

This dongle works with ATI / AMD Radeon 6xxx cards and above. Or maybe 5xxx, I haven't seen
reviews. It fixes the contents of the GPU registers in such a way that the card becomes
properly initialized, and the MacOSX drivers work with it as needed. Card
turns on at startup, and when you wake up after sleep. Thanks to vit9696 and Mieze.

Page 85

<key> NVCAP </key>

<string> 040000000000003000C0000000000000A00000000 </string>

This parameter is for NVidia video cards, configures the types and assignments of video ports. In this a line of 40 hexadecimal digits in capital letters. There is no theory here, there is empiricism, and even with conflicting results. There is such a plate, but its correctness is disputed.

The first byte is always 04 (in MacBook 05!). The second byte is LID = 01 for laptops.

You can find other ways to calculate the correct value for this string in the forums. AND Clover himself is trying to figure out from the BIOS.

Page 86

<key> display-cfg </key>

<string> 03010300FFFF0001 </string>

This is also a parameter only for NVidia cards. See the discussion for details.
http://www.insanelymac.com/forum/topic/215236-nvidia-injection/

However, the information provided there is controversial. Real configs can be look in the topic http://www.projectosx.com/forum/index.php?showtopic=370 AND actually, the default config that Clover creates seems to be the best option. Just do not specify this parameter at all, give Clover the opportunity to calculate.

<key> NvidiaGeneric </key>

<true />

If true, then instead of the name Gigabyte Geforce 7300LE the name will be NVIDIA Geforce 7300LE
Why - I don't know. Maybe someone needs real data, but someone looks like a native.

<key> NvidiaSingle </key>

<false />

From the same series of obscure patches. If it is worth it, then inject only the first card, the second is not.

<key> NvidiaNoEFI </key>

<false />

Adds the NVDA, noEFI property to Nvidia's injection
Explanations from FredWst http://www.insanelymac.com/forum/topic/306156-clover-bugissue-report-and-patch / page-107? p = 2443062 # entry2443062
Claims that without it there are artifacts on the screen on his GT640.

<key> ig-plajorm-id </key>

<string> 0x01620005 </string>

This parameter is required to run the Intel HDxxxx graphics card, dispute about specific values did not lead to a single rule, so the parameter was simply moved to the config - pick up. By the way, Clover himself will offer some meaning.
Now I have a result. My Skylake started only with parameter 0x193b0000, it matches a configuration that has an HDMI output like mine.

KernelAndKextPatches

This is a group of parameters for implementing binary patches on the fly. It should be noted that this is feasible only if loading occurs via kernelcache or via parameter ForceKextsToLoad. If the kext has not loaded and is not present in the cache, then these fixes are not work. Starting from version 5119, patches occur according to internal algorithms that are not depend on the system version. And for your patches, you can search by symbols.

<key> Debug </key>

<true />

If you want to watch on the move how the patch of kexts happens. Actually, this key for developers.

<key> KernelCpu </key>
<true />

Page 87

Prevents kernel panic on unsupported CPU, in particular Yonah, Atom, Haswell
for older systems. Or some Broadwell-E. Today this patch should be considered
obsolete, and use FakeCPUID and other kernel patches instead.
You need to understand that there are other algorithms in the kernel that will not work correctly with
unsupported CPU, so don't expect this patch to fix all of your problems. Very
it is doubtful if this will work with Pentium M, Pentium 4 or AMD, for such cases
better still find a specially made kernel.

<key> FakeCPUID </key>
<string> 0x010676 </string>

This patch, introduced since revision 2748, replaces KernelCpu. It doesn't just block
kernel panic, it replaces the processor ID so that in all calls it responds as
maintained. In particular, it affects the AppleIntelCPUPowerManagement.kext cache as well. IN
in this example, he substitutes the processor ID Penrin, which is supported by all
versions of OSX from 10.5 up to 10.14, in which it came to an end.

<key> AppleIntelCPUPM </key>
<true />

It turns out that BIOS on motherboards ASUS (which time ASUS spoils us
mood?) writes register 0xE2 bit 14 to MSR, and the register becomes ReadOnly, but it
used in AppleIntelCPUPowerManagement **cache** , and used by write.
The authors of this fix did not think of anything better how to fix the kext itself, because to return
register E2 its former functionality can only be rebooted.
Set Yes if at system start you have a panic about this kext. (yes, register E2 has
the WriteOnce property, i.e. you can write to it only once before rebooting). Actual
for Sandy processors and above. Or reflash the BIOS. And like other operating systems in
this case? They say that it is good for Windows too.

<key> AppleRTC </key>
<true />

The OSX operating system somehow does not work with CMOS as it is provided by the BIOS,
as a result, when you wake up from sleep or when you reboot, the CMOS is reset. Not at
of all, motherboards from Gigabyte are most often seen in this sin. Moreover, often this problem
solved simply by patching DSDT: Device (RTC) which Clover does.
However, in some cases this patch does not help either. Then you can correct the kext itself
AppleRTC, which is done here. Deprecated! vit9696 investigated the problem, and corrected it in
Clover operations with RTC, now the recommended key value is <false />, as it is
affects hibernation. Although, a moot point, I still have a hibernation key
saved to NVRAM.

<key> KernelLapic </key>
<false />

On HP laptops there is a problem with lapic, which is solved by starting with cpus = 1, or now with
with this patch <true />. The problem and solution has existed for a very long time, since the time of the Chameleon, but
until new developers made their way to HP laptops to figure out more seriously what
here the trick.

<key> KernelPM </key>
<false />

It turns out that since system 10.9 there is some CPUPM control embedded right in

nucleus. This patch prevents kernel panic when 0xE2 is locked in BIOS.

Page 88

<key> KernelXCPM </key>
<false />
On 10.12+, XCPM support for IvyBridge processors has been discontinued. Nothing terrible, you can go the Apple way, but those who are used to XCPM can put this key in <true />.

<key> DellSMBIOSPatch </key>
<true />
It has been noticed that on Dell laptops with a Skylake processor and above, the UEFI BIOS itself spoils our ready SMBIOS. It is not clear why, but this must be fought. The patch is tricky, manually so not do it. Only needed on Dell laptops with a Skylake processor (or higher?).

<key> KextsToPatch </key>
<array>
In addition to specific patches, you can make a patch of any other kext, the principle simple: hex string, what to look for, and string, what to replace. Since revision 5095 you can make patches using a mask, see the paragraph Patching with Mask.
Sample: patching VoodooHDA to replace Headphones with Telephones.
Condition - the number of letters must be the same. Or less and padded with zeros.

This method has been successfully applied to enable Trim support for SSD
http://www.applelife.ru/threads/clover.32052/page-539#post-310105
Here's another very useful patch: fighting yellow icons and a broken DVD turntable (which does not work for external drives):
Original theme http://www.applelife.ru/threads/ Change-external- to- internal.38111/

```
<dict>
    <key> Name </key>
    <string> AppleAHCIPort </string>
    <key> Find </key>
    <data> RXh0ZXJuYWw = </data>
    <key> Replace </key>
    <data> SW50ZXJuYWw = </data>
</dict>
```

To choose the MacPro4,1 or 5.1 model without having memory with ECC. AppleTyMCEDriver patch
The simpler method **-nehalem_error_disable disables the AppleTyMCEDriver is also** noticed. Thanks to AkimoA.

One of the most useful patches - removing the limitation on the number of ports in the controller USB3. The problem is that the kext numbers all ports, first as USB2, then as USB3, and

Page 89

the total amount, according to Apple, should not exceed 15 ports. I illustrate the result on your computer

That is, having counted 14 HS ports (usb2), it received only one SS port (usb3), while chipset spec there are 10 of them.
The patch is next (depends on the system version)

```
<dict>
    <key> Find </key>
    <data> g32UDw + DlwQ = </data>
    <key> Comment </key>
    <string> USB 3.0 limit High Sierra 10.13.4 </string>
    <key> Disabled </key>
    <false />
    <key> MatchOS </key>
    <string> 10.13 </string>
    <key> Name </key>
    <string> com.apple.driver.usb.AppleUSBXHCI </string>
    <key> Replace </key>
    <data> g32UGA + DlwQ = </data>
</dict>
```

The sample, by the way, illustrates additional patch keys:

```
<key> Disabled </key>
<true />
```

You can disable dubious keys in the config, then enable them in the interface Clover.

```
<key> Comment </key>
<string> USB 3.0 limit High Sierra 10.13.4 </string>
```

The comment is also visible in the Clover interface, but has no effect on the system.

```
<key> MatchOS </key>
<string> 10.13 </string>
```

It often happens that this patch is applicable only to a specific version of the system, for another version needs a different patch. We write both, and put down the version, including the full 10.13.5 or shortened, as in the example. However, you are unlikely to have two systems 10.13.2 and 10.13.4, so the full version does not make much sense, just remember to update the patches along with system update.

Page 90

It may be necessary to edit not the binary part of the text, but its info.plist. In this case the section looks like this

```
<dict>
    <key> Name </key>
    <string> AppleHDAController </string>
    <key> Comment </key>
    <string> Patch_to_not_load_this_driver </string>
    <key> InfoPlistPatch </key>
    <true />
    <key> Find </key>
    <string> 0x04020000 </string>
    <key> Replace </key>
    <string> 0x44220000 </string>
</dict>
```

There is one complication here. The patch is supposed to be done in kernelcache, but if we do the info-plate patch so that the kext is loaded, this kext is not there yet, since it is not yet loaded. Therefore, you need to boot twice. First time ignoring cache (key NoCache), then FSInject will load this cache, and the second time with the cache, where it will be patch successfully. Put ForceKextsToLoad in the config.
In revision 3154, and then in 3256, the info-sheet patch was fixed (thanks to solstice). Now multiple strings can be included in the search, excluding all invisible characters such as line feed and tabulation. Now you need to set the search in the form **<data>** , because the service characters such as "<" cannot be specified as text. Search and replace string lengths may differ, but the length must be the same, padded with spaces. Example

```
<dict>
    <key> Comment </key>
    <string> Power state 1 - 0 </string>
    <key> Name </key>
    <string> AppleIntelHDGraphicsFB </string>
    <key> InfoPlistPatch </key>
    <true />
    <key> Find </key>
    <data> PGtleT5Qb3dlclN0YXRlczwva2V5PjxpbnRlZ2VyPjE8L2ludGVnZXI + </data>
    <key> Replace </key>
    <data> PGtleT5Qb3dlclN0YXRlczwva2V5PjxpbnRlZ2VyPjA8L2ludGVnZXI + </data>
</dict>
```

If you have a sample config with many patches, but in a specific case use only some, then the extra ones can be prohibited

```
    <key> Disable </key>
    <true />
```

It was very convenient to do this with the Property List Editor, in which <true /> represented by just a check mark. In Clover revisions 3990+, these patches can be enabled and disable in the Clover menu by checking this box.

Patching with Mask

Starting with revision 5095, the ability to implement binary patches with a mask has been introduced. This applies to KextPatches, KernelPatches and BootPatches
I'll tell you in a separate place, keeping in mind all three points.
It looks like this (specific data is unreal, just like a method.
So, in addition to the hexadecimal string Find, we can also set the MaskFind mask, bitwise. If some bit = 1, then we are looking for an exact match, if = 0, then we ignore difference.
And for the Replace line, the MaskReplace mask means that bit = 1 - we make a replacement, bit = 0 - we leave it as it was.

Example:

Page 91

1. Find all lines in the specified cxt, {also works in the kernel, or in boot.efi} clever or Clever. The difference is in the first letter, it differs by bit 0x20. That is, we set the search mask DF FF FF FF FF FF. This means that we are trying to match all bytes (letters), except the first one, in which we ignore the capital letter or the small letter. Search mask can be shorten, because by default all missing bytes are assumed to be FF. It means that unspecified bytes must match
2. Replace the third letter in the found words with "o", that is, we get clover or Clover respectively.
MaskReplace = 00 00 FF 00 00 00
The string can be abbreviated to the right, since it is assumed to be filled with zero. Moreover, those bytes which we do not replace, it would be possible not to indicate in Replace, but there is a requirement exact length match.
To maintain backward compatibility of the new Clover with the old config the unspecified mask (missing) is assumed to be all FFFFFF, i.e. exact match to the search string, and complete replacement of all bytes with the specified ones.

Symbolic patching.

Starting with revision 5119 we have more search and patch capabilities.
General syntax

```
<dict>
        <key> Comment </key>
        <string> Symbolic patch example got lapic panic </string>
        <key> MatchOS </key>
        <string> All </string>
        <key> Disabled </key>
        <true />
        <key> Procedure </key>
        <string> _lapic_interrupt </string>
        <key> RangeFind </key>
        <integer> 200 </integer>
        <key> StartPattern </key>
        <data> ACnHeAAx241H + oM = </data>
        <key> MaskStart </key>
        <data> ///// wA = </data>
        <key> Find </key>
        <data> 6AAA // + DAAAAAAAA </data>
        <key> MaskFind </key>
        <data> / wAA //// AAAAAP // </data>
        <key> Replace </key>
        <data> 6AAA // 8xwJCQkJCQ </data>
        <key> MaskReplace </key>
        <data> / wAA /////////// </data>
</dict>
```

Set MatchOS to All, since we consider this patching method to be version independent systems.
Disabled is still true as this is not a realistic example.
Procedure here we write the name of the procedure we are looking for. Real name could be longer, but the comparison is based on the presence of a substring. Be sure that such a substring occurs only in this procedure.
RangeFind is the length of the codes to search. In general, just the size of this procedure, or smaller. This way we speed up the search without going through all the millions of rows.
StartPattern was invented before the symbolic patch. This is the starting point from where to look for our pattern. If we know the name of the procedure, then StartPattern is hardly needed anymore. Nevertheless let it be. **RangeFind** also applies to **it** .

MaskStart is the mask for the starting point, that is, for the **StartPattern** .
And then the **Find** / **MaskFind** and **Replace** / **MaskReplace pairs** .

Page 92

Starting with revision 2814, it became possible to force loading of kexts

```
<key> ForceKextsToLoad </key>
<array>
    <string> \ System \ Library \ Extensions \ AppleHDA.kext </string>
</array>
```

Thus, the unwillingness of the cakes to load is overcome. Required for loading IOXXXFamily, which is needed to load the main cache, but it depends on this surnames. For example, IONetworkFamily.
Or even the whole \ Extra \ Extensions folder (revision 2816+). This refers to folders on main partition, no other partitions / volumes / disks are provided.
Pay attention to the slash slash! This function requires a driver FSInject.efi.

```
<key> ATIConnectorsController </key>
<string> 6000 </string>
```

To fully run ATI (AMD) Radeon 5000 and 6000 series cards, it is not enough inject properties into the registry, you still need to adjust the connectors in the corresponding controller. In this case, we point to the 6000 controller. The following two properties indicate what to find and what to change to.

```
<key> ATIConnectorsData </key>

<string> 0004000004030000000100002103020404000000140200000001000000000403100000001000000000
0001000000000001 </string>
```

```
<key> ATIConnectorsPatch </key>

<string> 0400000014020000000100000000040400040000040300000001000011020105000000000000000000
0000000000000000 </string>
```

This method only works for systems 10.7 and up.
In 10.12, the connectors will be different, so this method should be deprecated, although the calculation method is still the same.
I'll tell you more about how to get these numbers.
Original article from bcc9
http://www.insanelymac.com/forum/index.php?showtopic=249642
Full recipe from Xmedik in Russian with discussions
http://www.applelife.ru/threads/Zavod-ati-hd-6xxx-5xxx-4xxx.28890/
Here I will briefly outline, taking into account the specifics of Clover.

1. First of all, you need to get your own video BIOS. Boot into CloverGUI and press F6.
   Your BIOS will be saved in the file /EFI/CLOVER/misc/c0000.bin, if, of course, Clover is installed on a partition with the FAT32 file system.
2. Download the radeon_bios_decode program from one of these links. In the same folder with with this utility put the BIOS file c0000.bin. Let's say this is the ~ / RadeonPatch folder
   We execute the following commands in the terminal
   cd ~ / RadeonPatch
   ./radeon_bios_decode <c0000.bin
3. On the screen you will receive information on your connectors, which is copy / take pictures for future use.

Here is what I have
iMac: test slice $ ./radeon_bios_decode <c0000.bin
ATOM BIOS Rom:

Page 93

SubsystemVendorID: 0x1458 SubsystemID: 0x2557
IOBaseAddress: 0xe000
Filename: R667D32I.F1
BIOS Bootup Message:
GV-R667D3-2GI / F1

PCI ID: 1002: 6758
Connector at index 0
Type [@offset 44282]: HDMI-A (11)
Encoder [@offset 44286]: INTERNAL_UNIPHY2 (0x21)
i2cid [@offset 44356]: 0x92, OSX senseid: 0x3
Connector at index 1
Type [@offset 44292]: DVI-D (3)
Encoder [@offset 44296]: INTERNAL_UNIPHY (0x1e)
i2cid [@offset 44383]: 0x95, OSX senseid: 0x6
Connector at index 2
Type [@offset 44302]: VGA (1)
Encoder [@offset 44306]: INTERNAL_KLDSCP_DAC1 (0x15)
i2cid [@offset 44410]: 0x90, OSX senseid: 0x1

4. Download the script **ati-personality.pl** from one of the links
5. Put in the same folder, and run in the terminal
   perl ati-personality.pl -386> frames.txt
   if you are doing this for a 32 bit system, or

   perl ati-personality.pl> frames.txt
   for 64-bit.
   Attention! In Sierra, the texts have changed, so that the patch is obtained systemically dependent.

6. Now you need to decide on the choice of a suitable framebuffer. Apple suggests we have a wide choice: birds, fish, and even monkeys. But the real differences are there in mostly in the connectors, which we are going to change. If not too think about it, then a simple selection option:

5000 series: mobile - Alouatta, desktop - Baboon
6000 series: mobile - Cattail, desktop - Ipomoea
7000 series: mobile - Pondweed, desktop - Futomaki.

7. For the selected framebuffer we take a printout of the connectors from our file frames.txt obtained in step 5.
   0000000 **00 04** 00 00 04 03 00 00 00 01 00 00 **12 04** 01 **05**
   0000010 00 08 00 00 04 02 00 00 00 01 00 00 **11 02** 04 03
   0000020 10 00 00 00 10 00 00 00 00 01 00 00 **00 00** 00 **02**
   The numbers that need to be edited are highlighted in red. Blue numbers - just addresses to be dropped. The third digit from the end is encoderid, the last digit - senseid. The first 4 digits on each line are monitor type (more precisely, the connector type).

   ConnectorType
   02 00 00 00 LVDS
   04 00 00 00 DVI_DL (Dual Link)
   00 02 00 00 DVI_SL (Single Link)

10 00 00 00 VGA
80 00 00 00 S-Video
00 04 00 00 DP

Page 94

00 08 00 00 HDMI

8. senseid we got in step 3 for each of our connectors. encoder can just to zero everywhere. We do not pay attention to other figures. We get the following table:
   0000000 **04 00** 00 00 04 03 00 00 00 01 00 00 **10 00** 01 **06**
   0000020 10 00 00 00 10 00 00 00 00 01 00 00 **00 00** 00 **01**
   0000010 00 08 00 00 04 02 00 00 00 01 00 00 **12 00** 04 03

Those. The first line is DVI-D, the second is VGA, the third is HDMI, and all with my values senseid.

9. And another recipe from Sergey_Galan. http://www.applelife.ru/threads/mobility-ati-radeon-hd5650m-hd5470m-hd4570m-hd4650m.29028 / page-58 # post-379044
   The second digit from the end of the HotPlugID must be in order 00, 01, 02. This affects to sleep and wake up. (highlighted in red)
   0000000 **04 00** 00 00 04 03 00 00 00 01 00 00 **10 00 00 06**
   0000020 10 00 00 00 10 00 00 00 00 01 00 00 **00 00 01 01**
   0000010 00 08 00 00 04 02 00 00 00 01 00 00 **12 00 02** 03
10. Having discarded the blue numbers, we enter the rest into config.plist without spaces and hyphens lines. The original table in ATIConnectorsData, after our edits in ATIConnectorsPatch. See the sample above.
11. I also saw a situation where the VGA connector was presented among the connectors as DVI-I (DVI-SL). And the patch worked with this use.
12. Recipe from eierfrucht https://applelife.ru/threads/sony-vaio-vpceb3m1r.522504/page-3#post-537081 :
    The most interesting point is the FEATURES bits of the LVDS connector, I advise brute force try 08 01, 08 00, 09 01 and 09 00, on one of them everything should start waking up normally. (Highlighted in orange)
    0000000 02 00 00 00 40 00 00 00 **09 01** 00 00 00 00 00 **07**
    0000010 00 04 00 00 04 06 00 00 00 73 00 00 11 02 01 01
13. Ibid, " ***Senseid LVDS panels put 0x7*** ", because Sony is VAIO.

In a new way you need to do this

```
<key> ForceKextsToLoad </key>
<array>
    <string> \ System \ Library \ Extensions \ AMD6000Controller.kext </string>
    <string> \ System \ Library \ Extensions \ AMDFramebuffer.kext </string>
</array>
<key> KextsToPatch </key>
<array>
    <dict>
        <key> Comment </key>
        <string> ATI Connector patch new way </string>
        <key> Disabled </key>
        <false />
        <key> Find </key>
        <data> AAQAAAQDAAAAAQAAIQMCBAQAAAAUAgAAAAEAAAAABAMQAAAAEAAAAAABAAAAAAAB </data>
        <key> MatchOS </key>
        <string> 10.9,10.10,10.11 </string>
        <key> Name </key>
        <string> AMD6000Controller </string>
        <key> Replace </key>
        <data> BAAAABQCAAAAAQAAAAAEBAAEAAAEAwAAAAEAABECAQUAAAAAAAAAAAAAAAAAAAAA </data>
    </dict>
</array>
```

For 10.12, duplicate the entire <dict> with other Find / Replace strings, they are longer.

We can also set binary patches for boot.efi or for the kernel. Way
is the same, so I'll show you with one example

Page 95

<key> KernelToPatch </key>

<key> BootPatches </key>

For more information on masks, see the **Patching with Mask** paragraph .
I want to note that this group of patches is practically not used by anyone, except for the actual
developers, in order to look for errors in difficult cases that are not obvious when
external examination. So we looked for hibernation problems, so we looked for the moment when E2
locks in the kernel.

Devices

A group of parameters for other PCI devices and buses in general.

<key> Inject </key>

<false />

Set this value to true - all internal injection is replaced with a single line input
Properties, which corresponds to the Apple injection via the APPLE_GETVAR_PROTOCOL protocol
with GUID = {0x91BD12FE, 0xF6C3, 0x44FB, {0xA5, 0xB7, 0x51, 0x22, 0xAB, 0x30, 0x3A, 0xE0}}; and
used on real Macs. In the old days, hackers called it EFIstrings.

<key> Proper_es </key>

<string> 0207364862FA54HG345 </string>

Deprecated! Starting from revision 4497, we do not a string of hexadecimal characters, but a dictionary of
the gfxutil rules.

```
<dict>
    <key> PciRoot (0x0) / Pci (0x14,0x0) </key>
    <dict>
        <key> AAPL, clock-id </key>
        <data> AA == </data>
        <key> AAPL, current-available </key>
        <data> sAQ = </data>
        <key> AAPL, current-extra </key>
        <data> vAI = </data>
        <key> AAPL, current-in-sleep </key>
        <data> 6 AM= </data>
        <key> built-in </key>
        <data> AA == </data>
        <key> device_type </key>
        <string> XHCI </string>
    </dict>
    <key> PciRoot (0x0) / Pci (0x19,0x0) </key>
    <dict>
        <key> built-in </key>
        <data> AQ == </data>
    </dict>
```

You can, for example, use DarwinDumper to see which playlist Clover generated
by default, that is, make automatic injections, or as you have done
old, boot into the system, and in the system call in the terminal
clover-genconfig> config-gen.plist
There you look for such a dictionary, and copy it into your config, and the old injection methods are all
turn off. You can also see such a dictionary in the DarwinDumper report from the present
Maca of a similar configuration. The syntax is the same. Only Clover understands a little more
it understands <string>, <integer>, <true>, <false>, <real>.

Today clover-genconfig function and plist editor are included in Clover.app.
Use it!

Page 96

Basically, the same result is achieved by inserting _DSM methods into DSDT, if it is already
is, and if you are engaged in its improvement. It's not the same for everybody. Basically Properties is better,
than _DSM, it is native for real people, and works before the kernel starts.

<key> PCIRootUID </key>
<integer> 0 </integer>

It turns out that the injection of video card properties depends on what number is in
DevicePath = PciRoot (0x0) or PciRoot (0x1). Previously thought to be hardware
characteristic. However, even at the dawn of hacking, it became clear that this number is simply
the identifier registered in the DSDT. Here:

```
Device (PCI0)
{
    Name (_HID, EisaId ("PNP0A08"))
    Name (_CID, EisaId ("PNP0A03"))
    Name (_ADR, Zero)

    Name (_UID, Zero)
```

_UID = Zero - means 0, if equal to **One** , then 1.
Moreover, if this number is changed by force, it will change and will work successfully. So
so, real Macs always have 0. And accordingly boot.efi always assumes 0,
so it's better if you correct your DSDT, Clover does it by default, and such a key
no more in the config.

<key> Audio </key>

```
<dict>
    <key> Inject </key>
    <string> 887 </string>
    <key> ResetHDA </key>
    <true />
    <key> AFGLowPowerState </key>
    <true />
</dict>
```

Inject
Sound card properties injection. True, this only works if the device is in DSDT
called HDEF, but if you rename it, then the rest can be done by another
way to inject. Also, this effort is unnecessary when using the driver.
VoodooHDA.
The value options are as follows:
NO - nothing is injected, for example, if you inject properties yourself via
Properties
Detect - automatic detection of the installed sound microcircuit so that its ID
use as a layout. Generally nonsense, but very popular. In many cases, not
interferes with, and affects the display of the sound card in the System-Profiler.
883 - in decimal form layout number. I mean Realtek ALC883.
0x0373 - the same thing in 16-bit form becomes unrecognizable.
In fact, these numbers are incorrect, the correct layout, for example, 12 = 0x0C, but, no matter how
strangely, are valid.
With the advent of AppleALC, this parameter has taken on new life. Now it really is
layout number, and you need to select it from the list of recommended ones for your sound
codec. See the documentation for this kext.
ResetHDA - if the sound chip for some reason does not turn on, then this key can help with

initial launch. Affects Windows as well. The need is seen after reboots from Windows to Mac.

Page 97

AFGLowPowerState - affects the AppleHDA driver, and should seem to solve the problem with clicks in the speakers when the sound card falls asleep / wakes up. Confirmations the effectiveness of the patch is not enough.

<key> USB </key>

```
<dict>
    <key> Inject </key>
    <true />
    <key> AddClockID </key>
    <true />
    <key> FixOwnership </key>
    <true />
        <key> HighCurrent </key>
      <true />
</dict>
```

Inject
You can put **false** if for some reason you want to refuse the injection of USB properties, for example, if you yourself inject properties through Properties.
FixOwnership
BIOS seizes control of YUSB, and before starting the kernel, we must tear YUSB away from BIOS. For UEFI downloads, it seems to be not relevant, therefore, by default, it is enabled for legacy-boot, and off for UEFI. Relevant!
AddClockID
In the presence of such a property, the YUSB controller falls asleep tightly, and does not wake up the computer. If you want to wake up from the YUSB mouse, put false here. But be prepared that your computer will wake up spontaneously, for example, from the built-in camera.
HighCurrent
The increased current on this YUSB controller is needed to charge the iPad, but I did not do this is the default.

A group of parameters for disguising their devices as native for OSX. (1971)

<key> FakeID </key>

```
<dict>
        <key> ATI </key>
        <string> 0x67501002 </string>
        <key> IntelGFX </key>
        <string> 0x01268086 </string>
        <key> NVidia </key>
        <string> 0x0FE210DE </string>
        <key> LAN </key>
        <string> 0x436311AB </string>
        <key> SATA </key>
        <string> 0x25628086 </string>
        <key> WIFI </key>
        <string> 0x431214E4 </string>
        <key> XHCI </key>
        <string> 0x1E318086 </string>
        <key> IMEI </key>
        <string> 0x1E3A8086 </string>
  </dict>
```

In this group of parameters you can set the relabeling of your unsupported devices to supported. Examples:

- AMDRadeonHD7850 has DeviceID = 0x6819, which is not supported by kexts

ATI7000Controller and ATIRadeonX3000 in 10.8 system. But there is support for
DeviceID = 0x6818. We make a substitution. This fake is required for it to take effect

Page 98

somehow inject. For video cards, there are two options: either Inject-> ATI = true, or DsdtFixMask includes 0x0100 (FixDisplay).

- NVidia GTX660 has DeviceID = 0x1183, the card works anyway, but AGPM for it is not provided. We make a substitution for 0x0fe0, and AGPM turns on. Since for such a card Inject-> NVidia = false, then ID substitution can be done only through DSDT patch with mask 0x0100 (FixDisplay).
- WiFi card in Dell laptop is called Dell Wireless 1595, DeviceID = 0x4315, in fact, it is Broadcom, which supports 4312, 4331, and a number of others. We make a substitution. DSDT patch with mask 0x4000 (FixAirport).
- The common network card Marvell 80E8056 DeviceID = 0x4353 is just not works, but works with the AppleYukon2 driver, if you change the ID to 0x4363. DSDT patch with mask 0x2000 (FixLan).
- IMEI - this device works together with Intel HD3000 / 4000, however, not the fact that your the chipset has the correct ID. The substitutions are as follows:
  SandyBridge = 0x1C3A8086
  IvyBridge = 0x1E3A8086
  Haswell = 0x8C3A8086
  Works with DSDT patch fix AddIMEI_80000 (AddIMEI)

This camouflage works in two cases: when injecting, or when patching DSDT. However, if we do not want a full injection in the version as conceived by Clover, then we can set the following property:

```
<key> NoDefaultProper_es </key>
<true />
```

In this case, the line for the injection is created, but so far it does not contain any new properties. For example, such a property would be FakeID.
Again, this way of doing FakeID is deprecated, it is better to do it through Properties as follows way

You can add your other properties, for example model, in the following array of dictionaries
Deprecated! Use Properties!

```
<key> AddProper_es </key>
<array>
    <dict>
        <key> Device </key>
        <string> NVidia </string>
        <key> Key </key>
        <string> AAPL, HasPanel </string>
        <key> Value </key>
        <data> AQAAAA == </data>
    </dict>
...
</array>
```

Value can be <data> or a hexadecimal string. You can't just string. Then

is instead of <string> ABC .... you must write <string> 0x414243 ....
Convert via PlistEditor or Xcode.

Page 99

The first **Device** key determines which device this device will be added to.
property. Device List:
ATI
Nvidia
IntelGFX
LAN
WIFI
Firewire
SATA
IDE
HDA
HDMI
LPC
SmBUS
USB
Names should be exactly like this, letter by letter. I think no explanation is needed here.
In this way, different properties can be injected for analog audio Device = HDA, and
for digital Device = HDMI. To distinguish Clover will, alas, not be very correct, according to the vendor.
If Intel, then HDA, if ATI or Nvidia, then HDMI. For example, Hazvel has Intel
HDMI sound. This option is not yet provided in Clover. It is provided that with Intel
the graphics on the HDMI output will use the chipset HDA sound. This is what
parameter

<key> UseIntelHDMI </key>
<true />
Affects this parameter on the injection of audio properties transmitted over HDMI, as well as on
DSDT patch. However, as far as I know, sound drivers that VoodooHDA, that
AppleHDA do not fully work with HDMI output. According to new information,
VoodooHDA works only with NVIDIA HDMI output, and as for AMD, Apple in
systems 10.13+ has created a new driver AppleGFXHDA.kext. Explore its capabilities.

<key> HDMIInjec_on </key>
<false />
Completely disable the injection of HDMI device properties.

Starting with revision 3262, a new method of injecting device properties not by name is introduced,
as it was before, but by their location on the PCI bus. Here is the list that gives
Clover in boot-log
4: 432 0: 000 PCI (00 | 00: 00.00): 8086 2E30 class = 060000
4: 432 0: 000 PCI (00 | 00: 02.00): 8086 2E32 class = 030000
4: 432 0: 000 Found GFX model = Unknown
4: 432 0: 000 PCI (00 | 00: 02.01): 8086 2E33 class = 038000
4: 432 0: 000 PCI (00 | 00: 1B.00): 8086 27D8 class = 040300
4: 432 0: 000 PCI (00 | 00: 1C.00): 8086 27D0 class = 060400
4: 432 0: 000 PCI (00 | 01: 00.00): 10DE 01D1 class = 030000
4: 432 0: 000 Found NVidia model = Gigabyte GeForce 7300 LE
4: 432 0: 000 PCI (00 | 00: 1D.00): 8086 27C8 class = 0C0300
4: 432 0: 000 PCI (00 | 00: 1D.01): 8086 27C9 class = 0C0300
4: 432 0: 000 PCI (00 | 00: 1D.02): 8086 27CA class = 0C0300
4: 432 0: 000 PCI (00 | 00: 1D.03): 8086 27CB class = 0C0300
4: 432 0: 000 PCI (00 | 02: 05.00): 10EC 8167 class = 020000

Here we see two video cards and four USB UHCI devices. Device type recognized by its class class = 040300 is an HDA standard audio device. In this case

Page 100

is located at 00: 1B.00 and the class 020000 network card is located at 02: 05.00
Bus = 02
Device = 05
Function = 00
Use this value to inject properties
Deprecated! Use Properties!

<key> Arbitrary </key>

```
<array>
    <dict>
        <key> PciAddr </key>
        <string> 02: 05.00 </string>
        <key> Comment </key>
        <string> Realtek LAN 8167 </string>
        <key> CustomProperties </key>
        <array>
            <dict>
                <key> Disabled </key>
                <true />
                <key> Key </key>
                <string> model </string>
                <key> Value </key>
                <string> Realtek 8169 Gigabit Ethernet Controller </string>
            </dict>
            <dict>
                <key> Key </key>
                <string> built-in </string>
                <key> Value </key>
                <data> AQAAAA == </data>
            </dict>
        </array>
        </dict>
        <dict>
            <key> PciAddr </key>
            <string> 01: 00.00 </string>
            <key> Comment </key>
            <string> Nvidia Geforce card in PCIe slot </string>
            <key> CustomProperties </key>
            <array>
                <dict>
                    <key> Key </key>
                    <string> model </string>
                    <key> Value </key>
                    <string> Gigabyte GeForce 7300 LE </string>
                </dict>
                <dict>
                    <key> Disabled </key>
                    <true />
                    <key> Key </key>
                    <string> AAPL, boot-device </string>
                    <key> Value </key>
                    <data> AQAAAA == </data>
                </dict>
            </array>
        </dict>
    </array>
```

Thus, the Arbitrary section is an array of dictionaries, each of which corresponds to one device with a given address, and for the description of each device an array of CustomProperties consisting of Key-Value pairs is used. Also specific the property can be disabled with the **Disabled** key **.**
The property can be turned on or off dynamically in the Clover menu.
Key must be string <string>
Value can be string, number or data <string>, <integer>, <data>

<key> ForceHPET </key>

<true />

It turns out that there are still computers where HPET is turned off by default, and in BIOS there is no jackdaw to enable it. This key will help (revision 2789+)

<key> SetIntelBacklight </key>

<false />

The key was introduced in revision 3298. In previous systems, the screen brightness was raised by special cakes IntelBacklight or ACPIBacklight, they do not work in the Captain, but it turned out to be very just do it in Clover at the stage of starting the system, and you do not need an additional kext, just put <true />. A trick from Ramalama and Rehabman.
And additional keys

<key> SetIntelMaxBacklight </key>

<true />

The value from the following key will be written into the chip register:

<key> IntelMaxValue </key>

<integer> 1808 </integer>

or the default value most suitable for this chip. Sherlock brought in Clover values for almost all Intel chips, based on the analysis of dumps from real devices. Whether this is always correct, I don't know, but Clover will offer default values.

<key> DisableFunc_ons </key>

<string> 0x18F6 </string>

This is the mask that is superimposed in RCBA 0x3418 - sets additional bits, prohibition of some devices in the Intell chipset. For very serious hackers.

<key> LANInjec_on </key>

<false />

By default, the built-in property is injected for a network card. With these parameters you can prohibit such injection.

RtVariables

The next two parameters were introduced starting with revision 980 and are intended for iMessage registration permissions.
Starting from revision 1129, the parameters are taken from the SMBIOS, and are not needed here.
MLB = BoardSerialNumber
ROM = last digits of SmUUID or Mac address.

<key> MLB </key>

<string> XXXXXXXXXX </string>

Numbers and letters, 17 characters long, indicating the serial number of the motherboard. There is no pattern. Most reliable, take a real number, and change the middle numbers, for example, write ... SLICE ... Who has a fantasy.

<key> ROM </key>

<data> AAAAAAAA </data>

Six pairs of 16-digit numbers, often matching the Mac address of a network card. There are, however,

messages that the service works with arbitrary numbers. Starting with revision 3051 you can

Page 102

write **<string> UseMacAddr0 </string>** , and Clover will determine the Macaddress of your network cards. The procedure does not work for everyone, so check it out.
And most importantly, registration in iMessage implies a paid service, you must specify a real bank card from which you will be debited $ 1. Who is trying to enter for free, receive messages like "Call Apple".

In 2015, 10.11 ElCapitan appeared with new security requirements. SIP = System Integrity Protection. By default, protection is enabled and does not allow loading your keksty and install your system utilities. To turn it off, Clover gives the ability to set new parameters in NVRAM

```
        <key> CsrAc_veCon2g </key>
    <string> 0x3E7 </string>

        <key> BooterCon2g </key>
    <string> 0x28 </string>
```

These are bit masks with possible bit values

```
/ * Rootless configuration flags * /
#define CSR_ALLOW_UNTRUSTED_KEXTS ( 1 << 0 )
#define CSR_ALLOW_UNRESTRICTED_FS ( 1 << 1 )
#define CSR_ALLOW_TASK_FOR_PID ( 1 << 2 )
#define CSR_ALLOW_KERNEL_DEBUGGER ( 1 << 3 )
#define CSR_ALLOW_APPLE_INTERNAL ( 1 << 4 )
#define CSR_ALLOW_DESTRUCTIVE_DTRACE ( 1 << 5 ) / * name deprecated * /
#define CSR_ALLOW_UNRESTRICTED_DTRACE ( 1 << 5 )
#define CSR_ALLOW_UNRESTRICTED_NVRAM ( 1 << 6 )
#define CSR_ALLOW_DEVICE_CONFIGURATION ( 1 << 7 )
#define CSR_ALLOW_ANY_RECOVERY_OS ( 1 << 8 )
#define CSR_ALLOW_UNAPPROVED_KEXTS ( 1 << 9 )
#define CSR_ALLOW_EXECUTABLE_POLICY_OVERRIDE ( 1 << 10 )


/ * Bitfields for boot_args-> flags * /
#define kBootArgsFlagRebootOnPanic ( 1 << 0 )
#define kBootArgsFlagHiDPI ( 1 << 1 )
#define kBootArgsFlagBlack ( 1 << 2 )
#define kBootArgsFlagCSRActiveConfig ( 1 << 3 )
#define kBootArgsFlagCSRPendingConfig ( 1 << 4 )
#define kBootArgsFlagCSRBoot ( 1 << 5 )
#define kBootArgsFlagBlackBg ( 1 << 6 )
#define kBootArgsFlagLoginUI ( 1 << 7 )
```

The default values correspond to disabling protection.
If, due to some paranoia, you need to enable protection, set the value to 0 (zero).

The following three parameters are excluded, because they must be set by the system from the Control Clover panels so there is no conflict.

```
    <key> MountEFI </key>
    <string> Yes </string>
```

This parameter tells the startup script to mount ESP (EFI System Partition) partition. This parameter is unnecessary for most people. or temporary, it is worth prescribing No in the config, and setting Yes in the menu, if necessary. Another possible value is disk1 if you have multiple disks and each has its own partition EFI.

<key> LogEveryBoot </key>
<string> Yes </string>

Page 103

The download log is needed by the developers, and ordinary users can also put No. Here instead of Yes, there can be a number, how many logs to keep in the system.

<key> LogLineCount </key>
<string> 3000 </string>

The number of lines in this log, then the old lines are replaced by new ones so that there is no unlimited growth of this file.
In fact, saving these logs is not interesting. You can always get the latest log with the command
$ bdmesg> ~ / Desktop / boot-log.txt

DisableDrivers

```
<key> DisableDrivers </key>
<array>
    <string> CsmVideoDxe </string>
    <string> VBoxExt4 </string>
</array>
```

The point of this section is to have different config.plist in different OEM folders, but since the drivers folder is common, you need to somehow distinguish which set of drivers is used on this or different configuration. One, for example, needs OsxAptioFixDxe, the other needs EmuVariableDxe. Deprecated! Now all boards use a common set.

Quirks

The history of this section is as follows. Start of development of Clover for UEFI download by Dmazar it was a driver development to adjust the memory that UEFI reserves BIOS Aptio (American Megatrend). The fact is that the Allocate function in this BIOS allocates memory at the bottom, and to boot macOS you need to have the bottom memory free. The conflict affects not only memory, boot.efi also does address virtualization, and this affects pointers, functions, and so on. I will not go into more details, this is not my job, this is Dmazar step by step found all the conflicts and figured out how to resolve them. It became a driver OsxAptioFixDrv.efi. 32 and 64 bit options with all addressing differences.
Note that there was no problem with Legacy Clover, because in this case Allocate from EDK2, and it allocates memory from top to bottom. Legacy Clover works without it driver.

For a long time after Dmazar left, no one touched this driver, except, perhaps, individual one-liners like Free2000. And now vit9696 took up the driver alteration thoroughly. First of all, he made a change to allow the use of native NVRAM on many chipsets (BIOS) that this didn't work with before. And then he split the driver into semantic parts (quirks), which could now be included and turn off at the request of the user if the OpenCore bootloader is used.
But there was also a ReddestDream programmer who decided to highlight all these developments from OpenCore into a separate driver OcQuirks.efi that works in conjunction with the driver OpenRuntime.efi and all settings are recorded in the OcQuirks.plist file to use all it is with Clover bootloader.

And then it's my turn. I need to have all sources in one repo so that I can do bisection. And since the license for the whole thing allows you to use the source in their own mind, and historically everything returned to their historical homeland, I copied and modernized so that instead of a separate .plist file the same Clover config.plist was used, and these settings can also be changed just from GUI Clover.

Page 104

That is, if you cannot boot immediately, you can try to go to this menu and change some setting yes or no.
Now details about each item. The default values are specified.

<key> AvoidRun_meDefrag </key>

<true />

Prevents memory defragmentation for runtime services such as NVRAM support. Recommended for everyone except Apple and VMware.

<key> DevirtualiseMmio </key>

<false />

Removes the runtime attribute from some known MMIO areas. Not recommended for systems older than Sandy Bridge. The list of known regions can still be expanded in the section pickaxes MmioWhitelist. However, according to my observations, it is always disabled. Approved a must for the Z390 chipset.

<key> MmioWhitelist </key>

<array />

The list of regions is specified as an array of dictionaries

```
<dict>
    <key> Comment </key>
    <string> This is such and such a region </string>
    <key> Address </key>
    <string> 0xffe00000 </string>
    <key> Enabled </key>
    <true />
</dict>
```

<key> DisableSingleUser </key>

<false />
Prevents the use of command line mode because it is unsafe. ;)

Page 105

<key> DisableVariableWrite </key>

<false />
Denies access from MacOS to NVRAM by recording, for safety.

<key> DiscardHibernateMap </key>

<false />
When waking up from hibernation, use an old memory card. Use, only
if you are sure that this is your case. Almost always not.

<key> EnableSafeModeSlide </key>

<true />
By default, loading in safe mode gives slide = 0. This patch allows you to use
another value set in the pickaxe ProvideCustomSlide.

<key> ProvideCustomSlide </key>

<false />
Whether or not to automatically calculate the slide = *** value. The need is visible
from the log, if there is an OCABC message : Only N / 256 slide values are usable!

<key> ProvideMaxSlide </key>

<integer> 0 </integer>
A value from 1 to 254, for the case indicated above. The value 255 is set after
default and this can lead to memory allocation errors.

<key> EnableWriteUnprotector </key>

<true />
Clears the write protect bit on the services Runtime page. Due to the insecurity of this pick
there is another RebuildAppleMemoryMap but if your ACPI suite contains MATS.

<key> ForceExitBootServices </key>

<false />
Retry ExitBootServices with a new memory card. This is the moment when
kernel and kext patches occur. Use only if you are sure what you are doing.

<key> ProtectMemoryRegions </key>

<false />
Changes the attributes of some regions when there is a conflict between the concepts of macOS and
BIOS. This pick includes several fixes, including one developed by me personally,
the one that protects the CSM region. In particular, if your video card does not have UEFI
VBIOS, this Kirk should be enabled. On the other hand, in the OsxAptioFix3Drv driver it
definitely included, and never raised any complaints. However, it is still off by
default.

<key> ProtectSecureBoot </key>

<false />
Protects encryption keys from being overwritten by the operating system. Secure technology
Boot, which we somehow don't really need. The need for this pickaxe is observed on
some BIOSes Insyde. The rest are not needed.

<key> ProtectUe2Services </key>

<false />

Page 106

Some BIOSes, such as VMware, try to rewrite pointers to Runtime services.
We don't need this, so we protect them. Rediskin claims that this is needed on the Z390 chipset.

<key> ProvideConsoleGopEnable </key>

<true />

Creates a GOP protocol for console mode, that is, so that the text output is not in text mode, as they used to do in the PC BIOS, and in the graphic mode, as Apple does. Kirk absolutely necessary.

<key> RebuildAppleMemoryMap </key>

<false />

Generate MacOS compatible memory card. The fact is that Apple has slightly different representations of how and what to do than in our UEFI BIOS. The problem, however, is that our the memory card corresponds to our hardware. Therefore, we turn off the pick. Other picks do the necessary part of this work. This kirk seems to replace EnableWriteUnprotector for systems that support MAT.
Also this Kirk requires SyncRuntimePermissions to be enabled. However, we have it enabled by default.

<key> SetupVirtualMap </key>

<true />

Kirk deals with the order in which virtual addresses are assigned and used.
Some BIOSes like OVMF do not support this Kirk. Hmm, why this driver OcQuirks.efi on a system with OVMF ?! She works without him, like Legacy Clover.
So turn it on.

<key> SignalAppleOS </key>

<false />

Tells BIOS that we are loading the macOS system, although we are loading Windows. Needed for some MacBooks, but not us.

<key> SyncRun_mePermissions </key>

<true />

Refreshes the permission flags in the runtime scope. Of course you need to.

The general situation is that for my not very new computers, these are the settings that defaults were found to be sufficient. Rediskin has a different list, and he claims that his the list matches the behavior of AptioMemoryFix. However, this is not true. With his set mine the computer does not boot, while the old AptioMemoryFix is fine. So in Clover's standard set of picks is different from the original.

ACPI

A group of parameters that regulate the correction of various ACPI tables.
And the point is not only that the Mac has its own requirements, but also just different versions of the ADC specifications, and elementary laziness of manufacturers, and there is simply no motherboard in the BIOS information about the installed cards and about the CPU (and dynamically determine weakly? Clover it does!).

Page 107

Parameters for the FADT table

<key> ResetAddress </key>

<string> 0x64 </string>

<key> ResetValue </key>

<string> 0xFE </string>

These two parameters serve one very valuable fix - the restart fix. These
the values should be in the FADT table, but for some reason they are not always there, moreover,
sometimes the table itself is shorter than necessary, so shorter that these values are
discarded. The default is the value already present in the FACP, however, if there
there is nothing, then the pair **0x64 / 0xFE is used,** which means restart through the PS2 controller.
Practice has shown that this does not work for everyone, another possible pair of values is **0x0CF9 / 0x06,**
which means restart via PCI bus. This pair is also used on the native, but not always
works on hackintoshes. The difference is clear, there is also a PS2 controller on hackintoshes,
which can prevent a restart if not reset. Another option is **0x92 / 0x01** , I don't know,
can someone help.

<key> HaltEnabler </key>

<true />

And this is a fix for the problem with shutting down / going to sleep during UEFI boot. Fix
is done once, before calling boot.efi, so 100% efficiency is not
guaranteed. Nevertheless, it is quite safe, at least on Intel chipsets.
For me personally, this is salvation!

<key> smartUPS </key>

<false />

Actually, this parameter is intended to register a profile in the FADT table
power supply = 3. The logic is as follows:
PM = 1 - desktop, mains powered
PM = 2 - notebook, mains or battery powered
PM = 3 - server powered by SmartUPS, about which MacOSX also knows something.
The choice between 1 and 2 Clover will make based on the analysis of the mobility bit, but there is also
the **Mobile** parameter in the SMBIOS section. You can, for example, say that we have McMini, and that he
mobile. The value 3 will be substituted if **smartUPS** = Yes.

MADT Adjustment (APIC)

<key> PatchAPIC </key>

<false />

On some computers, you can boot the system only with cpus = 1, or with a special
patched kernel (Lapic NMI patch). The simplest analysis showed that they have the wrong
table MADT, namely, it lacks NMI partitions. This parameter is used to
adjusting such tables on the fly. Nothing is bad for a healthy computer
will happen. However, I have not seen any reports that helped someone with something. Yes there is
the developer who needs it, and who corrected this patch in the new versions of Clover.
There is a corresponding patch in the KernelAndKextPatches section, also for solving this

problems, but by other means.

Page 108

Other ADC tables:

```
<key> DropTables </key>
<array>
    <dict>
        <key> Signature </key>
        <string> DMAR </string>
    </dict>
    <dict>
        <key> Signature </key>
        <string> MCFG </string>
    </dict>
    <dict>
        <key> Signature </key>
        <string> SSDT </string>
        <key> TableId </key>
        <string> CpuPm </string>
        <key> Length </key>
        <string> 0x0fe1 </string>
    </dict>
</array>
```

In this array, we list the tables we want to drop.
DMAR - because Mac is not friendly with VT-d technology. Rather, there is another table.
MCFG - because by setting a MacBookPro or MacMini we get violent
brakes. A more correct method has already been invented.

```
<key> FixMCFG </key>
<true />
```

In this case, the table is not discarded, but corrected. The author of the patch is vit9696 again. However
less method of dropping the table is still in stock.
Back to the story about DropTables
SSDTs are different, and we additionally indicate the TableId which we will discard,
because we are going to generate our SSDT tables, built according to Apple rules, and
not Gigabyte, or God forgive me, ASUS. You can view it in the table header, or in the boot-
Clover's log. For example, here's a table that shouldn't be dropped.
DefinitionBlock ("SSDT-0.aml", "SSDT", 1, "SataRe", **"SataTabl"** , 0x00001000)
In this case, the rule for binary patches will apply to the saved tables
DSDT, that is, these tables will also be modified, which is logical.
If all SSDT tables for some reason have the same TableID, then you can specify the length
the table we want to drop. The length can be set in a hex, as above, in
<integer> as a decimal number.

```
<key> DisableASPM </key>
<false />
```

This affects the settings of the ACPI system itself, such as Apple's ASPM management
does not work as we have planned. For example, a non-native chipset. In which cases
need to be applied, and what it affects I do not remember.

```
<key> SSDT </key>

    <key> DropOem </key>
    <true />
```

Since we are going to create or load our SSDT tables dynamically, then
unnecessary intersection of interests should be avoided. This option allows you to drop **all**

Page 109

native tables, in favor of new ones. Or, you categorically want to avoid patching tables
SSDT. You have this option: add native tables with minor edits to the folder
EFI / OEM / xxx / ACPI / patched /, and unpatched drops are dropped. (ugh!) uncorrected tables
discard. It is better, however, to use the above selective drop method.

<key> Generate </key>

```
<dict>
    <key> CStates </key>
    <true />
    <key> PStates </key>
    <true />
</dict>
```

Here we define that two additional tables will be generated for C-states and for P-
states, according to the rules developed by the hack community.
For C-states, the table taking into account the C2, C4, C6, Latency parameters mentioned in the CPU section.
You can also specify parameters in the SSDT section, which is logical

```
<key> EnableC7 </key>
<true />
<key> EnableC6 </key>
<true />
<key> EnableC4 </key>
<false />
<key> EnableC2 </key>
<false />
<key> C3Latency </key>
<integer> 67 </integer>
```

The fact that this generation took effect is controlled by the kernel log. Without this method
there is an error *ACPI_SMC_PlatformPlugin :: pushCPU_CSTData - _CST evaluation failed* .
A separate word about C3Latency . This value appears in real Macs, for iMacs
about 200, for MacPro about 10. In my opinion, iMaki are regulated by P-states, MacPro
- C-states. It also depends on the chipset, whether your chipset will adequately respond to
C-state of the team from MakOS. The easiest option is not to write this parameter, that's all
will work like that.
For P-states, the table complements the processor section with the _PPC, _PCT and _PSS methods.
_PCT - Performance Control - speed step control control.
_PPC - Performance Present Capabilities - speedstep capabilities, This function returns
one number that means frequency limiting. Details below, in the parameter
PLimitDict .
_PSS - Performance Supported States - a set of possible processor states - P-states.
This array is formed based on the processor data that Clover has already calculated, and
also taking into account user parameters:

<key> PLimitDict </key>

```
<string> 1 </string>
```

The essence of the parameter is very simple - to limit the maximum processor frequency. Value 0 -
work to the maximum, 1 - one step less than the maximum, 2 - two steps. Example:
Core2Duo T8300 2400MHz operates at a maximum frequency of 2000, if limited to two
steps. What for? Yes, so that the laptop does not overheat, there are many CPU capabilities
exceed the cooling capacity. The exact same parameter is present in
plist platforms, for example:
System / Library / Extensions / IOPla ormPluginFamily.kext / Contents / PlugIns / ACPI_SMC_Pla ormPlugin.kext / Contents /
Resources / MacBook5_1.plist

We will discuss these plists further below.

Page 110

For some processors, for example Core2Quad, PlimitDict has been noticed to work
on the contrary, and the best option = 1. It is possible that this is just a mistake in the DSMT.
For example, because they did not want to make a Darwin patch

<key> UnderVoltStep </key>

<string> 1 </string>

Additional option to lower CPU temperature by lowering it
operating voltage. Possible values are 0, 1, 2, 3 ... the more, the more we cool,
until the computer freezes. In this place, protection against the fool works, Clover does not
will allow you to put the value outside the acceptable range, or rather, write what you want, and
only what is allowed will work. However, the permitted values can give
unstable work. The effect of this parameter is actually observed. However, only for
Penrin.

<key> DoubleFirstState </key>

<true />

Found that for a successful speed step, you need to duplicate the first one in the P-states table
state. After the introduction of other parameters, the need for this became doubtful. Also this
true only for EvyBridge, undoubtedly cancel the rest.

<key> MinMul plier </key>

<integer> 7 </integer>

Minimum processor multiplier. He himself reports that he is 16, and prefers to work for
frequency 1600, however, for speedstep should be set in the table states down to 800 or even
700. Empiricism.

<key> MaxMul plier </key>

<integer> 30 </integer>

Introduced by analogy with the minimum, but it seems in vain. It should not be entered. However, he
somehow affects the number of P-states, so you can experiment, however, without
there is no special need to do this.

<key> Generate </key>

```
<dict>
    <key> CStates </key>
    <true />
    <key> PluginType </key>
    <false />
    <key> APLF </key>
    <false />
    <key> APSN </key>
    <false />
    <key> PStates </key>
    <true />
</dict>
```

In the new Clover, this group of parameters is combined into one section, and PluginType is now
just true or false. Because there are no other options. APLF and APSN parameters like
affect speedstep, but for those who know what they are for.

<key> PluginType </key>

<integer> 0 </integer>

For IvyBridge, Haswell (and higher?) Processors, set 1, for the rest 0.

Page 111

Large section on tuning and patching DSDT.

<key> DSDT </key>

<dict>

<key> Debug </key>

<false />

this parameter allows you to see what happens to the DSDT during its patch, if after
we cannot boot the system. The original version is saved first
/EFI/CLOVER/ACPI/origin/DSDT-or.aml, then the procedure for all patches is done (by the way
it leaves quite a few messages in debug.log if it is also connected), and then it is saved
file /EFI/CLOVER/ACPI/origin/DSDT-pa0.aml, if such a file already exists with
the previous attempt, the next one by number DSDT-pa1.aml, DSDT-
pa2.aml ..., they won't overwrite each other. Do not forget at the end of all exercises
clean the folder.

<key> Name </key>

<string> DSDT.aml </string>

You can have multiple versions of the DSDT file with arbitrary names, or you can
write a non-existent name, for example, the standard BIOS.aml, and then it will be taken as a basis
the DSDT that is in the BIOS. The file priorities are as follows:
1. The highest priority is the DSDT.aml file located in the root of the bootable system. Logic in
to load different computers with one flash drive, each of which has its own
DSDT.
2. If there is no such file, look for it on the USB stick in the OEM section:
/EFI/CLOVER/OEM/p8b/ACPI/patched/DSDT.aml
3. If it is not there, then we look in the shared folder
/EFI/CLOVER/ACPI/patched/DSDT.aml

<key> FixMask </key>

<string> 0xFFFFFFFF </string>

This parameter hides 32 patches for the DSDT table at once, according to the number of bits in
mask.
To calculate how the sum of the bits adds up to this or that mask, you can call
system calculator, convert it to the programmer's view, and switch to 16-digit
numbers. And now by clicking on bits from 0 to 31 we will type the required mask.
There is a more visual option: CloverFixDsdtMaskCalculator by Cvad
http://www.applelife.ru/attachments/cloverfixdsdtmaskcalculator-app-zip.43973/

Page 112

Starting from revision 2184 patches can (and should) be made bit by bit in the next section

<key> Fixes </key>

```
<dict>
  <key> AddDTGP </key>
  <true />
  <key> FixDarwin </key>
  <true />
  <key> FixShutdown </key>
  <true />
  <key> AddMCHC </key>
  <false />
  <key> FixHPET </key>
  <true />
  <key> FakeLPC </key>
  <false />
  <key> FixIPIC </key>
  <true />
  <key> FixSBUS </key>
  <true />
  <key> FixDisplay </key>
  <true />
  <key> FixIDE </key>
  <false />
  <key> FixSATA </key>
  <false />
  <key> FixFirewire </key>
  <true />
  <key> FixUSB </key>
  <true />
  <key> FixLAN </key>
  <true />
  <key> FixAirport </key>
```

```
<true />
<key> FixHDA </key>
<true />
```


Page 113

```
<key> FixDarwin7 </key>
<true />
<key> FixRTC </key>
<true />
<key> FixTMR </key>
<true />
<key> AddIMEI </key>
<true />
<key> FixIntelGFX </key>
<false />
<key> FixWAK </key>
<true />
<key> DeleteUnused </key>
<true />
<key> FixADP1 </key>
<true />
<key> AddPNLF </key>
<true />
<key> FixS3D </key>
<true />
<key> FixACST </key>
<true />
<key> AddHDMI </key>
<true />
<key> FixRegions </key>
<true />
</dict>
```

If this section is present, the mask fix key will be ignored.
But in order to tell what these fixes mean, you will have to open a new chapter.

More keys to help solve some problems with automatic patching.

<key> ReuseFFFF </key>

```
<false />
```

In some cases, an attempt to make a display patch is limited by the presence in the original DSDT device type

```
Device (PEGP)
{
    Name (_ADR, 0xFFFF)
    Name (_SUN, One)
}
```

He can change the address to 0, but this does not always work. <true /> - trying to change address, <false /> - we leave and don't try to patch it.

This fix has been removed from the new Clover since 5116.

```
<key> DropOEM_DSM </key>
<dict>
<key> ATI </key>
<true />
<key> NVidia </key>
<true />
<key> IntelGFX </key>
<true />
<key> HDA </key>
<true />
<key> HDMI </key>
<true />
<key> LAN </key>
<true />
```

```
        <key> WIFI </key>
        <true />
        <key> SATA </key>
```


Page 114

```
        <true />
        <key> IDE </key>
        <true />
        <key> USB </key>
        <true />
        <key> LPC </key>
        <false />
        <key> SmBUS </key>
        <false />
        <key> Firewire </key>
        <true />
    </dict>
```

In some cases, the device that we want to automatically patch already has
OEM method _DSM. You can't duplicate, so there are two options:
<true /> - the original method will be discarded, and ours will be generated instead,
<false /> - upon meeting the original method we retreat without doing anything.
What's in the original method? Yes, hardly what we would like to see, and hardly what
needed for OSX. Usually, BIOS manufacturers only think about Windows.
If it seems to you that there is something important in that method, then inject your properties into
this device using strings (see chapter Devices-> Inject). So I would
recommended dropping all these OEM DSMs, except when you
put your custom DSDT, and apply more fixes to it from among
automatic, but you don't want to override your _DSM methods with automatic
generated.

On my new computer, DSDT turned out to be insanely bad in terms of macOS. In him
many DSM methods, both in the parent device and in the daughters. Result -
panic. DropOEM_DSM is already powerless here. The patch of the following kind helped
Patches → find: _DSM, replace: ZDSM. See below.

```
    <key> SuspendOverride </key>
    <false />
```

The shutdown patch only works for the type 5 state - shutdown, however, we may want
to extend this patch to states 3 and 4, set SuspendOverride = true.
It helped me with going to sleep during UEFI boot. Otherwise, the screen goes out, and the lights and
the fans continued to run.

More advanced hackers can make their own DSDT patches using the replacement for
binary level:

```
    <key> Patches </key>
        <array>
        <dict>
                <key> Find </key>
                <data> UFhTWAhfQURSAAhfUFJXEgYC </data>
                <key> Replace </key>
                <data> UFhTWAhfQURSAAhfU1VOCgQIX1BSVxIGAg == </data>
        <key> TgtBridge </key>
        <data> UFhTW </data>
        </dict>
    <dict>
        <key> Comment </key>
        <string> Rename oem _DSM to ZDSM </string>
        <key> Disabled </key>
        <false />
        <key> Find </key>
```

```
<data>
    X0RTTQ ==
</data>
```


Page 115

```
<key> Replace </key>
<data>
    WkRTTQ ==
</data>
</dict>
  </array>
```

TgtBridge specifies that the patch will only work within one block defined by
by this name
Device (RP02) {..}
however, this method is inaccurate, because blocks can be fragmented, better
use the RenameDevices method (below)
Specific figures are yours, from your designs, if you know what to do. Line lengths
may not coincide, Clover will correctly take into account the change in length, with one exception: so that
it didn't happen inside an If or Else statement. If you require such a change,
replace the entire operator.

Some explanations. The Comment key serves not only to remind you that here
written, it is also used in the Clover menu to compose menus by
connecting / disconnecting these fixes. Initial value included or not
defined by the lines Disabled = false . The default is allowed. If you have
someone else's set of patches, it is better to first register them as Disabled = true, and then in their menu
allow one at a time.

Other ACPI tables

Clover has a service for loading other tables. In particular, it is very common
making a whole library of different SSDT-xxx.aml for different devices and for speedstep.
Starting with revision 3088, the rule for loading such tables has changed.
Place the tables in a folder
/ EFI / CLOVER / OEM / xxx / ACPI / patched /
if there is no such folder, then the shared folder is considered
/ EFI / CLOVER / ACPI / patched /
All files with the ".aml" extension that do not start with a dot "." Will be loaded from this folder.
and do not contain the string "DSDT" in their name, because the DSDT is one of different
options are loaded using a different algorithm.
Loading order is not guaranteed. If we want a strictly defined order
downloads, we must write this explicitly in the config

```
<key> ACPI </key>
<dict>
    <key> SortedOrder </key>
    <array>
        <string> SSDT-3.aml </string>
        <string> SSDT-1.aml </string>
        <string> SSDT-2.aml </string>
    </array>
```

And if such an array is present, only these tables will be loaded, and strictly in this
okay.
Another problem with tables is the name. OEM does not shy away from using national
alphabet, or just the absence of a name, but for Apple this is unacceptable. Name must be 4
symbols of the Latin alphabet. The next fix will fix this

<key> FixHeaders </key>

<true />
The exact same fix was in the DSDT Fixes section, however, since it is by no means related to DSDT, it is placed in a separate item of the ACPI section.

<key> RenameDevices </key>

```
<dict>
        <key> _SB.PCI0.RP02.PSXS </key>
        <string> ARPT </string>
        <key> _SB.PCI0.EHC1 </key>
        <string> EH01 </string>
        <key> _SB.PCI0.POP2.PEGP </key>
        <string> GFX0 </string>
</dict>
```

Unlike binary ACPI patches, which, by the way, affect not only DSDT, but also SSDT also, both native and bootable, this method serves as a replacement for a patch like Find-> PXSX, Replace-> ARPT. But if in the DSDT-> Patches section such a replacement affects all space, then in the RenameDevices method the algorithm will search for only those devices which lie on the indicated bridge.
Here is such a complex example for the first replacement

```
Scope (\ _ SB)
{
    Device (PCI0)
    {
        Device (RP02)
        {
            Device (PSXS) <- change here
            {
                Method (_ON)
                {
                }
                Method (_OFF)
                {
                }
             }
             PSXS._ON () <- change here
        }
        Scope (RP02)
        {
                PSXS._OFF () <- change here
        }
        Device (RP03)
        {
                Device (PSXS) <- don't change here
                {
                }
                PSXS._ON () <- do not change here
        }
```

DSDT correction

DSDT - Differentiated System Description Table - the largest and most complex ADC table. The minimum length is 36 bytes, the real length is 20kb or more, 400kb. This table describes devices and methods of accessing them. Access methods can contain arithmetic and logical expressions, and thus represent a program in a programming language similar to C in its curly braces. Fix this table means to understand something in programming. Clover suggests an option automatic editing, but you need to understand that artificial intelligence has not yet been created, and automatic program correction is still far from perfect. Man will do it's better.

Why should we fix it? The entire DSDT patch, since its inception, has been aimed at
First of all, to fix the HPET device - High Precision Events Timer. The thing is,

Page 117

that the OSX system has an AppleIntelCPUPowerManagement cache that serves
to manage the processor power (speedstep), and which strictly needs to
the system was HPET having IRQ interrupts. Without this condition, the kext goes into a panic.
You can work only by prohibiting, or by removing this kekst. But there is another option -
adjust the DSDT and the HPET will turn on as expected! However, in the system
10.9 requirements have changed. Nonsense! The patch is still needed and has the same effect. In new
systems on the new hardware has already lost its relevance, primarily due to the fact that
the functionality of the kext has gone into the core.

This is patch **# 1** , a vital necessity. Does MacOS only need this HPET? Not,
of course, but BIOS manufacturers are just beginning to realize this and prescribe
correct parameters, it is still rare to find DSDT working without a patch.

Moment **number 2** . In DSDT, you can see some dependence on the operating room
systems, "Windows 98", "Windows 2001", "Windows 2006", "Linux", MacOS has
identifier "Darwin", and, as a rule, the DSDT is not designed for it. And even if
is designed for a version like FreeBSD. MacOSX is a serious ACPI system, i.e.
uses DSDT to the maximum, the way Windows 2001 uses it, but not Linux, not
Windows 98, and not Windows 2006. The best way is to mimic Windows 2001. And
even if you already have Darwin, make sure it runs like Windows 2001. **On the
many BIOSes this corresponds to the value OSYS = 0x07D2. But not 0x2410, like
it is written in the native. Although there are reports that 7D6 or 7D9 for some
configurations are somewhat better.** You need to look at the algorithm.
On my 2017 skylake, the best value was 0x7DA. Therefore, an additional
fix DSDT FixDarwin7_10000, which makes mimicry under Windows 2009, that is
Windows 7. It corresponds to 0x7D9 for me, and this affects the operation of USB3.0, for some reason in
BIOS decided to turn it off for WindowsXP. Again, the patch may not be needed,
if this identification is not used further in the DSDT. Have you checked it?

Moment **number 3** . The manufacturer of the motherboard, and with it its BIOS and its DSDT, cannot
foresee which processor will be installed, which video card and other PCI
devices. But they should be registered in the DSDT! Conversely, to exclude such
devices like speaker, floppy disk drive, parallel port. There are no drivers for them and
are needed. It is also often necessary to add or subtract connectors / ports for some
devices, for example, a video card, or a SATA controller.

DSDT lies in BIOS and is used in the system in the AML binary code, exists
an IASL compiler / decompiler that translates codes into human-readable language
DSL. The human way of editing such AML-> DSL-> edit-> DSL-> AML. And then there is
moment **number 4** . The last compilation becomes impossible due to errors, syntax
and logical, originally present in the OEM DSDT. The editing process requires them
fix. Well, at the same time, correct semantic errors, due to which, for example,
the computer cannot sleep, or cannot wake up. And maybe new devices too
register. (In general, it is strange, but compilation / decompilation is not strictly the opposite.
operations, back and forth changes the table, or even in general, it goes there, there is no back - it is required
intervention. Looking from my bell tower, this means that the decompiler is written with
errors, such programmers worked on it. And non-compliance with standards is necessary
mark them as warnings, not errors. If in the original AML
something is wrong, then you need to understand that the computer is working, then this
the wrong can somehow be interpreted).

When we have come all this way, we can slip our fixed
DSDT by putting it in the / EFI / CLOVER / OEM / xxx / ACPI / patched folder, or if the OEM name

the computer is not yet known to the / EFI / CLOVER / ACPI / patched folder, or the system being booted itself has its own version of DSDT, which lies at the root of the system disk.

Page 118

Where can I get the original DSDT that needs to be patched? There are options to get it using Windows, Linux or even OSX. If Clover somehow managed to start, then now he himself provides such an opportunity. You need to enter the graphical menu and click **F4** key . If Clover is installed on a FAT32 partition, then it will be able to save all OEM ACPI tables, including untouched DSDT and FADT.
Be patient, if saving to a USB flash drive, and there are many tables, the process may take noticeable time. In the current revision, Clover retrieves the set tables previously inaccessible by other means, including in AIDA64. There is also a way save a version of the prepaired DSDT to disk. To do this, in the Clover interface, enter Options Menu, change the DSDT mask, then exit the menu and press **F5** . Clover will keep your DSDT corrected for the current mask, with a name like **DSDT-F597.aml** , i.e. patched with mask 0xF597. You can make several options in order to compare later.

Now you can take a DSDT file and edit ... Well, for those who are not strong in the language ASL, Clover suggests doing some fixes automatically. I will immediately answer this question: "Why, after editing by Clover, DSDT still compiles with errors?"
Yes, DSMD is a collection of descriptions and methods, many of which we don't need. Clover does not touch them, even if we set the maximum mask. Errors can be laid in those places, untouched, and they remain there. But this does not prevent everything from working the rest, because the DSDT does not work as a single whole, but simply as a set of descriptions and methods.

Let's consider fixes in more detail. In revision 4300+ the fixes description has been simplified.

AddDTGP bit (0):

To describe device properties, except for the DeviceProperties option discussed above, there is a variant with the _DSM method prescribed in DSDT.
_DSM - Device Specific Method - a well-known template for this method that works in MacOSX since version 10.5, this method contains an array with a device description and a call universal **DTGP** method , which is the same for all devices.
This fix simply adds this method so that it can then be applied to other fixes.
It does not matter on its own. I saw the advice to put a mask 0x31, they say, other fixes are not needed. But then (1) is not needed!

FixDarwin bit (1):

Mimicry OS Darwin for Windows XP. Many sleep and brightness problems growing with their feet from a misidentified system.

FixDarwin7 bit (16)

Likewise, only mimicry for the Windows 7 system may not be in the old DSDT checks for such a system. You have options.

FixShutdown bit (2):

A condition is added to the _PTS function: if argument = 5 (off), then no no other action is required. Strange, why? However, there are repeated confirmation of the effectiveness of this patch for ASUS boards, maybe for others. In some DSDT already has such a check, in this case such a fix should be disabled. If in the config SuspendOverride = true, then this fix will be extended to arguments 3 and 4. That is, leaving to sleep (Suspend). On the other hand, if HaltEnabler = true, then this patch is probably no longer needed.

AddMCHC bit (3):
Such a device of class 0x060000 is usually absent from the DSDT, but for some chipsets, this device is serviceable, and therefore it must be prescribed to properly wire the PCI bus power management. Question about the need for a patch solved experimentally. Another experience, this device was needed on a mother with a chipset Z77, otherwise kernel panic at the initial stage of launch. Conversely, on the G41M (ICH7) chipset this fix causes panic. Unfortunately, there is no general rule in sight.

FixHPET bit (4):
As already mentioned, this is the main fix needed.
Thus, the minimum required DSDT patch mask looks like 0x0010

FakeLPC bit (5):
Replaces the DeviceID of the LPC controller so that the AppleLPC kext clings to it. Needed for those cases where the chipset is not provided for OSX (for example ICH9). However, dear the list of Intel and NForce chipsets is so large that the need for such a patch is very rare. It is checked in the system whether the AppleLPC kernel has been loaded, if not, a patch is needed. Although, this also not yet a fact. It happens that the kext itself is unloaded from memory as unnecessary, although chipset supported.

FixIPIC bit (6):
Removes the interrupt from the IPIC device. This fix affects the operation of the Power key (pop-up window with options Reset, Sleep, Shutdown).

FixSBUS bit (7):
Adds SMBusController to the device tree, thereby removing the warning about it absence from the system log. And also creates the correct power control wiring tires, it also affects sleep.

FixDisplay bit (8):
Produces a number of video card patches. Injects properties and devices themselves, if there is none of them. Injects FakeID if ordered. Adds custom properties. The same fix adds an HDAU device for audio output over HDMI. If the FakeID parameter is specified, then it will be injected via the _DSM method. (1974)
Patches for all video cards, only for non-Intel. For embedded intellects other bit. FIX_INTELGFX_100000
An HDMI device (HDAU) is also added if needed.

FixIDE bit (9):
In the 10.6.1 system, there was a panic for the AppleIntelPIIXATA cache. Two solutions problems - using the corrected kext, or fix the device in the DSDT. And for more modern systems? Let it be, if there is such a controller.

FixSATA bit (10):
Fixes some problems with SATA, and removes the yellowness of disk icons in the system by mimicry under ICH6. Actually a controversial method, but without this fix I have a DVD discs cannot be played, and for a DVD, the drive must not be removable. Those. just a replacement icons are not an option!
There is an alternative, solved by adding a fix with the AppleAHCIport.kext kext.

See the chapter on patching kexts.

Page 120

And, accordingly, this bit can be omitted! One of those few bits that I
I recommend not to put it.

FixFirewire bit (11):
Adds the "fwhub" property to the Firewire controller, if present. If not, then nothing
will happen. You can bet if you don't know if you need to or not.

FixUSB bit (12):
Attempts to solve numerous USB problems. For XHCI controller, with
using native or patched IOUSBFamily, such a DSDT patch is indispensable.
The Apple driver specifically uses ACPI, and the DSDT script must be
correct. The prescription in the DSDT does not conflict with thongs.

FixLAN bit (13):
Injection of the "built-in" property for the network card is necessary for correct
work. Also a card model is injected - for cosmetics.

FixAirport bit (14):
Similar to LAN, in addition, the device itself is created, if not already registered in DSDT.
For some well-known models, the DeviceID is replaced with a supported one. AND
the airport is included without other patches.

FixHDA bit (15):
Correction of the description of the sound card in DSDT so that the native driver works
AppleHDA. Renaming AZAL -> HDEF is performed, layout-id is injected and
PinConfiguration.

FixRTC
Removes interrupt from _RTC device. It is necessary, and it is very strange that someone
refuses such a patch. If there is no interruption in the original, then it's okay
this patch won't do. However, the question arose about the need to edit the length of the region. To
there was no CMOS reset, you need to set the length to 2, but at the same time the phrase appears in the kernel log
like ... only single bank ...
I don't know what is wrong with this message, it can be excluded if the length is set to 8. But in
In this case, there will be a risk of getting a BIOS reset after sleep. Therefore, an additional
the key to enable this trick is:

```
<key> ACPI </key>
<dict>
        <key> DSDT </key>
        <dict>
                <key> Rtc8Allowed </key>
                <false />
        </dict>
```

true - the length of the region will remain 8 bytes, if there was one,
false - will be corrected by 2 bytes, which more reliably prevents the CMOS from being reset.
As investigated by vit9696, the length of the region must still be left 8, because it is needed for
save the hibernation key. And the fix itself is useful. Well, on desktops, hibernation is not
is needed, so he might think about resetting the CMOS.

FixTMR

Removes the interrupt from the _TMR timer in the same way. It is deprecated and not used by Mac.

Page 121

AddIMEI

Required patch for SandyBridge and above, adding a device
IMEI to the device tree, if it was not already there.

FixIntelGfx

The Intel integrated graphics patch is separate from the rest of the graphics cards, i.e.
you can put injection for Intel and not for Nvidia.

FixWAK

Adds Return to the _WAK method. He has to be, but for some reason often DSDT it is not
contain. Apparently the authors adhered to some other standards. Anyway this
the fix is completely safe.

DeleteUnused

Removes unused floppy devices. Why bother? On the
in fact, CRT and DVI devices are also removed here - an absolutely necessary condition
running IntelX3100 on Dell laptops. Otherwise black screen, tested by hundreds
users.

FixADP1

Adjusts the ADP1 device (power supply) as needed for proper sleep
laptop when plugged into or unplugged.

AddPNLF

Inserts PNLF (Backlight) device, which is necessary for correct control
screen brightness, and, oddly enough, helps to solve the problem with sleep, including for
desktop.

FixS3D

Likewise, this patch solves the problem with sleep.

FixACST

Some DSDTs contain a device or method or variable named ACST,
but this name is used by MacOSX 10.8+ for c-state management. Completely implicit
conflict with very obscure behavior. This fix renames all occurrences of such
name into something safe.

Can't figure out how you can ignore this patch series ?! Don't you need it well
working computer?

AddHDMI

Adds HDAU device to DSDT corresponding to HDMI output on ATI graphics card
or Nvidia. It is clear that since the card was bought separately from the motherboard, then in
native DSDT simply does not have such a device. In addition, the device is injected
property hda-gfx = onboard-1 or onboard-2 as appropriate:

-1 if UseIntelHDMI = false
-2 if there is an Intel port that occupied port 1.

Page 122

FixRegions

This is a very special patch. If the rest of the patches were meant to be edited BIOS.aml, to create a good DSDT out of nothing, then this fix is intended for final alignment of a well-done custom DSDT.aml, and for BIOS.aml it useless. The point is this.

In DSDT there are regions that have their own addresses, for example:

OperationRegion (GNVS , SystemMemory , 0xDE6A5E18 , 0x01CD)
The problem is that the address of this region is created by BIOS dynamically, and it can be different from download to download. This was primarily noticed when changing the total amount of memory, then when changing the BIOS settings, and on my computer even depends on the download history, for example, on the amount of occupied NVRAM. It is clear that in custom DSDT.aml this number is fixed, which means it may not correspond to the truth. The simplest observation is lack of sleep. After correcting the region, the dream appears, but until the next offset.

This fix fixes all regions in the custom DSDT to the values in the BIOS DSDT, and so way mask

```
<key> Fixes </key>
<dict>
    <key> FixRegions </key>
    <true />
</dict>
```

is a mask enough if you have a well-done DSDT with all your necessary fixes.

There is one more patch, but it does not concern DSDT, but in general all ACPI tables, and occupies here the place is illegal.

FixHeaders

It will check the headers of not only DSDT, but all ACPI tables in general, solving the problem of Chinese characters in the names of tables that MacOS does not tolerate, immediately panicking. Whether you have a problem with tables or not, it's safer to enable this fix.

Selecting patches

How to choose which patches are needed, which are harmless, and which are dangerous? Well computer you will not ruin in any way. All this happens only in RAM, and will forgotten after reboot. You can test a set of fixes by correcting the mask in the graphical menu, and saving the result by pressing F5 - "Save DSDT-xxxx.aml, adjusted to the current mask ".
You can try and load with the current mask. So that the real one does not interfere patched DSDT already present in the system can be specified in the menu DSDT name: BIOS.aml
Not finding such a file, the system will take the OEM DSDT from the BIOS and fix it on it, according to the set mask. If unsuccessful after restarting the computer, the current the settings will be lost and the default settings you have will take effect workable.

Mask 0xFFFFFFFF corresponds to enabling all fixes, and if the OS after that will be loaded, the work of programmers is not wasted. From the description above, you have already realized that

some fixes you just do not need (for example WIFI). Since revision 1992 there have been work to prevent panic attacks on a dual patch, so don't be afraid to ask unnecessary

Page 123

bits. Two fixes so far I would not recommend using: FIX_SATA, bit 0x0400, it is better to use a binary patch of the kext, and FIX_SHUTDOWN, bit 0x04, because instead, setting HaltEnabler = true almost always works, which works more correctly. Also a dangerous patch AddMCHC_0008, someone definitely needs it, but someone is categorically contraindicated.

To see how DSDT patches affected the result, let's say you couldn't boot, you can write the following key in the config, in the ACPI section:

```
<key> DSDT </key>
<dict>
    <key> Debug </key>
      <true />
```

In this case, before starting the system, two files will be saved to the disk in the folder / EFI / CLOVER / ACPI / origin /
DSDT-or.aml
DSDT-pa15.aml
or igin is your DSDT loaded from disk, or taken from BIOS, before applying patches.
pa tched - after applying patches.
Since you could not boot, you will try again and again, and patched files will be numbered sequentially without overwriting old information. 15 in this case - this is the 15th attempt to download, apparently successful, you need to look at what the problem was 14th try.
Still, I recommend avoiding a double patch. A situation may also arise when the double patch occurs because the _DSM method is already present in the OEM DSDT when we want to inject our own. So you need to set the bits in the mask DropOEM_DSM. See chapter "Configuring Hardware" → ACPI → DSDT → DropOEM_DSM. No, you need to rename all OEM DSMs.

Manual DSDT dressing

Prerequisites

You need to have some kind of computer with some kind of operating system so that you can was engaged in editing text files. Or are you already on this computer installed macOS somehow, and now you want to improve DSDT.

There is a command line compiler iasl for any operating system. Including for Windows, iasl.exe is run on the command line like in a Mac, has the same functionality, and gives the same results. Editing texts in Windows is inconvenient, notepad has no backlight syntax and line numbering, it's better to take Notepad +. Mac is full of options, both Xcode and BBEdit, and others. The new version of the compiler for Mac can be taken hereOptimization Dsdt. Newest Compiler.
For Windows on acpica.org Windows Binary Tools
You also need to stock up on the description of the ADCI language ACPI_6_2.pdf. Better to take a new one version, since you need to deal with the DSDT that the fresh BIOS slipped you.

Create a blank

First, make yourself a USB flash drive with Clover, bootable, even without a system. Important, so that it is in FAT32 format. We boot on this computer from this flash drive to

Clover interface. Press the "O" key (Latin letter Oy) or select the icon in the menu
Options. We go to the ACPI-> section there we find DSDT Name: and enter BIOS.aml. it

Page 124

exactly the DSDT that you could get in Windows through Aida. Going down the menu
and select DSDT fixes -> there you can check the boxes, with the keyboard or the mouse. We put almost
all the checkboxes that do not cause rejection in you. For example, you don't need Firewire, Airport,
IDE if you know you don't have such devices. Everything unfamiliar is better placed.
Return from this submenu, and press the keys F2, F4, F5 in sequence.

You can turn off the subject, and carry the flash drive to the computer on which you will
edit files. The files of interest to us are located on the USB flash drive in folders

EFI \ CLOVER \ ACPI \ origin
EFI \ CLOVER \ misc

We copy these folders to our working folder on the computer where iasl is. At least with Mac, at least with
Windows. The instructions are the same, except for the subtleties like slash slashes.

Decompilation

The resulting DSDT.aml is a binary file, its text editor is not
look. I strongly advise against specialized editors maciASL,
DSDTEditor and the like, since the purpose of the topic is not that a good uncle rehab is for you
he will do everything, but to see for himself what is happening and how.

Run the command line: cmd.exe in Windows, Terminal.app in the Mac, what's there in Linux
not in the know, like bash is called.

```
> cd WorkingDirectory \ origin
```

that is, the transition to the very copy of the folder that was removed from the flash drive. Put in the same folder
iasl.exe if you are on Windows, or install iasl on the system if you are on Mac

```
$ sudo cp ~ / Downloads / iasl / usr / local / bin
```

Now, finally, you can decompile, that is, get the DSDT in a readable language.

```
$ iasl -da SSDT * .aml DSDT - *. aml
```

The -da option does the magic, it looks for characters in all files at once, so in the end
there should be no unknown characters.

We decompile the DSDT that Clover patched for us, and thus we already have
a lot of useful fixes. Similarly, you can decompile the original file

```
$ iasl -da SSDT * .aml DSDT.aml
```

And to be able to compare what was and what became. And we got a file with a long
DSDT-1234567.dsl, you will have different numbers and letters. This is the original blank,
which you want to rename to DSDT.dsl, and the original one to DSDT-origin.dsl, edit
and compile in an endless iterative process:

1. Editing with a text editor.
2. Compile, get DSDT.aml

Page 125

compile command
$ iasl -ta DSDT.dsl
3. We test, that is, put it in the EFI \ CLOVER \ ACPI \ patched folder and start the system.
4. Returning to point 1.

Unfortunately, decompilation may end with the following error
Code:

```
Input file SSDT-10x.aml, Length 0x37F (895) bytes
ACPI: SSDT 0x0000000000000000 00037F (v02 PmRef Cpu0Cst
00003001 INTL 20120913)
Pass 1 parse of [SSDT]
ACPI Error: [C3ST] Namespace lookup failure, AE_ALREADY_EXISTS
(20160729 / dswload-462)
ACPI Exception: AE_ALREADY_EXISTS, During name lookup / catalog
(20160729 / psobject-310)
Could not parse external ACPI tables, AE_ALREADY_EXISTS
```

This means that the description of the C3ST symbol was found in two different SSDTs, the last one is this is SSDT-10x.
In my case, it turned out to be similar to SSDT-5x, differing in that 5x is ACPI1.0, and 10x is ACPI2.0. And the names are the same! That's what it is pouring in my kernel log, something like this. it BIOS error!
There are also two identical tables in the laptop. I traced, they really both present in BIOS, this is not my fault.
What to do? Before compilation, remove the backup, and in real work, do it in the config Clover drops unnecessary tables. In case of duplicates, both will fly away.

What to fix

1. Syntax errors made by the manufacturer of this computer
2. Conceptual errors
3. Tricks found on the internet. About this point, everyone thinks that it is the only thing to do. No, the first two points are also important.

Syntax errors

We look at the **DSDT** file **itself**.dsl and in the file itself we see the first problems

```
External (_SB_.PCI0.PEG0.VID_.LCD_, UnknownObj)
```

This means that somewhere in the text there is a reference to an object, but the object itself is nowhere to be found. Are looking for. is he available in SSDT-7. It turns out that it was not exported from there. It is for this reason that the this laptop is offered to include this **SSDT** inside the general DSDT. Simple copying text from that SSDT from the first curly brace to the last to the tail of DSDT.dsl before the last parenthesis.

Second error with missing characters
External (HNOT, MethodObj) // Warning: Unknown method, guessing 1 arguments

Page 126

Search reveals that the method is mentioned in such a construction

Code:

```
If (CondRefOf (HNOT))
 {
    HNOT (Arg0)
 }
 Else
 {
    Notify (GFX0, 0x80) // Status Change
 }
```

Then everything is in order, if the method is undefined, then it is not executed. I don't understand how then compiles? Better to remove this, leave only Notify ...

I start compiling the file I just received (I renamed DSDT-1F2C3B4D.dsl to simpler DSDT1.dsl as first try)

```
$ iasl -ta DSDT1.dsl
1 errors, 14 warnings, 91 remarks, 109 optimisations
```

The error must be corrected, otherwise there is no result.
Warnings also need to be corrected, considering why the compiler swears.
This is still a good situation, I have seen hundreds of errors on the first compilation.

In this case, the error is not critical, in the line
Name (_HID, "* pnp0c14")
the format of the character string is invalid, corrected by science like this:
Name (_HID, EisaId ("PNP0C14") / * Windows Management Instrumentation Device * /)
You need to fix it, otherwise we will not compile, but this does not affect the work, this is a Windows lotion.

Warnings are actually more critical, here are some sample fixes.
- what happened
+ what did

```
- CreateDWordField (BUF0, \ _SB.PCI0._Y0F._LEN, MSLN) // _LEN: Length
+ CreateQWordField (BUF0, \ _SB.PCI0._Y0F._LEN, MSLN) // _LEN: Length
```

The hint was in the compilation log
ResourceTag larger then field (size mismatch tag 64bit, Field 32 bit)
The rule is simple, tag is what you need, field is what you need to fix
tag = 1 => CreateBitField
tag = 8 => CreateByteField
tag = 16 => CreateWordField
tag = 32 => CreateDWordField
tag = 64 => CreateQWordField

It is necessary to fix it so that in real work the sent value does not run into the neighboring one

Page 127

field. You can't catch such a glitch.

Such a varning
Not all control path return a value
Logic error, the method must return something, but it turns out that in some cases
will return nothing. AND? Mac, of course, will not fall, but do not expect adequacy either.
And what to write there? Return (Zero) or Return (Local0)?
To calm down the compiler, this will do, but in general you need to look at the logic.

A similar glitch
Reserved method should not return a value
The code is like this
Code:

```
        Method (_SRS, 1, Serialized) // _SRS: Set Resource
Settings
        {
            Return (BUF2) / * \ _SB_.PCI0.A_CC.BUF2 * /
        }
```

Open the PDF, mentioned above, the ADCI specification, find the _SRS method by searching, and read what he should do. Returns are not provided there.

Therefore, we redo as follows
Code:

```
        Method (_SRS, 1, Serialized) // _SRS: Set Resource
Settings
        {
           BUF2 = Arg0
        }
```

So it seems more logical.

Well, well-known varning
Name (_T_0, Zero)
Use of compiler reserved name _T_0

The new iasl compiler understands this construct just fine and you don't need anything
do. The old compiler required renaming to TT_0

Conceptual errors

Sorry, but here you have to be at least a little programmer to catch them. Hints
there is nowhere to wait.

In the original **DSDT** see
External (CFGD, IntObj)
And we find this variable in **SSDT** CpuPm. And then it's time to remember that we are this table

drop, along with this variable!
You need to copy it to DSDT. As well as others that might come in handy.
Code:

Page 128

```
+                Name (CFGD, 0x0066F6FF)
+                Name (PDC0,0x80000000)
+                Name (PDC1, 0x80000000)
+                Name (PDC2, 0x80000000)
+                Name (PDC3, 0x80000000)
+                Name (PDC4, 0x80000000)
+                Name (PDC5, 0x80000000)
+                Name (PDC6, 0x80000000)
+                Name (PDC7, 0x80000000)
+                Name (SDTL, Zero)
```

A common mistake is the OEM methods of _DSM.
This is not a mistake by the manufacturer, this is what he wrote for Windows, but we have a hackintosh.
Example
Code:

```
-                Method (_DSM, 4, Serialized) // _DSM: Device-
Specific Method
                 {
-                                        Name (_T_0, Zero) // _T_x:
Emitted by ASL Compiler
                    If (Arg0 == ToUUID ("a5fc708f-8775-4ba6-bd0c-
ba90a1ec72f8 "))
                    {
                        While (One)
                        {
```

See the UUID? This is from the Windows registry, there is no such thing in Mac., And nothing is executed will be.
It would be half a disaster, but if it is normal for Windows to have _DSM for both the device and its bridge, then in Mac it causes a crash.
We kill all strangers _DSM! Simple option
Code:

```
- Method (_DSM, 4, Serialized) // _DSM: Device-Specific
Method
+ Method (ZDSM, 4, Serialized)
```

The method itself has survived, but no one will turn to it, and there will be no panic.

Here is such a curious piece
Code:

```
            OperationRegion (DXHC, SystemMemory, 0xFED1F418, 0x04)
              XHCD, 1
            }

          If ((OSYS <0x07D6) && (OSYS> 0x03E8))
```

```
{
    XHCD = One
```


Page 129

```
    Notify (XHC, Zero) // Bus Check
}
```

Looking at the address, I realized that this is a Function Disable Bit.
The point of the operation is that for systems lower than Windows Vista, disable USB3.
In my opinion, this piece should be cut out altogether.

But this is already a mess of writers
Code:

```
If ((OSYS> 0x07D0) || (OSYS <0x07D6))
```

Ranges add up and cover any value at all
Rather there should be
Code:

```
If ((OSYS> 0x07D0) && (OSYS <0x07D6))
```

Then the ranges overlap.
But then you need to look at what is in If and what is in Else.
I ended up with something more correct in Then, so the original construction
Code:

```
If ((OSYS> 0x07D0) || (OSYS <0x07D6))
{
    Notify (PCI0, Arg1)
}
Else
{
   Notify (GFX0, Arg1)
}
```

I've reduced it to just one operator
Notify (PCI0, Arg1)
The point is, this If checks if the system is WindowsXP and does what in Then.
for systems like Windows7,8,10 does Else.
What is the difference? Optimus works on new systems.
In **macOS,** we need type 1 notification, for the entire bus. And similarly you need to look elsewhere DSDT.

A small detective story on the topic, maybe it will lead someone to the right idea.
So the challenge.
The laptop sleeps and wakes up if not plugged into an outlet. But then the battery runs out.
If he is in an outlet, then immediately, after a second asleep, he wakes up. What's the matter?
It is in the DSDT!
So, we are looking for what distinguishes **ACPI** state in an outlet and not. Device
Code:

```
Device (AC)
{
    Name (_HID, "ACPI0003" / * Power Source Device * /) //
_HID: Hardware ID
```

The decompiler helpfully prompted us that this is the power supply.
The _PSR method determines whether this device is a power source or is resting.

Page 130

Code:

```
Method (_PSR, 0, NotSerialized) // _PSR: Power Source
{
    Local0 = ECG5 ()
    Local0 & = One
    If (Local0! = PWRS)
    {
        PWRS = Local0
        PNOT ()
    }

    Return (Local0)
}
```

You can read about it in the book ACPIspec.pdf, any year of publication

An Integer containing the power source status
0 - Off-line (not on AC power)
1 - On-line

That is, in my case, some hardware ECG5 () method produces a certain number in which the bit 0 means plugged in.
And stores it in the PWRS variable if the value is different there.
Now, throughout the code, we look at where PWRS is mentioned, and we find in the _PTS (Prepare To Sleep), in the one that is responsible for falling asleep.

Code:

```
Method (_PTS, 1, NotSerialized) // _PTS: Prepare To Sleep
{
    P80D = Zero
    P8XH (Zero, Arg0)
    PTS (Arg0)
    If (AOAC & One) {}
    If (Arg0 == 0x03)
    {
        If (PWRS == Zero)
        {
            \ _SB.PCI0.XHC.PMEB = Zero
            \ _SB.PCI0.EHC1.PMEB = Zero
            \ _SB.PCI0.EHC2.PMEB = Zero
            If (\ _SB.PCI0.XHC.PMST == One)
            {
                \ _SB.PCI0.XHC.PMST = One
            }
```

```
If (\ _SB.PCI0.EHC1.PMST == One)
```


Page 131

```
        {
            \ _SB.PCI0.EHC1.PMST = One
        }

        If (\ _SB.PCI0.EHC2.PMST == One)
        {
            \ _SB.PCI0.EHC2.PMST = One
        }
    }
```

Interesting here!
Condition: if the device is not plugged into the outlet, then do something in YUSB2 and YUSB3.
And if stuck, then nothing is needed.
In the unexpected awakening log, we see XHC, EHC2.
That is, the creators of this DSDT believe that a laptop can be plugged into an outlet only during working hours. Otherwise, pull it out.
Removing this condition If (PWRS == Zero) I got a solution to my problem. I have now the laptop sleeps and wakes up without being disturbed by the power adapter.

Good luck in creating the minimum correct DSDT!

Native speedstep

It is more correct to say Power and Processor Frequency Management (PPiP), in English it will be EIST - Enhanced Intel Speedstep Technology, hence the Russian word "Speedstep".

Actually, this topic is not so much for the bootloader as for setting up HakOS in general, but since Clover takes some steps, we will describe it in a separate chapter.
Clover does not do everything that needs to be done, it takes a little work with his hands.

What is it for? The meaning is this: the processor in idle works on minimum frequency with minimum voltage, under load speed and voltage grow. (And why is the voltage? But because the pulse front becomes steeper, and therefore picks up the level faster, quickly passes from state 0 to state 1).

PPiChP can be done in two ways: with a specialized utility, such as CoolBukController, or GenericCPUPM, or understand native speedstep, good for MacOS he knows how to do it.

The following steps are required:

1. The HPET must be corrected in the DSDT, which is being successfully done Clover with mask 0x0010.
2. There must be a correct processor section, which is done by Clover when key GeneratePStates = Yes (and in addition DropSsdt)
3. MacModel should be chosen as a sample of your SMBIOS for which EIST technology is provided. It turns out not for all models. TO For example, for the MacBook1,1 model, speedstep will not work, but for MacBook5,1 - will be.

Point 3 can be rethought as follows: let the model be more

is similar in configuration to the real one, but we will fix its platform-plist so that the speedstep appeared.

Page 132

Each model has its own plist, see here
System / Library / Extensions / IOPlatformPluginFamily.kext / Contents / PlugIns /
ACPI_SMC_PlatformPlugin.kext / Contents / Resources / MacBook5_1.plist
We look at the similarities and differences between different plists, and correct ours in the right direction.

Con2gArray

```
<key> ConfigArray </key>
<array>
    <dict>
        <key> WWEN </key>
        <true />
        <key> model </key>
        <string> MacBook4,1 </string>
        <key> restart-action </key>
        <dict>
            <key> cpu-p-state </key>
            <integer> 0 </integer>
        </dict>
    </dict>
</array>
```

This restart-action key means which P-State the CPU should fall on during restart. Only with this key, sleep and shutdown of the computer started!

CtrlLoopArray

```
<key> CtrlLoopArray </key>
<array>
    <dict>
        <key> Description </key>
        <string> SMC_CPU_Control_Loop </string>
...
        <key> PLimitDict </key>
        <dict>
            <key> MacBook4,1 </key>
            <integer> 0 </integer>
        </dict>
```

This PLimitDict key was already mentioned in generating P-states. Again: this is a limitation maximum processor speed. 0 - maximum speed, 1 - one step lower maximum. If this key is missing here, then the processor will get stuck at minimum frequency.

CStateDict

```
<key> CStateDict </key>
<dict>
    <key> MacBook4,1 </key>
    <string> CSD3 </string>
    <key> CSD3 </key>
    <dict>
        <key> C6 </key>
        <dict>
            <key> enable </key>
            <true />
```

Practice shows that it is better to remove the entire section for the control to work. powered by PState, not CState. Although, as anyone, maybe this option is worth work out.
Symptom - the processor is at the maximum frequency, does not fall. After deleting a section begins to vary the frequency.

With Skylake processors, there is a new technique called Processor Frequency Control to they call it SpeedShift. I haven't figured it out yet, preliminary results:

Page 133

In the CPU section we register two keys (the inventor is goodwin_c)

```
<key> CPU </key>
<dict>
    <key> HWPEnable </key>
    <true />
    <key> HWPValue </key>
    <string> 0x30002a01 </string>
```

The first key puts 1 in the MSR register 0x770.
The second key writes the specified value to the MSR register 0x774.

Sleep problem

What is the problem of sleep? When all of the above is done, the computer will lie down sleep and wake up like an obedient child. The most important thing you need to do this, Clover already done: corrected FADT and FACS. It remains only to fix the DSDT, start speedstep, use only good cakes, and you will be happy.

Any device can interfere with a good sleep, including an uninstalled PCI device, or partially wound. For example, AppleHDA. It categorically interferes with sleep NullCPUPM.kext. You may not need speedstep, but you should do the HPET patch this way, to start native AppleCPUPM, and zero was not needed. And for those who do not have a processor lets you use AppleCPUPM, you can try SleepEnabler - sometimes it helps, or a patched kernel.

DSDT has a group of _GPE methods with notifications for each device that needs to be awakened after sleep. The computer itself woke up, but it may turn out that video / network / sound / mouse forgot to wake up. Watch DSDT, teach theory how to do it.

There was also a problem with sleeping when UEFI booting into a 10.8 system. Fixed **issue in revision 1942.** Change in OsxAptioFixDxe driver: sleep / wake up works even in 10.8, even with CsmVideoDxe. Today it is the ProtectMemoryRegions pick.

Next trick for UEFI boot

```
<key> ACPI </key>
<dict>
    <key> HaltEnabler </key>
    <true />
```

This corrects the state of the chipset, which was incorrectly initialized by the UEFI-bios. For legacy boot CloverEFI does everything correctly, there was no such problem. Symptoms - does not go to sleep, the screen goes out, but the fans do not.

And one more trick.

Page 134

Without checking the "Start up automatically ..." checkbox, I could not achieve waking up after sleep.

Hibernate

It is also called deep sleep, but in general it looks more like clinical death. The essence in the fact that the system seems to be sent to sleep, but it saves its state in a file sleepimage, and just turns off the computer, so that later, when turned on, it just restored his condition and woke up. For laptops, this is critical. When during normal sleep, the computer does not turn off completely, and continues to use electricity, although less than in working condition, is still noticeable that the battery is completely discharged during sleep. When hibernating, the battery does not is used and discharges only due to its leaks.

Once upon a time, hibernate worked with Chameleon, but only up to version 10.7.2 (sort of), then, due to some changes in the system, this technique stopped working. In Clover managed to make a hibernate, but under the following conditions:

- Loading either CloverEFI (legacy), or InsydeEFI, or Phoenix 2.3.1. In the current revision 2915+, the OsxAptioFix2Drv driver appeared, which allows you to have hibernate with AMI UEFI on system 10.9.1+. But the 10.7.5 system does not boot at all from this option. And God bless her!
- The system is either 10.7.5 or 10.9.1+. Other systems do not wake up yet.
- Fashion 21 or better 29 or even 57, although Apple insists on 25.
  sudo pmset -a hibernatemode 29
  Starting with the Captain, it is impossible to put 29 like that. The trick is as follows:
  - copy /Library/Preferences/com.apple.PowerManagement.511CE201-1000-4000-9999-120361221216.plist to the Desktop. There are several such files, choose the one which matches the UUID in the System Information. Editing this file, to bring there 29

<key> Hibernate Mode </key>
<integer> 29 </integer>

Page 135

And copy the edited file using the terminal back
sudo cp ~ / Desktop / com.apple.PowerManagement.511CE201-1000-4000-9999-120361221216.plist / Library / Preferences /
After that, a reboot is required, only then the change will take effect.

- If you have a real, iron NVRAM working, then you can make mod 25. And then we prescribe in the config
<key> Boot </key>
<dict>
<key> StrictHibernate </key>
<true />
- For system 10.13 and higher, you need to enable
<key> RtcHibernateAware </key>
<true />
because the encryption key can be registered in CMOS, and because Clover should generate additional NVRAM variables. (about the last one more need to think.

It works like this:
1. We put the fashion 29 (or 25), if it is not already set. There is no need to repeat.
2. We send the computer to sleep either through the menu, or by closing the lid, or by pressing the button power supply, if configured. After a minute, the computer will turn off completely.
3. To wake up, just turn it on as usual. We see the BIOS splash screen, enter Clover menu. And here we see that our system is marked (hibernated)

Whereas other systems are not. When you click on this icon, the system boots from image, a few seconds, progress is visible below, and the system turns on. It is much faster than normal system boot, especially for laptops, and especially when a large number of open applications.
It should be noted that if the file system of the volume has been modified after hibernate, for example from a system booted from the second partition, then a serious the danger of damage to the file system, because the sleeping system has a cache on the other

structure. For system 10.9, this complexity is automatically overcome by comparing modification dates. In system 10.7.5 this does not work, please check the correctness manually.

Page 136

You can cancel waking up from image by pressing the space bar on this icon, and by selecting "Cancel hibernation".
If the system continues to think that it needs to wake up, then it will have to be prescribed to config:

```
<key> Boot </key>
<dict>
   <key> NeverHibernate </key>
   <true />
```

How to use

First meeting

First, boot into the Clover GUI and try living here first. press different keys, move the mouse.

The top row of buttons are the intended operating systems that you can download. There are two of them in this picture, Lion and Windows, as you can see from the pictures. Really, I I remind you that Clover is not a bootloader of operating systems, he is a manager their own bootloaders. Namely, for Mac, the bootloader is /System/Library/CoreServices/boot.efi. For Windows, in this case, /EFI/microsoft/boot/bootmgfw.efi

The bottom row of buttons - additional functions: command line (Shell), Options menu, information about the bootloader and environment, restart and exit from Clover. Exit where? Back Wednesday EFI, UEFI BIOS or CloverEFI, respectively.

Page 137

It is very useful for initial acquaintance to press F1 (who would have thought ?!). If in the config indicated

```
<key> GUI </key>
<dict>
    <key> Language </key>
    <string> ru: 0 </string>
```

then the certificate will be in Russian
The command line is something like DOS, with the ability to copy and delete files. How and why is beyond the scope of this book. This is Shell.efi with its help.

The Options menu allows you to change some settings that will affect the progress system boot.

Some of them are set in the config.plist file, but it could well turn out that it says wrong, and in order not to edit this file yet, the settings can be changed already at running Clover.

What exactly to change and why, it depends on the task, what exactly needs to be obtained. Very annoying requests like "I have a hdd western digital and memory corsair, help me set up config ". The config is configured not by hardware, but by the result. If you can't right away boot, try to determine what the problem may be, and fix it
menu.

Several special download methods can be obtained by pressing the spacebar on the icon

bootloader. (again, ENTER - loading the system, SPACE - entering the additional menu downloads).

Page 138

In particular, this is the only way to prevent the exit from the hibernate if it is undesirable.

Why is Clover starting so slowly?

Some can not even wait for the launch, they report that Clover is not working.
Let's take a closer look at this issue.

1. **Debug.log is exposed** .

```
<key> Boot </key>
<dict>
    <key> Debug </key>
    <true />
```

Yes, this is very valuable information to find out what is not working and why. But keep in mind, that it slows down the launch very much. If Clover is installed on YUSB (for a sample, so to speak, we do we believe that Clover is capable of working at all?!), then launching with debug can last 10 minutes. Yes really. This happens because the debug log opens, closes and is overwritten on the USB stick on each line. This ensures that you receive information on the success of the launch even with a forced cut, so if you let's go to debug, wait! Or put <false /> for the first try.

2. **Too many disks, partitions and files on them** .

Clover, in order to compose the startup menu, must scan all disks, all partitions, and everything

files on them to find which systems you can suggest to run. Well wait! Or
cancel the scan,

```
<key> GUI </key>
```


Page 139

```
<dict>
    <key> Scan </key>
    <dict>
        <key> Entries </key>
        <false />
        <key> Legacy </key>
        <false />
        <key> Tool </key>
        <false />
    </dict>
```

and compose the menu manually

```
<key> GUI </key>
<dict>
    <key> Custom </key>
    <dict>
        <key> Entries </key>
        <array>
            <dict>
```

True, it takes some mental effort to figure out what to write there. Or
leave the bootloader scanning for now to boot somewhere.

```
        <key> Scan </key>
        <dict>
            <key> Entries </key>
            <true />
```

3. A **huge Windows or Linux partition, and not even one** .
Scanning a Windows partition is performed with an NTFS driver. And in this section,
typically a million files, and among them we're going to look for bootmgr.efi.
I would recommend installing Windows so that this file is on the EFI partition, and
remove the NTFS.efi driver altogether, and thus do not scan Windows partitions.
Likewise with the Linux partition and the VBoxExt2.efi driver.
4. **Too many drivers in the / EFI / CLOVER / drivers / folder**
I foresee a situation where people will start producing their own drivers of this type, and there will be
willing to try them. At this point, consider if you need extra drivers
keyboard, mouse, LowMemoryFix ... The fact is that if you start with YUSB, then reading
all of these files may take time.
5. **Unsupported mouse** .
Unfortunately, not all mice are supported by the EFI driver we have. Bad
the mouse may behave incorrectly on the screen, or give hard brakes.
Ban it for testing, or even forever if it's wrong

```
        <key> Mouse </key>
        <dict>
            <key> Enabled </key>
            <false />
            <key> Speed </key>
            <integer> 0 </integer>
        </dict>
```

6. **Slow HFS + driver** .
The official Clover comes with the VboxHFS.efi driver, which is licensed
clean and understands links, but works slower than Apple's HFSplus.efi. Download where-
Anything this unofficial but fast driver, and put it in a folder
/ EFI / CLOVER / drivers / UEFI /. This also applies to legacy (.. **drivers / BIOS /**) boot.
7. A **monstrously beautiful design theme has been selected** .
The richer the theme is in colors and animation, the longer it takes to load. Choose built-in
topic, it is the fastest

```
<key> GUI </key>
<dict>
```

```
<key> Theme </key>
<string> embedded </string>
```


Page 140

8. **Use the newest Clover** .
Whatever kind advisers whisper, but the new version is better than the old one, and it has been fixed bugs, in particular, causing Clover to freeze. And since revision 3063 messages directly on the screen about the loading process.

The lettering is crooked because a good font hasn't been loaded yet and this has caused quite a few unflattering responses "How to remove the inscriptions on the screen?". Put timeout = 0 and no labels will be. And to begin with, they are very useful, many new users realized that Clover is all- it works, just slowly. This chapter is written for them.
Or so

```
<key> Boot </key>
<dict>
    <key> NoEarlyProgress </key>
    <true />
```

Next, we will consider some tricks, special patches and methods of work, collected by the principle of a reverse dictionary. There is a problem -> here is the solution.

Analyzing Debug.log / Preboot.log

What for? Clover will give you information about the hardware that you will not see from the characteristics on the site, and will also talk about how he deals with the configuration before start the system.

How do I get it? Let me make a reservation right away that this is the same thing, with a difference in success and speed. To get the preboot.log, go to the Clover Shell (GUI) and press the F2 key. File preboot.log will be saved in the Clover folder EFI / CLOVER / misc /. File ends with a phrase Enter GUI, which is logical. However, if you boot the system, the file will end with running the system itself, that is, everything that is possible.
If Clover hangs, then debug.log will help. To get it, put it in the config

```
<key> Boot </key>
<dict>
    <key> Debug </key>
    <true />
```

In this case, the file is created line by line, and it is guaranteed that you find the point in it stop. Example:

222: 890 0: 001 DefaultIndex = -1 and MainMenu.EntryCount = 7
222: 892 0: 001 SetScreenResolution: 1366x768 - already set
0: 100 0: 100 MemLog inited, TSC freq: 2591578780
Khaki clover. Version 5.0, revision 5120 140
Moscow, 2020

Page 141

0: 100 0: 000 CPU was calibrated with ACPI PM Timer
Here we see that after the message about changing the screen resolution, a new log started. I.e the error must be looked for in Clover's text after this message, and before the next message, which was not. Tell these lines to the developer, he will search.

Let's analyze now what we see in the log.

MemLog inited, TSC freq: 2591583140
Here is the starting clock speed of the processor. According to the standard, the processor starts with one core at maximum non-turbo frequency. If this is not the case, ring the bells urgently! Work will not be. Such situations are rare, and we deal with each individually with Clover's special amendments. The figure is not round, and rightly so! It will only be round in utilities that perform rounding for ease of display.

Now is 14.7.2019, 08: 47: 7 (GMT)
Launch date and time GMT. In Moscow, the local time will be +3. Distinguishing the old log from the new.

0: 100 0: 000 Starting Clover revision: 5120 (master, commit dddceaae3) on American Megatrends EFI
Here and below we find out which version of Clover the user used, as well as his UEFI boot, as in this case, or legacy (it would be Clover EFI).

=== [Get Smbios] ====
Next comes the decoding of memory modules, as the BIOS recognized them. Usually correct, so we put in the config

```
<key> SMBIOS </key>
<dict>
    <key> Trust </key>
    <true />
```

And if you don't like the values, then false. Clover then himself will determine by reading SPD or XMP. In SMBIOS, values are also taken from reading SPD, according to algorithms BIOS. We have good algorithms, but universal, and in BIOS they are specific for this motherboard, so it's not a fact who is more accurate.

```
<key> Boot </key>
<dict>
    <key> XMPDetection </key>
    <string> -1 </string>
```

See the chapter on configuration.

Running on: 'Latitude E6430' with board '0H3MT5'
And this information is what your computer is called.

=== [GetCPUProperties] ====
Here is information about the processor, and CPUID, and its own name
BrandString = Intel (R) Core (TM) i5-3320M CPU @ 2.60GHz
And other important characteristics.
MSR 0xE2 before patch 1E008404
See the eight? This is a lock, a terrible sin for Mac. If possible,
Flash the BIOS to the unlocked one. If not, then it is quite possible to live by installing patches

```
<key> KernelAndKextPatches </key>
<dict>
    <key> KernelPm </key>
    <true />
    <key> AppleIntelCPUPM </key>
```

<true />
Clover, in theory, will do it automatically, but it's better if you do it manually.

Page 142

Turbo: 31/31/31/33
These numbers mean that 3300MHz frequency is achievable on one core, on two cores or more only 3100. Since all cores are included in the Mac, there is no need to make a claim that the processor in the Mac is not overclocked to the maximum possible frequency of 3300.
I, however, have one more line
The CPU not supported turbo
This is most likely the BIOS banned the use of turbo frequencies. We must look at the settings BIOS.

=== [GetDevices] ===
I love this section very much, it tells what PCI (PCIe) devices are found in computer, along with their addresses and DeviceID / VendorID. It helps a lot to find out can I find drivers for this device

```
0: 100 0: 000 PCI (00 | 00: 00.00): 8086 0154 class = 060000 MCH
0: 100 0: 000 PCI (00 | 00: 01.00): 8086 0151 class = 060400
0: 100 0: 000 PCI (00 | 00: 02.00): 8086 0166 class = 030000 VideoCard
0: 100 0: 000 - GFX: Model = Intel HD Graphics 4000 (Intel)
0: 100 0: 000 PCI (00 | 00: 14.00): 8086 1E31 class = 0C0330 USB3.0
0: 100 0: 000 PCI (00 | 00: 16.00): 8086 1E3A class = 078000 IMEI
0: 100 0: 000 PCI (00 | 00: 16.01): FFFF FFFF class = FFFFFF
0: 100 0: 000 PCI (00 | 00: 16.03): 8086 1E3D class = 070002 SerialPort
0: 100 0: 000 PCI (00 | 00: 19.00): 8086 1502 class = 020000 LAN
0: 100 0: 000 - LAN: 0 Vendor = Intel
0: 100 0: 000 PCI (00 | 00: 1A.00): 8086 1E2D class = 0C0320 USB2.0
0: 100 0: 000 PCI (00 | 00: 1B.00): 8086 1E20 class = 040300 HDA
0: 100 0: 000 PCI (00 | 00: 1C.00): 8086 1E10 class = 060400
0: 100 0: 000 PCI (00 | 00: 1C.01): 8086 1E12 class = 060400
0: 100 0: 000 PCI (00 | 03: 00.00): 14E4 4353 class = 028000 WiFi
0: 100 0: 000 - WIFI: Vendor = Broadcom
0: 100 0: 000 PCI (00 | 00: 1C.02): 8086 1E14 class = 060400
0: 100 0: 000 PCI (00 | 00: 1C.03): 8086 1E16 class = 060400
0: 100 0: 000 PCI (00 | 00: 1C.05): 8086 1E1A class = 060400
0: 100 0: 000 PCI (00 | 0C: 00.00): 1217 8221 class = 080501 SD-reader
0: 100 0: 000 PCI (00 | 00: 1D.00): 8086 1E26 class = 0C0320 USB2.0
0: 100 0: 000 PCI (00 | 00: 1F.00): 8086 1E55 class = 060100 LPC
0: 100 0: 000 PCI (00 | 00: 1F.02): 8086 1E03 class = 010601 SATA AHCI
0: 100 0: 000 PCI (00 | 00: 1F.03): 8086 1E22 class = 0C0500 SMBUS
0: 100 0: 000 PCI (00 | 00: 1F.06): FFFF FFFF class = FFFFFF
```

some devices are also commented.
FFFF means that the device is not connected, although it is somehow present.
The device class is Video, Audio, USB2.0, USB3.0, LAN, WiFi, and so on. All transcripts known from PCI specifications. I added the class decryptions in the right column, in the log they not. Class 060400 is a bridge, a slot where you can insert a device.
There are no USB devices here. Clover is not old enough to scan them.
And here's another moment, from someone else's log
- GFX: Model = GeForce GTX 760 family CE (Fermi)
What do we see ?! 760 must be Kepler! Why does Clover say she's Fermi?
Clover is right, he cannot be fooled by BIOS rewiring, information about the video card family is taken not from the BIOS, but from the functioning of the video core.

```
0: 128 0: 027 EFI \ CLOVER \ #. Plist not loaded with name from LoadOptions: Not Found
0: 143 0: 015 EFI \ CLOVER \ config.plist loaded: Success
```

This ability of Clover is practically not used by anyone. The thing is that in BIOS there is a place where to flash the name of the config.plist file, and thus you can directly in the BIOS choose with which config to load Clover. In this case, no such name was found, and therefore the standard config.plist is loaded. In general, you can use it if you remember

2014 instructions from Dmazar.
Below in the log there is another list

Page 143

```
0: 346 0: 003 === [Found config plists] ================================
0: 403 0: 057 - config0.plist
0: 403 0: 000 - config1.plist
0: 403 0: 000 - config.plist
```

This is already for the Clover menu, in order to change the config from there. But the Boot, GUI and KernelAndKextPatches will not change anymore. Late!

```
0: 143 0: 000 === [GetListOfThemes] ================================
0: 162 0: 018 - [00]: embedded
0: 176 0: 014 - [00]: random
0: 177 0: 001 - [00]: metal
0: 188 0: 010 - [01]: BGM
0: 223 0: 035 - [02]: CESIUM
0: 285 0: 061 - [03]: METAL @ 2X
0: 298 0: 012 - [04]: Clovy
0: 343 0: 044 - [05]: BOOTCAMP
```

This is understandable, a list of everything established. Is it just to make sure that the config is selected a topic that really is.

```
KextsToPatch: 13 requested
KernelToPatch: 1 requested
```

A small list of what is set in the config for the kernels and kernels patches. We look into a stranger config, and criticize why he made these or those patches.

```
0: 749 0: 002 === [LoadDrivers] ================================== =
0: 861 0: 111 Loading ApfsDriverLoader.efi status = Success
0: 883 0: 021 - driver needs connecting
0: 885 0: 002 Loading AudioDxe.efi status = Success
0: 893 0: 007 - driver needs connecting
0: 895 0: 002 Loading DataHubDxe.efi status = Success
0: 917 0: 022 Loading EnglishDxe.efi status = Success
0: 926 0: 009 Loading Fat.efi status = Success
0: 935 0: 008 - driver needs connecting
0: 937 0: 002 Loading FSInject.efi status = Success
0: 944 0: 007 Loading OsxAptioFix3Drv.efi status = Success
0: 952 0: 007 Loading SMCHelper.efi status = Success
0: 959 0: 007 Loading VBoxHfs.efi status = Success
0: 966 0: 007 - driver needs connecting
0: 968 0: 002 4 drivers needs connecting ...
0: 970 0: 002 PlatformDriverOverrideProtocol not found. Installing ... Success
0: 974 0: 004 APFS driver loaded
0: 978 0: 003 Searching for invalid DiskIo BY_DRIVER connects: not found, all ok
```

And the next reason for criticism is why the user loads these drivers, and why not others.

```
SetScreenResolution: 1366x768 - already set
```

I ordered this permission for my screen. And it was successfully displayed.
For those whose desire is at odds with reality, see the chapter on configuration.

```
4: 481 0: 002 === [GetMacAddress] ================================
4: 561 0: 080 MAC address of LAN # 0 = D4: BE: D9: 6C: 86: CD:
```

Clover can read the MAC address of almost any network card. We use this information to set its value to the ROM variable. On some UEFI BIOS it doesn't work, we are looking for other ways.

```
=== [ScanSPD] ====
```

Checking memory modules if we do not trust BIOS.

=== [GetAcpiTablesList] ====

Page 144

List of ACPI tables found in BIOS. Useful if you want to throw away some
(drop)
- [06]: SSDT CpuPm len = 2850
There is also an ID = CpuPm and a length of 2850. You can drop it this way or that, depending on uniqueness.

=== [GetUserSettings] ===
some selective information, what exactly is exposed in the config, is especially useful for reading other people's logs.

=== [ScanVolumes] ===
List of volumes with their addresses and UUIDs. A volume is either a partition or a whole disk. Helpful to see when "Clover can't see my section!"

=== [InitTheme] ====
Further information on the successful creation of a graphical interface with the selected theme. For example, I see
OSicon os_mav not parsed
that is, my chosen theme does not have a Mavericks icon, just an icon will be used Maca.
There is also information about the start sound, which depends on the theme.

6: 511 0: 002 === [Dump SMC keys from NVRAM] ==========================
6: 570 0: 059 found AppleSMC protocol
6: 584 0: 014 Registered 17 SMC keys
In most cases, SMC keys do not matter at the start. They are strictly necessary for FileVault2 and Hibernation. Provided by the SMChelper.efi driver and infrastructure Clover. (there is another version of VirtualSMC with its own infrastructure).
For the installation of the system, it seems, are not needed, but ... who knows!

=== [ScanLoader] ===
And here is already a list of what you can download from. As well as information, if any the system is in hibernation state.

=== [GetEfiBootDeviceFromNvram] ====
It depends on the success of this operation whether Clover autostart on timeout. We look See instructions in the corresponding chapter.
Success looks like this
Boot redirected to Entry 3. 'Boot macOS from HighHD'
That is, my system boots from the HighHD disk by timeout.

=== [StartLoader] ===
We start loading the system
GetOSVersion: 10.13.6 (17G7024)
To see which system the user is talking about loading.

The following is information about what Clover does before loading the selected system, which patches, which properties are generated, which kexts are loaded, and the last line unlock USB2.0 if needed.
USB EHCI Ownership for device 1E26 value = 1000001

In version 5120, the log continues with values from the kernel patch procedures and kexts. It is for

developers.

Page 145

Running OSX on unsupported hardware

Actually, the whole book is about THIS. I'll tell you partly here, starting from the question.

Unsupported BIOS. Still would! It is about Hackintosh that we are talking about. And in first of all, this is data in DMI, which contains the manufacturer's name (must be Apple inc.), model and serial number, numbers and letters in which are not random, they mean something, in particular the model and production date. In its simplest form, since the days of Netkas, the model was installed for everyone MacPro3,1, and a certain serial, one for all, that worked. Now Clover, having analyzed the hardware, offers dozens of options that are workable. Nevertheless, it is recommended to generate your own serials, or maybe take a model different from default models.

Unsupported processor . Yes, different versions of MacOS support different sets CPU and your processor might not be supported.
Here is a table for old systems:

CPU name	CPUID	10.4 10.5 . 8	10.6 . 3	10.6 . 8	10.7 . 2	10.7 . five	10.8 . five	10.9 . five
Yonah	0x0006E6	1 1 1 1 1 1 0 0						
Conroe	0x0006F2	1 1 1 1 1 1 1 1						
Penryn	0x010676	0 1 1 1 1 1 1 1						
Nehalem	0x0106A2	0 1 1 1 1 1 1 1						
Atom	0x0106C2	0 0 0 0 0 0 0 0						
XeonMP	0x0106D0	0 0 0 1 0 0 0 0						
Linnfield	0x0106E0	0 0 1 1 1 1 1 1						
Havendale	0x0106F0	0 0 1 1 1 1 1 1						
Clarkdale	0x020650	0 0 0 1 1 1 1 1						
AtomSandy	0x020660	0 0 0 0 0 0 0 0						
Lincroft	0x020670	0 0 0 0 0 0 0 0						
SandyBridge	0x0206A0	0 0 0 1 1 1 1 1						
Westmere	0x0206C0	0 0 0 1 1 1 1 1						
Jaketown	0x0206D0	0 0 0 1 1 1 1 1						
NehalemEx	0x0206E0	0 0 1 1 1 1 1 1						
WestmereEx	0x0206F0	0 0 0 1 1 1 1 1						
Atom2000	0x030660	0 0 0 0 0 0 0 0						
IvyBridge	0x0306A0	0 0 0 0 0 1 1 1						
Haswell	0x0306C0	0 0 0 0 0 0 1 1						
IvyBridgeE5	0x0306E0	0 0 0 0 0 0 0 1						
HaswellMB	0x0306F0	0 0 0 0 0 0 1 1						
HaswellULT	0x040650	0 0 0 0 0 0 1 1						
CrystalWell	0x040660	0 0 0 0 0 0 1 1						
etc								

That is, support for Yonah and XeonMP has ended; the newer the processor the newer the system required; Atom has never been supported, although it looks like an ordinary Intel processor. Plate obsolete, see XNU source. Skylake, for example, is supported in 10.11.6 and higher.

When you start the system on an unsupported processor, you get a kernel panic. For her to prevent the patch KernelCpu = true. It just replaces the panic call with an empty one operator, and everything continues to work. How correct is it? Well, at least it works! In new Clover revisions I made a patch FakeCPUID = 0x010676. Or other numbers suitable for your system, and those close to your processor (about the same generation, for example The atom should be replaced by Penrin, or even Conroy). The substitution occurs in the core at the level call the get_cpu_info () procedure and thus will affect those cakes that access the CPU for information instead of calling CPUID themselves. for example this is how AppleIntelCPUPowerManagement.kext works and is affected by this patch.
Example:

<key> KernelAndKextPatches </key>
<dict>
<key> FakeCPUID </key>

Page 146

<string> 0x010676 </string>

Unsupported graphics card.
Intel . Supported: GMA950, X3100, HD3000, and above. Alas, no substitutions
help. Each option has its own set of patches, and if the video card is different, then
at best, you will have a picture, without the possibility of changing the resolution, and without any
3D effects. You can live, in principle, but the impossibility of calibrating the screen color
I am not satisfied, because even with photographs on such a computer to work
impossible. HD4000 and higher are supported on 10.14. But HD4000 does not support
10 bits / color, but Skylake HD530 is already yes!
Nvidia . 7300-7600 cards are only supported up to system 10.7.5 in 32-bit mode.
It's probably useless to talk about older cards.
There are some questions for the Fermi cards of the 4xx / 5xx series. They should also be attributed to
partially supported, and only up to 10.11.6. In the case of Nvidia also
control the AppleGPUPowerManagement cache, it may also have the ID of your or
a similar map.
For systems 10.12 and higher, only **Kepler** , the GK family, works natively . For newer
cards need a WEB driver, which exists only before system 10.13.6. These are GTX cards
6xx-7xx, but not all, there are also Fermi among them, then bummer. For Nvidia's work
requires a trick with MacModel, or with substitution of BoardID. For systems 10.14 and up, the web
there are no drivers, that is, Maxwells and Pascals are generally in flight. On the other side someone
starts old Tesla in the Mojave, apparently Apple left a loophole for its old
computers.
ATI / AMD . The whole story. And about how I started the Radeon9000IGP, and about dong's kekst for
X1500, and about Callisto cakes, and complex recipes for patching connectors for modern
cards. See this book. A lot has been done for the Radeons, look, read, do not
be teapots! On systems 10.13 and above, Radeon 6000 series and below do not work. They have
there is no support for Metal, and therefore the drivers in the system are defective, or even not at all
included.

Now here's another WhateverGreen by vit9696 - "the driver of all video cards, just put
the most recent version as well as the most recent version of Leelu, and don't think about anything. IN
Clover needs to turn off everything related to video cards. "But I don't play like that! Who doesn't want
understand, please use. For others, we analyze step by step what is needed for
video card factory.

5700 cards only work starting from Catalina. For the Mojave, for example, a list
possible cards: 550-590 and Vega.

Sound card. Professional cards usually have Mac drivers.
Chipset HDA codecs are supported all with VoodooHDA cakes. To relatives
AppleHDA codec does not support any codec on the market. Once upon a time
was ALC885, but now it is not found. But hackers have developed a patch technique
AppleHDA so that it supports almost any real-tech chip you need (then
there is ALCxxx). Clover helps to fix DSDT for this kext, and suggests ways to patch
keksta on the fly. What exactly to patch and how to read on the forums. HDMI Audio works with
Clover's DSDT patches, however, does not work with some cards
AMD. But in 10.13+ systems there is a new driver AppleGFXHDA.kext, study it!

LAN card. First, Apple drivers support a variety of chips. In-
secondly, for network cards, programmers have learned to write kexts, and drivers exist
for most known cards. In some cases, it is enough to make a FakeID for the card,
so that it will be included in the list of supported by native drivers, but in most cases

needs a separate kekst.

Page 147

WiFi . And here everything is very sad. Some Broadcom, Atheros and
Ralink. Look at the forums for information about each specific model. Intel in general
no way. Clover can help with FakeID, for example, in my version of replacing Boadcom4315 with
supported by 4312. As well as Atheros with neighboring numbers.
Since 2020, a driver for some Intel WiFi cards has appeared on the network. There are chances!

Kext blocking

I happened to install Geenna.kext into the SLE system folder. Panic on the screen
after reboot, this kekst is loaded in the first place, and then there is panic. so what
do now? It needs to be removed, but there is no other system on this computer yet. For this
goals in Clover introduced an additional function: in the Options Menu in the third line, enter
Block kext: Geenna and calmly boot the system into single user mode (space on the icon
system). The kext will not have time to load, because it is blocked. In this text mode

```
fsck -fy
mount -uw /
rm -r -v /S*/L*/Ex*/Geenna.kext
reboot
```

Reboot is necessary here, otherwise the kernel will still load this kext with the next step, and again
there will be panic.
The new Clover has a new technique: we put all our cakes in a folder
EFI / CLOVER / kexts / Other. And in the Clover Details menu (the "space" key) we will find a submenu by
blocking / unblocking kexts.

Starting with revision 5052, it became possible to store unnecessary kexts in the *Off* folder , and
connect them in the same menu as needed

Page 148

At the same time, kexts in the Other folder are enabled by default, and in the Off folder are disabled, however visible to Clover for connectivity.

Slot name (AAPL, slot-name)

This is mainly cosmetics, although there are claims that it is necessary in some cases.

Where does the system get the slot name from? In the old way, they tried to inject it through _DSM property "AAPL, slot-name", but this is a completely wrong method because it heals the consequence instead of a reason. This property is exposed by the AppleSMBIOS system cache based on ACPI properties _SUN and DMI tables. That is, _SUN sets an ID in the range 0-255, by which there is an SMBIOS table type 9 with the corresponding ID, where the slot name comes from and others its properties.
See the chapter on filling in the config, section SMBIOS-> Slots

HDMI sound

Everything AppleHDA Needs Has Been Investigated By Toleda, But Not Everyone Will Want To Track It Down explanations in English. I made patches for DSDT, with his participation, to maximize get closer to its result.

There are basically two options for an HDMI device.
1. On an external video card ATI or NVidia. It is listed in the system as a sound device HDA class = 0x0403, and is served by the same sound driver. You just need to video card and HDMI had the same property "hda-gfx = onboard-1". Or maybe not earn! Unsupported device.
2. The built-in Intel card has an HDMI connector, but there is no such device, sound is used from the chipset HDA. In this case, you need to register in the config

<key> Devices </key>

Page 149

<dict>
<key> UseIntelHDMI </key>
<true />

In this case, the sound from ATI or NVidia will become "onboard-2".
Required fixes for DSDT: FixDisplay_0100, FixHDA_8000, AddHDMI_8000000
3. Option that the insert is used only for IQSV. Then (iMac18.3)
HDEF device has No-hda-gfx property
IGPU has nothing (inline for IQSV)
In GFX0 (which is Radeon) hda-gfx = "onboard-1"
And in HDAU it is the same.
Clover does not automatically do this yet, use the Properties array.
Note that all these additional properties are needed only for AppleHDA. Driver
VoodooHDA does not need external prompts.

Computer startup sound

This is the invention of Goldfish64. He wrote an EFI driver for HDA audio, and made utilities for sound tuning, testing and dumping an audio codec. https://github.com/Goldfish64 / AudioPkg
But he inserted sound at the time boot.efi was launched, intercepting its call by the bootloader. I.e, idea so that it works not only with Clover, but with any EFI bootloader, without touching its internal codes. And it's more interesting for me that the sound works before entering the Clover interface, or even while I stroll through his menu. The license is open, so I remade it to fit your desires. All settings and tests can be performed by Clover himself, with its graphical menu, and I included the driver in Clover's repository so that it not lost, and so that in the future it could be improved without asking the author, which can disappear, not today, so tomorrow.

So, for sound to work, you need:
1. Use Clover revision 4871+. It worked in previous revisions, but in bugs.
2. Put the AudioDxe.efi driver in the EFI / CLOVER / drivers / BIOS or drivers / UEFI folder respectively. Or both. This driver supplied with Clover is already different from the author's original is not important yet, but I would recommend my own version.
3. Put sound files named sound.wav and sound_night.wav in the ones you use Topics. Thus, the starting sound depends on the selected theme. sound_night.wav not is obligatory, if not, then sound.wav will play at night. These files should be RIFF / WAV format, 2 channels, 16bit little endian unsigned int, sampling rate can be 8, 44.1, 48kHz, the file size depends on it. The sound itself supports 44 and 48kHz, or even more. For compactness, I allowed 8 kHz, and Clover converts this file on the fly at 48kHz. The quality is inevitably lower, but for such a case it is not very special and need to. But the sound of this format was packaged directly into Clover, and it is used for testing the output.
4. Go to the Clover interface, Startup sound output →, and test which of the outputs will play

Page 150

In the first line, we adjust the sound volume from 0 to 100. This is a percentage, more than 100 is not it happens. A value of 0 means no sound will be played. That is, it is not what screams with closed mouth, and this is that no attempt is made to meow. By the way, I don't know, the scale linear or logarithmic.
The following lines are combined from the chip model and its output. If you have several sound cards, as often happens in addition to the built-in HDMI, then you will see everything in this list, with all their outputs. Choose, press F7, listen. After getting out of this menu, the selected setting will be saved in NVRAM, including the one emulated in variables Clover.SoundVolume, Clover.SoundDevice, Clover.SoundIndex. Here I have unlike goldfish, it saves the settings in a private area with its UUID, which impossible for emulated memory, for Legacy Clover. My settings will be visible from systems can be removed or modified from the system, and the Clover prefix ensures that there is no conflict with Apple's interests. At this point, Clover will read fish settings, if there are no own ones yet, but in the future it will use only its own.
On the next reboot, you will hear a sound before loading the Clover shell, but after captions ... scan entries ...
In the config, these settings are not taken out, to nothing. In any case, you must first test, and therefore already write the settings to NVRAM. But there is one in the config setting remaining from the test period. PlayAsync = true.
If the false, then while the sound is playing, nothing works. It seems so in the real world.
If true, then the sound is played in the background, without interfering with everything else. I put long sound file, and listened to music. GUI Clover popped up, music was playing, I went into menu, and chose to load in verbose. Music plays. I clicked "load system" and look messages: boot.efi has worked, music is playing, kernel has started, music continues

play! And only after a few loaded cakes did she shut up, probably it was VoodooHDA, which reinitialized the sound chip. On a loaded system, no

Page 151

found no problems. In revision 4862, it was impossible to use asynchronous sound, it hangs. Revision 4871 resolves the problem, and now you can use asynchronous audio for default.

NVRAM, iMessage, mul_boot

Question about the system using non-volatile memory NVRAM using I actually raised the GetVariable () and SetVariable () functions back in 2010 http://www.projectosx.com/forum/index.php?showtopic=1504 Then I tried to implement work with him in the Chameleon in my own branch, but no did not receive support. No one needed this, although my argument about the test the Boot Disk panel remained compelling. Then the gurus explained to me what it is in the bootloader DUET, therefore, starting a Clover project based on DUET, I first of all set out to provide this functionality.

Chameleon has these functions, but they are made very simply by "return Unsupported", so so that the system launched with the Chameleon does not panic and simply does not respond to the call these functions. This worked for the time being, except for the StartupDisk panel. But here iMessage has already refused to work in this version. No substitution worked and emulation. I bow my head to Meklorth, who, within a month, nevertheless came up with a way to do this kind of functionality in Chameleon, using the FakeNVRAM.dylib module and some mother.

What is meant by NVRAM health? If the system wants to keep some variable until next reboot, it writes it to NVRAM with the SetVariable (...) functions. We can also save our variables using the utility nvram:

```
sudo nvram MyVar = qu-qa-re-ku
```

after reboot, this variable must be known in the system using the command reading

```
nvram MyVar
```

How does Clover make this service work?

1. For legacy loading, the EmuVariableDxe functions are used. This, of course, is not real non-volatile memory, due to the fact that Legacy-Clover is intended for those computers where there is no such memory at all, as well as no own EFI with the necessary services. This driver writes variables simply to memory, but this memory is available for using macOSX in its native interface. At system shutdown the rc.shutdown.local script is called, which saves all this memory to the nvram.plist file in the root of the system drive. Clover looks for this file at startup, and sets all the variables from there again to the RAM emulating NVRAM. The method is defective, because this way, only variables with AppleBootGuid are saved, however, this is enough to select the Start Disk.
2. For UEFI downloads we rely on our own service VariableDxe, which provided by OEM UEFI. In revision 2837, Dmazar corrected the work with this service, so that for most users it now works natively. For those who still do not works, the EmuVariableUEFI emulation driver is provided, which works similarly to Legacy driver, and also requires scripts and the nvram.plist file. New times have come! Again vit9696 corrected the OsxAptioFix driver so that the iron HBRAM works, but here on new chipsets 360, 390 and it doesn't work. Change represented by driver OsxAptioFix3Dxe, and vit9696 itself offers a more advanced version of AptioMemoryFix, now included in the Clover repository.

EmuVariable in both cases is not a full-fledged emulation., For example, not
panic.log is saved, simply because the script does not have time to run. Not saved

Page 152

also the variable boot0082, which is required for hibernation, but we bypassed this problem
in other ways. But the presence of panic.log, the old dream of hackintoshars, remains
Clover's prerogative with real NVRAM. And again, hibernation in mod 25 requires
saving the encryption key online, that is, only with a real NVRAM.

iMessage is an instant messaging system from Apple itself. Since December 2012
the rules of registration and use have changed, and all hackintoshares are out of work. FROM
She would have worked as a Clover if in September, having dealt with the iCloud service, we had not
made a mistake in the number of digits, it was necessary to leave 17, but we left 12. We understood the error
only in January, and thus the Chameleonites understood what was going on, only they did not have
NVRAM, without which all this was impossible. Namely, for successful registration
iMessage needs to be written to NVRAM with ROM and MLB variables unique to each
computer, and the computer is recognized by its HardwareUUID, which, accordingly, is also
must be unique. For all dummies, I made the generation of these properties based on
DMI data, but also the recommendation to enter the appropriate values in config.plist, for
those who think a little more. At the same time, it turned out that the iMessage service was paid, and
the user needs to register his account in the app store, from which Apple can
write off $ 1 to verify that the bank account is valid. It also follows from this
the need for uniqueness of the account. No need to use someone else's ROM, MLB and UUID,
and even more so, someone else's bank card. When everything is different, ROM has 12 digits, MLB has 17
digits, the UUID is nonzero, and all this is unique, the account is tied to a valid account, on
with money, iMessage will work. And don't listen to any speculation about
en0, section formatting, and the like. I have listed all the conditions.

Boot Disk is a service that allows you to select in the control panel which
we want to reboot the system, press restart, and just leave.

The computer will do everything by itself. This service requires the disk to be GPT partitioned. So it is possible
switch between 10.9 and 10.7, for example.
Remember the general rule: **dynamic data takes precedence over static data.**
Data from NVRAM takes precedence over data from config.plist.

Using multiple configurations

Possible problem: You have multiple systems, but these systems must boot with a different set of patches written in the config, for example, defining

Page 153

The Radeon framebuffer in the new system is not the same as in the old one. But how to do it if config in clover one? Starting from revision 3266 this possibility is provided.

Here's a config

```
<key> GUI </key>
<dict>
    <key> Custom </key>
    <dict>
        <key> Entries </key>
        <array>
            <dict>
                <key> FullTitle </key>
                <string> Lion special </string>
                <key> Settings </key>
                <string> config-special </string>
                <key> Volume </key>
                <string> EE9CCC69-EE7F-358F-B120-BCD07AD78282 </string>
                <key> SubEntries </key>
                <array>
                    <dict>
                        <key> FullTitle </key>
                        <string> Boot Lion with own settings </string>
                        <key> CommonSettings </key>
                        <false />
                    </dict>
                    <dict>
                        <key> FullTitle </key>
                        <string> Boot Lion with common settings </string>
                        <key> CommonSettings </key>
                        <true />
                    </dict>
                </array>
            </dict>
            <dict>
                <key> FullTitle </key>
                <string> Lion default </string>
                <key> Volume </key>
                <string> EE9CCC69-EE7F-358F-B120-BCD07AD78282 </string>
                <key> Type </key>
                <string> OSX </string>
            </dict>
        </array>
    </dict>
```

The following is described here: we assigned our own main menu items (Entries) with the names "Capitan special" and "Capitan default". The second point, as usual, allows shipping system with a common config.plist, taking into account the changes made in the Options menu of Clover. The first item creates a new icon for the same system, but it will load with another config config-special.plist, as specified in the Settings key.

Page 154

But that is not all. By pressing the spacebar, we will enter the launch menu, and then we will find our inputs, spelled out as SubEntries

Compare the config with the above pictures for clarity of what is happening.
The second item in this menu means rejection of the special config in order to use
general.
It is clear that since a special config is connected after Clover is launched, then
the Boot and GUI sections are no longer needed in it, they can only be in the general config. I, personally,
used this opportunity to test new configs for launching the captain, having

one proven worker. The worker is special, and the general config is experimental, therefore that the general config can be changed through the menu, and the special one is used as is.

Now in Clover it is possible to switch configs right in the menu. Here is a picture:

The limitation of this method is that the Boot section does not change. Also it doesn't change KernelAndKextPatches. The new config will take effect after exiting the main menu.
This is where the confusion arises, why change the config at all, if not for KextPatches? Alas, not works, the reason is somewhere deep in Clover's algorithms. On the other hand, write all patches in one config, the main one, and in the Clover interface you can enable them with a tick or turn off. The use of different configs, except in different SMBIOS sections.

How to prevent boot.e2 from spamming too much on the screen?

Enter this menu

and
write log = 0 there. Other values are possible, researched by vit9696

Page 156

log = value , output direction
1 - AppleLoggingConOutOrErrSet / AppleLoggingConOutOrErrPrint (classical ConOut or StdErr on failure)

2 — AppleLoggingStdErrSet / AppleLoggingStdErrPrint (StdErr or serial?)

4 — AppleLoggingFileSet / AppleLoggingFilePrint (BOOTER.LOG / BOOTER.OLD file on EFI partition)

debug = value
1 - enables print something to BOOTER.LOG (stripped code implies there may be a crash)

2 - enables perf logging to / efi / debug-log in the device three

4 - enables timestamp printing for styled printf calls

level = value - error level
kc-read-size = value - log size
For our purposes, log = 0 is enough, which is done by default in Clover.

FAQ

Cha hundred Asked **During** asks.

Q. I want to try Clover, where do I start?

A. From reading this book.
Shl. It's strange to write this inside a book, but maybe these FAQs will end up outside its pages.

Q. Which version of Clover works best for my hardware?

A. Last. Not even discussed.
Shl. Here is the Bug log for some revisions, which is finally fixed:
3514: DDR4 support
3471: global bug using va_args
3362: bug of SMBIOS when lines are duplicated in the original
3358: fixed calculation of the number of cores for many Xeons
3336: fixed bug with fixing regions
3333: added new processors
3259: Kernel patch allowing loading kernels in ElCapitan
3168: changing the config in the menu was not accepted by Clover
3164: Fixed IDE driver to work correctly in UDMA mode.
3162: Fixed XHCI driver to disable legacy and enable ports.
3157: AHCI speed increased significantly.
3154: Prevented looping on InfoPlist.
3147: BiosBlockIO driver bugs fixed.
3144: fixed bugs with reading config and setting from the menu, interdependencies.
3138: a bug with the launch of Windows.
3128: bug with reading SPD.
3121: starting from exFAT partition.
3116: Several installer fixes.
3100: Yossi kernel patch capability.
3090: Bug VboxHFS.efi is reading a file that is not the requested one.

3086: bug with reading the mac-address hanging on the new chipset.

Page 157

3074: USB Legacy Support freeze. Known initially, but different solutions
developers were exactly opposite and incompatible.
3060: AHCI driver. The patch came from its creators - from Intel.
3057: Overlapping table of addresses and EBDA, making it impossible to start at
some BIOSes.
3053: Procedures using VA_ARG macros must have EFIAPI, otherwise they are possible
bugs in work. And they were really observed! Patch from Intel.
3041: Added injection of new video cards.
3036: Fixed kernel patch for 10.10. Author - Rehabman.
3035: Fixed DSDT patch causing freezes.
And so on ... All this is not counting the amendments to the compilation and installation processes, to cosmetics and design, as well as support for new hardware and new OSes.

Q. Doesn't work.

A. The fool himself.
Shl. Well, what else can you answer?

Q. I installed Clover, but I get a black screen.

A. Booting the OS takes place in eight steps (see page 6). Please specify on which
it is at this stage that the stop occurs. And in your report, be sure to indicate "Installed
the installer with a choice of such options. " Then there will be a conversation.
The most common mistakes are:

- CsmVideoDxe does not work with some BIOSes, uninstall it;
- it happens that PatchVBios = Yes leads to a black screen, try to turn it off,
- worth Boot-> Debug = true. Everything works, but slowly, not enough patience to wait.

For a better diagnosis of what is happening, put

```
<key> Boot </key>
<dict>
    <key> Debug </key>
    <true />
```

in the config.plist file. Loading will be very slow, because at each step
/EFI/CLOVER/misc/debug.log will be updated, but after the final hang you
get information exactly what happened. In reality, when booting from a USB flash drive, speech can
walk about ten minutes to the entrance to the GUI. Since revision 3063 the screen is no longer black if
CloverGUI started loading, then you will see the inscriptions on the screen, by which you will understand that
exactly what happens.

Q. I see 6_ on the screen and nothing else happens.

A. This is the most severe case of iron incompatibility. Doesn't meet now, isn't
what about the AMD processor. Only a programmer who can
insert debug messages into the Clover codes, and do reboot after reboot until complete
clarifying the problem. Alas, there is nothing to advise ordinary users. Read the chapter
about slow Clover, can you wait?
Is that playing around with BIOS settings, sometimes it helps. Try instead of the boot file
use boot7 (Clover BiosBlockIO). Or reinstall the boot1 sector.

B. It loads only up to the text analogue of BIOS with five points, the top one - Con_nue>

A. This means that the boot file has loaded successfully and is working, but cannot find the file CloverX64.efi. Either he does not see that section, or devices in general - you need to understand further by walking through the options in this menu. For example, the file may be missing

Page 158

HFSPlus.efi, and you have Clover installed on the HFS + partition. It's strange actually why do UEFI boot from HFS + partition.

Q. I installed Clover on a USB flash drive, booted from it, and I don't see my HDD.

A. First, the HDD must be inserted into the Sata0 port. In the future it may be already fixed. [5] Secondly, I understand if you have a well-functioning Ham, Chimera, HRC, in short, BBH (**B** uter on **B** ukwu **X**), you don't want to kill him, but you want to try Clover, then such an act seems natural. But, nevertheless, there are options for installing Clover to the hard drive that do not kill the old bootloader, and in this situation, the sounded error will disappear.
Also try the boot7 file if you have any unusual SATA / SAS / RAID controller.
On a UEFI boot, this may also mean that the PartitionDxe.efi files are missing and HFSPlus.efi.

Q. When loading UEFI, I do not see a section with MacOS, only Legacy.

A. This means that there is no HFSPlus.efi in / EFI / CLOVER / drivers / UEFI folder or its legal analogue of VboxHFS.efi.

Q. When UEFI boots, Windows looks like legacy, although it is EFI.

A. The same, the NTFS.efi driver is missing
Shl. These two drivers are missing from the repository for licensing reasons, you need find this file somewhere on the Internet. Now there is a legal analogue GrubNTFS.efi. Available in Clover's installer.

V. Set the native resolution in the bootloader, but the screen is in a black frame.

A. Can't fix it. In any case, the Clover developers could not come up with anything, and no one will answer this question. There is one option: if you have a UEFI BIOS, then you need to do UEFI download, and flash the video card to UEFI VideoBIOS. In BIOS we make the settings:
- OS: Windows 8 WHQL
- CSM: Never
- Full screen logo: Disabled
Nothing can be done for legacy loading. Don't like the funeral frame - make more low resolution.

B. When trying to start the OS, it freezes on a black screen

A. At this point, the DSDT patch happens to your mask. Yes, ideally it shouldn't be here hang. But the problem is that a lot of BIOS manufacturers do not comply standards, do not know how to program, and do not want to polish their DSDT to fit the needs OSX. It is very easy to make sure that the operation is decompiled - not compiled again passes - DSDT curve. Clover would like to fix all this, but alas, the number of bad options are not even amenable to review. Therefore, you are required to choose such a mask fix DSDT, so that the bootloader does not hang, and then so that the OS does not hang, and ideally, so that she also worked. It's real. Or abandon the autopatch (mask = 0), and do DSDT manually. See the chapter on debugging dsdt. And I highly recommend using the latest version of Clover, because such bugs are found from time to time and are being corrected.
And there is also an option: set KernelPM = true

5 There was such a bug in the Intel SATA driver, which has now been fixed.

Page 159

B. The kernel starts to load, but panics after the tenth line Unable To 2nd driver for this plajorm \ "ACPI \".

A. It is a missing or incorrect DSDT. If the auto patch fails, add
DSDT made by hand. Pay attention to the auto patch options, as well as the keys
ReuseFFFF and DropOEM_DSM.

Q. The system starts to boot, but stuck at s_ll wai_ng for root device….

A. In addition to the usual advice for such cases, enable AHCI in BIOS, or, if this is not the case,
find the correct driver (in the sense of kext) for your IDE controller, here is some more advice
boot with the WithKexts key (in new revisions of NoCaches), then the download will go
slower, and the controller will have time to turn on. By the way, such an error can only occur
if Clover and the system are on different devices.

B. The system boots up to the message: Wai_ng for DSMOS….

A. Missing FakeSMC. Maybe with the Chameleon you had this cake in Extra, and
Clover does not see this folder. The folder /EFI/CLOVER/kexts/10.x or
others. Don't forget about the InjectKexts key as well. Disabled by default!
At the second stage of the installation, Clover does not know the version of the system (it has not yet been determined),
so put FakeSMC in / EFI / CLOVER / kexts / Other / folder
In newer versions, the InjectKexts key is set to "Detect", which should automatically
to cope with this situation, check what is written in your config.

B. The system passes this message, but nothing further changes, although the hard drive buzzes as if the system is booting.

A. A typical situation when the video card did not turn on. Try GraphicInjector = Yes in
config, or vice versa = No. In the second version, the Radeons are launched on the "native
factory ", which even allows you to work in the system, with a few exceptions,
for example DVDplayer won't work. For the complete installation of Radeon, it is also required
fix the connectors. For other cases, you can try to boot the system with the key -
x, and log in to the desktop in VESA mode. Not very cool, but it will fix something.
Another variant of the brake in this place is observed if you choose the MacMini model or
MacBookPro. The problem is solved with the installation of the key DropMCFG = Yes or FixMCFG

Q. The system boots up to the message: [Bluetooth controller ....

A. Same thing. See the previous point. Blue tooth has nothing to do with it.

Q. The system has booted, everything is fine, but there are errors in the System Profiler ...

A. In general, this is cosmetics, it does not affect functionality.
About PCI cards. See the chapter on AAPL, slot-name
About memory. There are two speeds, nominal and actual, and they are often not
match. Which one to show in the profiler? I put the first one - they yelled that it was not true.
I put the second one, these fell silent, other users yelled that it was wrong ...
See page 47 for how to write custom memory values in the config.

Conclusion

Clover, of course, is still far from ideal, but the process of improving programs never
is never complete. There will be new revisions, there will be new functions, but so far.

Clover's biggest flaw is that it tries to be versatile.
The programmer can make his own version from the source code, suitable for his own iron. For the rest, there is a config with hundreds of settings, and it is too complicated for

Page 160

average mind, despite the presence of automation, instructions, descriptions and a lot of advice from experts. Chameleon works by means of BIOS drivers, and therefore it has more chances run on arbitrary hardware, but no one keeps statistics, in what percentage Clover's cases work better.

Clover's development is completed, but the project is not dead, it continues to remain, and there will still be develop.

About the Chameleon.

Great respect to all the creators of this project, which made the Mac possible on the usual PiSi. Clover borrowed a lot of technology from him, for he was created for the same purposes (injection of video cards, efi-strings, adsp patch, ssdt generator, smbios patch, but only all this is already on a completely different level).

I was also among the Chameleon developers and offered my patches / improvements, however, the project admins ignored me. There are a lot of shortcomings and just bugs that have not been fixed. http://www.projectosx.com/forum/index.php?showtopic=1106
When the Chameleon is not working, they don't talk about it, they just ignore it.

The first hit came in the spring of 2011, when the 10.7 system came out, and the Chameleon was unable to download it. Then Gyk discovered that the system could be loaded by XPC, which EFI-loader. This was the start for the Clover project, an open source EFI downloader, in unlike private XPC. The reason for Chameleon's failure was in the BootArgs structure which changed in the new system as well as legacy interrupts. Respect to netkas and cparm who found a way to fix the Chameleon to load the new system.

The second hit came in January 2013 when iMessage demanded availability of ROM and MLB variables in NVRAM. Clover got over it back in September, but with a small error in the length of the line, which was only fixed in January. Then with Clover earned iMessage, but for Chameleon it turned out to be impossible to repeat. The principle of operation is completely different. It took Meklorth and Cosmo1 a month to overcome this bar. Since that winter, the number of Clover users has exceeded the number of Chameleon users. But the Chameleon is fully functional again, and they remain ardent adherents of it. "Everything works with a chameleon!"

The chameleonists ignored the third blow, like "no and don't." In January 2014 we made hibernation - deep sleep. It only worked with chameleon up to system 10.7 for some reason. There was no one to investigate why and how. Meklort retired, the rest developers in the team can only add new names for video cards. Clover ended up the only bootloader with which hibernation works at least on a 10.9 system.

I can also remind you that the problems of floating regions are not solved with the Chameleon, the name of the slot, and a lot of not particularly necessary little things. In addition, the Chameleon has a lot mistakes that there is simply no one to fix.

The last hit came in June 2014. Apple released the 10.10 Yosemite system. which Clover can load, and the necessary patches have already been made, starting with the revision 2695. But for the Chameleon, the end seems to come ... Looking back at history, you understand that one cannot renounce, everything in this world is possible, it is possible that one of the developers is all he will overcome this bar, and some of the fans will remain with the Chameleon.
Happy to stay!

PS: Yes, we figured out this problem, the Chameleon is now loading Yosu, but for some reason there were problems with 10.9.4, problems with NVRAM, and therefore with iMesyadzh. And judging by activity on the forum, Chameleon / Chimera have only those who somehow once installed

system, and is not going to change anything.

Page 161

Another blow, the appearance of the apfs file system. For Clover, there is native Apple apfs.efi, but it only works in the EFI environment, but not in the Chameleon. Well, again, two years later there was a programmer who made a legacy APFS driver for the Chameleon. Good luck!

Chimera is a stripped-down branch of the Chameleon, with its own theme and with a "different video injection". Then there is for a video card factory you need to apply a DSDT patch or a kext like natita.

Revobut is a stripped-down Chameleon in which you need to compile your DSDT. I.e Everyone should compile for themselves. According to the creators (Master Chief and his "daughter" Revogirl) this allows you to reduce the download time for the time of reading the DSDT file. Rave! The rest of the improvements are even more questionable. Currently supported Pike R.Alpha ("son" of the chief, brother of this girl), who, in particular, managed to download Yosemite. For himself, of course, he can make everything work (and someone can confirm?). But for other users, there is nothing to offer.

Yes, the Chameleon has the right to life due to the fact that he is purely legal bootloader, and can work where Clover has a problem with legacy boot, old computers, left chipset, and the like. I do not quite understand the relationship with AMD CPU. Have someone works, while others do not even try, they just use ready-made solutions with Chameleon. He seems to work with Clover, but no one is investigating.

In short, I'm tired of talking about the Chameleon. I've argued for years that Clover is better someone was not convinced, do not care. The topic is closed.

Other EFI downloaders.

The XPC loader was announced in 2009, a team gathered and even created a website project. I don't know what happened to them. The last message states that "due to spammers we will not make the project open. "I did not understand what spammers were and how they prevented them. The project was frozen, the team scattered. There is still iPhoneTom, actually the founder, which did not go to any cooperation anymore, and did not open the source code. The finest hour of the project came when in the spring of 2011 Gyk installed 10.7 with the help of XPC, which was impossible with the Chameleon, as I said above. Tom came to life, but not cooperate became, but only allowed testers to send their reports and wishes to the IRS. I do not have HRC it worked on any computer, so I started my project, this was the start Clover. So, the alignment for the fall of 2011: most users use Chameleon, which overcame this problem and began to develop rapidly. Some have tried XPC, and became his ardent adherents: "What the hell do you better help Tom with his bootloader. He is quite an adequate guy, and he listens to criticism. "I, however, a programmer, I I can work on my own, and not sit by the KFM, waiting for a good uncle to fix something. And I, while alone, I began to make a bootloader based on DUET, and in the first month I received some results are better than XPC. War is so war, I didn't give my know-how to Tom. AND a small initial advantage - the support of Russian users, of which there are more than any others.

Clover version 1 used the interface from Ninzy, who "stole" it from an early XPC versions. In such a situation, it was impossible to develop Clover, and at the beginning of 2012, when I understood all the necessary technologies, I started making the Clover version 2 interface on based on the rEFIt project, open source. I want to note that XPC comes from him, so the claims are more likely to him, which Tom has the right to close the source, if he himself uses open source. Clover is now licensed clean, and has risen to a level where one could talk about competition. Spring 2012. " XPC is so far surpassed in functionality failed . "However, he remained unresolved

a problem with the system-type, which in the case of a laptop interfered with sleep. And also the board-id, which interfered on some configurations with a 10.7+ system installation. And on Clover I have these

Page 162

there were no problems, because I initially chose other patches, for different ideas, and which of
they were so influenced was completely unobvious, looking at my sources. I knew something, but firmly
decided not to explain to anyone. Users don't need it. Works in Clover, so
you will use Clover.

This is how the bareBoot project was born. Author of SunKi, an ardent supporter of XPC and the best helper
Tom in his project, I decided to get to the bottom of the truth. He was repeatedly interested in
Clover, why and how it was done, but never made any suggestions for improvement
Clover, on his further promotion. Realizing that I'm not going to tell my
secrets, he opened his project, they say, I want to combine CloverEFI + patches into one
file, and use the existing SetupBrowser as a GUI, with modifications for
booting multiple systems, so we got a text menu in which you can select
system to boot. I agree, work has been done, and not a little one. However, to this
time already Dmazar made a UEFI download, and the CloverEFI + GUI combination turned out to be
unacceptable. Barebut is designed exclusively for legacy downloads. However, Sanka does not
the goal was to make an attractive bootloader, its goal was to decrypt technologies
Clover. He started with pure Duet, and started adding patches from Clover step by step,
checking what influences what (and after all, he could start with the finished Clover!). But Clover is not worth it
in place. We, already with Dmazar, rapidly improved and transformed the codes, so
keeping track of us was not easy, just as it was not easy to compare what was and what became. And Sankey
I could not find in any way how the system-type was made in Clover. Meanwhile Tom has stopped
to engage in a project, but there were no trump cards in the barebutt to attract users. Lack of
graphics? Okay, we will make a purely text interface in Clover too, if anyone has an allergy
on the schedule. Download speed? Let's compete. Meanwhile in Clover
new functions appear that are not so easy to copy into barebut, in particular patches
DSDT, kexts and kernels, not to mention the UEFI download. Users left to reap
shoulders "And why barebut is needed at all?"

Meanwhile, another notable event took place in the hackintosh world. A certain company
QUO has crafted a motherboard based on the Gigabyte Z77, making changes to
better compatibility with Hackintosh. But most importantly, they suggested stitching the bootloader
Mac directly into BIOS. One of the founders of this bootloader, THeKiNG constantly
was present in the Clover topic, and diligently asked what and how, but also nothing from
did not bring himself into Clover. And now we see a certain Ozmosis bootloader, which is stitched into
BIOS, and contains modules taken from Clover. Stitched there in BIOS and some kind of stripped down
variant of FakeSMC. And thus, on this motherboard, you can run pure OSX, without
a single hacker file, no downloaders, no extra kexts. True, in my opinion, all this
only true if nothing is updated. If you update the system, then you will have to BIOS
alter, and in general you can reach the brick. About updating the fake and sensors too
a huge question. And, of course, this bootloader is not designed for other motherboards
boards.

Recently, King dropped another phrase "oz is not acceptable for laptops." And I
I guess that the point is not only that there is a danger of getting a brick with
overburning the BIOS. In fact, Oz was leveled precisely for the Gigabyte Z77 board, and
work on other hardware is questionable. Happy sailing!

With the release of ElCapitan, while still in beta, the Ozmosis bootloader also experienced
shock. In this system, kexts from the outside, for example from BIOS, are not loaded. For Clover we
this problem was solved (thanks to solstice), but it's in Clover's body itself. And the body of Ozmosis
No one to edit, the sources are closed. Stay with the old systems, comrades!

Khaki clover. Version 5.0, revision 5120
Moscow, 2020