Клевер цвета хаки

Автор Slice

Содержание

Предисловие

О чем идет речь? Да уж разумеется не о цветочке, растущему на лугу на радость коровам. Речь идет о программном обеспечении, о загрузчике нового типа, который позволяет на обычном компьютере запустить необычную операционную систему – Mac OSX. Apple этого делать не разрешает, в первую очередь мотивируя тем, что “мы не можем обеспечить работоспособность на компьютерах, произведенными не компанией Apple”. Что ж, ставим систему на свой страх и риск. Ну и не стоит получать какую-то коммерческую выгоду из этого, во избежание других юридических сложностей. Не-эппловский компьютер с установленной системой MacOSX называется Хакинтош, происхождение слова понятно.

Чтобы запустить Хакинтош, нужен специальный загрузчик, их много разных, но по своей основе можно разделить на два класса: FakeEFI и RealEFI.
FakeEFI изобретен David Elliot много лет назад, и действует по-простому принципу: сделаем вид, что у нас ЕФИ уже отработала, оставим в памяти следы его деятельности (boot-args и все дерево таблиц), оставим в памяти EfiRuntime в упрощенном виде "Неподдерживается", и запустим ядро mach_kernel. Так работает Хамелеон, и работает успешно, но за небольшими исключениями типа панели "Загрузочный диск". Не исключено, что со временем Эппл даст нам и другие проблемы, связанные с отсутствием Рантайм Сервисов. Январь 2013: это случилось! iMessage перестал работать, потому что ему обязательно нужен SetVariable().
RealEFI должен был бы быть прошит вместо БИОСа, но для тех, у кого материнская плата на основе БИОС, придуман загружаемый ЕФИ. Эта система, загрузка EFI на машине с BIOS придумана Intel, и сейчас находится в активной разработке с открытыми исходными кодами на сайте tianocore.org. Собственно этот загрузчик называется DUET. Да вот беда. ЕФИ-то он загружает, а вот загрузка операционной системы МакОС там не предусмотрена. Требуется следующий шаг, приспособить ДУЕТ под требования МакОС. На новых материнских платах EFI уже есть, но он так же не пригоден для загрузки Хакинтоша.

Название Clover данный загрузчик получил от одного из первооснователей проекта kabyl’a, который увидел сходство клавиши “Command”, существующей только на Маках, с четырехлистным клевером.

Четырёхли́стный кле́вер *— одиночное растение клевера, обладающее по крайней мере одним четырёхпластинчатым листом, в отличие от обычных трёхпластинчатых. В западной традиции существует поверье, что такое растение приносит удачу нашедшему, в особенности если оно найдено случайно*[1] *[1]. По легенде каждая из пластинок четырёхпластинчатого листа представляет что-то конкретное: первая — надежду, вторая — веру, третья — любовь, а четвёртая — удачу[2].*

В русском варианте мы называем загрузчик «Кловер».
Разработка проекта идет на форумах
http://www.projectosx.com/forum/index.php?showtopic=2562&st=0
http://www.applelife.ru/threads/clover.32052/

1 Интересно, а как можно найти клевер случайно? Регулярно щипать травку на лугу?

Тактико-технические характеристики

EFI – Extensible Firmware Interface – расширяемый интерфейс доступа к аппаратно-зависимым функциям. В отличии от БИОС, который занимает 64кб и написан в 16-битных кодах, ЕФИ занимает от 4Мб, написан в 32 или 64-битных кодах, и позиционируется как аппаратно-независимый, хотя… конечно, чудес не бывает, и 100% совместимости с любой платформой добиться невозможно.

Clover – это EFI загрузчик операционных систем, для компьютеров уже имеющих UEFI BIOS, и для компьютеров, не имеющих такового. При этом сами операционные системы могут поддерживать EFI- загрузку (OSX, Windows 7-64EFI, Linux), либо нет (Windows XP), в последнем случае предусмотрен legacy-boot – возврат к старой схеме BIOS-загрузки через бутовые сектора.

EFI – это не только начальный этап загрузки ОС, она создает также таблицы и сервисы, которые доступны для использования в ОС, и её работоспособность зависит от корректности ЕФИ. Нельзя на встроенном UEFI загрузить OSX, также, как нельзя загрузить OSX из чистого Дуета. CloverEFI и CloverGUI выполняют немалую работу по корректировке встроенных таблиц для возможности запуска OSX:

- таблица SMBIOS (DMI) заполняется данными, эмулирующими реальные компьютеры Apple Macintosh – условие запуска OSX. Серийные номера выдуманные, но подходящие;
- ACPI таблицы, содержащиеся в ROM компьютера, как правило содержат ошибки и недостатки, чаще всего из-за лени производителей: в таблице APIC неправильное число ядер ЦПУ, отсутствуют данные NMI, в таблице FACP отсутствует регистр Reset, неправильный профиль питания, в таблицах SSDT отсутствуют данные для EIST, а уж про DSDT вообще длинный разговор. Clover пытается все это поправить;
- OSX также стремится получить от загрузчика данные о дополнительных устройствах, таких как видеокарта, сетевая, звуковая и т.д. через механизм EFI-string. Clover формирует такую информацию;
- для компьютеров на основе BIOS характерно использование USB на начальной стадии загрузки в режиме Легаси, что становится неприемлемым при передаче управления в ОС. Загрузчик осуществляет переключение режима работы USB;
- также OSX обменивается с EFI информацией через специальную память NVRAM, доступ к которой осуществляется через RuntimeSevices, отсутствующие в легаси-загрузчиках. Кловер предоставляет такой обмен информацией, причем двухсторонний, что дает правильную работу Firewire и возможность использование панели управления "Стартовый Диск" для автоматической перезагрузки в другую систему. Еще NVRAM нужен для возможности регистрации в сервисах iCloud и iMessage;
- необходим протокол ConsoleControl, отсутствующий в Дуете;
- необходимо заполнить некоторые данные в EFI/Platform через протокол DataHub, отсутствующий в Дуете, и не всегда присутствующий в УЕФИ. Наиболее серьезная величина FSBFrequency, определение которой является задачей загрузчика, данные в ДМИ неточные, либо вообще отсутствуют;
- перед началом работы CPU должен быть правильно проинициализирован, но, поскольку системные платы изготавливают универсальными, на целый круг разных ЦПУ, во внутренних таблицах отсутствуют данные по процессору, либо представлены какие-то универсальные, некорректные в частном случае. Кловер осуществляет полный детект установленного ЦПУ и делает

необходимые коррекции в таблицах, и в самом ЦПУ. Как один из результатов - включается режим Турбо.

- еще одна мелочь. Исходники ДУЕТа, да и всего ЕДК2 написаны универсальными под разное железо...., но зависимость от железа сделана через константы. То есть предполагается, что пользователь компилирует Дует под одну конкретную конфигурацию. Цель Кловера — быть универсальным, с автодетектом платформы.

Все это осуществляется автоматически, и новичок может использовать Кловер даже ничего не понимая в указанных проблемах. Ну а продвинутому пользователю Кловер предоставляет возможности по изменению вручную множества параметров. Их рассмотрение и представляет собой цель этой книги.

Что есть что?

При включении или при перезагрузке компьютера загрузка операционной системы с помощью Кловера происходит по следующему пути.
Вариант А. Компьютер основанный на БИОС (старая схема)
`BIOS->MBR->PBR->boot->CLOVERX64.efi->OSloader` (boot.efi в случае MacOSX, bootmgr.efi для Windows).
Вариант Б. Компьютер, основанный на УЕФИ БИОС (новая схема)
`UEFI BIOS->CLOVERX64.efi->OSloader`

Чтобы это осуществлялось, следующие файлы должны быть прописаны в следующих местах:

MBR сектор

нулевой блок внешнего носителя, с которого происходит загрузка (HDD, SSD, USB Stick, USB HDD, DVD). В этот блок следует записать в первые 440 байт один из вариантов:
boot0 – его роль в поиске активного раздела в MBR разметке диска, и передача управления на его сектор PBR. Возможен вариант гибридной схемы разделов MBR/GPT. Если разметка — чистая GPT, то происходит передача управления на EFI раздел. Нынче называется boot0af (Active First).
boot0hfs – поиск первого раздела с сигнатурой 0xAF, т.е. HFS+ раздела с установленной OSX, и передача управления в его PBR. Таким образом можно загрузить систему с HFS+ раздела на GPT размеченном диске, но только с первого такого раздела. Нынче называется boot0ss (Scan Signature).
boot0ab — поиск раздела с сигнатурой 0xAB — Apple Boot Partition.
boot0md – комбинированный вариант, который ищет HFS+ раздел по нескольким дискам, а не только по главному.

PBR сектор

первые блоки каждого раздела на выбранном носителе. Сюда записывается загрузчик фаза два. Этот загрузчик знает файловую систему своего раздела, и способен отыскать там файл с именем boot, чтобы загрузить его, и передать в него управление. Соответственно, существуют варианты под разные файловые системы.
boot1h2 - для файловой системы HFS+ и поддержкой длины файла boot до 472кб. Старый вариант boot1h, распространяемый с загрузчиком «Хамелеон» поддерживает

только 440кб файл (надо 472). Имеет паузу на 2 секунды, чтобы переключить загрузчик.
boot1h — тоже, но без паузы.
boot1f32alt - для файловой системы FAT32. Эта файловая система имеет поддержку записи, поэтому очень удобна для установки на нее загрузчика. Можно использовать EFI раздел, можно использовать флешку, как она есть, поскольку в продажу поступают флешки уже отформатированные в FAT32[2]. Имеет паузу 2 сек.
boot0f32 — вариант без паузы.
Эти сектора (точнее сказать загрузчики фазы 2) имеют еще одну полезную функцию. Они имеют начальную задержку на старте длиной в две секунды в ожидании ввода с клавиатуры. Введенная цифра будет присоединена к имени файла, т.е. нажав клавишу 1 мы загрузим файл boot1, нажав клавишу 3 мы загрузим файл boot3, нажав клавишу 6 мы загрузим файл boot6. Смысл в том, чтобы держать в одном месте разные варианты загрузчиков, или даже разные загрузчики, просто дав им разные цифры. К примеру:
boot — Clover, текущая версия, или тестовая версия
boot1 — Хамелеон
boot3 — Clover-32bit, проверенная, рабочая версия
boot6 — Clover-64bit, проверенная, рабочая версия
boot7 — Clover-64bit с драйвером BiosBlockIO, работающий с любыми контроллерами, которые поддерживаются BIOSом.
Кроме этих вариантов в секторе PBR могут находиться boot manager Windows, знающий файловую систему NTFS, GRUB, знающий файловую систему EXT4, и другие, не имеющие отношения к Кловеру. По-крайней мере на данном этапе.

Boot или CloverEFI

В случае с Хамелеоном это и есть весь загрузчик. В случае с Кловером вариант А в этом файле лежит вся система EFI, а также бут-сервис для передачи управления в следующий этап. Все это, в варианте Б следует считать уже присутствующим в ROM компьютера. Реально получается, что там есть не все, и кое-какие детали следует загружать дополнительно. Детали, уже собранные в файл boot варианта А.
В отличие от предыдущих стадий файл boot уже отличается по битности, т.е. отдельные варианты для 32 и 64 битной загрузки. В большинстве случаев выбирать следует 64битный вариант, если процессор поддерживает этот набор инструкций. Впрочем, если по каким-то причинам предполагается работать только в 32-битной ОС, то имеет смысл грузить именно EFI32. Она меньше по размеру на 20%, и соответственно быстрее, но несовместима с Windows 7 EFI, которая всегда 64бита.
По своей сути файл boot представляет собой модифицированный DUET. Причём исправлений по существу вряд ли наберётся на 1%, но этот процент обеспечивает кардинальное отличие от Дуета – Кловер работает для той цели, для которой предназначен. Если кто-то полагает, что я весь год занимался ..нёй, редактируя ДУЕТ, и что достаточно взять ванильный, и добавить к нему AppleSim, то счастливого пути! Это далеко не так, начиная с того, что ДУЕТ компилируется под один конкретный компьютер, заданный через PCD константы, а Кловер должен работать у всех. Но у меня нет цели описывать все тонкости этой разработки. Дело уже сделано.
В дальнейшем называем эту программу обобщенно CloverEFI.

2: по историческим причинам рекомендуют переформатировать каждую флешку, да еще и обязательно в системе Windows. Нынче эта рекомендация устарела, и не имеет под собой основания. Фабричный формат пригоден для установки Кловера.

CLOVERIA32.efi и CLOVERX64.efi или CloverGUI

Этот файл, в двух вариантах разной битности, представляет собой графическую оболочку загрузчика для выбора операционной системы, для установки дополнительных опций, для подгрузки дополнительных драйверов и собственно для запуска ОС. Графика и меню изначально основаны на проекте rEFIt[3], что отражено в названии директории и в меню About. На настоящий момент оригинальная часть составляет едва ли 10% от всей программы, да и то в исправленном виде.
В дальнейшем обобщенно называем эту программу CloverGUI.

Структура папок

Кроме того, загрузчику нужны дополнительные файлы, структура папок будет следующая.

```
EFI:
	BOOT:
		BOOTX64.efi
	CLOVER:
		ACPI:
			WINDOWS:
				SLIC.aml
			origin:
			patched:
				DSDT.aml
		CLOVERIA32.efi
		CLOVERX64.efi
		themes:
			black_green:
				BoG_LucidaConsole_10W_NA.png
				icons:
					func_about.png
					os_clover.icns
					os_unknown.icns
				banner.png
				background.png
				Selection_big.png
				Selection_small.png
				theme.plist
			hellfire:
			metal:
		config.plist
		drivers32:
			FSInject-32.efi
		drivers64:
			FSInject-64.efi
			NTFS.efi
		drivers64UEFI:
			CsmVideoDxe.efi
			DataHubDxe.efi
			FSInject-64.efi
			HFSPlus.efi
			NTFS.efi
			OsxAptioFixDrv-64.efi
			OsxFatBinaryDrv-64.efi
			PartitionDxe.efi-64.efi
		kexts:
			10.6:
```

3 Часто спрашивают «Почему не rEFInd?». Отвечаю «Потому что он появился позже Кловера»

```
        10.7:
        10.8:
        Other:
misc:
OEM:
   Inspiron 1525:
        ACPI:
              origin:
              patched:
                    DSDT.aml
        config.plist
        kexts:
              10.5:
              10.6:
                    Injector.kext
                    VoodooSDHC.kext
              10.7:
              Other:
        UEFI:
              ACPI:
                    origin:
                    patched:
                          DSDT.aml
              config.plist
              kexts:
ROM:
tools:
        Shell32.efi
        Shell64.efi
        Shell64U.efi
```

То есть, файл CLOVERX64.efi должен находиться по адресу /EFI/CLOVER/, а шрифт BoG_LucidaConsole_10W_NA.png в папке /EFI/CLOVER/themes/black_green/. Реально эти, а также и другие папки более наполнены содержимым. По ходу повествования будем описывать подробнее, что для чего служит.

Несколько слов про папку /EFI/CLOVER/OEM/

Папка предназначена для хранения вариантов загрузок для разных конфигураций. Типичная ситуация когда мы создаем загрузочную флешку, и на ней кроме общего конфига /EFI/CLOVER/config.plist, есть еще и хорошо выверенные /EFI/CLOVER/OEM/Inspiron 1525/config.plist и /EFI/CLOVER/OEM/H61M-S1/UEFI/config.plist, а также свои проработанные DSDT.aml, разные для разных компьютеров.

Название папки определяется из SMBIOS и вы можете посмотреть по boot.log, как именно называется ваш компьютер:

```
10:061  0:000  Clover revision: 1709  running on Inspiron 1525
10:061  0:000  ... with board 0U990C
```

В первой строчке название всей системы, характерно наличие для ноутбуков, но на десктопах там что-то абстрактное. Во второй строчке модель материнской платы, удобно для десктопов, но не для ноутбуков. Для названия своей папки годятся оба имени, выбирайте более понятное.

Еще в вашей папке может содержаться папка UEFI, чтобы иметь разные конфиги для UEFI (вариант B) и для легаси загрузки (вариант A) на одном компьютере (хотя я лично сомневаюсь, что это кому-то надо).

Драйвера EFI

Отдельно упомяну папки drivers32, drivers64, drivers64UEFI, соответственно для 32, для 64битной загрузки по варианту А – BIOS boot, и для UEFI загрузки по варианту Б. Состав этих папок будет отличаться для разных ревизий БИОСа, а также для разных конфигураций разделов.

Стоит заметить, что эти драйвера имеют силу только на период работы загрузчика. На загруженную операционную систему они не влияют, разве что косвенно.

Что следует положить в эти папки? На выбор пользователя.

- **NTFS.efi** – драйвер файловой системы NTFS, для возможности грузить Windows EFI.
- **HFSPlus.efi** – драйвер файловой системы HFS+, необходим для запуска MacOSX. Необходим для варианта Б, а вот в А он уже присутствует в файле boot.
- **VboxHFS.efi** — легальная альтернатива для HFSPlus.efi, отличается меньшей скоростью.
- **VBoxExt2.efi** – драйвер файловой системы EXT2/3, необходим для запуска Linux EFI. Аналогично **VboxExt4.efi**
- **FSInject.efi** – драйвер перехватывающий файловую систему, для возможности инжектировать внешние кексты в систему. Сложно для понимания? Позже к этому вопросу еще вернемся, когда будем рассматривать ключ WithKexts
- **PartitionDxe.efi** – вообще-то, такой драйвер есть в CloverEFI, да и в UEFI он есть, но только тот не рассчитан ни на Apple partition, ни на гибрид MBR/GPT. Вывод: в варианте Б драйвер нужен.
- **OsxFatBinaryDrv.efi** – необходимый драйвер для варианта Б, обеспечивает запуск толстых (Fat) модулей, каким является boot.efi.
- **OsxAptioFixDrv.efi** – особый драйвер, предназначенный для коррекции карты памяти, которую создает AMI AptioEFI, иначе запуск OS невозможен.
- **OsxLowMemFix.efi** — упрощенный вариант AptioFix, они не должны использоваться одновременно.
- **Usb*.efi, UHCI.efi, EHCI.efi, XHCI.efi** – набор драйверов USB, для тех случаев варианта Б, когда встроенные драйвера почему-то работают плохо. С чего бы вдруг? Возможно, есть какая-то завязка на другие функции, которые пришлось отключить.
- **PS2Mouse..., PS2MouseAbsolute..., UsbMouse...** - набор драйверов для поддержки указателя мыши/трекпада/тачпада в интерфейсе CloverGUI. На операционную систему эти драйвера не влияют.
- **DataHubDxe.efi** – этот драйвер уже присутствует в варианте А, и вполне возможно, есть и в UEFI. Но на случай если его там нет, стоит загрузить внешний. Конфликта не возникнет, зато будет уверенность, что он есть.
- **CsmVideoDxe.efi** — драйвер видео, который обеспечивает больший диапазон размеров экрана, чем встроенный в UEFI, нужен для варианта Б.

Разработка

Проект не имеет коммерческой значимости по лицензионным причинам, и еще и очень велик по объему, чтобы делать его в одиночку, поэтому самое разумное

решение сделать его с открытыми исходниками[4], и пусть все желающие внесут свой вклад. В настоящий момент проект базируется на сервере sf.net в репозитарии http://cloverefiboot.sourceforge.net/

Лирическое отступление. В создании проекта, да еще такого большого, требуется труд по следующим пунктам:

- Сбор документации, даташитов, спецификаций, образцов программ по поставленной задаче. У меня была хорошая стартовая точка — Хамелеон, в котором «все» работает (в кавычках, однако!). Правда, время идет, и добавляются новые процессора, новые видеокарточки, новые требования к новым версиям OSX, и нужно снова искать описания, или делать новые тесты. Стартовую точку систематизировал Кабыл, поклон, в первую очередь, ему.
- Когда есть постановка задачи, входные данные и что желательно получить на выходе, требуется написать алгоритм, желательно компактный и быстрый, желательно безошибочный и безопасный. Такие задачи программисты любят, и большинство поклонов в данном проекте отвешено именно таким помощникам. Особо хочу выделить Дмазара и Гыка, они сделали реально сложные вещи.
- Наступает время нудной и тяжкой, но несложной (?) работы по написанию тысяч строк кода, где особенно думать и не надо, копи-паст с исправлениями. А вот тут-то, извините, мне практически никто не помогал все два года. Твой проект — ты и вкалывай. Разве что Апианти вложился, отдельное спасибо ему.
- Далее идет работа по тестированию и выявлению ошибок. Тут у меня полно помощников, всех даже и не вспомнишь. 20000 постов только на applelife.ru. Тестирование бывает разным, от простого нытья, что ничего не получается, до конкретных указаний, что нужно изменить в коде. Все это полезно, даже нытье, ибо заставляет подумать, как сделать, чтобы его не было (неважно, решил проблему или нет, нытье надо прекратить!).
- Самый плохой вид ошибки, это принципиальная проблема. Три первых проблемы я решал в одиночку больше полугода: 1. КП в младших системах, 2. Отсутствие сна, 3. Нет старта на ноутбуке. Следующие проблемы решали вместе с Дмазаром и Пене: 4. iCloud, 5. iMessage. И еще две проблемы висит — ребут после сна при УЕФИ-загрузке и легаси-бут. Не вижу желающих поработать.
- Еще стоит упомянуть дополнительные вещи: скрипты компиляции, инсталлятор, системные скрипты и приложения, которые имеют прямое отношение к проекту, хотя и не являются частью загрузчика. Вся эта огромная работа практически лежит вне моей компетенции, хотя я и начинал. Основной вклад здесь Jadran, Crasybirdy, JrCs, да и apianti приложился.
- Ну и программа должна быть документирована. Опять-таки кропотливый и не слишком интересный труд. И опять-таки, кроме меня никто не возьмется, ни у нас, ни за рубежом.

Остальные разработчики и вкладчики так или иначе упоминаются в исходниках и в инсталляторе.

Пишу эту главу в расчете на то, что будут еще желающие поработать над проектом, а для этого надо научиться хотя бы компилировать.

4Проекты с открытыми исходниками тем не менее имеют некоторые ограничения в использовании, так сказать “лицензирование”. И, в частности, лицензия GPL примененная для этого проекта, развита в применении к Линуксу. Одно “но”. Эта лицензия плоха и опасна, Интел не рекомендует ее использовать.

Для того, чтобы скомпилировать проект, необходимы еще компилятор, и библиотеки – азбучная истина. Что в данном случае? В роли библиотек выступают исходные модули EDK2. Как это правильно назвать? Framework? Environment? По-русски наверно надо сказать **СРЕДА**. Скачивается с того же сервера. http://sourceforge.net/projects/edk2/ Поскольку весь проект создавался для Хакинтоша и ради Хакинтоша, то в первую очередь рассматривается компиляция проекта в среде MacOSX. Это, однако, не единственная возможность. Сам по себе EDK2 предусматривает компиляцию также в Windows, Linux, и в каких-то других системах и средах. Для Windows нужно иметь Visual Studio 20xx, для MacOSX должен быть установлен Xcode Command Line Tools, а в Linux компилятор gcc входит по-умолчанию. Встроенные средства MacOSX тем не менее недостаточны для компиляции проекта, поэтому предлагается, как и для Linux, скачать и собрать новую версию gcc, используя скрипт buildcc.sh в папке Кловера. Почему не Кланг? Потому что Кланг до сих пор не выдает работоспособных кодов. Ждемс.

Теперь к делу. Читатель, заглянувший в эту главу, не может по определению быть простым юзером, и уж ему не требуется рассказывать, как пользоваться терминалом.

1. Скачиваем исходники и готовим среду:

```
cd ~
mkdir src
cd src
svn co https://edk2.svn.sourceforge.net/svnroot/edk2/trunk/edk2 edk2
cd edk2
make -C BaseTools/Source/C
svn co svn://svn.code.sf.net/p/cloverefiboot/code/ Clover
cd Clover
```

2. Собираем компилятор. gcc-4.7.2 способен компилировать и 32 и 64 бита.

```
./buildgcc.sh –x64 -all
```

3. Исправляем среду EDK2 под наши нужды

```
cp ~/src/edk2/Clover/Patches_for_EDK2/build_rule.txt ~/src/edk2/Conf/
cp ~/src/edk2/Clover/Patches_for_EDK2/tools_def.txt ~/src/edk2/Conf/
```

4. Теперь можно и сам CloverEFI собирать. Например так:

```
./ebuild.sh -64
./ebuild.sh -mc
./ebuild.sh --ia32
```

Созданы и другие скрипты компиляции, как правило самодокументированные. Смотрите, выбирайте, пользуйтесь.

5. Одна тонкость. В репозитарии отсутствуют файлы HFSPlus.efi для 32 и 64 бит, ввиду их приватности. Два варианта: раздобыть эти файлы из других источников, либо изменить определения проекта файлы .fdf таким образом, чтобы вместо приватных драйверов использовались свободные VboxHfs. Этот драйвер работоспособен, но медленный и с недостатками, которые в будущем, возможно будут исправлены.

Было

```
      # foreign file system support
#INF  Clover/VBoxFsDxe/VBoxHfs.inf
INF  RuleOverride=BINARY Clover/HFSPlus/HFSPlus.inf
```

Сделать

```
      # foreign file system support
INF  Clover/VBoxFsDxe/VBoxHfs.inf
#INF  RuleOverride=BINARY Clover/HFSPlus/HFSPlus.inf
```

6. Проект не стоит на месте, поэтому эта инструкция со временем может оказаться неработоспособной из-за какого-то мелкого изменения. Этот проект для тех кто думает, кто сможет разобраться, в чем дело, и как быть.

7. Ну а теперь собираем Инсталлятор

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

Все сделано! Какие-то шаги можно было опустить, если вы делаете для себя и не в первый раз. Например, вместо скачивания всего проекта, сделать просто обновление svn up, исключить 32-битную компиляцию, и не собирать пакет.
Для чайников приготовлены полностью автоматизированные скрипты CloverGrower и его Pro версия. Смешно. Скрипт, чтобы помочь чайнику самому скомпилировать пакет. Чайнику лучше скачать готовый инсталлятор.
А теперь, когда все готово, можно приступать к инсталляции.

Инсталляция

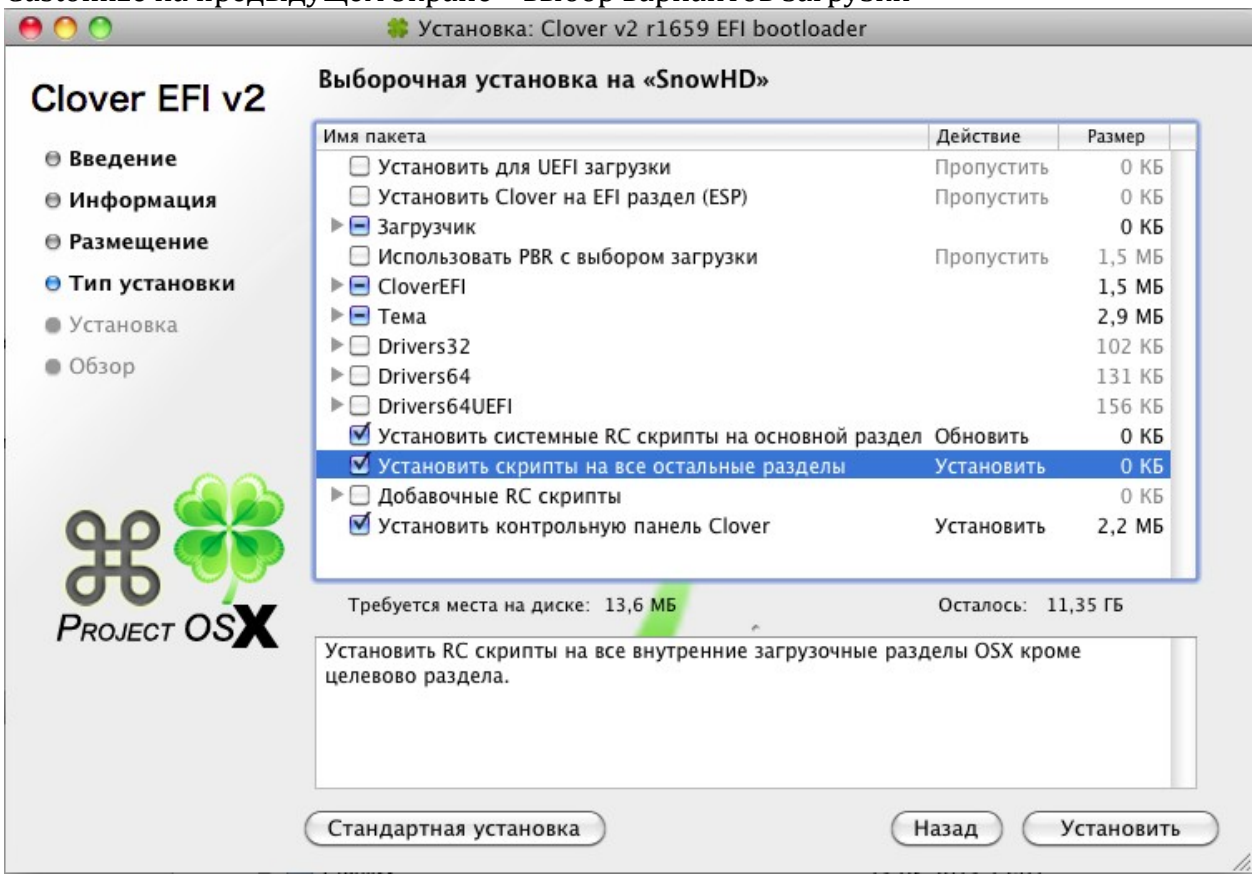
Использование инсталлятора

Для чего сделан инсталлятор? Чтобы инсталлировать программу! Зачем же это делать вручную, инсталлятор все сделает точнее, чем вы сами! Единственное условие, что у вас на этом компьютере уже есть MacOSX. Один из вариантов, что вы запустили установочный DVD с другим загрузчиком, и из интерфейса установки MacOSX запустили инсталлятор. В зависимости от языка ОС инсталлятор будет работать по-русски, по-английски, или даже по-китайски. Здесь приведены инструкции для английского варианта, поскольку по-русски и так разберетесь, а по-китайски и я не знаю. В текущей версии имеется 17 языков, в том числе индонезийский, может кому надо.

Итак,
Следуем по клавишам Continue и OK, читаем и соглашаемся с лицензионными соглашениями (хм, а они там есть?), и приходим к выбору, что мы устанавливаем, куда и зачем.

Change Install Location – выбор куда именно ставить загрузчик

Customize на предыдущем экране – выбор вариантов загрузки



Если поставить курсор на одну из строк, то в нижнем поле будет краткое описание этого варианта.

Установить для UEFI загрузки — этот вариант отменяет установку файлов boot. Кому-то они очень мешают!!!

Install Clover in the ESP (Установить Clover на EFI раздел ESP) — лучший вариант, когда присутствует такой раздел (схема разделов GPT). Инсталлятору не виден этот раздел, поэтому в меню выбора дисков указываем на раздел, который лежит на том же диске, на ESP которого мы хотим поставить загрузчик. Предполагаем, что на этом разделе есть MacOSX, куда будут установлены скрипты, контрольная панель и апдейтер (по-русски слишком длинно «программа автоматического обновления»).

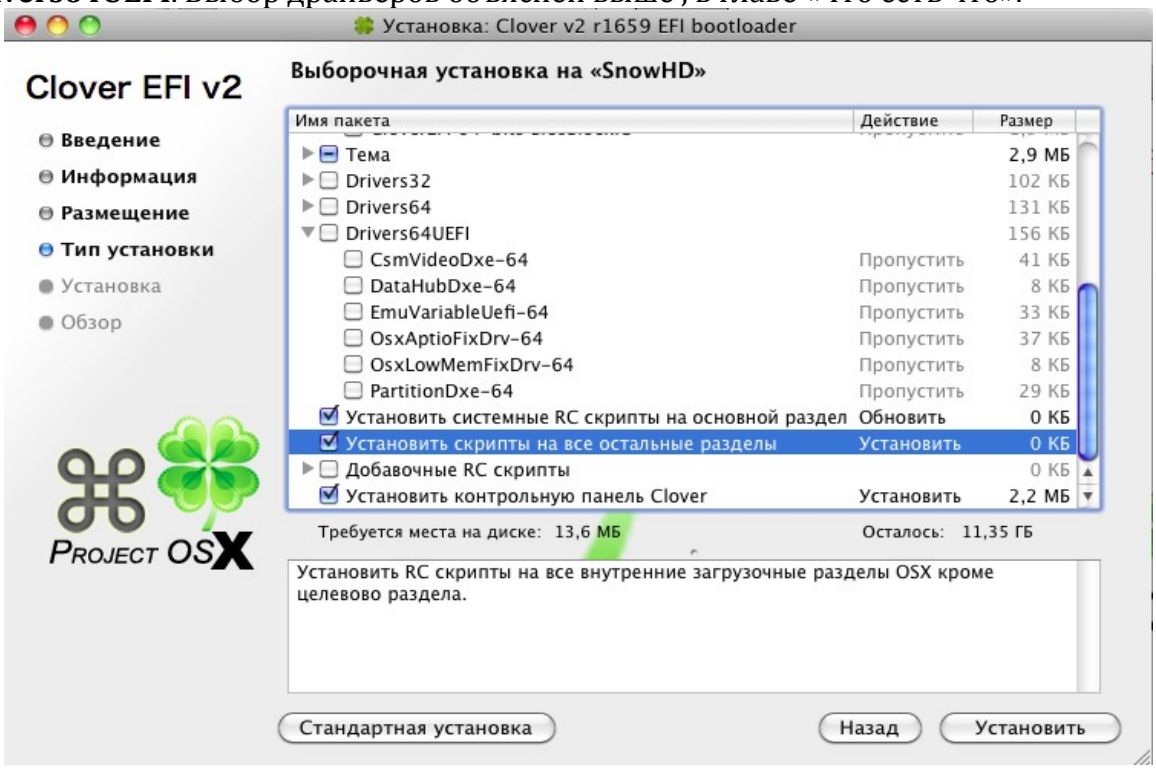
Bootloader (Загрузчик) - это вариант с БИОС (вариант А), при котором используется CloverEFI, или с UEFI (вариант Б).

- **Don't update MBR and PBR sectors** — не обновлять сектора по причине, что они уже есть, или просто для варианта Б;
- **Install boot0af in MBR** – загрузка с использованием boot0af, т.е. поиск активного раздела. Инсталлятор сделает выбранный раздел активным.
- **Install boot0ss in MBR** – загрузка с использованием boot0ss, т.е. поиск раздела HFS+, даже если он неактивный. Инсталлятор не меняет текущий активный раздел. Это сделано для конфигурации с активным Виндоус разделом – ему это надо.

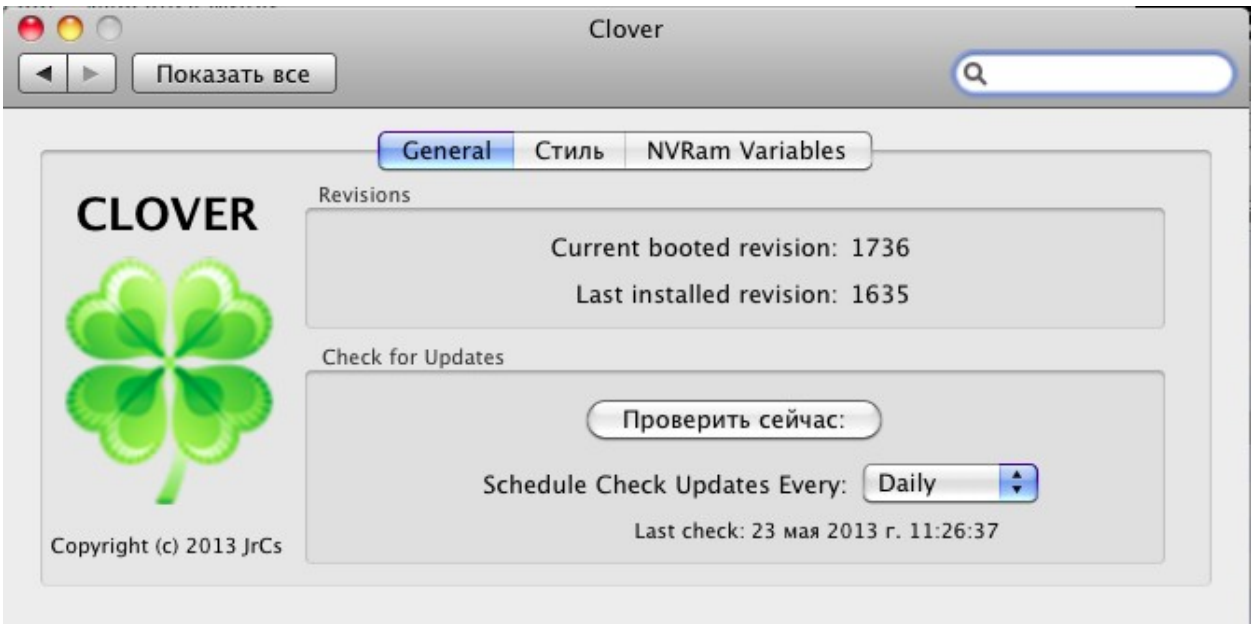
Использовать PBR с выбором загрузки — как говорилось в главе «Что есть что», сектор PBR может быть с паузой для нажатия клавиш 1-9, или без нее. С этой опцией мы установим сектор с паузой.

CloverEFI - это, как видно из списка, выбор битности загрузчика. Либо 32 бита, либо 64 бита. Также здесь специальный вариант **BiosBlockIO**. Это такой вариант

CloverEFI-64, который имеет специальное имя boot7, и предназначен для компьютеров, имеющих нестандартный SATA-контроллер. Драйвер этот работает через БИОС, и, как правило, работает с любым контроллером (БИОС же должен с ними работать!). Но бывают и осечки, например Dell Inspiron 1525.
Drivers64UEFI. Выбор драйверов объяснен выше , в главе «Что есть что».



Установить системные скрипты RC на основной раздел - Это скрипты rc.local и rc.shutdown.local, которые исполняются системой OSX при входе и выходе — необходимая часть всей концепции Кловера. Вы можете их не ставить, если не предполагаете далее использовать Кловер (тогда что вы здесь вообще делаете?).
Установить скрипты на все остальные разделы — если на компьютере больше одного раздела с MacOSX. Инсталлятор достаточно умен, чтобы не ставить их на разделы с Виндоус или Линуксом.
Вы также можете не ставить скрипты, если уверены, что знаете, что вы делаете.
Установить контрольную панель Clover — эта контрольная панель помогает обновлению Кловера, выбору темы и установке NVRAM переменных.



Установка загрузчика вручную

Нужна в двух случаях~~: при ловле блох и при поносе.~~ Во-первых, когда человек хорошо знает, что он делает, и хочет контроллировать каждый шаг, не веря инсталлятору (а зря!), и во-вторых, при установке из-под другой ОС, где запуск инсталлятора невозможен.

OSX

Очень не рекомендуется заниматься этим тому, кто не знает, что такое терминал.
Установка на раздел HFS+ в MBR или гибридной разбивке. Почему MBR? Это очень стандартная ситуация, когда компьютер уже существует, и уже с информацией, ничего терять нельзя, можно только поставить новый загрузчик.

Установка сектора MBR

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

Что в этой команде?
fdisk440 – специальная версия утилиты fdisk, поправленная так, чтобы использовала только 440 байт нулевого сектора, есть сведения, что это необходимо для совместимости с Windows (проблема просыпания), о чем Apple не позаботилась.
boot0 – файл, описанный выше в главе "Что есть что"
rdisk0 – физическое устройство, на которое вы собираетесь ставить загрузчик. Убедитесь, что оно действительно имеет номер 0.
Эти файлы поставляются вместе с Кловером.

Установка сектора PBR

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

boot1h2 – файл сектора PBR для файловой системы HFS+, отличается от аналогичных поддержкой больших файлов boot, и возможностью выбора boot1,3,6 по горячей клавише. Подробности в главе "Что есть что".
rdisk0s9 – девятый раздел на выбранном устройстве… Почему девятый? А чтобы дураки ничего не попортили, тупо повторяя написанные команды, такого раздела наверняка нету. А ставить нужно реальную цифру, например первый раздел.

Ну и после того, как сектора MBR и PBR успешно записаны на выбранное устройство/выбранный раздел, следует этот раздел сделать активным

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

Девятка во второй строке – это опять номер раздела (их всего четыре!) – делайте вывод.
Теперь можно на этот раздел скопировать файл boot и папку EFI в корень раздела.

Установка на раздел FAT32.
В отличии от предыдущего метода здесь есть одна тонкость. Сектор PBR должен содержать геометрию раздела. Эти сведения туда заносятся в процессе разбивки на разделы, поэтому потеря такой информации чревата последствиями. Сам же метод установки сектора усложняется

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

boot1f32alt - уже упоминался в главе "Что есть что" – сектор для установки на раздел FAT32. Но не FAT16! Будьте внимательны!
rdisk1s9 – опять девятый раздел на первом устройстве. Подставьте свои цифры.
Остальные буквы и цифры в этом рецепте обсуждению и пересмотру не подлежат.
Остальные действия аналогичны установке на HFS+.

Linux

Под линуксом также имеется терминал, и почти такие же команды, но установка возможна только на FAT32. Отличия следующие.

- вместо rdisk1 будет sdb – смотрите в вашей версии Линукса точнее.
- вместо fdisk440 для записи MBR нужно использовать тот же 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

Из-под Виндоуз также есть смысл ставить загрузчик только на флешку FAT32, для этого достаточно запустить скрипт

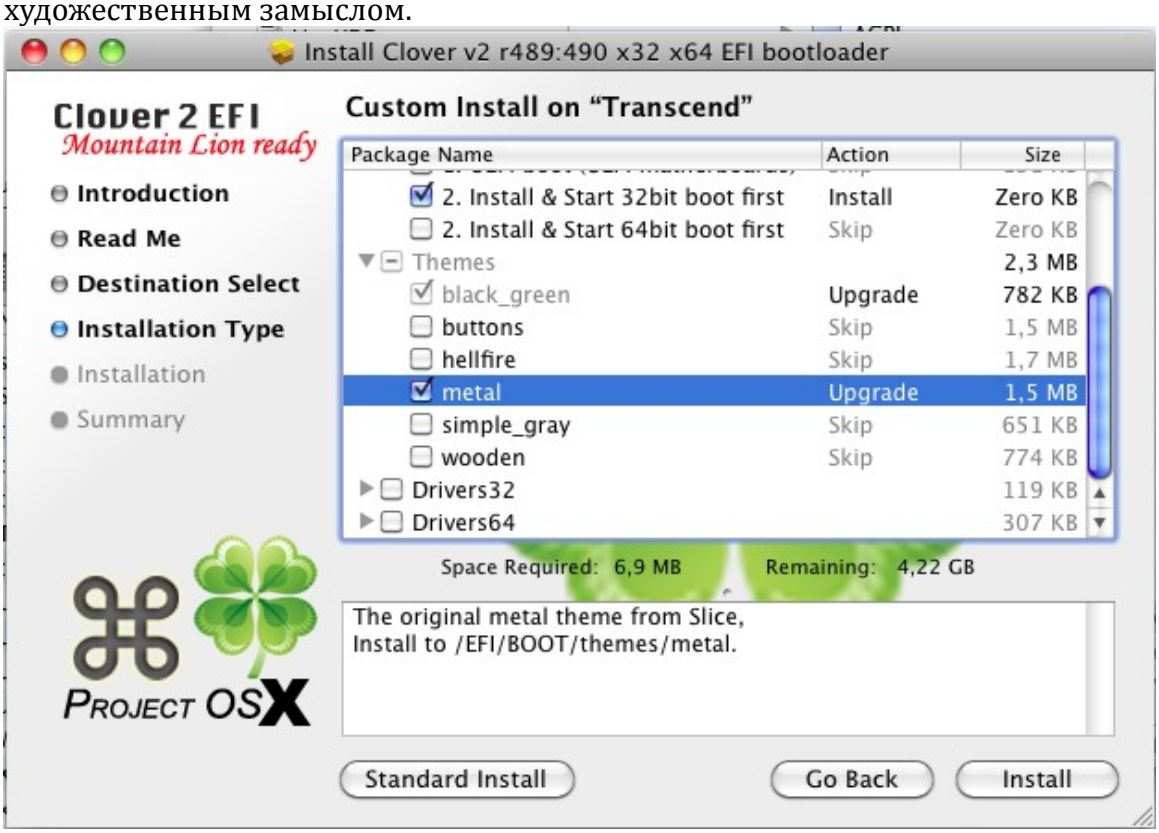
```
makeusb.bat E:
```

где E: - это буква вашей флешки. Наличие нескольких разделов на ней не предполагается. Это же Виндоус!
Все файлы, необходимые для скрипта прилагаются в комплекте с Кловером. Однако, инсталлятор надо предварительно распаковать, либо получить эти файлы непосредственно с svn. Они лежат в папке
edk2/Clover/CloverPackage/CloverV2/BootSectors
После выполнения скрипта флешку нужно извлечь и вставить заново, затем скопировать на нее файл boot и папку EFI.
Еще лучше воспользоваться BootDiskUtility.exe by Cvad которая поможет сформировать флешку из-под Windows.
http://www.applelife.ru/threads/bootdiskutility-exe-создаем-clover-boot-usb-flash-disk-под-windows.37189/

Оформление

Выбор темы

Теперь выбираем тему. Что есть тема? Это элементы оформления: баннер, фоновое изображение, рисунки иконок и кнопок, шрифт, объединенные единым художественным замыслом.



В инсталляторе можно указать и все темы, чтобы позже менять их по настроению. Например, это можно сделать в контрольной панели.
Сделаем краткий обзор тем. Реально набор тем гораздо шире, смотрите на форумах, кто что предлагает.
http://www.applelife.ru/threads/themes-темы-для-загрузчика-clover.36074/
http://www.insanelymac.com/forum/topic/288685-clover-themes/

Темы загрузчика Кловер

Metal. Автор Slice. Тема №1

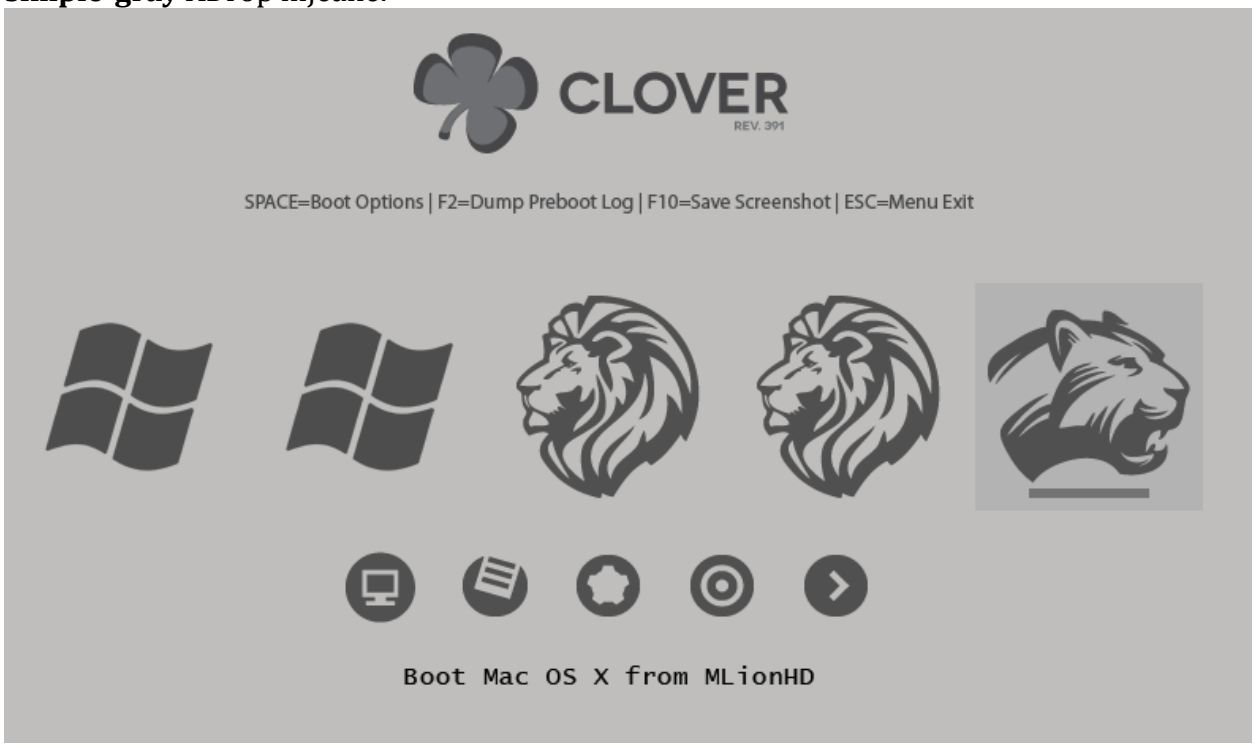
dawn. Автор Slice. Не включена в стандартный инсталлятор

Black-green. Автор blackosx. Тема, идущая в этом инсталляторе по умолчанию.

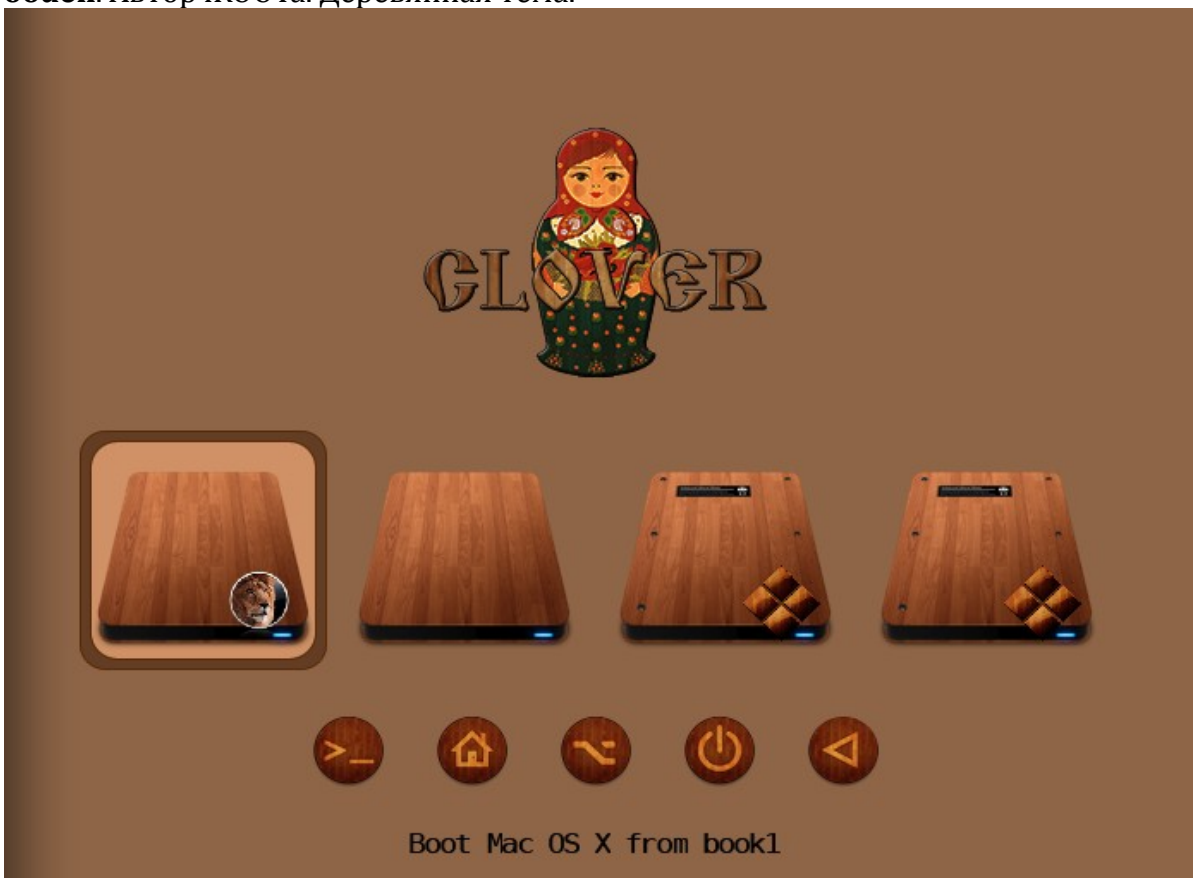
steampunk. Автор Xmedik. Тема включена во второй инсталлятор по умолчанию.

hellfire Автор lllevelll.

simple-gray Автор hijeane.

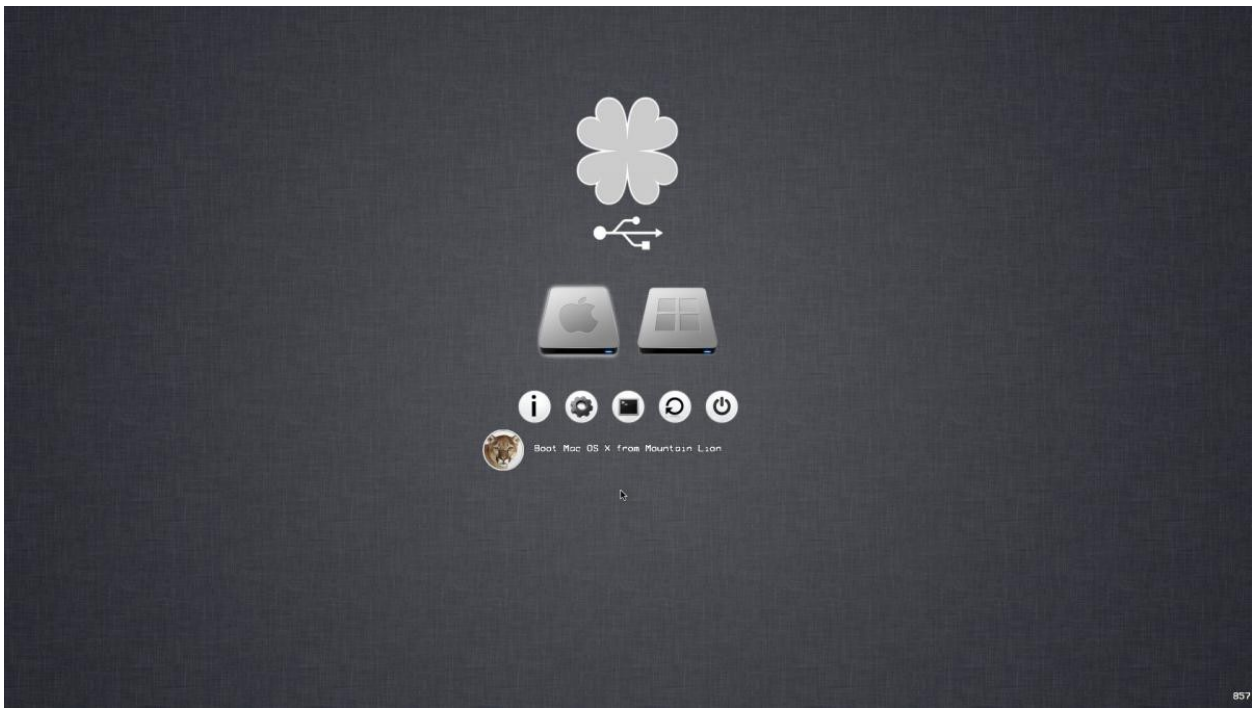


Wooden. Автор iROOTa. Деревянная тема.



aluminium. Автор iROOTa.

iClover. Автор winlog.



AppleStyle by Eps

Настройка интерфейса в config.plist

К темам относится и ряд параметров, прописанных в файле ~~refit.conf, находящимся по адресу EFI/BOOT/~~. Эта инструкция для ревизии 1659 и выше, в котором настройки оформления сделаны по-другому. Для старых версий смотрите старые варианты инструкций.

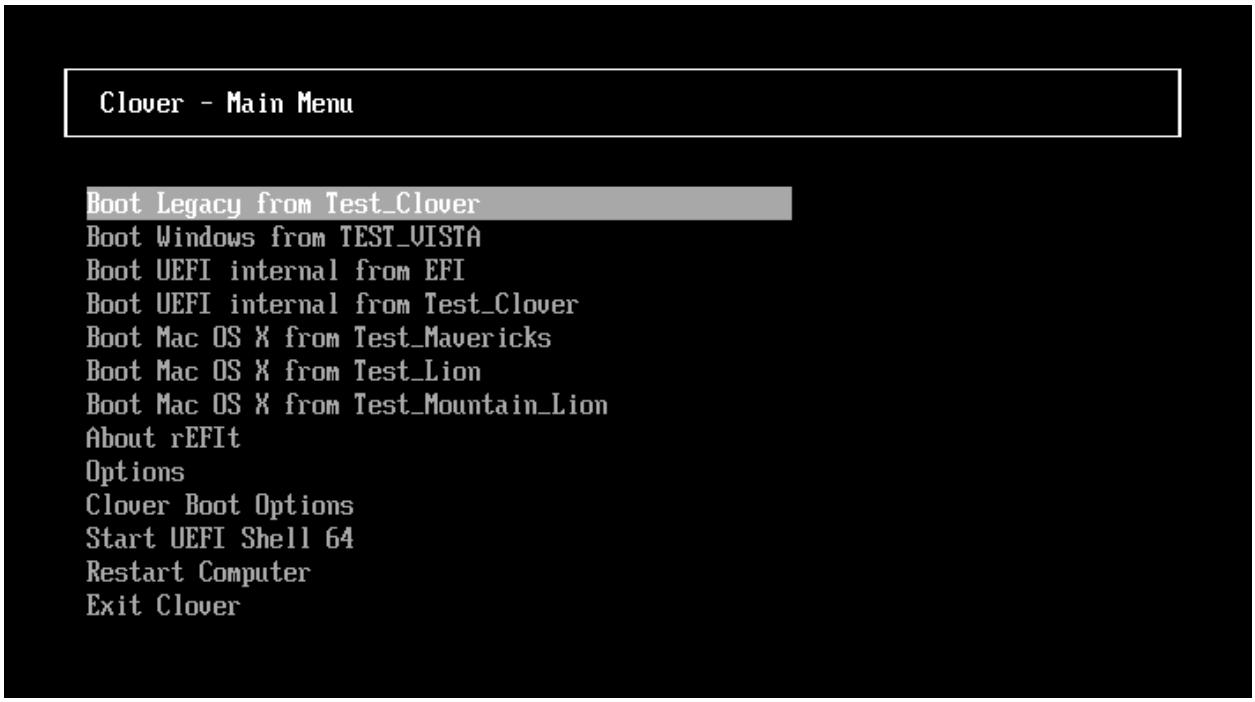
Настройки интерфейса делаются в EFI/CLOVER/config.plist в разделе GUI

```
<key>GUI</key>
<dict>
```

Интерфейс может быть графическим, а может быть и текстовым (начиная с ревизии 1764). Для этого пишется

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

Наверно, только в России живут любители текстового интерфейса, Total Commander, Volkov Commander, DOS и т.д. и т.п. К вашим услугам!



Если стоит false, то Кловер работает в графическом режиме.

Оформление графической оболочки зависит от выбранной темы. Тема по-умолчанию выбирается в переменной

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

Однако, тему можно также выбрать в Контрольной панели, и тот выбор будет определяющим. Если же там указана неправильная тема (нет такого файла theme.plist по указанному пути), то будет выбрана тема из плиста. Если же и там указана несуществующая тема, то на экране будет нечто черно-полосатое, но работоспособное.

Кроме того, в секции GUI определяются параметры поведения интерфейса:

```
<key>Timeout</key>
<integer>5</integer>
```

загрузчик вошел в графический интерфейс и сделал паузу в 5 секунд перед стартом системы по-умолчанию. Если в течении этого времени юзер нажмет какую-либо клавишу, отчет времени прекратится. Варианты: если 0 – ГУИ не вызывается, система сразу стартует, однако, если до этого нажать пробел — зайдем в ГУИ.
-1 (минус один) – загрузчик входит в меню, попыток старта не делает.
Пауза на 25 секунд сделана, чтобы юзер полюбовался анимацией.
Вариант с Timeout=0 можно заменить на вариант

```
<key>FastBoot</key>
<true/>
```

В этом случае производится дополнительная экономия времени загрузки на том, чтобы не загружать интерфейс и его элементы. Т.е. уже без шансов зайти в GUI.
По окончанию отчета времени начнет грузиться система с раздела, заданного в следующем параметре

```
<key>DefaultBootVolume</key>
<string>MacHDD</string>
```

имя раздела, как вы его назвали, как у вас отображается в логе загрузчика. Однако, имя может быть также задано в NVRAM после перезагрузки из контрольной панели «Загрузочный том» («Startup Disk»). Имя, заданное там, учитывается в первую очередь.

```
<key>ScreenResolution</key>
<string>1024x768</string>
```

вы можете установить желаемое разрешение экрана, больше, чем стандартное 1024x768, если в параметрах видеокарты и собственно экрана есть такой режим. Кловер пытается выставить наибольшее возможное разрешение, однако, он может и ошибиться. Проверяйте список доступных режимов по бут-логу.
Если в секции графики стоит PatchVBios=Yes, то у вас появится максимальное разрешение, доступное для данного монитора. В этом случае параметр ScreenResolution может оказаться лишним.

```
<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 — бывают конфигурации, когда мышь не работает, или вообще виснет, ну что ж, тогда ее можно запретить.
Speed 2 — скорость перемещения курсора, разумные значения 2 — 8. Для некоторых мышей требуется отрицательная скорость, перемещение в обратном направлении. Значение 0 означает, что мышь отключена.
Mirror — а также сделать обратное направление только по одной координате.
DoubleClick 500 — пауза в миллисекундах на определение двойного клика. Значение 500 подходило до сих пор всем.

В интерфейсе Кловера можно увидеть легаси и уефи загрузчики для установленных операционных систем. При этом на одном раделе может быть и несколько загрузчиков. Может быть, вам и не нужно все, что нашел Кловер, вам достаточно указания на реальную пару систем. Вы можете скрыть из интерфейса как отдельные разделы, так и целые классы загрузчиков. Следующие разделы в конфиге:

```
<key>Volumes</key>
<dict>
  <key>Legacy</key>
  <string>First</string>
  <key>Hide</key>
  <array>
        <string>WindowsHDD</string>
        <string>HD(1,GPT,E223FF7F-F2DA-4DBB-B765-756F2D95B0FE)</string>
  </array>
</dict>
```

Для **Legacy** (то есть загрузчики, запускаемые из PBR), есть варианты значений No, First, Last — не показывать вообще, расположить в начале списка, или в конце.
Hide — скрыть тома по имени, или по их UUID.

```
<key>HideEntries</key>
<dict>
        <key>OSXInstall</key>
        <true/>
        <key>Recovery</key>
        <true/>
        <key>Duplicate</key>
        <true/>
        <key>WindowsEFI</key>
        <false/>
        <key>Grub</key>
        <false/>
        <key>Gentoo</key>
        <false/>
        <key>Ubuntu</key>
        <false/>
        <key>OpticalUEFI</key>
        <true/>
        <key>InternalUEFI</key>
        <true/>
        <key>ExternalUEFI</key>
        <true/>
</dict>
```

Таким способом вы можете скрыть загрузчики отдельных операционных систем (а зачем тогда вы их ставили?), или служебные дубли, такие как Recovery, UEFI файлы, находящиеся на разных носителях.

Еще один параметр отнесен к разделу GUI, хотя это не о том, как его отображать, а для целей отладки его.

```
<key>DebugLog</key>
<false/>
```
Установка значения в <true/> серьезно замедлит работу, зато дает возможность после перезагрузки узнать, в чем состояла проблема.

Оформление: theme.plist

Теперь собственно оформление в соответствии с выбранной темой. Файл theme.plist грузится из папки с темой, и является уникальным для каждой из них. Путь для темы metal таков:
/EFI/CLOVER/themes/metal/theme.plist
Первые параметры темы это копирайт, типа такого

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

Далее идет секция с параметрами оформления

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

Формат всех упоминаемых картинок PNG, причем желательно с корректным заголовком. К примеру программа «Просмотр» сохраняет файлы в правильном формате.

```
<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>
```

Параметр **Path** задает имя файла (а точнее путь!) в котором лежит фоновое изображение на весь экран. При этом, экран может оказаться меньше или больше изображения, и что с этим делать определяется параметром
Type
Crop — обрезать большое изображение под размер экрана, или заполнить фоном.
Tile — замостить мозаикой из плиток.
Scale — растянуть пропорционально, чтобы изображение заняло весь экран и больше, под обрезку.
При обыкновенном растяжении получаются квадратные пиксели, поэтому обычно применяется некоторое сглаживание, однако, такое сглаживание портит края.
В Кловере сделан детект краев, его величина опередеяется параметром
Sharp
Если 0 — нет детекта, края размыты. Максимальное значение 0xFF = 255 — нет размытия. Также в паре с ним работает параметр
Dark

Если <true/> подразумевается, что у вас темное изображение с белыми линиями, <false/> - светлое изображение с темными линиями.

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

Баннер — это центральная картинка, на нее есть ограничения на размер, зависящие от размеров экрана. К примеру, в теме dawn картинка имеет размер 672 × 190 pixels. Эту цифру можно рассматривать как максимальную. Логотип следует либо сделать непрозрачным, если мы не собираемся использовать фоновый рисунок. Тогда первый пиксель логотипа определяет цвет фона. Либо на логотипе есть непрозрачный элемент на прозрачном фоне, а весь экран покрыт фоновым изображением. Трюк от Eps: левый верхний пиксел сделать непрозрачным на 1%.

```
<key>Selection</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>
</dict>
```

Color — цвет выделения строки в меню. Художник задает цвет в соответствии с общим тоном темы. Значение 0x11223380 означает цвет red=0x11, green=0x22, blue=0x33, alfa=0x80. Последнее число — степень непрозрачности, 0x80 соответствует 50%. 0x00 будет означать отсутствие выделения. 0xFF закроет фоновое изображение (буквы на непрозрачном баре).
Big и **Small** — это рисунки, выделяющие иконки в главном меню в верхнем ряду - большие, и в нижнем — маленькие.

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

Type — тип шрифта. Есть два встроенных шрифта **Alfa** и **Gray**, и десяток загружаемых – **Load**. В этом случае имя файла указывается в следующем параметре **Path - BoG_LucidaConsole_10W_NA.png**
Для каждой темы её автор подобрал шрифт, наиболее соответствующий его замыслу, следует смотреть в прилагаемом файле.
Для названий шрифтов приняты следующие соглашения (blackosx)
BoG — Black On Gray — черный на сером фоне.
LucidaConsole — название оригинального шрифта.
10W — ширина буквы
NA — No Antialiasing. Тоже продумано.

Размер одного символа в файле 16 пикселей, однако, сами символы занимают меньше места, поэтому следующим параметром указывается оптимальная ширина, и это, опять-таки, зависит от замысла автора.
CharWidth 10 — можно использовать ширину, рекомендованную автором шрифта, а можно изменить на свой лад. 9 — потеснее, 11 — пореже.

```
<key>Badges</key>
<dict>
        <key>Show</key>
        <true/>
        <key>Inline</key>
        <true/>
        <key>Swap</key>
        <false/>
</dict>
```

Баджик — это маленький рисунок в правом нижнем углу основной картинки. Первоначально задумано, что основная иконка изображает диск (как в буткампе), а баджик сообщает, какая там операционная система.
Show — показывать ли баджик.
Swap — поменять смысл иконки и баджика. Теперь иконка изображает ОС, а баджик — устройство (в этом случае его и неинтересно показывать).
Inline — показать баджик в строке с информацией о выбранной иконке. Здесь всегда ОС, независимо от параметра Swap. Смотрите скриншот темы iClover.

```
<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>
```

Поскольку меню настроек может оказаться длинее, чем вертикальный размер экрана, то в меню появляется полоса прокрутки (Scroll), ее параметры заданы темой, и есть параметры по-умолчанию, для картинок, включенных в тему.

```
<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/>
```

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

В составе темы могут быть анимированные изображения (клипы). Поддерживается серия PNG рисунков с последовательными номерами.

ID — определяет использование данного клипа.

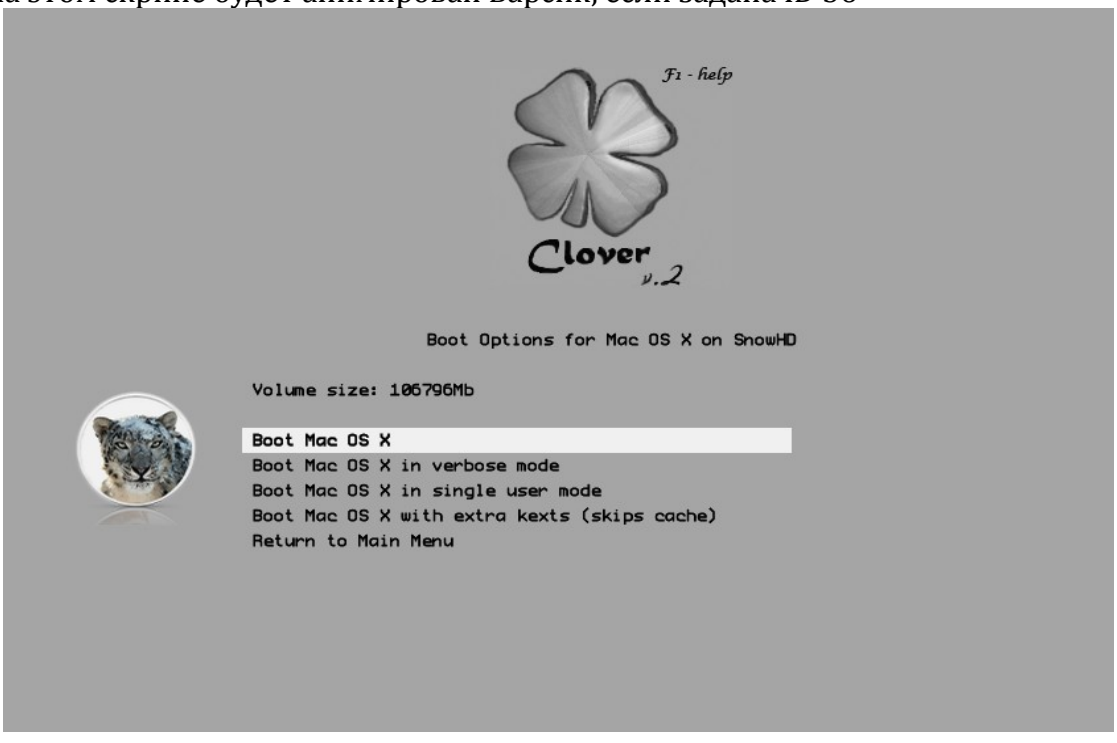
#Logo	(1)
#About	(2)
#Help	(3)
#Options	(4)
#Graphics	(5)
#CPU	(6)
#Binaries	(7)
#DSDT	(8)
#BOOT Sequence	(9)
#SMBIOS	(10)
#Apple	(21)
#WinXP	(22)
#Clover	(23)
#Linux	(24)
#BootX64.efi	(25)
#Vista	(26)
#Recovery	(30)
#Tiger	(34)
#Leopard	(35)
#Snow Leopard	(36)
#Lion	(37)
#Mountain Lion	(38)
#Lynx	(39)

Анимируются заголовочные изображения в каждом субменю, а также эта анимация воспроизводится на выбранном пункте главного меню.

1-10 — список существующих субменю настроек.

21-26, 30-39 — это меню подробностей «Опции загрузки», вызываемом пробелом на иконке в главном меню, либо правым щелчком мыши.

Т.е. на этом скрине будет анимирован Барсик, если задана ID 36



Кто-нибудь сделает тему Full Anime, со всеми анимированными элементами?

Path - ML_Anim **—** Название анимации, определяет имя папки, в которой лежат отдельные кадры с именами
ML_Anim_000.png
ML_Anim_001.png
ML_Anim_008.png
ML_Anim_014.png
В случае пропущенных кадров будет использован последний действующий, т.е. в качестве кадров 002-007 будет использован кадр 001, а в качестве 009-013 — кадр 008. Это удобно, если по сюжету в этот период времени картинка не меняется.
Frames — 15 — общее число кадров в анимации. Недостающие будут заполнены по вышеуказанному алгоритму.
FrameTime - 100 — временной интервал между кадрами в мс. Переменный интервал реализуется с помощью пропущенных кадров.
Once — если указано <true/>, то анимация будет сыграна всего один раз, до выхода из главного меню (щелчок правой клавишей мыши в молоко на главном экране, либо клавиша Escape). Если указано <false/>, то анимация проигрывается по бесконечному циклу, за последним кадром идет нулевой после такого же интервала, без дополнительной паузы.

Конфигурирование аппаратной части

Создание файла config.plist

Вообще, Кловер проделывает конфигурирование автоматически. Но автомат никогда не бывает совершенным, поэтому пользователь имеет возможность менять различные параметры через файл config.plist, либо просто в меню графического интерфейса.
Это файл формата xml, однако, в данный момент удобно его рассматривать как текстовый файл. Редактировать этот файл можно текстовым редактором или специализированной программой типа PListEditor. Вместе в Кловером распространяется два варианта этого файла – максимальный и почти минимальный. Совсем минимальный файл – пустой.
Общее правило – если вы не знаете, какое значение следует дать какому-то параметру, исключите этот параметр вообще из файла. Не оставляйте параметр без значения! И уж тем более, не ставьте значения, которого не понимаете!
Предлагается следующий вариант изготовления такого конфига под свой компьютер:

- установить поставляемый по-умолчанию почти минимальный файл, в нем заложены только безопасные параметры;
- загрузиться в графическую оболочку Кловера и зайти в меню Options (есть такая кнопка в нижнем ряду, или просто по клавише «O»;
- клавишами вверх/вниз/enter/escape погулять по всему меню, и попытаться вникнуть, что там пишут, и зачем;
- что понятно исправляем, непонятное оставляем как есть.
- загружаемся в систему. Если не удалось, повторяем операцию, но уже поменяв параметры, до полного успеха.

Войдя в систему, заходим в терминал, и набираем команду

```
cd ~/App/getconfig >config.plist
```

Предполагая, что вы предварительно положили утилиту getconfig в папку ~/App.

Таким способом вы получите почти полный config.plist с вашими, наиболее удачными параметрами, с которыми вам удалось загрузиться.
Еще немного ручной работы для полного перфекционизма. Ниже приводится описание параметров конфига.
Все параметры объединены по группам: SystemParameters, SMBIOS, CPU, Graphics, PCI, ACPI, KernelAndKextPatches, RtVariables, DisableDrivers.

SystemParameters

<key>boot-args</key>
<string>-v arch=i386</string>

Это аргументы, которые передаются в boot.efi, а он, в свою очередь, часть их передает ядру системы. Конкретный список аргументов ядра следует искать в документации Apple. Список аргументов, необходимых самому boot.efi можно узнать из мануала по com.apple.Boot.plist. Наиболее известны следующие
`Kernel=mach_kernel`
`MKext=extensions.mkext`

Для UEFI загрузки в систему 10.8 или 10.9 необходим ключ slide=0. Начиная с ревизии 1887 он добавляется автоматически.

<key>prev-lang:kbd</key>
<string>ru:0</string>

На данный момент установка языка имеет смысл только для меню "Help" вызываемого по клавише F1. Впрочем, это значение передается в систему, и может повлиять на язык по-умолчанию.

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

Уникальный идентификационный номер вашего компьютера. Если вы не поставите этот ключ, будет сгенерен какой-то из аппаратных сведений, если же вы хотите полный контроль над происходящим, напишите свои 16-чные цифры.
Но, ради бога, не копируйте мои образцовые цифры! Они давно уже не уникальны, дураков полно, которые их скопировали!

<key>InjectSystemID</key>
<false/>

Этот же номер будет инжектирован другим способом, и в свойствах системы будет преобразован во что-то другое. Смысл в этой операции в том, чтобы точно повторить UUID, сгенеренный Хамелеоном. Для этого ставим <true/>, а в качестве CustomUUID используем то значение, которое присутствует с Хамелеоном в реестре IODeviceTree:/efi/platform=>system-id. Тогда в профайлере мы увидим другое значение, но такое же, как и раньше с Хамелеоном.

<key>LegacyBoot</key>
<string>PBR</string>

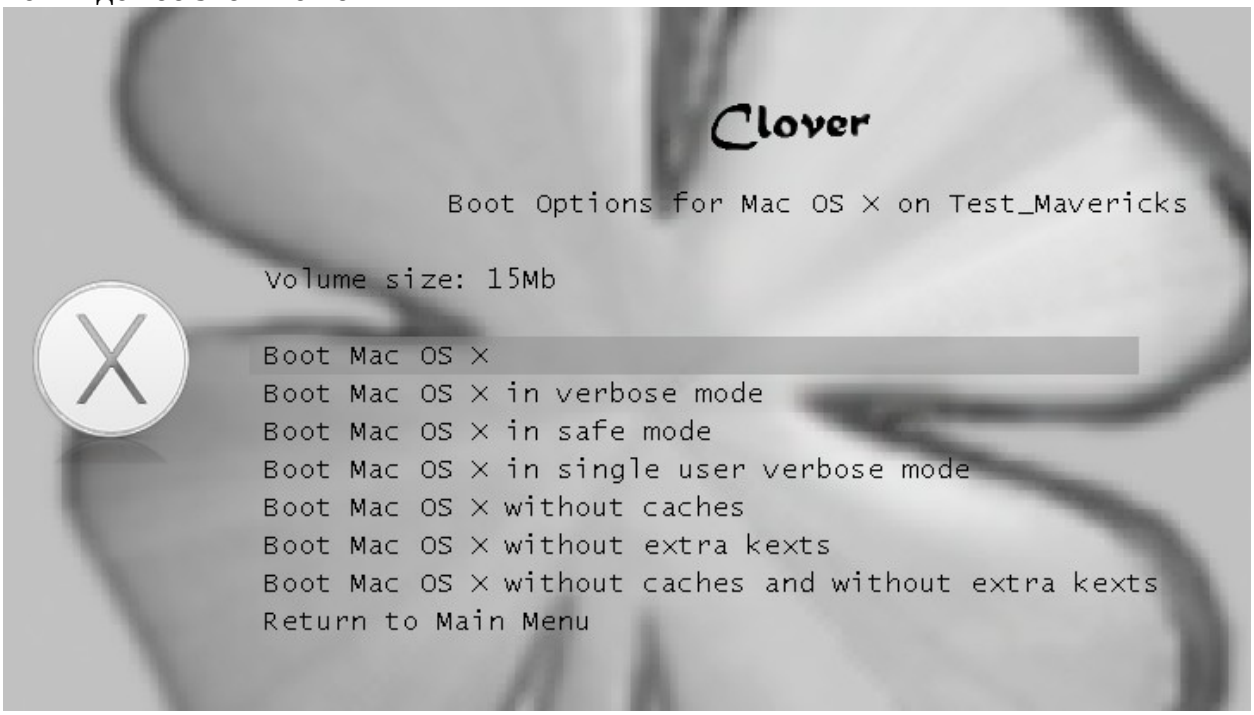
LegacyBoot, необходимый для запуска старых версий Windows и Linux, очень сильно зависит от аппаратной части, от построения БИОСа, поэтому разработаны несколько алгоритмов, и выбор алгоритма производится в этом ключе. Варианты:
LegacyBiosDefault – для тех UEFI BIOS, где есть протокол LegacyBios.
PBRtest , **PBR** – варианты алгоритма PBR boot, кому с каким повезет.

```
<key>BacklightLevel</key>
<string>0x0101</string>
```
Это свойство инжектируется в систему, и система знает о его существовании. Однако влияние заметно только на очень редких конфигурациях. Что это? Яркость монитора… как следует из названия. Это свойство также считывается из NVRAM, и, по-умолчанию, используется то значение, которое выставила система. Значение же, прописаное в конфиге, или выставленное в меню, будет перебивать значение по-умолчанию.

Начиная с ревизии 1865 введены дополнительные ключи Кловера:

```
<key>InjectKexts</key>
<true/>
```
если нужно загрузить дополнительные кексты из EFI/CLOVER/kexts. Если ключ задан, то происходит загрузка кекстов из папки EFI/CLOVER/kexts/10.7 – конкретная папка выбирается в соответствии с версией загружаемой ОС.
Внимание! По умолчанию ключ НЕ задан, и загрузка кекстов НЕ происходит.

```
<key>NoCaches</key>
<false/>
```
Ставим **true** если нужно пересоздать системные кэшы, например, после установки новых драйверов, тогда все кексты загружаются заново. В отличие от системного игнорирования кеша , в этом варианте Кловер сможет применить патч кекстов, предусмотренных в его конфиге.
Какой смысл прописывать этот ключ в конфиг? Он нужен только для однократной загрузки. На иконке ОС нажимаем пробел и выбираем загрузку без кеша.
Речь идет об этом меню:



Эти ключи анализируется драйвером FSInject.efi, его наличие обязательно.

SMBIOS

Эта группа параметров нужна для мимикрии вашего PC под Mac. Кловер это сделает автоматически, основываясь на обнаруженной модели CPU, видеокарте, и признаке мобильности. Однако, вы можете захотеть и другой выбор. Берите программу MacTracker и подбирайте модель Мака, которая вам больше нравится, а затем ищите по интернету, или по знакомым все номера и серийники от этой модели. Комментировать тут особо нечего. Эти параметры не для чайников. Знаете их – меняйте, наугад не получится. Вычислить их тоже нельзя.

<key>ProductName</key>
<string>MacBook1,1</string>
SMBIOS.table1->ProductName
Вы можете указать только имя продукта, и Кловер вычислит по собственным таблицам все остальные параметры, соответствующие этой модели. Остальные параметры можно и не вводить, однако, если вы хотите другие параметры, не по умолчанию, введите и их тоже. Новые параметры будут приоритетнее. Однако, список имен, знакомых Кловеру, ограничен:
"MacBook1,1",
"MacBook2,1",
"MacBook4,1",
"MacBook5,2",
"MacBookPro5,1",
"MacBookPro8,1",
"MacBookPro8,3",
"MacBookPro9,2",
"MacBookAir3,1",
"MacBookAir5,2",
"Macmini2,1",
"Macmini5,1",
"Macmini6,2",
"iMac8,1",
"iMac10,1",
"iMac11,1",
"iMac11,2",
"iMac11,3",
"iMac12,1",
"iMac12,2",
"iMac13,1",
"iMac13,2",
"MacPro3,1",
"MacPro4,1",
"MacPro5,1"
Для других вариантов заполняйте все поля вручную.
Если модель не задана, то Кловер подставит что-то из этого списка, смотрите в меню, насколько вас устраивает такой выбор. Меняйте на свое усмотрение.

<key>SmUUID</key>
<string>00000000-0000-1000-8000-010203040506</string>
SMBIOS.table1->Uuid
Похоже, есть смысл прописать сюда мак-адрес вашей сетевой карты (последние шесть пар символов). Этот GUID также будет использован, если CustomUUID не задан.

<key>FirmwareFeatures</key>
<string>0xC0001403</string>

SMBIOS.table128->FirmwareFeatures
Эти цифры выходят за рамки стандарта SMBIOS, это нечто, специфичное для Эппл. В разных настоящих Маках можно встретить разные цифры, описания нигде никакого нет, разве что в исходниках bless можно найти

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

Следовательно, и нам надо иметь здесь нечетное число.

```
<key>BoardSerialNumber</key>
<string>C02032101R5DC771H</string>
```

SMBIOS.table2->SerialNumber
Этот параметр Кловер поставляет какой-то один определенный. Вы должны подставить свои цифры. Он нужен для того, чтобы работали iCloud и iMessage. Длина обязательно 17 букв, заглавные латинские и цифры. Номер, заложенный в Кловере, скорее всего уже давно забаннен.

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

SMBIOS.table2->BoardType
Этот параметр введен для МакПро, у которого здесь стоит не 10 — Motherboard, а 11 — ProcessorBoard, видимо по историческим причинам. Смысл неочевиден, но на Систем Профайлере это заметно.

```
<key>Mobile</key>
<true/>
```

Вообще-то, Кловер всегда правильно вычисляет, является ли данная платформа мобильной (т.е. с питанием от аккумуляторов, требующее экономии энергии), или же нет. А параметр нужен, если по какой-то причине мы хотим обмануть систему, указать, что никакой батарейки у нас нет, либо наоборот.

```
<key>ChassisType</key>
<string>0x10</string>
```

SMBIOS.table3->Type
Этот параметр служит косвенным указанием, мобильная ли у нас платформа Вот таблица по стандарту SMBIOS

```
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,
```

Кловер подбирает значение, как выставлено в настоящих Маках, в соответствии с выбранной вами моделью. На что это влияет, кроме мобильности — я не знаю.

```
<key>ChassisAssetTag</key>
<string>LatitudeD420</string>
```
SMBIOS.table3->`AssetTag`
Это поле в реальных Маках никогда не заполнено, поэтому мы можем использовать для своих нужд, например, привяжем к проекту HWSensors.

```
<key>Trust</key>
<true/>
```
Параметр служит для разрешения спора между SMBIOS и SPD, чьи параметры памяти признать более точными, помимо того, что там проводятся еще внутренние проверки. По умолчанию идет true.

Если же, ни с true, ни с false вы не можете получить «правильное» отображение памяти в системе, для вас сделана возможность прописать все вручную (начиная с ревизии 1896)

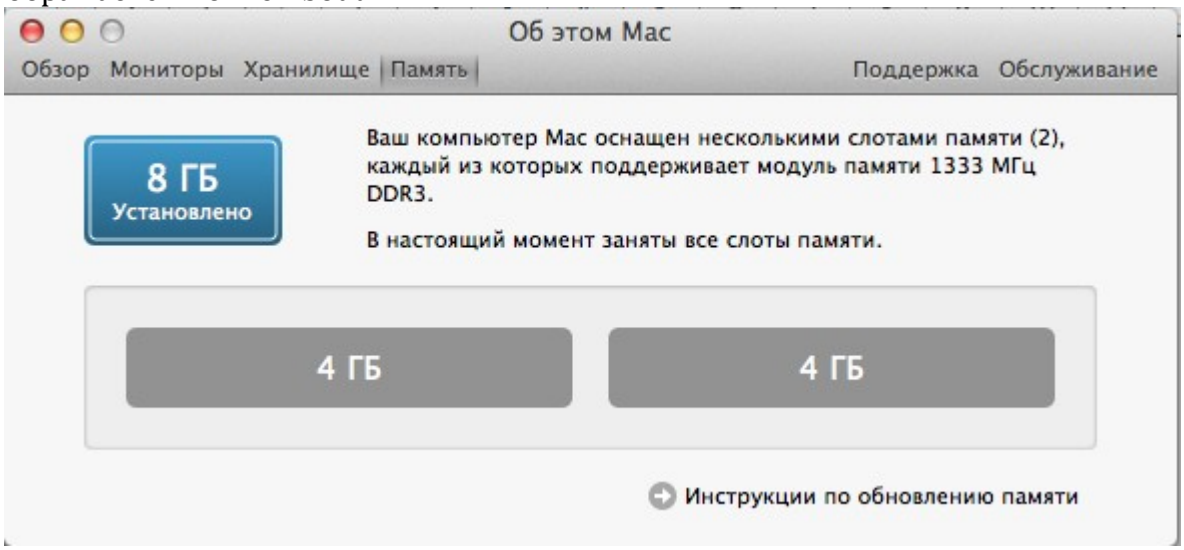
```
<key>Memory</key>
<dict>
  <key>Channels</key>
  <integer>1/2/3/4</integer> OR <string>1/2/3/4</string>
  <key>SlotCount</key>
  <integer>24</integer> OR <string>24</string>
  <key>Modules</key>
  <array>
    <dict>
      <key>Slot</key>
      <integer>0</integer> OR <string>5</string>
      <key>Size</key>
      <integer>2048</integer> OR <string>4096</string>
      <key>Frequency</key>
      <integer>1600</integer> OR <string>1333</string>
      <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>
```

Некоторые пояснения:

Channels — количество каналов памяти. На очень старых компьютерах был один канал. На современных два. Существуют отдельные конфигурации (Кларкдейл, например) где три канала.

SlotCount — общее количество слотов, куда можно вставить планки памяти. Отображается в окне About.



Теперь рисуем массив модулей, описываем только занятые слоты. Пустые даже не упоминаем. В ключе Slot пишем его номер от 0.

Размер пишем в мегабайтах и скорость в мегагерцах. Пустых полей не оставляем.

В серийном номере (Serial) и в инвентарном номере (Part) разрешены только заглавные буквы, цифры, знаки минус и точка.

На этом разрешите закрыть вопрос с правильностью отображения памяти в системе.

CPU

Эта группа параметров помогает с определением ЦПУ, когда внутренние алгоритмы не справляются.

```
<key>Turbo</key>
<string>Yes</string>
```

Есть процессора, которые официально не поддерживают эту технологию. В настоящей версии имеется детект, поддерживает ли ваш процессор Турбо. Смотрите бут-лог. На процессорах ниже Нехалема, Турбо не поддерживается системой, даже если есть на самом процессоре.

В текущей версии Кловера параметр исключен. ЦПУ либо поддерживает турбо режим, либо нет, насильно не изменишь.

```
<key>CpuFrequencyMHz</key>
<string>3200</string>
```

Базовая частота процессора в МГц. Обычно Кловер получает это значение из DMI, но если там неверно, можно подставить через этот ключ. Неправильное значение может вызвать неправильную работу системы – рассинхронизацию, тормоза и т.п. Лучше вообще ничего не пишите, даже самого ключа.
Похоже на то, что этот ключ следует исключить из Кловера, как бесполезный и вредный.

```
<key>BusSpeedkHz</key>
<string>133330</string>
```

Этот параметр – базовая частота шины, является критически важной для работы системы, и передается из загрузчика в ядро. Если частота неправильная, ядро вообще не запускается, если частота немного не соответствует, могут возникнуть проблемы с часами, и очень странное поведение системы.
Значение в DMI хранится в МГц, и это неточно, более правильно вычисленное из частоты ЦПУ, ну а вы можете подобрать значение более точное, и прописать его в этом ключе в килогерцах. К примеру, у меня в ДМИ написано 100МГц, а для часов лучше стало, когда я прописал 99790кГц.
Один момент. Некоторые производители имеют другое понятие, что есть BusSpeed, а что есть FSBSpeed, и вписывают в БИОС значение в четыре раза больше. Разобраться в правильности можно по диапазону: оно должно быть от 100 до 400МГц, либо по формуле ЧастотаЦПУ=ЧастотаШины*МножительЦПУ.
Понятно, что если АСУС пишет частоту шины 1600МГц, да множитель процессора 8, то формула не сойдется, процессоров на 12,8ГГц не существует. Реально надо делить на 4.
Начиная с ревизии 1060 имеется автодетект частоты основанный на АЦПИ таймере, и эти два значения он вычисляет лучше, чем прописано в DMI.
Внимание! При UEFI загрузке используется плохой таймер, грубый, и частота шины получается неточной. CloverEFI дает гораздо более точное значение. Загрузитесь с ним, запишите, и поставьте в конфиге именно это значение для UEFI загрузки.

```
<key>QPI</key>
<string>4800</string>
```

В системном профайлере эта величина называется Processor Bus Speed или просто Bus Speed. В Хамелеоне есть алгоритм ее вычисления для процессоров семейства Nehalem (да и тот неправильный!). В Кловере сделан поправленный алгоритм по даташитам от Интел. В исходниках кекста AppleSmbios рассматриваются два варианта: либо значение уже прописано в SMBIOS, как там изготовитель прописал, либо просто вычисляется BusSpeed*4. После долгих споров эта величина вынесена в конфиг – пишите что вам нравится (МГц). На работу это никак не влияет – чистой воды косметика. По последним сведениям QPI имеет смысл только для Нехалемов, для всех остальных здесь необходимо иметь BusSpeed*4. Или вообще ничего.

```
<key>ProcessorType</key>
<string>0x0201</string>
```

Этот параметр придуман Apple и используется в окошке «Об этом Маке», внутренними средствами переводящим такую константу в обозначение процессора. Иначе будет показан «Неизвестный процессор». ~~Почему нельзя было вызвать CPUID?~~ (потому что был еще PowerPC). Ну или в SMBIOS посмотреть в таблице 4? Нет, у Эппл свое мировоззрение, а нам приходится приспосабливаться, какой процессор как зашифрован. В основном Кловер знает все шифры, но, поскольку прогресс не стоит на месте, то оставлена возможность вручную изменить этот параметр. Правильность

установки этого параметра контролируется в окошке «About this Mac». Опять-таки, косметика чистой воды.

Graphics

Эта группа параметров служит для инжектирования свойств видеокарточки, как это делает, к примеру, Natit.kext. Параметров, которые реально инжектируются много, но это в основном константы, некоторые вычисляемые, некоторые заданы во внутренней таблице, и только совсем отдельные параметры вводятся через конфиг.

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

Собственно включение этой функции инжекции. Кстати, по умолчанию она включена, ибо инжекция должна работать при чистом конфиге – условие запуска системы. Выключать инжекцию стоит в том случае, если вы знаете лучший способ. Для некоторых современных карт, как Нвидия 6xx или Радеон 6xxx, инжекция по умолчанию отключена, потому что работает нативная заводка. Неполноценная, зато на рабочий стол можно войти.

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

Объем видеопамяти в Мб. Вообще-то он и автоматически определяется, но если пропишете правильное значение – никто не пострадает. Реально, однако, я не помню ни одного случая, чтобы этот параметр кому-то в чем-то помог.

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

Загрузка видеобиоса из файла, который должен лежать в папке EFI/CLOVER/OEM/xxx/ROM или EFI/CLOVER/ROM и иметь имя файла vendor_device.rom, например 1002_68d8.rom. Это иногда имеет смысл, если использовать патченный видеобиос. Также бывают проблемы, что видеокарточка не показывает системе свой видеобиос, хотя система и требует, например в случае мобильных радеонов. В этом случае можно поставить этот параметр в Yes, но никакого файла не подсовывать. Кловер возьмет ВидеоБИОС из легаси-памяти по адресу 0xc0000, как ни странно, он там практически всегда есть, и теперь Кловер его инжектирует в систему, и мобильный радеон включается!
Еще уточнение. Оказывается, БИОС, прошитый в РОМ карточки не совпадает с тем, что формируется на адресе 0xc0000 — тень рома. Так вот, нам нужен именно он, теневой, а не тот БИОС, который прожигаем программатором.
Короче. Для мобильных радеоном ставим Yes, хотя никакого файла нет, для остальных карточек No. Других вариантов история не зафиксировала.
А вот и наступили новые времена. Для компьютеров с UEFI-only БИОСом, на легаси-адресе нет никакого ВидеоБиоса. Подкладываем в файле и ждем новых решений.

```
<key>DualLink</key>
<integer>0</integer>
```

По умолчанию инжектируется значение 1, но для некоторых старых конфигураций этот параметр=1 приводит к учетверению экрана. Помогает установка в 0, как в приведенном примере.

```
<key>PatchVBios</key>
<true/>
```

Кловер вносит исправление в тень ВидеоБиоса по адресу 0xC0000, чтобы он поддерживал тот видеорежим, который является максимальным для подключенного монитора. Например, в EDID монитора есть мода 1920x1080, а в ВидеоБиосе такой нету. Кловер пропишет ее в качестве первой моды и запустит в использование. Если монитор сам не формирует EDID, его можно инжектировать, как показано ниже.
Либо для патча используется значение из файла config.plist

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

Если автоматика ошиблась, можно прописать патч ВидеоБиоса вручную, по стандартному алгоритму Найти/Заменить.

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

Можно в одном БИОСе сделать несколько патчей 0,1,2... К примеру успех с Нвидия пришел при четырех патчах.
Этот пример из ВидеоБиоса ATIRadeon HD6670, заменяющий моду 1920x1440 на более приемлимую 1920x1080. При таком способе следует выбирать моду с такой же горизонталью.

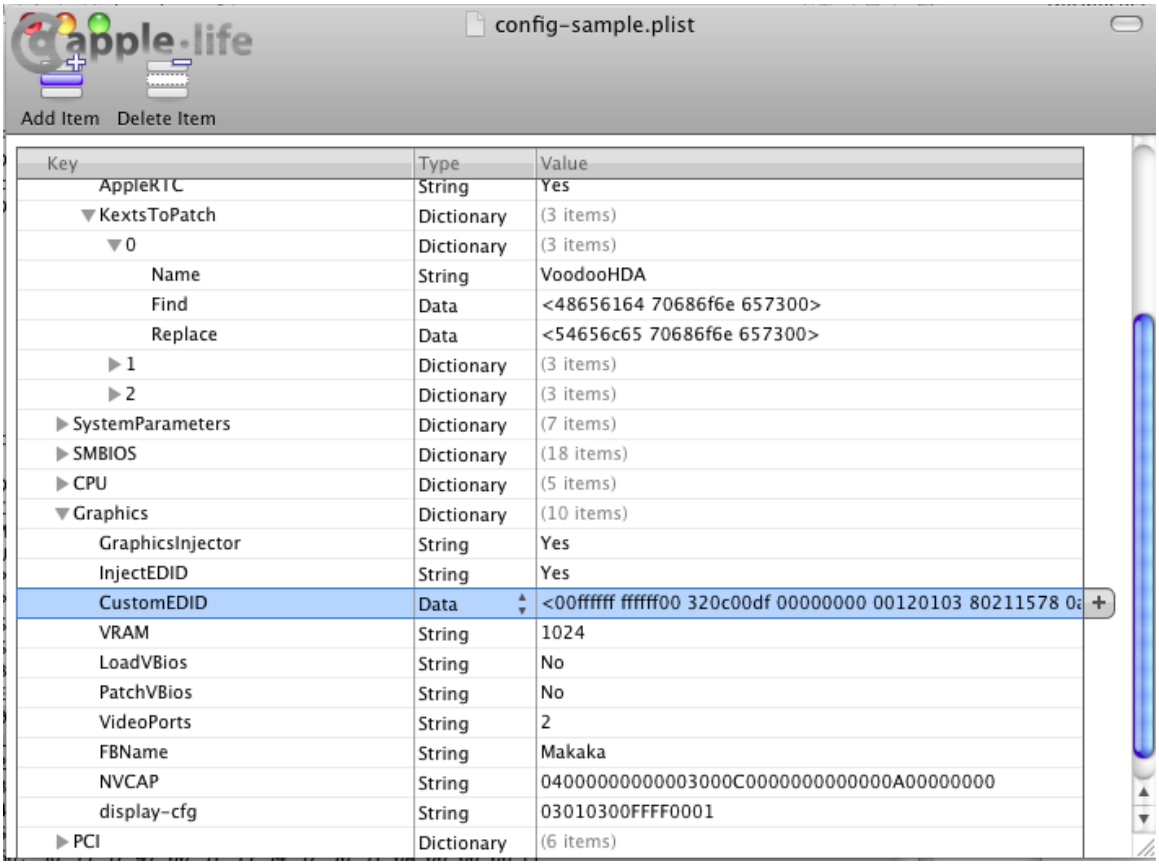
```
<key>InjectEDID</key>
<true/>
```

1. Существуют мониторы без DDC, например панели ноутбуков.
2. Существует варианты, когда EDID есть, но Эппловские драйвера его не видят.

На второй вариант мы ставим просто InjectEDID=yes, и Кловер сам извлечет EDID , и подсунет его драйверам. Необходимость такого действия подмечена в теме про мобильные радеоны. Для компьютеров UEFI-only считаем первый вариант, поскольку Кловер не извлечет ВидеоБиос.
На первый вариант мы еще должны прописать новый EDID ручками. Пишем так:

```
<key>CustomEDID</key>
<data>AP///////wAyDADfAAAAAAASAQOAIRV4CunVmVlTjigm
UFQAAAABAQEBAQEBAQEBAQEBAQEB3iGgcFCEHzAgIFYAS88QAAAY3iGgcFCEHzAgIFYAS
88QAAAAAAAA/gBXNjU3RwAxNTRXUDEKAAAA/gAjMz1IZYSq/wIBCiAgAJo=</data>
```

Кстати, искусственно подсунутый EDID будет также использован для автоматического патча видеобиоса. См.выше.
Откуда взять EDID? Ну раз из этого компа он никак не извлекается, то берем чужой. Главное условие - чтобы было правильное максимальное разрешение.
В свой образец конфиг-плиста я ставлю EDID от Делла Инспирон. Матрица 1440x900.
Буковки в этом примере — это стандартная шифровка XML, если смотреть через PlistEditor, то видим более человеческую картину



Еще вариант изготовления ЕДИДа — воспользоваться программой ViewSonic EDID Editor (версия 3.1.5), которая, при желании, легко портируется в OSX. Но это уже не касается собственно Кловера. Изучайте теорию. Кловер дает вам возможность инжектировать свой EDID, хороший, качественный.

```
<key>VideoPorts</key>
<integer>2</integer>
```

Количество видеовыходов на карте, включая TVO и/или HDMI. Выбранный фрейм из Эпповского списка может не соответствовать нашей реальной карте.

```
<key>FBName</key>
<string>Makaka</string>
```

Этот параметр специфичен для ATI Radeon, к которым имеется три десятка разных фреймбуферов без какой-либо закономерности кому что. Кловер автоматически выбирает из таблицы на большинство известных карточек наиболее подходящее имя. Однако, другие пользователи точно такой же карточки оспаривают, им нужно другое имя. Вот и напишите в этот параметр то, что вам кажется наиболее правильным. Общее правило: не знаете, что писать, вообще сотрите параметр.
Но не пишите уж эту макаку! Специально прописал для абсурда – нет, все равно копируют в свой конфиг!
Простой вариант подбора:
5000 серия: мобильный — Alouatta, десктоп — Baboon
6000 серия: мобильный — Cattail, десктоп — Ipomoea
7000 серия: мобильный — Pondweed, десктоп — Futomaki.

<key>NVCAP</key>
<string>0400000000000003000C0000000000000A00000000</string>

Это параметр для видеокарт NVidia, конфигурирует типы и назначения видеопортов. В этой строке 40 шестнадцатеричных цифр заглавными буквами. Теория здесь отсутствует, есть эмпирика, да еще и с противоречивыми результатами. Есть вот такая табличка, но ее правильность оспаривается.

Yellow = Desktop 1 (primary)	Green = Desktop 2 (secondary)

DVI+VGA

	TV	DVI 2	VGA 2	VGA 1	Bin string	Conversion to hex
Desktop 1	0	0	0	1	1	1
Desktop 2	1	1	1	0	1110	0e
NVCAP	TV + DVI + VGA2			VGA1	04000000 00000100 0e000000 00000007 00000000	

DUAL DVI

	DVI 2	VGA 2	DVI 1	VGA 1	Bin string	Conversion to hex
Desktop 1	0	0	1	1	11	3
Desktop 2	1	1	0	0	1100	0c
NVCAP	DVI2+VGA2		DVI1+VGA1		04000000 00000300 0c000000 00000007 00000000	

DUAL DVI + TV on hardware channel 2 (maybe not supported)

	TV	DVI 2	VGA 2	DVI 1	VGA 1	Bin string	hex
Desktop 1	0	0	0	1	1	11	3
Desktop 2	1	1	1	0	0	11100	1c
NVCAP	DVI2+VGA2+TV			DVI1+VGA1		04000000 00000300 1c000000 00000007 00000000	

DUAL DVI + TV on hardware channel 1 (maybe not supported)

	DVI 2	VGA 2	TV	DVI 1	VGA 1	Bin string	hex
Desktop 1	1	1	0	0	0	11000	18
Desktop 2	0	0	1	1	1	111	7
NVCAP	DVI2+VGA2		DVI1+VGA1+TV			04000000 00001800 07000000 00000007 00000000	

The hardware channel and desktops are intentionnaly swapped so that TV out stays in use for secondary desktop.

DUAL DVI + TV on hardware channel 1 & 2

	TV2	DVI2	VGA2	TV1	DVI1	VGA1	Bin string	to hex
Desktop 1	0	0	0	1	1	1	111	7
Desktop 2	1	1	1	0	0	0	111000	38
NVCAP	TV2+DVI2+VGA2			TV1+DVI1+VGA1			04000000 00000700 38000000 00000007 00000000	

Laptop with VGA + TV out

	TV	DVI 2	VGA 2	LVDS	Bin string	Conversion to hex
Desktop 1	0	0	0	1	1	1
Desktop 2	1	0	1	0	1010	5
NVCAP	External VGA+TV			LCD	04000000 00000100 05000000 00000007 00000000	

Laptop with DVI + TV out

	TV	DVI 2	VGA 2	LVDS	Bin string	Conversion to hex
Desktop 1	0	0	0	1	1	1
Desktop 2	1	1	1	0	1110	0e
NVCAP	External DVI+VGA+TV			LCD	04000000 00000100 0e000000 00000007 00000000	

На форумах можно найти другие способы вычисления правильного значения этой строки. А Кловер и сам пытается вычислить из БИОСа.

<key>display-cfg</key>
<string>03010300FFFF0001</string>

Это тоже параметр только для карт NVidia. Подробности смотрите в обсуждениях http://www.projectosx.com/forum/index.php?showtopic=1105

Однако, сведения, приведенные там являются спорными. Реальные конфиги можно посмотреть в теме http://www.projectosx.com/forum/index.php?showtopic=370
А вообще-то, конфиг по-умолчанию, который создает Кловер, похоже и является лучшим вариантом. Просто не указывайте вообще этот параметр, дайте возможность Кловеру его вычислить.

```
<key>ig-platform-id</key>
<string>0x01620005</string>
```

Этот параметр необходим для запуска видеокарточки Intel HD4000, спор о конкретных значениях не привел к единому правилу, поэтому параметр просто вынесен в конфиг — подбирайте. Кстати, Кловер и сам предложит некое значение.

KernelAndKextPatches

Эта группа параметров для осуществления бинарных патчей на лету. Надо заметить, что это осуществимо, только если загрузка происходит через kernelcache либо через параметр **NoCache**. Если кеш не загрузился по другим причинам, то эти фиксы не работают.

```
<key>Debug</key>
<true/>
```

Если вы захотите понаблюдать на ходом, как происходит патч кекстов. Вообще-то, этот ключ для разработчиков.

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

Предотвращает панику ядра на неподдерживаемом ЦПУ, в частности Yonah, Atom, Haswell для старых систем.
Нужно понимать, что в ядре есть и другие алгоритмы, которые будут неправильно работать с неподдерживаемым ЦПУ, поэтому не ждите, что этот патч решит все ваши проблемы. Очень сомнительно, что это будет работать с Pentium M, Pentium 4 или AMD, для таких случаев лучше все же найти специально сделанное ядро.

```
<key>AsusAICPUPM</key>
<true/>
```

Оказывается, БИОС на материнских платах АСУС (который раз нам АСУС портит настроение?) что-то делает, что MSR регистр 0xE2 становится ReadOnly, но он используется в кексте **AppleIntelCPUPowerManagement**, причем используется по записи. Авторы этого фикса не придумали ничего лучшего, как исправить сам кекст, ибо вернуть регистру E2 его былую функциональность можно только перезагрузкой. Ставьте Yes, если при старте системы вы имеете панику на этот кекст. (да, регистр E2 имеет свойство WriteOnce, т.е. записать в него можно только один раз до перезагрузки). Актуально для процессоров Sandy и Ivy Bridge. Либо перепрошивайте БИОС. А как другие операционки в этом случае?

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

Операционная система OSX как-то не так работает с CMOS, как это предусмотрено BIOSом, в результате при пробуждении из сна или при перезагрузке происходит сброс CMOS. Не у всех, больше в этом грехе замечены платы от Gigabyte. Более того, часто эта проблема решается просто патчем DSDT: Device(RTC) что делает и Кловер.

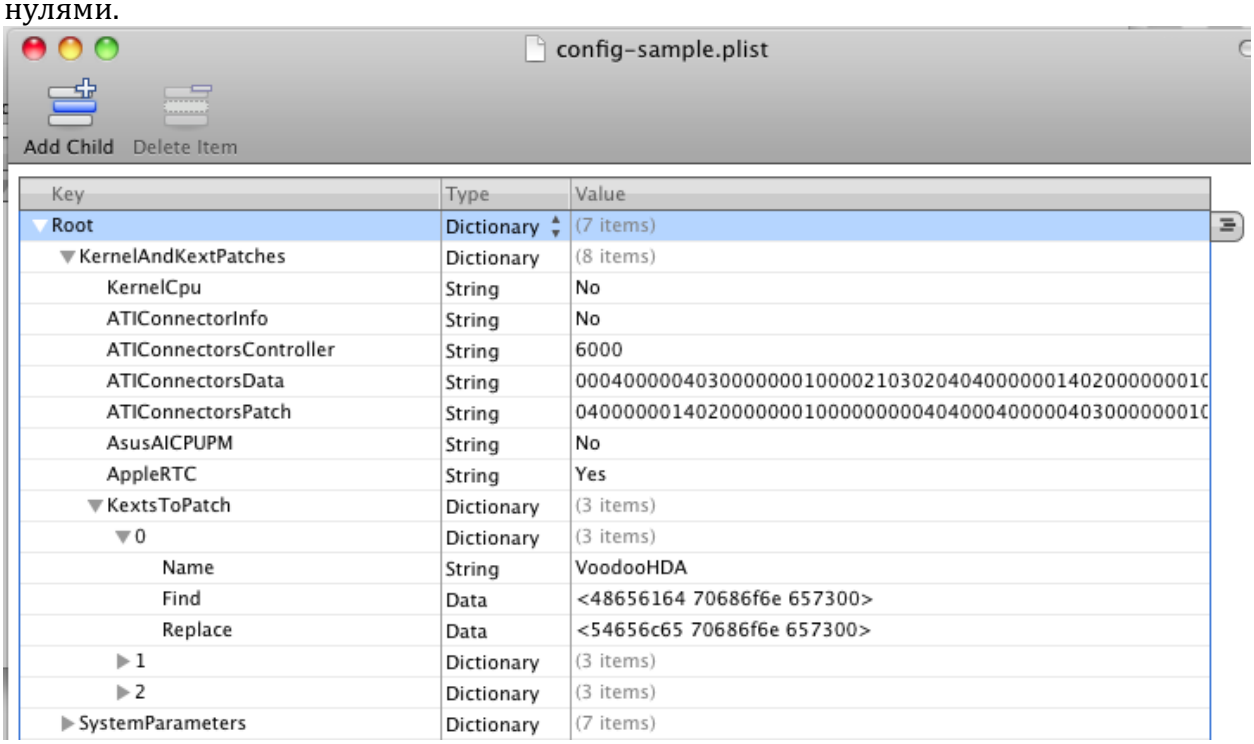
Однако, в некоторых случаях и этот патч не помогает. Тогда можно поправить сам кекст AppleRTC, что здесь и делается.

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

На ноутбуках HP есть проблема с lapic, которая решается запуском с cpus=1, или теперь с этим патчем <true/>

```
<key>KextsToPatch</key>
<array>
```

Помимо специфических патчей, можно сделать патч любого другого кекста, принцип простой: 16 ричная строка, что искать, и строка, на что заменить. Образец: патчим VoodooHDA на предмет замены названия Headphones на Telephones.
Условие — количество букв должно быть таким же. Либо меньше и дополнить нулями.



Этот метод успешно применяется для включения поддержки Trim для SSD
http://www.applelife.ru/threads/clover.32052/page-539#post-310105
Вот еще один очень полезный патч: борьба с желтыми иконками и нерабочим DVD проигрывателем (который не работает для внешних приводов):
Оригинальная тема http://www.applelife.ru/threads/Меняем-external-на-internal.38111/

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

Чтобы выбрать модель MacPro4,1 или 5,1, не имея памяти с ECC. AppleTyMCEDriver patch

```
<key>Name</key>
<string>AppleTyMCEDriver</string>
<key>Find</key>
<data>cgoATWFjUHJvNCwxAE1hY1BybzUsMQBY</data>
<key>Replace</key>
<data>cgoAAAAAAAAAAAAAAAAAAAAAAAAAABY</data>
```

Бывает необходимость править не бинарную часть кекста, а его info.plist. В этом случае секция выглядит следующим образом

```
<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>
```

Здесь есть одно осложнение. Патч предполагается делать в кернелкэше, но, если мы делаем патч инфо-плиста, чтобы кекст грузился, там этого кекста еще нет, поскольку он еще не загрузился. Поэтому загружаться нужно дважды. Первый раз с игнорированием кэша (ключ NoCache), тогда FSInject загрузит этот кекст, и второй раз уже с кешем, где он и будет успешно патчиться

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

Для полноценного запуска карточек ATI (AMD) Radeon 5000 и 6000 серий недостаточно инжектировать свойства в реестр, необходимо еще подкорректировать коннекторы в соответствующем контроллере. В данном случае указываем на 6000 контроллер. Следующие два свойства указывают, что найти, и на что изменить.

```
<key>ATIConnectorsData</key>
<string>00040000040300000001000021030204040000001402000000010000000004031000000010000000000010000000000001</string>
```

```
<key>ATIConnectorsPatch</key>
<string>040000001402000000010000000004040004000004030000000100001102010500000000000000000000000000000000</string>
```

Этот метод работает только для систем 10.7 и выше.
Расскажу подробнее, как получить эти цифры.
Оригинальная статья от bcc9
http://www.insanelymac.com/forum/index.php?showtopic=249642
Полный рецепт от Xmedik на русском языке с обсуждениями
http://www.applelife.ru/threads/Завод-ati-hd-6xxx-5xxx-4xxx.28890/
Здесь изложу короче, с учетом специфики Кловера.

1. Прежде всего, надо получить свой видеобиос. Загрузиться в CloverGUI и нажать F6. Ваш Биос будет сохранен в файле /EFI/CLOVER/misc/c0000.bin, если, конечно, Кловер установлен в раздел с файловой системой FAT32.

2. Загрузите по одной из этих ссылок программу radeon_bios_decode. В ту же папку с этой утилитой положите файл биоса c0000.bin. Допустим, это папка `~/RadeonPatch`

   Выполняем в терминале следующие команды
   ```
   cd ~/RadeonPatch
   ./radeon_bios_decode < c0000.bin
   ```
3. На экране вы получите информацию по вашим коннекторам, которую стоит скопировать/сфотографировать для дальнейшего использования.
   Вот что у меня
   ```
   iMac:test slice$ ./radeon_bios_decode <c0000.bin
   ATOM BIOS Rom:
       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. Загрузите по одной из ссылок скрипт **ati-personality.pl**
5. Положите в эту же папку, и выполните в терминале
   ```
   perl ati-personality.pl -386 >frames.txt
   ```
   если вы делает это для 32-битной системы, или
   ```
   perl ati-personality.pl >frames.txt
   ```
   для 64-битной.

6. Теперь нужно определиться с выбором подходящего фреймбуфера. Эппл предлагает нам широкий выбор: и птички, и рыбки, и даже обезьяны. Но реальные отличия там в основном в коннекторах, которые мы и собираемся изменить. Если не слишком задумываться, то минимальные рекомендации следующие:
   ATI5000 – мобильный Galago, десктопный – Baboon
   ATI6000 – мобильный Cattail, десктопный - Ipomoea
7. Для выбранного фреймбуфера берем распечатку коннекторов из нашего файла frames.txt, полученного на шаге 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**
   Красным цветом выделены цифры, которые необходимо править. Синие цифры – просто адреса, нужно отбросить. Третья цифра с конца – encoderid, последняя цифра — senseid. Первые 4 цифры в каждой строке – тип монитора (точнее сказать тип коннектора).

```
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
00 08 00 00  HDMI
```

8. senseid мы получили на шаге 3 для каждого из наших коннекторов. encoder можно просто всюду занулить. На остальные цифры не обращаем внимания. Получаем следующую таблицу:

```
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
```

   T.e. Первая строка DVI-D, вторая – VGA, третья – HDMI, и все с моими значениями senseid.

9. Отбросив синие цифры остальные вписываем в config.plist без пробелов и переносов строк. Исходная таблица в ATIConnectorsData, после наших правок в ATIConnectorsPatch. Смотрите образец выше по тексту.

PCI

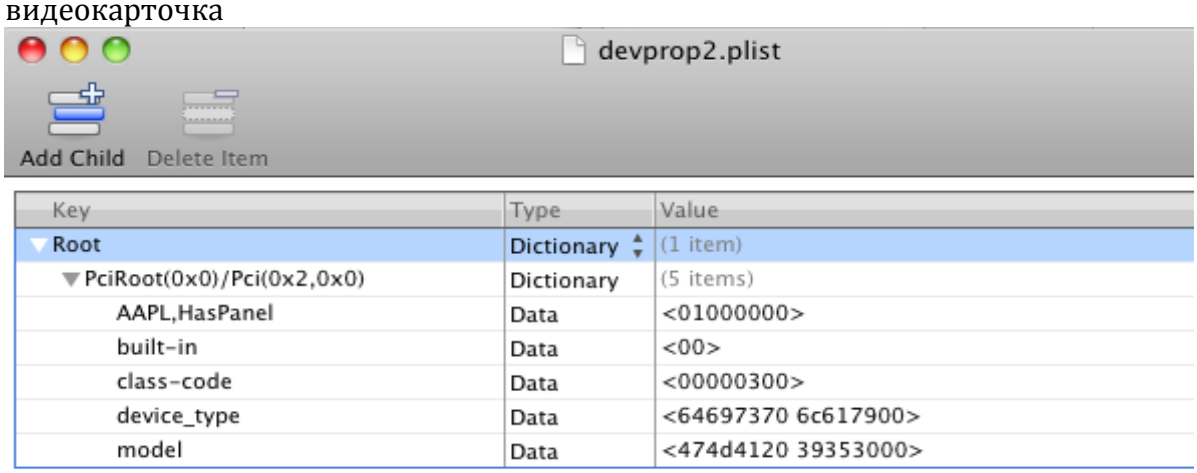
Группа параметров для остальных PCI устройств и шины вообще.

<key>StringInjector</key>
<false/>

Поставить это значение в Yes –вся внутренняя инжекция заменяется на ввод единой строки DeviceProperties

<key>DeviceProperties</key>
<string>0207364862FA54HG345</string>

Как сделать свою строку? Для этого вам нужен gfxutil, который не входит в стандартную поставку Кловера, зато входит в комплект DarwinDamper.
Сначала надо создать нужный xml файл с группами параметров, имеющими в качестве заголовка DevicePath по стандартной нотификации, соответствующий устройству, чьи свойства вы собрались инжектировать. Вот, к примеру, встроенная видеокарточка



Можно, к примеру, с помощью DarwinDamper посмотреть, какой плист сгенерировал Кловер по-умолчанию, затем поправить его, преобразовать в вид 16-чной строки командой

```
./gfxutil -i xml -o hex devprop.plist devprop.hex
```

и получится текстовый файл вида

```
d3000000010000000100000c700000005000000002010c00d041030a000000000101060000027fff04001
00000006d006f00640065006c0000000c000000474d4120393530001c000000640065007600690063006 5
005f007400790070006500000000c000000646973706c617900200000004100410050004c002c004800610
073005000610006e0065006c000000080000000100000016000000620075006900 6c0074002d0069006e00
000005000000001a00000063006c006100730073002d0063006f006400650000000800000000000300
```

Этот текст нужно скопировать в значение параметра DeviceProperties.
В принципе, такой же результат достигается и вставкой методов _DSM в DSDT, если он уже есть, и если заниматься его совершенствованиями. Кому как. Если же ДСДТ еще нет, а готовые строки для ваших карточек уже есть, то почему бы и не воспользоваться таким методом?

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

Оказывается, инжекция свойств видеокарты зависит от того, какое число стоит в DevicePath=PciRoot(0x0) или PciRoot(0x1). Ранее считалось, что это аппаратная характеристика. Однако, еще на заре хакинтошестроения выяснилось, что это число – просто идентификатор, прописанный в DSDT. Вот здесь:

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

    Name (_UID, Zero)
```

_UID=Zero – значит 0, если же равно **One**, значит 1.
Причем, если это число поменять насильно, оно поменяется, и будет успешно работать. Так вот, настоящие Маки имеют всегда 0. И соответственно boot.efi всегда предполагает 0, поэтому лучше, если поправить свой ДСДТ, чтобы там тоже получился 0, и ничего не писать в конфиге, значение 0 стоит по-умолчанию. И в таком раскладе наличие этого параметра следует считать устаревшим методом.

```
<key>HDAInjection</key>
<string>Detect</string>
```

Инжекция свойств звуковой карточки. Правда, это работает, только если устройство в DSDT называется HDEF, если же заниматься его переименованием, то и остальное можно другим способом инжектировать. Также эти усилия не нужны при использовании драйвера VoodooHDA.
Варианты следующие:
NO – ничего не инжектируется.
Detect – автоматическое определение установленной звуковой микросхемы, чтобы ее ID употребить в качестве лейаута. Вообще-то бред, но очень популярный. Во многих случаях не мешает, и оказывает влияние на отображание звуковой карточки в Систем-Профайлере.
883 – в десятичном виде номер лейаута. Имеется ввиду Realtek ALC883.
0x373 – тоже самое в 16-чном виде становится неузнаваемым.
На самом деле эти числа неправильные, правильный лэйаут 12 = 0x0C, но, как ни странно, являются допустимыми.

<key>USBInjection</key>
<true/>

Можно поставить No, если вы почему-то хотите отказаться от инжекции свойств USB. В некоторых конфигурациях они не нужны. Инжекция свойств ЮСБ также отключится, если прописана маска патча ДСДТ 0x1000. Смысл в том, чтобы не дублировать такую инжекцию и патч в ДСДТ.

<key>USBFixOwnership</key>
<true/>

Может быть, мы хотим оставить патчи ЮСБ, но только выключить фикс оунества. Для УЕФИ загрузки он вроде и не актуален.

<key>InjectClockID</key>
<true/>

В версиях до 904 этот параметр был Yes по умолчанию, затем стал No по умолчанию, но на некоторых конфигурациях без него нельзя. Настраивайте по-своему.
Yes — хороший, глубокий сон, из него нельзя вывести компьютер клавиатурой или мышкой.
No — если повезет, то компьютер будет спать, но пробудить его можно просто клавиатурой или мышкой. А у меня он просыпается от встроенной веб-камеры, то есть сразу после засыпания.

<key>LpcTune</key>
<true/>

Разрешать или нет инициализацию чипа контроллера LPC. Согласно документу от Интел в чипсете ICH8M есть устройство LPC controller, в котором регистр A0 имеет следующие значения битов:
12 -> 0 = Enable C4
11 –>1 = Enable C6
7 –>1 = Enter C4 when C3 invoked – выполнить C4, когда запрошен C3.
3 –>1 = Intel SpeedStep Enable — не путайте с Enchanced IntelSpeedstep Tecnology (EIST) — тот, который связан с C- и P-states. Это не оно!!!
Эти биты определены только для мобильного чипсета, поэтому на десктопах параметр не имеет смысла. А если честно, то и на ноутбуках влияния не замечено.

RtVariables

Эта два параметра введены начиная с ревизии 980 и предназначена для разрешения регистрации в сервисе iMessage.
Начиная с ревизии 1129 параметры берутся из СМБИОС, здесь не нужны.
MLB = BoardSerialNumber
ROM = последние цифры SmUUID

<key>MLB</key>
<string>XXXXXXXXXX</string>

Цифры и буквы, длиной 17 знаков, означающие серийный номер материнской платы. Закономерности нету. Самое надежное, взять реальный номер, и поменять средние цифры, например, написать ...SLICE... У кого какая фантазия.

<key>ROM</key>
<data>AAAAAAAA</data>

Двенадцать 16-чных цифр, часто совпадающих с Мак-адресом сетевой карты. Есть, однако, сообщения, что сервис работает с произвольными цифрами.

Ну и самое главное, регистрация в iMessage подразумевает платный сервис, вы должны указать реальную банковскую карту, с которой у вас будет списано 1$. Кто пытается войти нахаляву, получают сообщения типа «Позвоните в Эппл».

```
<key>MountEFI</key>
<false/>
```

Этот параметр сообщает стартовому скрипту, что при входе в систему следует смонтировать раздел ESP (EFI System Partition). Этот параметр для большинства людей является ненужным или временным, стоит в конфиге прописать No, а в меню при необходимости ставить Yes. Еще возможное значение disk1, если у вас несколько дисков, и на каждом есть свой раздел EFI.

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

Лог загрузки нужен разработчикам, а простым пользователям можно и No поставить. Здесь вместо Yes может быть число, сколько логов хранить в системе.

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

Количество строк в этом логе, дальше идет вытеснение старых строк новыми, чтобы не было неограниченного роста этого файла.

DisableDrivers

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

Суть этой секции в том, чтобы иметь разные config.plist в разных OEM папках, но, поскольку папка /drivers64UEFI общая, надо как-то различать, какой набор драйверов используется на той или иной конфигурации. На одной, к примеру, нужен OsxAptioFixDxe, на другой нужен EmuVariableDxe.

ACPI

Группа параметров, регулирующих коррекцию различных ACPI таблиц. И дело не только в том, что у Мака свои требования, но и просто разные версии АЦПИ спецификации, и элементарная лень производителей, и просто в БИОСе системной платы нет сведений об установленных картах и ЦПУ (а динамически определить слабо? Кловер же это делает!).

```
<key>DropOemSSDT</key>
<true/>
```

Поскольку мы собираемся создавать или подгружать динамически свои таблицы SSDT, то нужно избежать ненужного пересечения интересов. Этот параметр позволяет отбросить все родные таблицы, в пользу новых. У вас есть и такой вариант: подложить родные таблицы с небольшими правками в папку EFI/OEM/xxx/ACPI/patched/, а ~~непатченные дропнануть~~. (фу!) неисправленные таблицы отбросить.

Можно отбросить и другие таблицы, при условии, что вы знаете, что вы делаете.

<key>DropAPIC</key>
<false/>

<key>DropMCFG</key>
<false/>

Фантастика, но с моделью MacMini удается загрузиться только если дроппануть MCFG! Аналогично замечено с MacBookPro. Точной причины пока не известно.

<key>DropHPET</key>
<false/>

<key>DropBGRT</key>
<false/>

<key>DropECDT</key>
<false/>

<key>DropDMAR</key>
<true/>

А вот это уже особый вариант. Он помогает, когда есть ошибка с технологией VT-d.
http://www.applelife.ru/threads/Исправляем-pcifamily.17764/page-10

<key>GenerateCStates</key>
<true/>

Автоматическая генерация таблицы SSDT, дополняющей процессорную секцию методами _CST для каждого процессора. На формирование самого метода _CST влияют параметры **EnableC2**, **EnableC4**, **EnableC6**, **EnableISS, C3Latency.** Собственно эти параметры нечего комментировать, включено или нет, все равно все работает, так что экспериментируйте. Кроме того, Кловер уже вычислил, какой имеется процессор, и сколько у него ядер. То, что этот параметр вступил в силу, контроллируется по кернел-логу. Без этого метода присутствует ошибка *ACPI_SMC_PlatformPlugin::pushCPU_CSTData - _CST evaluation failed*.

Отдельное слово про

<key>C3Latency</key>
<string>0x03E9</string>

Это задержка на включение C3 state. Критическое значение 0x3E8=1000. Меньше — включается спидстеп, больше — не включается. На нативниках всегда 0x03E9, то есть спидстеп не работает. На Хаках приходится выбирать, что мы хотим, быть похожим на нативника, или включить управление питанием. Разумное значение во втором случае — 0x00FA, как встречается на некоторых ноутбуках.

<key>GeneratePStates</key>
<string>Yes</string>

Автоматическая генерации таблицы SSDT, дополняющей процессорную секцию методами _PPC, _PCT и _PSS.
_PCT – Performance Control – контроль за управлением спидстепом.
_PPC – Performance Present Capabilities – возможности спидстепа, Эта функция возвращает одно число, которое означает ограничение частоты. Подробности ниже, в параметре **PLimitDict**.

_PSS – Performance Supported States – набор возможных состояний процессора – P-states. Этот массив формируется на основе данных о процессоре, которые Кловер уже вычислил, а также с учетом параметров пользователя **PLimitDict, UnderVoltStep** и **Turbo**.

```
<key>PLimitDict</key>
<string>1</string>
```

Суть параметра очень проста – ограничить максимальную частоту процессора. Значение 0 – работа до максимума, 1 – на одну ступень меньше максимума, 2 – на две ступени. Пример: Core2Duo T8300 2400MHz работает на максимальной частоте 2000, если ограничить на две ступени. Зачем? Да чтобы ноутбук не перегревался, там возможности ЦПУ намного превышают возможности по охлаждению. Точно такой же параметр присутствует в платформ-плистах, например:
System/Library/Extensions/IOPlatformPluginFamily.kext/Contents/PlugIns/ACPI_SMC_PlatformPlugin.kext/Contents/Resources/MacBook5_1.plist
Далее мы еще обсудим эти плисты.
Для некоторых процессоров, например Core2Quad, замечено, что PlimitDict работает наоборот, и лучший вариант =1. Вполне возможно, что это просто ошибка в ДСДТ.

```
<key>UnderVoltStep</key>
<string>1</string>
```

Дополнительный параметр для снижения температуры процессора путем снижения его рабочего напряжения. Возможные значения 0, 1, 2, 3 … чем больше, тем сильнее охлаждаем, пока компьютер не повиснет. В этом месте работает защита против дурака, Кловер не позволит поставить значение вне допустимого диапазона, вернее пишите, что хотите, а работать будет только то, что дозволено. Впрочем и дозволенные значения могут давать неустойчивую работу. Эффект от этого параметра реально наблюдается.

Для процессоров IvyBridge законы спидстепа немного поменялись, в Кловер введены новые параметры и новые инструкции. Обобщенно все, что нужно, достигается одним ключом:

```
<key>GenerateIvyStates</key>
<string>Yes</string>
```

Разработка этого параметра еще не закончена, в ближайшем будущем возможны изменения функциональности. Сам по себе это не параметр, это сборник всех параметров, как их нужно выставить по-умолчанию конкретно для таких процессоров. Я, к примеру, еще не решил, а какие могут быть мотивы поставить этот параметр в No, если известно, что процессор IvyBridge?
Когда стоит Yes, следующие параметры принимают значение по умолчанию:

```
gSettings.GeneratePStates = TRUE;
gSettings.GenerateCStates = TRUE;
gSettings.EnableISS       = FALSE;
gSettings.PluginType      = 1;
gSettings.MinMultiplier   = 7;
gSettings.DoubleFirstState = FALSE;
gSettings.DropSSDT        = TRUE;
gSettings.C3Latency       = 0x3E7;
```

```
<key>DoubleFirstState</key>
<true/>
```

Найдено, что для успешного спидстепа нужно в таблице P-states продублировать первый стейт. После введения других параметров необходимость этого стала сомнительной.

```
<key>MinMultiplier</key>
<integer>7</integer>
```

Минимальный множитель процессора. Сам он рапортует, что 16, и предпочитает работать на частоте 1600, однако, для спидстепа следует задать в таблице стейты вниз до 800 или даже 700. Эмпирика.

```
<key>MaxMultiplier</key>
<integer>30</integer>
```

Введено по аналогии с минимальным, но, похоже, зря. Его не стоит вписывать.

```
<key>PluginType</key>
<integer>0</integer>
```

Для процессоров IvyBridge нужно ставить 1, для остальных 0.

```
<key>ResetAddress</key>
<string>0x64</string>
```

```
<key>ResetValue</key>
<string>0xFE</string>
```

Эти два параметра служат для одного очень ценного фикса – исправление рестарта. Эти значения должны быть в таблице FADT, но почему-то они там не всегда есть, более того, бывает и сама таблица короче необходимого, короче настолько, что эти значения оказались отброшены. По умолчанию идет эта пара **0x64/0xFE** что означает рестарт через PS2 контроллер. Практика показала, что это не у всех работает, другая возможная пара значений **0x0CF9/0x06,** что означает рестарт через PCI шину. Эта пара используется и на нативнике, но не всегда работает на хакинтошах. Разница понятна, на хакинтошах есть еще и PS2 контроллер, который может помешать рестарту, если его не сбросить. Эти два параметра можно поставить в **ноль**. В этом случае, если в FACP присутствует значение, оно не изменяется, если его нет, будет вписано значение по-умолчанию.

```
<key>smartUPS</key>
<string>No</string>
```

Вообще-то, этот параметр предназначен для того, чтобы прописать в таблице FADT профиль питания=3. Логика следующая:
PM=1 – desktop , питание от сети
PM=2 – notebook, питание от сети или от батарейки
PM=3 – server, питание от SmartUPS, про который MacOSX тоже что-то знает.
Выбор между 1 и 2 Кловер сделает на основе анализа бита мобильности, но также есть и параметр **Mobile** в секции SMBIOS. Можно, к примеру, сказать, что у нас МакМини, и что он мобильный. Значение же 3 будет подставлено, если **smartUPS**=Yes.

```
<key>PatchAPIC</key>
<string>No</string>
```

На некоторых компьютерах можно загрузить систему только с cpus=1, либо со специальным патченным ядром (Lapic NMI patch). Простейший анализ показал, что у них неправильная таблица MADT, а именно, в ней отсутствуют разделы NMI. Этот

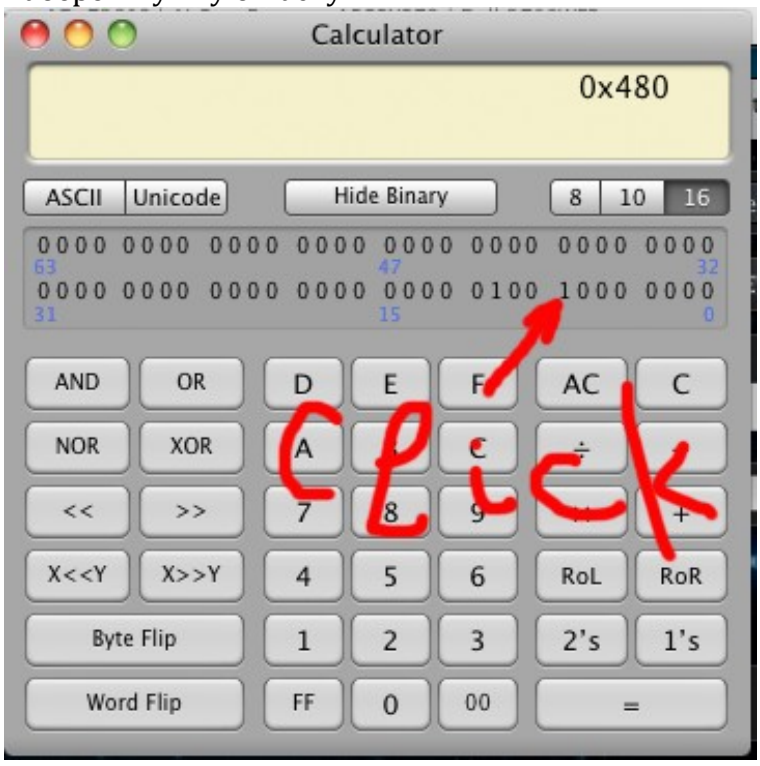
параметр служит для корректировки таких таблиц на лету. Для здорового компьютера ничего плохого не произойдет.

```
<key>FixDsdtMask</key>
<string>0xFFFF</string>
```

Под этим параметром скрывается сразу 16 патчей для таблицы DSDT, по числу битов в маске FFFF. Перечень патчей следующий

```
//0x00FF
#define FIX_DTGP      bit(0)
#define FIX_WARNING   bit(1)
#define FIX_SHUTDOWN  bit(2)
#define FIX_MCHC      bit(3)
#define FIX_HPET      bit(4)
#define FIX_LPC       bit(5)
#define FIX_IPIC      bit(6)
#define FIX_SBUS      bit(7)
//0xFF00
#define FIX_DISPLAY   bit(8)
#define FIX_IDE       bit(9)
#define FIX_SATA      bit(10)
#define FIX_FIREWIRE  bit(11)
#define FIX_USB       bit(12)
#define FIX_LAN       bit(13)
#define FIX_WIFI      bit(14)
#define FIX_HDA       bit(15)
```

Чтобы подсчитать, как сумма битов складывается в ту или иную маску, можно вызвать системный калькулятор, перевести его в вид для программиста, и переключить на 16-чные числа. А теперь щелкая мышью по битам с 0 по 15 мы наберем нужную маску.



Есть более наглядный вариант: CloverFixDsdtMaskCalculator by Cvad

http://www.applelife.ru/attachments/cloverfixdsdtmaskcalculator-app-zip.43973/

А вот для того, чтобы рассказать, что означают эти фиксы, придется открыть новую главу.

Корректировка DSDT

DSDT - Differentiated System Description Table – самая большая и самая сложная АЦПИ таблица. Минимальная длина 36 байт, реальная – 20кб, бывают варианты и больше. Эта таблица описывает устройства и методы доступа к ним. Методы доступа могут содержать арифметические и логические выражения, и, таким образом, представляют собой программу на некоем языке программирования, похожим на С своими фигурными скобками. Исправлять эту таблицу значит что-то понимать в программировании. Кловер предлагает некий вариант автоматической правки, но надо понимать, что искусственный интеллект еще не создан, и автоматическая корректировка программ пока еще далека до совершенства. Человек сделает лучше.

А зачем ее нужно исправлять? Весь патч ДСДТ, со времен его основания был нацелен в первую очередь на исправление устройства HPET – High Precision Events Timer. Дело в том, что в системе OSX присутствует кекст AppleIntelCPUPowerManagement, который служит чтобы управлять питанием процессора (спидстеп), и которому строго необходимо, чтобы в системе был HPET, имеющий прерывания IRQ0 и 8. Без этого условия кекст уходит в панику. Работать можно только запретив, или удалив этот кекст. Но есть и другой вариант – скорректировать DSDT, и устройство HPET включится как положено!

Это – патч **№1**, жизненная необходимость. Только ли МакОСу нужен этот HPET? Нет, конечно, но производители БИОСов только еще начинают это осознавать и прописывать правильные параметры, до сих пор редко встретишь, чтобы ДСДТ работал без патча.

Момент **№2**. В ДСДТ можно разглядеть некоторые зависимости от операционной системы, "Windows 98", "Windows 2001", "Windows 2006", "Linux", МакОС имеет идентификатор "Darwin", и, как правило, на него ДСДТ не рассчитан. А

даже если и рассчитан, то на версию типа FreeBSD. MacOSX является серьезной ACPI системой, т.е. использует ДСДТ по максимуму, так, как его использует Windows 2001, но не Linux, не Windows 98, и не Windows 2006. Правильнее всего сделать мимикрию под Windows 2001. И даже если у вас уже есть "Darwin", добейтесь, чтобы он работал как "Windows 2001". **На многих БИОСах этому соответствует значение OSYS = 0x07D2. Но не 7D6, не 7D9, и уж никак не 0x2410, как это прописано в нативнике.**

Момент **№3**. Производитель системной платы, а с ней и его БИОСа, и его ДСДТ, не может предусмотреть, какой процессор будет установлен, какая видеокарточка и другие PCI устройства. А ведь их стоит прописать в ДСДТ! И наоборот, исключить из ДСДТ такие устройства как спикер, флоппи дисковод, параллельный порт. Драйверов на них нет и не нужны. Также часто необходимо добавлять или убавлять коннекторы/порты у каких-то устройств, например, у видеокарты, или у SATA контроллера.

DSDT лежит в БИОСе и используется в системе в бинарном коде AML, существует компилятор/декомпиллятор IASL, который переводит коды в понятный для человека язык DSL. Человеческий путь правки такой AML->DSL->edit->DSL->AML. И тут возникает момент **№4**. Последняя компиляция становится невозможной из-за ошибок, синтаксических и логических, изначально присутствующих в OEM DSDT. В процессе правки требуется и их исправлять. Ну а заодно исправить и смысловые ошибки, из-за который, например, компьютер не может заснуть, или не может проснуться. А может еще и новые устройства прописать. (А вообще странно, но компиляция/декомпиляция не являются строго обратными операциями, туда-обратно меняет таблицу, а то и вообще, туда идет, обратно нет — требуется вмешательство. Взглядом с моей колокольни это означает, что декомпилятор написан с ошибками, такие уж программисты над ним работали. А несоответствие стандартам нужно отмечать как варнинги, а не эроры).

Когда мы проделали весь этот путь, мы можем подсунуть загрузчику наш исправленный ДСДТ, положив его в папку /EFI/CLOVER/OEM/xxx/ACPI/patched, или, если OEM имя компьютера еще не известно, в папку /EFI/CLOVER/ACPI/patched, или загружаемая система сама имеет свой вариант ДСДТ, лежащий в корне системного диска.

Где взять исходный ДСДТ, который необходимо патчить? Есть варианты добыть его используя Windows, Linux или даже OSX. Если Кловер как-то удалось запустить, но теперь он и сам предоставляет такую возможность. Надо войти в графическое меню и нажать клавишу **F4**. Если Кловер установлен на раздел FAT32, то ему удастся сохранить все OEM ACPI таблицы, включая нетронутые DSDT и FADT. Будьте терпеливы, если сохранение происходит на флешку, и таблиц много, процесс может занять заметное время. В текущей ревизии Кловер при таком способе извлекает набор таблиц, ранее недоступных другими способами, в том числе в AIDAe. Есть также способ сохранить на диске вариант препатченного DSDT. Для этого, в интерфейсе Кловера входим в Options Menu, меняем маску DSDT, затем выходим из меню и нажимаем **F5**. Кловер сохранит ваш ДСДТ, поправленный на текущую маску, с именем типа **DSDT-F597.aml**, т.е. патченный с маской 0xF597. Можно сделать несколько вариантов, чтобы потом сравнивать.

Теперь можно брать DSDT файл и редактировать… Ну а для тех, кто не силен в языке DSL, Кловер предлагает проделать некоторые фиксы автоматически. Рассмотрим подробнее.

FIX_DTGP bit(0)

Для описания свойств устройства, кроме варианта DeviceProperties, рассмотренном выше, есть вариант с методом _DSM, прописанным в DSDT.
_DSM - Device Specific Method – хорошо известна заготовка этого метода, который работает в MacOSX начиная в версии 10.5, этот метод содержит массив с описанием устройства и вызов универсального метода **DTGP**, который един для всех устройств. Данный фикс просто добавляет этот метод, чтобы потом его применять для других фиксов. Самостоятельного значения не имеет.

FIX_WARNING bit(1)

Мировой накопленный опыт по корректировке DSDT содержит ряд типичных ошибок типичных производителей (в основном АСУС, чьи ДСДТ на редкость кривые). Перечислять, пожалуй, не будем. Фикс рекомендуемый для всех. Этот фикс включает в себя также фикс Darwin, как сказано выше.

FIX_SHUTDOWN bit(2)

В функцию _PTS добавляется условие: если аргумент = 5 (выключение), то никаких других действий делать не надо. Странно, а почему? Тем не менее, есть неоднократные подтверждения эффективности это патча для плат АСУС, может и для других. В некоторых ДСДТ такая проверка уже есть, в этом случае такой фикс следует отключить.

FIX_MCHC bit(3)

Автор всей методики патча ДСДТ — pcj - поставил этот фикс для себя. Он создаёт устройство с DeviceID=0x0044, что соответствует Intel Clarkdale. Такое устройство класса 0x060000, как правило, отсутствует в ДСДТ, но для некоторых чипсетов это устройство является обслуживаемым, а поэтому его нужно прописать, чтобы правильно развести управление питанием PCI шины. Вопрос о необходимости патча решается экспериментально. Еще опыт, это устройство понадобилось на маме с чипсетом Z77, иначе паника ядра на начальной стадии запуска.

FIX_HPET bit(4)

Как уже сказано, это главный фикс, необходимый. Кроме собственно устройства HPET, этот фикс затрагивает также устройства RTC и TMR, и попутно решает проблему сброса БИОСа после пробуждения.
Таким образом, минимально необходимая маска патча ДСДТ выглядит как 0x0010

FIX_LPC bit(5)

Подменяет DeviceID LPC контроллера, чтобы кекст AppleLPC прицепился к нему. Нужен для тех случаев, когда чипсет не предусмотрен для OSX. Впрочем, родной список чипсетов Intel и NForce настолько большой, что необходимость такого патча очень редка. Проверяется в системе, загружен ли кекст AppleLPC, если нет – патч нужен. Хотя, это тоже еще не факт. Бывает, что кекст сам выгружается из памяти за ненадобностью, хотя чипсет поддерживаемый.

FIX_IPIC bit(6)

Удаляет прерывание из устройства IPIC. Назначение такого фикса спорно. Для ноутбуков скорее надо не удалять, а добавлять такое прерывание.

FIX_SBUS bit(7)

Добавляет SMBusController в дерево устройств, тем самым удаляя предупреждение об его отсутствии из системного лога. И также создает правильную разводку управления питанием шины.

FIX_DISPLAY bit(8)

Производит ряд патчей и инжекцию свойств для видеокарточки. В принципе, этот метод включения видео имеет больше возможностей, чем через DeviceProperties, поскольку имеет возможность не только прописать свойства через метод _DSM, но также и изменить другие АЦПИ свойства дисплея. Например, для ноутбуков Делл строго требуется удалять устройство CRT из описания видеокарточки. Этот же фикс добавляет устройство HDAU для вывода звука через HDMI.

FIX_IDE bit(9)

В системе 10.6.1 появилась паника на кекст AppleIntelPIIXATA. Два варианта решения проблемы – использование исправленного кекста, либо исправить устройство в ДСДТ. А для более современных систем? Не знаю, пусть будет.

FIX_SATA bit(10)

Фиксирует какие-то проблемы с SATA, и убирает желтизну иконок дисков в системе путем мимикрии под ICH6. Вообще-то спорный метод, однако без этого фикса у меня DVD-диски не проигрываются, а ля ДВД привод не должен быть съемным. Т.е. Просто замена иконки — это не вариант!
Есть альтернатива, решаемая добавлением фикса с кексту AppleAHCIport.kext. Смотрите главу про патч кекстов.
И, соответственно, этот бит можно не ставить!

FIX_FIREWIRE bit(11)

Добавляет свойство "fwhub" к контроллеру Firewire.

FIX_USB bit(12)

Попытки решения многочисленных проблем с USB. Инжектирование DeviceProperties в настоящее время дает более правильное решение.

FIX_LAN bit(13)

Инжектирование свойства "built-in" для сетевой карточки – необходимо для правильной работы. Также инжектируется модель карточки – для косметики.

FIX_WIFI bit(14)

Аналогично LAN, кроме того, создается само устройство, если еще не прописано в DSDT. Для некоторых известных моделей производится подмена DeviceID на поддерживаемую. И аэропорт включается без прочих патчей.

FIX_HDA bit(15)

Корректировка описания звуковой карточки в ДСДТ, чтобы работал родной драйвер AppleHDA. Производится переименование AZAL -> HDEF, инжектируется layout-id и PinConfiguration.

Послесловие

Как выбрать, какие патчи необходимы, какие безвредны, а какие опасны? Ну компьютер вы не погубите ни в каком случае. Все это происходит только в оперативной памяти, и будет забыто после перезагрузки. Можно испытывать набор

фиксов, исправляя маску в графическом меню, и сохраняя результат по клавише F5 – "Сохранить DSDT-xxxx.aml, скорректированный по текущей маске".
Можно попытаться и загрузиться с текущей маской. Чтобы не мешался настоящий, патченный ДСДТ, уже присутствующий в системе, можно указать в меню
`DSDT name: NO.aml`
Не найдя такого файла система возьмет OEM DSDT из БИОС и проделает над ним фиксы, согласно установленной маске. В случае неудачи после перезагрузки компьютера текущие установки будут утеряны, и в силу вступят установки по умолчанию, которые у вас работоспособные.

Маска 0xFFFF соответствует включению всех фиксов, и если OS после этого загрузится, труд программистов потрачен не зря. По описанию выше вы уже сообразили, что некоторые фиксы вам просто ни к чему (например WIFI), а вот попортить картину могут. Для большинства не очень новых, но и не очень старых конфигураций маска 0xA7D7 является достаточной, и даже избыточной.

Нативный спидстеп

Правильнее говорить Управление Питанием и Частотой Процессора (УПиЧП), по-английски это будет EIST – Enhanced Intel Speedstep Technology, откуда и русское слово "Спидстеп".

Собственно эта тема не столько для загрузчика, как вообще для настройки ХакОС, но поскольку Кловер делает некоторые шаги, то опишем отдельной главой. Кловер делает не все, что нужно, требуется и немного поработать руками.

Для чего это вообще нужно? Смысл такой: процессор в бездействии работает на минимальной частоте с минимальным напряжением, под нагрузкой скорость и напряжение растут. (А напряжение-то зачем? А потому что фронт импульса становится круче, и потому быстрее набирает уровень, быстрее переходит из состояния 0 в состояние 1).

УПиЧП можно осуществить двумя способами: специализированной утилитой, типа КулБукКонтроллер, или ДженерикЦПУПМ, или же понять нативный спидстеп, благо МакОС это умеет делать.

Следующие шаги необходимы:

1. В ДСДТ обязательно должен быть поправлен HPET, что успешно делается Кловером при маске 0x0010.
2. Должна быть правильная процессорная секция, что делается Кловером при ключе GeneratePStates=Yes (ну и вдобавок DropOemSsdt=Yes)
3. Должна быть выбрана MacModel как образец вашего SMBIOS, для которого предусмотрена технология EIST. Оказывается, не для всех моделей. К примеру для модели MacBook1,1 спидстеп работать не будет, а для MacBook5,1 – будет.

Пункт 3 можно переосмыслить следующим образом: пусть, все-таки модель будет более похожа по конфигурации к настоящей, но исправим ее платформ-плист так, чтобы спидстеп появился.
Для каждой модели существует свой плист, смотрите здесь
`System/Library/Extensions/IOPlatformPluginFamily.kext/Contents/PlugIns/ACPI_SMC_PlatformPlugin.kext/Contents/Resources/MacBook5_1.plist`
Смотрим сходства и различия разных плистов, и исправляем свой в правильную сторону.

ConfigArray

```
<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>
```

Этот ключ restart-action означает на какой P-State должен свалиться CPU при рестарте. Только при наличии этого ключа заработали сон и выключение компьютера!

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>
```

Этот ключ PLimitDict уже упоминался в генерации P-states. Повторим: это ограничение максимальной скорости процессора. 0 – скорость максимальна, 1- на одну ступень ниже максимальной. Если же этот ключ здесь отсутствует, то процессор застрянет на минимальной частоте.

CStateDict

```
<key>CStateDict</key>
<dict>
	<key>MacBook4,1</key>
	<string>CSD3</string>
	<key>CSD3</key>
	<dict>
		<key>C6</key>
		<dict>
			<key>enable</key>
			<true/>
```

Практика показывает, что эту секцию лучше всю удалить, чтобы работало управление питанием именно по PState, а не по CState. Хотя, кому как, может и этот вариант стоит проработать.
Симптом – процессор стоит на максимальной частоте, не падает. После удаления секции начинает варьировать частоту.

Проблема сна

А что проблема сна? Когда все вышесказанное будет сделано, компьютер будет ложиться спать и просыпаться, как послушный ребенок. Самое главное, необходимое для этого, Кловер уже проделал: скорректировал FADT и FACS. Осталось только поправить ДСДТ, завести спидстеп, пользоваться только хорошими кекстами, и будем вам счастье.

Хорошему сну может помешать любое устройство, в том числе незаведенное PCI устройство, или заведенное частично. К примеру, AppleHDA. Сну категорически мешает NullCPUPM.kext. Вам, может, спидстеп и не нужен, но вы должны так сделать патч HPET, чтобы запустился родной AppleCPUPM, и нуль был не нужен.

В ДСДТ есть группа методов _GPE с нотификациями на каждое устройство, которое нужно пробудить после сна. Сам-то компьютер проснулся, а вот может оказаться, что видео/сеть/звук/мышь забыли проснуться. Смотрите ДСДТ, учите теорию, как это делать.

Есть проблема со сном при UEFI загрузке в систему 10.8. Система 10.7 спит хорошо. Система 10.8 тоже хорошо спит при CloverEFI загрузке. А вот 10.8, загруженная UEFI, засыпает хорошо, а просыпается в ребут. Есть некоторые рецепты, как это поправить, в частности, удалить CsmVideoDxe.efi, но остается недоумение, чем 10.8 отличается от 10.7 при UEFI- загрузке. Вопрос пока открытый.

ЧаВо

Часто задаваемые **Во**просы.

В. Хочу попробовать Кловер, с чего начать?

О. С чтения этой книги.
ЗЫ. Странно писать это внутри книги, но может эти ЧаВо окажутся вне ее страниц.

В. Не работает.

О. Сам дурак.
ЗЫ. Ну а что тут еще ответишь?

В. Установил Кловер, но получаю черный экран.

О. Загрузка ОС происходит в восемь этапов (см. Стр.6). Будьте добры, уточните, на каком именно этапе происходит остановка. И в своем отчете обязательно укажите «Устанавливал инсталлятором с выбором таких опций». Тогда и будет разговор.
Наиболее распространенные ошибки:

– в файле refit.conf указана тема black-green, а реально папка с такой темой отсутствует. В текущей версии получаем синий экран с полосатыми квадратиками, но все работает;
– с некоторыми БИОСами CsmVideoDxe не работает, удалите его;
– бывает, что PatchVBios=Yes приводит к черному экрану, попытайтесь выключить.

Для лучшей диагностики происходящего поставьте

```
<key>DebugLog</key>
<true/>
```

в файле config.plist в секции GUI. Загрузка будет происходит очень медленно, поскольку на каждом шаге будет обновляться /EFI/CLOVER/misc/debug.log, зато после окончательного зависания вы получите информацию, что именно произошло.

В. Вижу на экране 7_ и больше ничего не происходит.

О. Это самый тяжелый случай несовместимости по железу. Сейчас уже не встречается. Продиагностировать сможет только программист, который сможет вставлять в коды Кловера отладочные сообщения, и делать ребут за ребутом до полного выяснения проблемы. Увы, простым пользователям посоветовать нечего. Разве что поиграться с установками БИОСа, иногда помогает. Пробуйте вместо файла boot использовать boot7.

В. Происходит загрузка только до текстового аналога БИОСа с пятью пунктами, верхний – Continue>

О. Это означает, что файл boot успешно загрузился, и работает, но не находит файла CloverX64.efi. То ли того раздела не видит, то ли вообще устройства – надо разбираться далее, прогулявшись по опциям этого меню.

В. Установил Кловер на флешку, загрузился с нее, и не вижу своего HDD.

О. Во-первых, HDD надо вставить в порт Sata0. В будущем может быть это будет исправлено.[5] Во-вторых я понимаю, если у вас есть хорошо работающий Хам, Химера, ХРС, короче, ББХ (**Б**утер на **Б**укву **Х**), вы не хотите его убивать, но хотите попробовать Кловер, то такой поступок кажется естественным. Но, тем не менее, есть варианты установки Кловера на жесткий диск, не убивающие старого загрузчика, и в таком раскладе озвученная ошибка пропадет.
Пробуйте также файл boot7, если у вас какой-то необычный SATA/SAS/RAID контроллер.

В. При УЕФИ-загрузке не вижу раздела с МакОСью, только легаси.

О. Это означает, что в папке /EFI/drivers64UEFI отсутствует HFSPlus.efi или его легальный аналог VboxHFS.efi.

В. При УЕФИ-загрузке Виндоус выглядит как легаси, хотя он EFI.

О. То же самое, отсутствует драйвер NTFS.efi
ЗЫ. Эти два драйвера отсутствуют в репозитарии по лицензионным причинам, Вам нужно отыскать этот файл где-то на просторах сети.

В. При попытке запуска ОСи зависает после синей строки с надписью rEFIt

О. В этот момент происходит патч ДСДТ с вашей маской. Да, в идеале тут не должно виснуть. Но проблема в том, что очень много производителей БИОСов не соблюдает стандарты, не умеют программировать, и не желают отшлифовывать свой ДСДТ под нужды OSX. Очень легко убедиться, что операция декомпилировать-снова скомпилировать не проходит – ДСДТ кривой. Кловер желал бы все это исправить, но увы, количество плохих вариантов пока не поддается даже обзору. Поэтому, от вас требуется подобрать такую маску фикса ДСДТ, чтобы загрузчик не повис, а затем и чтобы ОСь не повисла, а в идеале, чтобы она еще и работала. Это – реально. Либо отказаться от автопатча (маска = 0), а ДСДТ сделать вручную.

5 Была такая ошибка в САТА драйвере от Интел, в настоящее время исправлена.

В. Ядро начинает грузиться, но паникует после десятой строки.
О. Это отсутствующий, или неправильный ДСДТ. Если автопатчем не получается, добавьте ДСДТ, сделанный вручную.

В. Система начинает грузиться, но стопорится на still waiting for root device….
О. Кроме обычного для таких случаев совета включить AHCI в БИОСе, или, если такого нет, найти правильный драйвер (в смысле кекст) для вашего IDE контроллера, тут есть еще совет загрузиться с ключом WithKexts (в новых ревизиях NoCaches), тогда загрузка пойдет медленнее, и контроллер успеет включиться. Кстати, такая ошибка может возникнуть только если Кловер и система находятся на разных устройствах.

В. Система грузится до сообщения: Waiting for DSMOS….
О. Отсутствует FakeSMC. Может быть с Хамелеоном у вас этот кекст лежал в Экстре, а Кловер этой папки не видит. Для него предназначена папка /EFI/kexts/10.x или другие.

В. Система проходит это сообщение, но дальше ничего не меняется, хотя винчестер жужжит, как будто система грузится.
О. Типичная ситуация, когда не включилась видеокарта. Пробуйте GraphicInjector=Yes в конфиге, либо наоборот =No. Во втором варианте Радеоны запускаются на «нативной заводке», которая позволяет даже работать в системе, за небольшими исключениями, например DVDplayer не будет работать. Для полной же заводки Радеона требуется еще и коннекторы поправить. Для других случаев можно попытаться загрузить систему с ключом –x, и войти на десктоп в режиме VESA. Не очень здорово, но зато позволит что-то исправить.
Еще вариант тормоза в этом месте наблюдается, если выбираете модель MacMini. Проблема решается с установкой ключа DropMCFG=Yes

В. Система загрузилась, все хорошо, но в Систем Профайлере ошибки…
О. Вообще это косметика, на функциональность не влияет.
О платах PCI. Если желаете информацию, добавьте в ДСДТ в нужное устройство свойство Name (_SUN, 0x1234) – число на ваше усмотрение. Оно отобразится в профайлере.
О памяти. Есть две величины скорости, номинальная и фактическая, и они часто не совпадают. Какую показать в профайлере? Поставил первую – заорали, что это неверно. Поставил вторую, эти замолчали, другие пользователи заорали, что это неправильно....
Смотрите страницу 40 — как прописать свои значения памяти в конфиге.

Заключение

Кловер, конечно, еще далек до идеала, но процесс совершенствования программ никогда не бывает завершенным. Будут новые ревизии, будут новые функции, а пока так.