Hacking the Xbox: An Introduction to Reverse Engineering. Русский перевод

Русский перевод книги Andrew «bunnie» Huang о хакинге Xbox и введении в обратную разработку.

Больше крутых материалов в моём телеграм-канале @grogrigs

Фронтматтер, письмо читателю и благодарности

Передняя обложка

Hacking the Xbox: An Introduction to Reverse Engineering
Andrew "bunnie" Huang

Это практическое руководство по хакингу было отменено первоначальным издателем из страха перед судебными исками, связанными с DMCA. После самостоятельной публикации книги автором (за это время он продал тысячи экземпляров напрямую) *Hacking the Xbox* теперь представлена вам No Starch Press.

Hacking the Xbox начинается с нескольких пошаговых руководств по аппаратным модификациям, которые обучают базовым хакерским техникам, а также важнейшим навыкам обратной разработки. Затем книга переходит к обсуждению механизмов безопасности Xbox и других продвинутых хакерских тем, подчеркивая важные предметы компьютерной безопасности и обратной разработки. Книга включает многочисленные практические руководства, такие как где достать хакерское снаряжение, техники пайки, советы по отладке и аппаратный справочник Xbox.

Hacking the Xbox сталкивается с социальными и политическими вопросами, стоящими перед сегодняшним хакером, и знакомит читателей с людьми за хаками через несколько интервью с мастерами-хакерами. Она рассматривает потенциальное влияние сегодняшних правовых вызовов на законную деятельность по обратной разработке, которые дополнительно изучаются в главе, внесенной Lee Tien из Electronic Frontier Foundation (EFF), о правах и обязанностях хакеров. Книга завершается обсуждением последних тенденций и уязвимостей в безопасных PC-платформах.

Поторопитесь и получите *Hacking the Xbox*, пока этого не сделала Microsoft!

$24.99 ($34.99 CDN)
SHELVE IN: PC HARDWARE/GENERAL
Получите *Hacking the Xbox*, пока этого не сделала Microsoft!
ISBN 1-59327-029-1
THE FINEST IN GEEK ENTERTAINMENT(TM)
www.nostarch.com

Титульные страницы

Hacking the Xbox: An Introduction to Reverse Engineering
Unlimited Edition
Andrew "bunnie" Huang
No Starch Press, Inc.
San Francisco

Посвящение

Памяти Аарона Шварца.

Письмо читателю

Дорогой читатель!

Спасибо, что загрузили и читаете эту книгу.

No Starch Press и я решили выпустить эту бесплатную электронную версию книги *Hacking the Xbox* в память об Аароне Шварце. Читая эту книгу, я надеюсь, вы вспомните, насколько важна свобода для хакерского сообщества, и захотите поддержать те дела, в которые верил Аарон.

Я согласился выпустить эту книгу бесплатно отчасти потому, что отношение MIT к Аарону не кажется мне чем-то незнакомым. В этой книге вы найдёте рассказ о том времени, когда я был аспирантом MIT и извлекал ключи безопасности из оригинальной Microsoft Xbox. Вы также прочитаете о сокрушительном разочаровании, которое я испытал, получив письмо от юридического отдела MIT: в нём институт отказывался признавать какую-либо связь с моей работой и фактически оставлял меня один на один с Microsoft.

Разница была в том, что преподаватели моей лаборатории, Лаборатории искусственного интеллекта, были возмущены таким обращением. Они открыто пошли наперекор юридическому отделу MIT и пообещали опубликовать мою работу как официальный «меморандум AI Lab», тем самым дав мне более сильную позицию в переговорах с Microsoft. Microsoft, понимая возможную отрицательную реакцию общественного мнения на судебное преследование добросовестного академического исследователя, пришла со мной к цивилизованному взаимопониманию по этому вопросу.

Меня печалит, что так называемое американское правительство народа, избранное народом и для народа, проявляет меньше сострадания и просвещённости по отношению к ближнему, чем корпорация. Поскольку позже мне самому довелось столкнуться с юридическим давлением со стороны других организаций, я слишком хорошо знаю, насколько уродливым и выворачивающим душу может быть судебный процесс с высокими ставками. К счастью, в моих случаях ставки были не так высоки, а мои противники не были столь могущественны, как противники Аарона; иначе я тоже мог бы поддаться безнадёжности и страху. Несколько лет назад я начал заново строить свою жизнь за границей и нахожу некоторую толику утешения в мысли, что моё проживание за рубежом немного затрудняет вручение мне судебных документов.

Хотя правовая система США стремится к правосудию, её правила создают асимметричную войну, благоприятную для тех, у кого есть ресурсы. Один из самых эффективных способов принудить небольшого участника к исходу, правильному или неправильному, состоит в том, чтобы просто истощить его ресурсы и волю к борьбе досудебными манёврами. Вся ваша жизнь оказывается как будто под электронным микроскопом: каждый крошечный изъян раздувается до ожесточённой битвы ходатайств, ответных ходатайств, раскрытия доказательств, повесток и письменных показаний, а каждое действие добавляет десятки тысяч долларов к вашим судебным расходам.

Правительство США - безусловно самый хорошо финансируемый и грозный противник в мире, а закон об авторском праве связан с необычайно крупными, если не жестокими, наказаниями. Я никогда не знал Аарона, но чувствую, что масштаб давления, которому он подвергся, отразился в его решении уйти из жизни.

Я разделяю мысль Ларри Лессига о том, что правовой системе США необходимо чувство стыда. С точки зрения такого стороннего наблюдателя, как я, кажется, что некоторые прокуроры в правительстве США одержимы желанием сделать себе имя за счёт людей, которых они преследуют. Победы в делах дают им признание и репутацию, необходимые для повышения и назначения на всё более громкие процессы. Для них это не о правосудии - это о победе и самоутверждении.

Такая система стимулов способствует бесстыдному запугиванию отдельных людей и небольших организаций, у которых хватает смелости подняться и сделать что-то дерзкое. У людей отнимают волю и силы бороться за то, что они считают правильным, потому что сам факт преследования может стать наказанием не меньшим, чем приговор. В результате я опасаюсь, что эпоха гражданского неповиновения может подходить к концу.

Как люди, как отдельные личности, как хакеры, мы должны противостоять этой тенденции и продолжать делать то, что в глубине сердца считаем правильным. История Аарона закончилась трагически, но я надеюсь, что в этой книге вы найдёте ободряющую историю со счастливым концом. Без права разбирать, переделывать и исследовать мы рискуем стать рабами технологии; и чем чаще мы пользуемся правом на хакерство, тем труднее будет это право у нас отнять.

bunnie
Сингапур, март 2013 года

Благодарности

Я хотел бы поблагодарить моих преданных и заботливых родителей за то, что они воспитали меня человеком, которым я стал.

Я также хотел бы поблагодарить сетевое хакерское сообщество за советы и руководство, особенно тех его участников, которым приходится действовать анонимно из страха перед преследованием со стороны государства или ответными мерами работодателя.

Ли Тиен из Electronic Frontier Foundation, Джозеф Лю из Boston College Law School, а также доктор Том Найт и профессор Хэл Абельсон из Лаборатории искусственного интеллекта MIT заслуживают особой благодарности за помощь в процессе публикации моей первоначальной статьи о системе безопасности Xbox. Без их поддержки и советов я никогда бы её не опубликовал.

Я также в долгу перед командой Xbox-Linux: Майклом Штайлем, Милошем Мериаком, Францем Ленером (спасибо за такой подробный технический обзор!) и потрясающим Энди Грином (он же numbnut) - за то, что они дали так много понимания новейших хаков Xbox и предоставили столь интересный материал для книги. Огромное уважение вам, ребята; продолжайте в том же духе. Я также хотел бы поблагодарить Дэна Джонсона (он же SiliconIce), основателя BBS XboxHacker.net, за запуск XboxHacker.net BBS, за его интересные материалы для книги, а также за очень полезный технический обзор, советы и поддержку. Кроме того, спасибо Герхарду Фарфеледеру за предоставленную фотографию команды Xbox-Linux.

Спасибо Тимоти Чену из Via Technologies, Inc. за предоставленную материнскую плату P4M266 для сравнения Xbox и PC и за его увлекательный взгляд на индустрию PC. Я также хотел бы поблагодарить Xilinx за щедрые пожертвования FPGA в рамках Xilinx University Program.

Вы сами знаете, кто вы, и знаете, чем помогли мне: xor, adq, luc, head, visor, roastbeef, kgasper, xerox, lordvictory, pixel8, EI (GCN), tom from HK и sween (Scotch!).

Дополнительные благодарности связаны с unlimited edition этой книги: Билл Поллок из No Starch Press заслуживает особой благодарности за то, что смело взялся за печать и распространение книги; Пол Юн заслуживает горячей благодарности за многочисленные исправления опечаток; а Раэль Дорнфест и Тим О’Рейли из O’Reilly & Associates, Inc. также заслуживают особой благодарности за полезные советы и поддержку.

Пролог. README.1ST

Игровая консоль Xbox(TM) от Microsoft(R) - замечательное устройство, и не только потому, что на ней можно играть в новейшие видеоигры. Мощная и недорогая Xbox потенциально может использоваться как PC, универсальный медиаплеер или даже веб-сервер. К сожалению, книг, способных научить читателя исследовать и модифицировать современную электронную аппаратуру вроде Xbox, очень мало. Большинство учебников по электронике ориентированы на теорию и имеют узкую специализацию, тогда как настоящее хакерство требует широкого набора практических навыков и знаний. Кроме того, те немногие практические книги по аппаратному хакингу, которые вдохновляли меня в детстве, давно устарели из-за стремительного развития технологий. Эта книга призвана восполнить потребность в практическом руководстве по пониманию и обратной разработке современных компьютеров: в настольной книге для нового поколения хакеров.

Главная польза хакинга Xbox - его образовательная ценность, или, как говорится, «дай человеку рыбу - он будет сыт один день; научи ловить рыбу - будет сыт всю жизнь». Поэтому эта книга сосредоточена на знакомстве начинающих хакеров с базовыми приёмами хакинга - пайкой, обратной разработкой, отладкой, - одновременно предоставляя аппаратные справочные сведения и наблюдения, которые могут быть полезны более опытным хакерам. Xbox послужила обучающим примером и для сообщества специалистов по безопасности, и для хакерского сообщества: не потому, что она является выдающимся образцом безопасности, а потому, что это заметный массовый продукт крупной компании, председатель которой недавно определил безопасность как одно из направлений её деятельности. [1] Опыт Xbox показывает, что создавать доверенные клиенты в недружественной пользовательской среде трудно даже крупной, хорошо финансируемой компании. Одно из наблюдений состоит в том, что риск и сложность создания дешёвых доверенных аппаратных клиентов задают верхнюю границу важности секрета, который можно доверить такому клиентскому оборудованию. Кроме того, Xbox даёт последовательный учебный пример: на момент написания существовало почти 10 миллионов практически одинаковых устройств. Сходство архитектуры Xbox с обычным PC ещё больше повышает образовательную ценность хакинга Xbox, поскольку значительная часть обсуждения в этой книге напрямую применима и к гораздо более широкой теме PC.

Ещё одна интересная сторона хакинга Xbox - подпольное общество аппаратных хакеров, возникшее вокруг Xbox. Люди, взломавшие Xbox, и приобретённая ими экспертиза останутся значимыми ещё долго после того, как сама Xbox превратится в пыльную вещь с дворовой распродажи. Поэтому в этой книге сознательно присутствует социальный фокус. Я включил в неё профили нескольких заметных представителей Xbox-хакинга. Надежда в том, чтобы через ролевые модели вдохновить людей взять отвёртку и паяльник и начать хакерствовать. Привить такой исследовательский дух молодым поколениям будет важно в долгосрочной перспективе для сохранения пула талантливых инженеров, который довёл технологическую революцию до нынешнего состояния. Многие сегодняшние инженеры начинали с хакинга и возни с любительскими радиостанциями, телефонами и компьютерами, которые в те времена поставлялись с полным набором схем и исходного кода. Этот запас инженерного таланта необходим для поддержания здоровой экономики и сильной национальной безопасности в компьютерную эпоху.

Рынок игровых консолей

2002 год был отмечен потрясениями не только за рубежом, но и на технологическом рынке: продажи PC перестали расти, серверный бизнес сократился, а телекоммуникационный рынок, за немногими исключениями, выглядел мрачно. Несмотря на медвежий рынок технологий, рынок аппаратуры, программного обеспечения и аксессуаров для видеоигр пережил знаковый год: общий объём продаж достиг 10,3 миллиарда долларов - на 10% больше, чем в 2001 году. [2] Это сопоставимо с продажами звукозаписывающей индустрии в США в 2001 году, составлявшими 13 миллиардов долларов.

Хотя рынок видеоигр велик, вести прибыльный консольный бизнес чрезвычайно трудно. Покупатели видеоигр разборчивы, подвержены моде и бережливы. Они требуют высокопроизводительное, привлекательное консольное железо по цене хорошего семейного ужина или визита к врачу. Такое сочетание бережливости с ожиданием высокопроизводительной игровой аппаратуры заставляет поставщиков консолей продавать своё оборудование в убыток. В результате поставщики консолей используют бизнес-стратегию «закрытой консоли»: консоль продаётся как убыточный товар-приманка, а прибыль приходит от будущих продаж игровых тайтлов. Такая бизнес-стратегия требует больших первоначальных инвестиций в консольное оборудование и рекламу. Производитель консоли отвечает за создание рынка для своего оборудования, чтобы разработчики игр чувствовали себя достаточно уверенно, вкладывая время и деньги в платформу.

Уловка-22 состоит в том, что никто не хочет покупать консоль, для которой мало игр. Поэтому риск создания и развёртывания миллионов единиц оборудования, как и сотни миллионов долларов первоначальных убытков на этом оборудовании, почти полностью ложатся на производителя консоли. В результате на сегодняшнем рынке игровых консолей сейчас есть только три игрока: Sony, Nintendo и Microsoft. Из них Sony имеет огромный отрыв на рынке консолей, а Nintendo заняла рынок портативных устройств своей линейкой Gameboy. Microsoft - новый игрок на рынке игровых консолей. Гонка за второе место ещё не решена. В начале 2003 года продажи Gamecube опережали продажи Xbox в Японии и Европе, тогда как Xbox сохраняла преимущество над Gamecube на огромном североамериканском рынке.

Для успеха бизнес-модели закрытой консоли решающее значение имеет идея привязать потребителей к покупке только одобренных игровых тайтлов, с которых выплачиваются роялти. Иными словами, пиратство и неутверждённые игровые тайтлы способны уничтожить прибыльность бизнеса. Поэтому консоль должна применять механизмы безопасности, затрудняющие копирование игр, а также разработку и распространение неутверждённых игр. Провал Sega Dreamcast - показательный пример того, что происходит, когда механизмы безопасности не срабатывают.

Dreamcast была запущена в Японии в ноябре 1998 года. Производственные проблемы с микросхемой NEC PowerVR2 DC, графическим ускорителем Dreamcast, ограничили первоначальные поставки. Следующие три года стали для Dreamcast настоящими американскими горками. Популярные игры, такие как Soul Caliber, Dead or Alive 2, Resident Evil, Crazy Taxi и Shen Mue, поддерживали популярность Dreamcast, тогда как запуск Sony Playstation2 подтачивал продажи Dreamcast и в конечном счёте уверенность разработчиков программного обеспечения. По иронии, качество графики Dreamcast было равно качеству ранних тайтлов Playstation2, таких как Dead or Alive 2, или превосходило его, несмотря на дополнительную вычислительную мощь Playstation2. (Playstation2 трудна в программировании, и разработчикам потребовалась пара лет, чтобы раскрыть её полный потенциал.)

Последний гвоздь в крышку гроба Dreamcast был забит весной и летом 2000 года. Немецкая хакерская группа Team Utopia обнаружила чёрный ход в mask-ROM BIOS Dreamcast, позволявший

Dreamcast загружаться со стандартного CD-ROM. Номинально Dreamcast использует для распространения игр проприетарный формат под названием «GD-ROM». Формат GD-ROM нельзя скопировать стандартными пишущими CD- или DVD-приводами. Однако чёрный ход в ROM BIOS Dreamcast позволил пиратам в конце концов создавать монолитные CD-ROM-образы видеоигр, которые загружались без какой-либо аппаратной модификации. Кто стал бы платить за игру, если её можно бесплатно скачать из интернета? Последовавшее массовое пиратство снизило продажи игр, отбило у разработчиков желание выпускать продукты для консоли и нанесло ущерб бизнесу Sega. Было продано шесть миллионов устройств, и примерно через три года после запуска Dreamcast сняли с рынка. Теперь Sega занимается исключительно разработкой игр и выпускает игры даже для своих бывших конкурентов Sony и Nintendo, а также для Microsoft.

Из опыта Dreamcast можно извлечь много уроков, но главный вывод ясен: способность запускать код из почти бесплатных источников, таких как CD-R, DVD-R или сеть, без существенных аппаратных модификаций - смертельный поцелуй для любого консольного бизнеса, основанного на модели закрытой консоли. Для Microsoft Xbox это жёсткая проблема, поскольку она построена на стандартном PC-железе, изначально спроектированном как открытое и рассчитанное на запуск кода, загруженного из множества источников. Поэтому судьба Microsoft на консольном рынке тесно связана с успехом и стойкостью системы безопасности Xbox. Пока система безопасности держалась довольно неплохо: все найденные слабости требуют установить по меньшей мере не требующую пайки модификацию, аннулирующую гарантию. Необходимость аппаратных модификаций ограничивает практическое влияние этих слабостей, поскольку большинство пользователей боится снимать крышку со своей техники. Однако несколько групп - как законных, так и незаконных - очень сильно хотят заставить Xbox запускать код из произвольных источников без аппаратных модификаций.

Xbox стала жертвой собственного дизайна: выбор стандартного PC-железа резко повышает ценность «открытой» Xbox как для хакеров, так и для пиратов. Xbox является довольно приятной целью для хакеров выходного дня и любителей по той же причине, по которой Microsoft приняла PC-архитектуру для Xbox: существующие PC-программы легко переносятся на Xbox. Кроме того, существует широкая и глубокая база знаний о PC-железе, поэтому кривая обучения хакингу Xbox не так крута, как у других консолей. С другой стороны, Playstation2 и Gamecube имеют крутую кривую обучения, а также архитектурные ограничения, мешающие переносу большинства PC-приложений. Xbox также популярна у пиратов из-за простоты переноса старых игровых эмуляторов, а также из-за её известности и простоты получения совместимого отладочного и тестового оборудования.

Кроме того, сходство архитектуры Xbox с архитектурой PC делает Xbox хорошим учебным средством. Знания, полученные из этой книги, применимы не только к встраиваемому оборудованию или игровым консолям; большую часть этих знаний вы сможете напрямую применить к PC. К тому же огромные массивы документации, применимой к Xbox и унаследованной из мира PC, удобно индексируются веб-поисковиками. Готовая доступность документации поможет мотивированным читателям развить знания, содержащиеся в этой книге.

Xbox также является более привлекательным учебным примером, чем обычный PC. Аппаратные детали реализаций PC слишком разнообразны, чтобы по ним можно было составить полезные пошаговые руководства по хакингу PC, тогда как пошаговые руководства для Xbox гарантированно точны для миллионов устройств, которые удобно доступны для покупки почти в любом торговом центре или магазине электроники.

О хакерах и хакинге

Эта книга о хакинге в традиционном смысле: о процессе и методах исследования. Некоторые могут удивиться, что в этой книге нет глав, посвящённых копированию игр и исправлению конкретных проверок безопасности - в конце концов, разве не в этом суть хакинга? На самом деле термин «хакер» за годы довольно сильно изменился по мере того, как росла осведомлённость общества о технологиях, а склонные к сенсациям массовые медиа продолжали окрашивать общественное мнение о хакерах.

Изначально хакером был человек, который увлечённо работал ради любопытства и исследования. Были аппаратные хакеры, которые брали на себя задачу снимать крышки с компьютеров, чтобы оптимизировать их конструкцию (ранние компьютеры строились из дискретных компонентов, поэтому их можно было осмысленно модифицировать простыми инструментами), и были программные хакеры, которые трудились над созданием максимально компактного и изящного кода, поскольку вычислительные ресурсы были скудны и медленны. Были хакеры, исследовавшие все закоулки телефонной системы, и те, кто исследовал крыши и туннели зданий университетских кампусов. Довольно часто ранние хакеры занимались всеми этими вещами. Хакеры свободно делились друг с другом своими находками или результатами (хаками), поскольку их награда была не финансовой: она приходила от удовлетворения интеллектуального любопытства и энтузиазма сверстников. В результате хакеры склонны были объединяться в меритократические группы, где членство и продвижение полностью зависели от способности человека хакерствовать.

По мере развития технологий и роста скорости и интеграции компьютеров хакеры обнаружили, что усилия, связанные с аппаратным хакингом, уже не стоят получаемых преимуществ. Интересные части компьютеров быстро оказывались глубоко внутри герметично запечатанных керамических корпусов, вытравленные в кремниевых структурах, которые трудно рассмотреть даже в хороший микроскоп. Сложный аппаратный хак, способный удвоить производительность компьютера, уже через месяцы терял смысл из-за закона Мура. С другой стороны, программный хакинг начал всё больше сосредотачиваться на приложениях и всё меньше - на алгоритмах или оптимизации. Компактность или изящество программы больше не имели прямой важности, поскольку память и процессорная мощность стали дешёвыми и обильными. Кроме того, технология компиляторов тоже улучшилась до такой степени, что скомпилированный код работал почти так же быстро, как рукописный ассемблер. К концу 80-х термин «хакер» стал подразумевать человека, который способен писать тома кода на C во сне и создавать блестящие приложения за одну ночь. Старые аппаратные хакеры либо превращались в программных хакеров, либо отступали в университетские лаборатории и корпорации, способные поддерживать их дорогие увлечения. [3]

Термин «хакер» в то время всё чаще связывали с людьми, которые взламывали пароли и программы, чтобы получить доступ к машинам и программному обеспечению, которые иначе были закрыты. Отчасти за этот стереотип отвечал Голливуд, выпустивший поток фильмов, где подростки несколькими нажатиями клавиш подводили мир к грани ядерного уничтожения, а одинокие гении создавали у себя в подвале искусственно-интеллектуальных кибермонстров. [4] К сожалению, гиперболичность этих кинематографических сюжетов была потеряна для широкой публики, и это мрачное впечатление о хакерах в конце концов стало доминирующей частью хакерского стереотипа. Неточность этого стереотипа способствовала появлению термина для хакеров, сосредоточенного главным образом на взломе систем и программ, - «кракеры».

Технология формирует современного хакера не меньше, чем хакеры формировали технологию. Новым поколениям хакеров приходится усердно работать, чтобы пробиться сквозь «дружелюбные» пользовательские интерфейсы, медийный блеск и маркетинговую мишуру, окружающие компьютерные технологии сегодня. Все пользуются компьютерами и ожидают, что они будут работать безупречно и интуитивно, но мало кто по-настоящему понимает, что происходит под капотом.

Технология вычислений стала настолько сложной, что начинающие всё больше напоминают притчу о семи слепцах и слоне. Одни начинающие начнут своё хакерское путешествие с исследования интернета. Другие начнут с исследования операционной системы на своём компьютере. Третьи начнут с того, что заглянут под крышку своего компьютера. Каждый из них может потратить год на исследование своей грани, но к концу дня у каждого будет отчётливо иное представление о компьютерных технологиях.

Культурный разрыв между молодыми хакерами и старой гвардией стал для меня очевиден, когда первокурсник MIT, самопровозглашённая хакерская звезда, презрительно спросил: «Где все компьютеры с Windows[98]? ... у вас тут только эти убогие Sun, на которых даже AOL нет! Я думал, в MIT должен быть хороший доступ в интернет». Похоже, он совершенно не понимал, что эти «убогие Sun» были весьма мощными рабочими станциями под управлением одной из самых надёжных операционных систем в мире, и что за пределами AOL тоже существует интернет; более того, кампус MIT был одним из мест рождения интернета, имел права на большее количество IP-адресов, чем большинство ISP, и прямое подключение к магистрали интернета.

Проникновение компьютерных технологий в каждый уголок повседневной жизни усилило стереотипы о хакерах. В частности, медийное изображение хакеров как современных Робин Гудов каким-то образом необратимо связало хакинг с аспектами, касающимися безопасности или доступа к компьютерным ресурсам. Теперь стереотипный хакер отвечает за warez, Code Red и ping flood, тогда как «разработчики» отвечают за Linux и BSD. Хакеры - это `31337 d00ds`, которые `own your box` («владеют твоей коробкой»), а аппаратный хакер разгоняет свой компьютер и модифицирует его корпус неоновыми огнями. Хакинг стал модным, и многие стремятся соответствовать стереотипу, созданному медиа. Сегодня очень трудно убедить людей, что я хакнул Xbox исключительно потому, что она была здесь и её можно было хакнуть: это было сложно, и это было новым. Точно так же людям трудно понять, почему с тех пор я не работал над Xbox. После взлома безопасности Xbox всё, что остаётся, - это стандартный PC, который для меня не так уж интересен как объект работы и определённо не стоит риска судебного иска со стороны Microsoft.

Политика хакинга

Принятие в 1998 году Digital Millennium Copyright Act (DMCA) вывело криптографию из области, доступной хакеру: теперь закон прямо говорит, что исследовать криптографические методы защиты прав доступа к произведениям разрешено только исследователям, которые «заняты законным курсом обучения, работают по найму или имеют соответствующую подготовку либо опыт». [5] В результате хакинг Xbox стал политически заряженной темой. Это битва между хакерами и законодателями за то, чтобы сохранить криптографию в пределах законных прав хакеров.

Достойная похвалы реакция Microsoft на хакеров Xbox - то есть отсутствие преследования или попыток закрыть проекты по хакингу Xbox на сегодняшний день - будем надеяться, послужит примером для других, кто думает использовать DMCA, чтобы остановить хакерскую деятельность. Несмотря на все существующие хаки Xbox, Microsoft по-прежнему наслаждается устойчивыми продажами игр. Весь интерес и шум, созданные хакингом Xbox, могли увеличить продажи Microsoft сильнее, чем пиратство им навредило. (Конечно, я сочувствую хакерам, поэтому моя интерпретация ситуации пристрастна. Более субъективный и информированный юридический анализ обратной разработки можно найти в главе 12, «Caveat Hacker», написанной Ли Тиеном из Electronic Frontier Foundation.)

Самая тревожная для хакеров сторона DMCA состоит в том, что он воплощает ошибочное представление: будто единственные источники инноваций, полезных обществу, находятся в стенах исследовательских институтов и корпораций. Внезапно стало преступлением исследовать у себя

дома, ради собственного увлечения, криптографические методы, используемые для защиты прав доступа. Ограничение исследований такой технологии только признанными учреждениями исключает возможность разработки технологий независимыми людьми. Без свободы исследовать и разрабатывать технологии в собственном гараже где сегодня были бы такие люди, как Билл Хьюлетт и Дэйв Паккард или Стив Джобс и Стив Возняк? Были бы у нас Linux и netBSD, если бы право хакеров свободно выражать себя в коде регулировалось законом?

На каждую схему защиты авторских прав, побеждённую хакером, приходится кто-то, кто усвоил важный урок о том, как сделать лучшую схему защиты. Принимать законы, регулирующие исследование технологических мер, защищающих авторские права, и распространение результатов таких исследований - значит признавать, что технология защиты авторских прав сломана и никогда не может быть улучшена; что единственный возможный исход, если позволить обычным людям понимать технологии контроля авторских прав, - гибель этой технологии. Я предлагаю возражение против такого образа мыслей: некоторые из лучших экспертных отзывов, которые я получил по своей работе над хакингом Xbox, пришли не из академического сообщества. Они пришли от отдельных хакеров со всего мира - особенно из зарубежных стран, - которые были свободны исследовать и понимать технологии контроля доступа. Более строгие законы в США и склонность корпораций к судебным тяжбам уже отрицательно сказались на положении США в области электронной безопасности, и это только начало.

Социальное влияние DMCA ощущается хакерскими сообществами по всему миру. Во время работы над Xbox мне посчастливилось встретить блестящих хакеров по всему земному шару. Американские хакеры были одними из самых напуганных в этой группе, и хотя они были талантливыми инженерами, они не хотели применять свои навыки к таким задачам из страха преследования. В результате некоторые из самых интересных результатов в хакинге Xbox получены европейскими и азиатскими хакерскими сообществами. Существенно, что эти результаты мало известны в Америке, поскольку у таких хакеров почти нет мотивации прилагать усилия, чтобы делиться своими находками с американцами. Более того, многие зарубежные хакеры сознательно стараются не выпускать свои находки за пределы своих сообществ, в том числе из-за страха перед ответными действиями американских корпораций. Эта «утечка мозгов» мало помогает укреплять компетентность Америки в столь важной технологии, как справедливый и эффективный цифровой контроль авторских прав. А в сегодняшней глобальной экономике американские корпорации не могут выжить, делая вид, что ведут бизнес в вакууме.

Кто-то может указать на успешную публикацию моей статьи о системе безопасности Xbox как на пример того, как DMCA защищает и право на свободу слова, и экономические интересы в технологии контроля авторских прав. Моя ситуация не была типичной для большинства хакеров в США. Поскольку я тогда был аспирантом, у меня не было семьи, о которой нужно было бы заботиться, или значительных активов, которые я мог бы потерять, если бы оказался втянут в судебный процесс из-за своей работы. У меня также была щедрая юридическая помощь Electronic Frontier Foundation (EFF), которая помогла провести меня через юридическое минное поле. EFF помогла представить мою статью в максимально законном свете, сообщив мне о моих правах и обязанностях по DMCA.

Например, от меня требуется «добросовестно попытаться получить разрешение [от Microsoft] до обхода». [6] (Обратите внимание: требуется не разрешение, а именно добросовестная попытка.) EFF помогла мне составить такое письмо для исследования. Мне также пришлось бороться с MIT за то, чтобы моё исследование было разрешено к публикации как работа аффилированного с институтом лица. Все прямые усилия по обратной разработке безопасности Xbox финансировались из моего собственного кармана, выполнялись в моей квартире и делались в нерабочее время, за мой собственный счёт. MIT сначала воспользовался этим фактом, чтобы отделить себя от моей работы, вынудив меня обратиться за советом к EFF. В конце концов MIT уступил и разрешил мне опубликовать статью как студенту MIT после долгих уговоров со стороны сочувствующих

профессоров и после того, как я получил от Microsoft конструктивное, не угрожающее письмо по поводу моего исследования.

Свобода слова не должна требовать юриста, а свободная мысль не должна сопровождаться письмами с разрешением на исследование. Я боролся за публикацию своей статьи, потому что мне было нечего терять и потому что я верил: этим я заявляю о своих правах как хакера. К сожалению, существует молчаливое большинство хакеров, которым нужно кормить семьи и которые могут потерять работу, и не каждому может так повезти, что EFF придёт ему на помощь.

Книга, которую вы читаете, - ещё один пример того, как DMCA оказывает охлаждающее воздействие на свободу слова. Изначально заказанная техническим издательством John Wiley & Sons, Ltd., эта книга была отменена в последний момент из-за опасений судебных исков и ответной реакции Microsoft. Такая цензура расстраивает и обескураживает, и, возможно, некоторые авторы на этом остановились бы и позволили страху заглушить свой голос. Я беру на себя юридический и финансовый риск самостоятельной публикации этой книги, чтобы заявить о своём праве на свободную и беспрепятственную речь как хакера. Впрочем, даже этот путь не свободен от препятствий. Процесс предварительных заказов книги был приостановлен на второй день, потому что первоначальный поставщик электронной коммерции, Americart, «отказался предоставить [мне] сервис корзины для продажи хакерских материалов... 15 долларов в месяц не окупают для нас риск оказаться упомянутыми в иске по DMCA».

Я должен подчеркнуть, что эта книга не нарушает авторские права Microsoft, а знания, представленные в этой книге, нельзя напрямую применить для обхода защиты авторских прав. Чтобы совершить нарушающее действие, человеку пришлось бы отточить навыки и применить значительный объём дополнительного искусства и ноу-хау, специально направленного на обход контроля авторских прав. Утверждать, что эта книга является инструментом обхода, было бы равносильно утверждению, что все книги о печатных платах, встраиваемом ПО или криптографии тоже являются инструментами обхода.

Объём действия DMCA применительно к «добросовестному использованию» аппаратуры - ещё одна важная политическая тема с огромными экономическими последствиями. Незаконно ли модифицировать или обходить криптографически защищённую последовательность загрузки ради запуска альтернативного, законно приобретённого или созданного программного обеспечения? Этот вопрос может быть отчасти решён судьбой хакеров Xbox. Строгое толкование исключения DMCA для обратной разработки [7] показывает сильные аргументы в пользу признания таких действий по обходу незаконными. В частности, обратная разработка разрешена только для обеспечения совместимости, где совместимость означает «способность компьютерных программ обмениваться информацией и способность таких программ взаимно использовать информацию, которой они обменялись». Но это определение содержит две потенциальные мины. Во-первых, обход аппаратных мер безопасности, возможно, отличается от обхода мер безопасности программы (ПО). Технически это может быть не очень сильный аргумент, но, насколько мне известно, этот пункт ещё не проходил юридической проверки. Во-вторых, цель на самом деле состоит не в обмене информацией с аппаратными мерами безопасности, а в их обходе.

Последний аргумент против разрешения обратной разработки аппаратных механизмов безопасности - побочный обход авторских прав. Информация, полученная в процессе обратной разработки, может быть в равной степени применена для создания устройств обхода авторских прав. Иными словами, фундаментальное исследование, которое обеспечивает совместимость, по крайней мере в случае Xbox, может быть также косвенно применено теми, кто хочет создать устройства обхода. Как выясняется, некоторые очень конкретные конструктивные недостатки Xbox позволяют обходить безопасность загрузки, не обязательно позволяя обходить авторские права, хотя эти недостатки могут быть исправлены в ближайшем будущем, снова ставя нас лицом к лицу с исходным вопросом.

Если окажется, что «добросовестное использование» не охватывает обратную разработку безопасности Xbox ради запуска альтернативных приложений, это будет иметь серьёзные экономические последствия. Самое значительное из них состоит в том, что Microsoft сможет продавать конечным пользователям юридически ограниченное оборудование, привязывая пользователей к своей программной базе. Это можно использовать для создания нерушимой монополии на компьютерное оборудование и ПО. Например, Microsoft могла бы предлагать субсидии поставщикам, которые решат защитить своё оборудование так, чтобы оно запускало операционную систему Microsoft. Этот финансовый стимул будет передан покупателям, у которых появится мотивация покупать оборудование со скидкой. После того как значительная часть установленной базы оборудования окажется привязанной к операционным системам Microsoft, Microsoft сможет устанавливать цены на свои продукты на рынке без конкуренции, поскольку запуск любой другой операционной системы на заблокированном оборудовании будет незаконным.

В реальности Microsoft, возможно, было бы трудно реализовать такой сценарий, даже если бы DMCA действительно ограничивал добросовестное использование аппаратуры, поскольку государственные и общественные органы внимательно следят за деятельностью Microsoft на предмет монополистического поведения. Однако на других развивающихся рынках, таких как смартфоны, PDA и приставки set-top box, попытка поставщика получить преимущество над конкурентами такими демпинговыми приёмами может оказаться вполне реалистичной. По крайней мере, такие приёмы можно использовать, чтобы задержать конкуренцию на время судебного процесса, а этого может быть достаточно, чтобы нанести непоправимый вред рыночному положению конкурента. Именно из-за этих опасений многие хакеры Xbox сознательно действовали так, чтобы выражать свои политические убеждения инженерными усилиями.

Автор за своим рабочим местом.

Люди за хаками

На протяжении этой книги я включаю профили различных хакеров, которые согласились дать интервью. Этот набор хакеров ни в коем случае не является единственным набором хакеров; фактически это самоотобравшаяся группа, поскольку многие хакеры работают скрытно из страха преследования или потому, что наняты компаниями с сильными связями с Microsoft. Цель этих интервью - немного рассказать о людях, стоящих за хаками, и представить их мотивацию и методы, чтобы способствовать пониманию и вдохновить новых хакеров присоединиться к нашим рядам.

Позвольте мне начать этот процесс с представления самого себя. Я Эндрю «bunnie» Хуан; большинство людей зовут меня bunnie. На момент написания мне было 27 лет, я сын Эндрю К. и Маргарет Хуан. Я родился и вырос в Каламазу, штат Мичиган, но сейчас живу в Сан-Диего, штат Калифорния, со своей замечательной невестой Никки Джастис. Недавно я окончил MIT со степенью PhD по электротехнике. Одна из причин, по которой меня выбрали автором этой книги о хакинге Xbox, состоит в том, что я обнаружил и опубликовал первую известную слабость в системе безопасности Microsoft Xbox.

В целом я хакерствую потому, что довольно приятно знать: чья-то жизнь стала лучше благодаря тому, что я построил. Я чувствую, что обязан применять свои таланты и возвращать обществу то, что оно дало мне. Мне также нравится вызов исследования. Я хочу понимать электронику настолько глубоко, насколько могу. Чёрные ящики меня раздражают; ничто не пробуждает моё любопытство сильнее, чем коробка, которую мне не разрешают открыть или понять. В результате у меня есть доверительный интерес к криптографии и методам безопасности.

Я занимаюсь аппаратным хакингом потому, что мне нравится эстетика электроники; есть что-то удовлетворяющее в том, чтобы в конце дня иметь осязаемый артефакт, в отличие от эфемерных

битов программного кода. Это может звучать немного глупо, но одно из моих занятий - разбирать электронные устройства и «читать» печатные платы. В запахе совершенно нового электронного оборудования, только что вынутого из антистатических пакетов, есть что-то волнующее; думаю, это запах разворачивающегося нового приключения. Он манит, как стопка чистой бумаги: мне интересно, что я сделаю с этими пустыми страницами. Стопка чистой белой бумаги стоит передо мной и бросает вызов: заполни меня полезной информацией.

Моя любознательность берёт начало в детстве. Когда мне было около семи лет, отец купил клон Apple II. Он купил только материнскую плату, поэтому корпуса у неё не было. Я до сих пор помню, как он впервые достал её из коробки: зелёная печатная плата, блестящие микросхемы и все эти разноцветные резисторы и конденсаторы. Мне хотелось с ней играть! Как бы мне ни было любопытно устройство Apple II, мне не разрешали прикасаться к материнской плате. Разумеется, это означало, что всякий раз, когда родители не смотрели, я вынимал микросхемы из их панелек на материнской плате и делал глупости вроде установки их задом наперёд, чтобы посмотреть, что произойдёт.

После того как я несколько раз почти уничтожил компьютер, родители купили мне набор радиолюбителя 200-in-1 от Radio Shack и мою первую книгу по электронике, *Getting Started in Electronics* Форреста Мимса III. Для меня это было отличным введением в электронику, поскольку удовлетворяло моё желание играть со схемами и компонентами. Мой дядя также отдал мне свой старый экземпляр *The Art of Electronics* Хоровица и Хилла вместе с парой книг о микропроцессорах. Я подписался на журнал *Byte*, который в те времена регулярно публиковал колонки об аппаратных проектах со схемами и фотографиями.

В конце концов я развил достаточное чутьё к электронике, чтобы начать понимать схемы и листинги ROM, включённые в руководства пользователя Apple II. (Я по-прежнему считаю, что компьютеры должны поставляться с полными схемами и исходным кодом.) К восьмому классу я уже понимал достаточно, чтобы построить собственную плату расширения для Apple II. На плате был речевой синтезатор General Instruments SPO-256, который я купил в Radio Shack. Я также добавил к Apple II аналого-цифровой преобразователь и написал приложение, превратившее мой Apple II в говорящий вольтметр. Я продолжал строить аппаратные устройства, и до поступления в MIT собрал собственный рабочий встраиваемый компьютер на микропроцессоре 80188.

В годы бакалавриата в MIT я уклонялся от рутины учебных заданий, строя забавные небольшие проекты, такие как дистанционно управляемый выключатель света и реагирующая на музыку праздничная подсветка для моего братства ZBT. Именно в эти годы я впервые познакомился с доступными услугами прототипирования и CAD-инструментами для PCB, такими как обсуждаемые в приложении C, «Getting Into PCB Layout».

Появление услуг изготовления печатных плат, укладывающихся в бюджет студента колледжа, - знаковое событие для аппаратных хакеров. Наконец-то инструмент для монтажа накруткой можно убрать, а поверхностно-монтируемые компоненты и сложные схемы оказываются в пределах досягаемости обычных любителей.

На протяжении многих лет я старался описывать свои проекты на своей веб-странице (http://www.xenatera.com/bunnie), чтобы каждый мог извлечь пользу из моего опыта. Многие из моих проектов доступны со схемами, Gerber-файлами и исходным кодом, хотя некоторые из более поздних проектов были консультационными работами, поэтому, к сожалению, я не могу поделиться этими результатами с миром.

Пока у меня есть ваше внимание, я хотел бы прояснить один момент. Я не получил степень PhD в MIT за хакинг Xbox. Хакинг Xbox на самом деле был отвлечением от моей диссертации, косвенно связанным с ней, но не центральным для её темы.

Моя диссертация о суперкомпьютерах [8] была посвящена архитектуре для эффективной миграции кода и данных. Мой интерес к игровым консолям проистекает из природного любопытства ко всякому оборудованию в сочетании с поддержкой моего научного руководителя, доктора Тома Найта. Игровые консоли представляют вершину производительности на единицу стоимости, а стоимость сегодня является существенной проблемой для суперкомпьютеров. Поэтому меня поощряли изучать все игровые консоли, чтобы понять, что я могу узнать о построении экономически эффективного оборудования. То, что Xbox ещё и имела интересную систему безопасности, было бонусом: поскольку государственные ведомства очень интересуются суперкомпьютерными технологиями, безопасность суперкомпьютеров всегда является темой для рассмотрения. (Кстати, мои коллеги по исследовательской группе написали очень интересную статью о построении доверенных компьютеров; [9] рекомендую прочитать её, если вам любопытны альтернативы криптографически защищённым доверенным вычислительным платформам, таким как Palladium и TCPA.)

Мой лучший совет начинающим аппаратным хакерам - быть настойчивыми и тщательными. Важно, что настойчивость и тщательность приходят естественно, если вы любите то, чем занимаетесь. Кроме того, быть аппаратным хакером - значит отчасти быть барахольщиком. Покупать новое оборудование непомерно дорого, поэтому я привычно накапливаю сломанное и обесцененное оборудование и инструменты, даже если не знаю точно, что могу с ними сделать и смогу ли их починить. Оказывается, попытки чинить измерительное оборудование сами по себе являются учебным опытом и могут быть весьма вознаграждающими, даже если в итоге приходится разобрать эту проклятую штуку на запчасти.

Процитирую бывшего евангелиста Apple и нынешнего руководителя Garage Technology Ventures Гая Кавасаки: «ешь как птица, испражняйся как слон». Кавасаки указывает, что колибри каждый день съедает эквивалент 50% массы своего тела. Поэтому есть как птица означает, что у вас должен быть бесконечный аппетит к информации. Подписывайтесь на бесплатные отраслевые журналы по электронике, просматривайте веб (но избирательно относитесь к сайтам, которые просматриваете: вы есть то, что вы едите), ходите на бесплатные выставки и подписывайтесь на каждый каталог и периодическое издание, до которых сможете добраться; разбирайте каждое электронное устройство, которое принадлежит вам и вашим друзьям, и пытайтесь узнать всё возможное из его конструкции.

В аппаратном хакинге половину самых трудных проблем можно решить или облегчить, просто выбрав правильные компоненты или приёмы. «Испражняйся как слон» относится к тому, чтобы делиться информацией и открытиями с другими хакерами. Сколько бы информации вы ни переварили, вы никогда не сможете знать всё. Свободно делясь своими находками, вы приглашаете советы и добрую волю других хакеров, и это приводит к синергии умов. Особенно в аппаратном хакинге, где все результаты имеют основу в осязаемых артефактах, сокрытие ваших приёмов и результатов означает лишь, что другие люди в конце концов заново изобретут вашу работу без вашей помощи. С другой стороны, проявляйте некоторую рассудительность в том, что говорите или чем делитесь: у людей ограниченная пропускная способность, и они будут слушать внимательнее, если вы делитесь результатами, которые в каком-то отношении новы или интересны.

Итак, берите отвёртку, и начнём хакерствовать!

Сноски

[1] [назад] «Trustworthy Computing» Билла Гейтса, http://www.microsoft.com/mscorp/execmail/2002/07-18twc.asp
[2] [назад] Источник: NPDFunworld.
[3] [назад] Хорошая новость состоит в том, что в последнее время технологии аппаратного хакинга догоняют закон Мура, приводя к возрождению аппаратного хакинга. Появились доступные услуги изготовления печатных плат, а рождение интернета упростило процесс приобретения компонентов. Кроме того, такие сервисы, как полупроводниковая фабрика Mosis и услуги FIB (focused ion beam, сфокусированный ионный пучок), начали переводить хакинг интегральных схем в область финансово возможного для отдельных энтузиастов аппаратного хакинга.

[4] [назад] Родни Брукс, директор Лаборатории искусственного интеллекта MIT, однажды сказал, что голливудская идея чудаковатого изобретателя, создающего у себя в подвале искусственно-интеллектуальное существо, примерно равнозначна тому, как если бы кто-то построил у себя во дворе реактивный лайнер Boeing 747.
[5] [назад] 17 U.S.C § 1201(g)(3), Factors in determining exemption. Разумеется, значение выражения «соответствующая подготовка или опыт» не определено. Я считаю, что лучшая подготовка для прикладных исследований криптографии должна включать некоторый практический опыт хакинга реальных криптосистем.
[6] [назад] 17 U.S.C. § 1201(g)(2), Permissible acts of encryption research.
[7] [назад] 17 U.S.C § 1201(f), Reverse Engineering.
[8] [назад] Текст моей PhD-диссертации можно найти по адресу http://www.xenatera.com/bunnie/phdthesis.pdf
[9] [назад] «A Minimal Trusted Computing Base for Dynamically Ensuring Secure Information Flow» Джереми Брауна и Тома Найта можно найти по адресу http://www.ai.mit.edu/projects/aries/Documents/Memos/ARIES-15.pdf

Глава 1. Аннулирование гарантии

Инструменты ремесла

Аппаратный хакинг поначалу может казаться пугающим из-за сложных инструментов, которые требуются для некоторых проектов. К счастью, большинство базовых проектов можно выполнить, вложив в инструменты совсем немного денег - сумму, сравнимую с ценой одной-двух видеоигр. В приложении A, «Где купить оборудование», приведён рекомендуемый список начальных инструментов и инструкции по их заказу.

В этой главе речь пойдёт о базовых инструментах, которые понадобятся для серьёзного аппаратного хакинга: об инструментах, позволяющих вскрывать устройства, устанавливать и снимать электронные компоненты, диагностировать и прозванивать схемы, а также проектировать печатные платы. Качественные версии инструментов из первых двух групп можно купить по довольно разумным ценам. Диагностические и измерительные приборы, такие как осциллографы и логические анализаторы, стоят на вес золота, но вы обнаружите, что это золото очень тяжёлое, и такие приборы станут серьёзной инвестицией. Что касается инструментов для проектирования печатных плат, некоторые из лучших решений могут оказаться удивительно доступными.

Глава завершится пошаговым иллюстрированным руководством по вскрытию Xbox. Более опытные аппаратные хакеры могут пропустить следующие пару глав.

Инструменты для вскрытия устройств

Первый шаг в хакинге чего бы то ни было - снять крышку. Большинство электронных приборов можно открыть с помощью набора крестовых и плоских отвёрток, но самые интересные коробки потребуют набора специальных защитных бит.

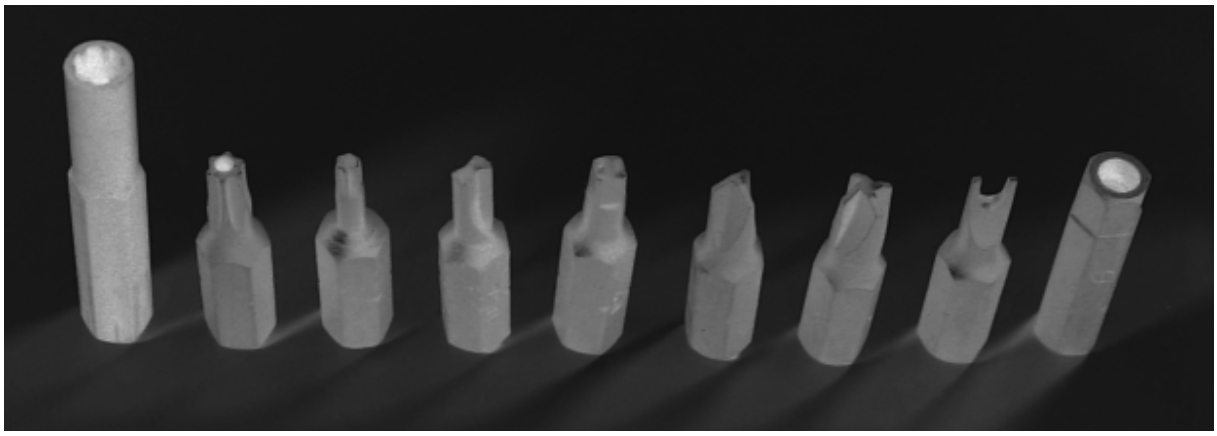


Рисунок 1-1. Набор защитных бит. Слева направо: Nintendo 4,5 мм, security torx, стандартный torx, clutch, Robertson или square, tri-wing, torq, spanner и security allen или hex.

На рисунке 1-1 показана линейка некоторых распространённых защитных бит. Удивительно, но наборы защитных бит доступны по цене и их легко достать. MCM Electronics (www.mcmelectronics.com) продаёт набор из 105 защитных бит (номер заказа MCM 22-3495) дешевле двадцати долларов и набор из 32 предметов (номер заказа MCM 22-1875) дешевле десяти долларов. Эти наборы вполне стоят вложений. Защитные биты Nintendo продаются

отдельно. Крупную защитную биту Nintendo, используемую в Nintendo Gamecube, можно купить за несколько долларов (номер заказа MCM 22-1150, «4.5mm Security Bit»). Меньшая версия этой биты (номер заказа MCM 22-1145, «3.8mm Security Bit») также используется в более старых системах Nintendo и их игровых картриджах.

Xbox использует стандартные биты torx (шестиконечная звезда) размеров T10, T15 и T20. Эти биты довольно распространены и продаются в хозяйственных магазинах, таких как Home Depot. Вам также может пригодиться магнитный удлинитель-держатель для бит, чтобы добраться до пары тесных мест вокруг жёсткого диска и DVD-привода в Xbox.

Не применяйте чрезмерную силу, когда снимаете крышку с оборудования. Если вам кажется, что вы выкрутили все винты, но крышка всё ещё держится, скорее всего, вы либо пропустили винт, либо нужно нажать на какие-то фрикционные защёлки. Кроме того, винты часто прячутся под резиновыми ножками на нижней стороне устройства или под наклейкой. Чтобы найти винты, скрытые наклейками, сильно потрите поверхность наклейки. Вы почувствуете мягкое место там, где под ней находится винт. (Повреждение такой наклейки для доступа к винту мгновенно аннулирует гарантию на оборудование, но не бойтесь: большинство устройств рассчитано на обслуживание, поэтому простое снятие крышки редко вызывает какие-либо повреждения.)

Иногда вам встретится упрямая сборка, которая отказывается разбираться. Если крышка или панель отгибается по краям или, кажется, имеет некоторую свободу движения, её может удерживать какой-то фрикционный замок. Фрикционные замки обычно представляют собой конструкции «язычок и паз», сформированные так, что вставить язычок намного проще, чем вынуть его. В таком случае найдите язычок, наблюдая, где корпус, похоже, застрял, и нажмите на язычок маленькой плоской отвёрткой, одновременно осторожно подтягивая корпус вверх. Если таких язычков несколько, вставьте какой-нибудь клин, например другую отвёртку или скрепку, чтобы язычок не защёлкнулся обратно, пока вы открываете остальные.

Если крышка или панель не двигается даже чуть-чуть при уверенном нажатии, она также может быть приклеена или даже заварена. Например, блоки питания wall-wart (квадратные чёрные коробочки, которые включаются прямо в розетку) часто запечатаны именно таким образом. Разборка такого оборудования может означать, что вы уже никогда не сможете собрать его обратно в исходном виде.

Инструменты для установки и снятия компонентов

Электронные компоненты крепятся к платам пайкой. При пайке легкоплавкий сплав, известный как припой, нагревается и растекается вокруг соединяемых металлов. Припой и металлы образуют локальный сплав. После остывания соединения компоненты оказываются электрически и механически соединены.

Базовые инструменты для пайки - паяльник, припой, флюс и оплётка для выпайки. (Пинцет с тонкими кончиками тоже очень удобен для работ с компонентами малого шага или мелкими деталями.) Паяльник - ручной инструмент, состоящий из нагревательного элемента и жала; жало используется для плавления припоев теплопроводностью или прямым контактом, в отличие от других инструментов, использующих горячие газы или интенсивное инфракрасное излучение. Тип жала паяльника, требуемый для оптимальной передачи тепла, зависит от ситуации. Например, плоское жало «chisel» или «conical chisel» будет работать лучше, чем простое заострённое жало, при пайке большинства небольших компонентов поверхностного монтажа.

Паяльники тоже бывают самых разных классов. Самые дешёвые стоят около десяти долларов, поставляются с большими, неудобными жалами и не имеют регулировки температуры: они просто

нагреваются настолько сильно, насколько могут. Более хорошие паяльники стоят дороже и имеют датчик, который активно регулирует температуру жала. Регулировка температуры делает действие инструмента более предсказуемым и продлевает срок службы жала. У лучших паяльников также шире выбор жал, среди которых могут быть очень тонкие, пригодные для работы с крошечными компонентами, встречающимися сегодня в большинстве электроники. Для лёгкого использования достаточно качественного паяльника, включаемого прямо в розетку, с хорошим жалом. Однако если вы планируете собирать платы и всерьёз заниматься аппаратным хакингом, сотня долларов за качественный паяльник с регулировкой температуры, такой как Weller WTCPT или Weller WES50, вполне стоит вложения.

Припои бывают самых разных видов. Для большинства задач достаточно эвтектической проволоки припоя Pb-Sn с сердечником из no-clean или водосмываемого флюса. Эвтектические сплавы желательны потому, что они переходят напрямую из жидкой фазы в однородное твёрдое состояние. Kester - крупный производитель припоев; их стандартные припои с сердечником, Formula 245 и 331, оба довольно хороши. Formula 245 использует no-clean флюс, но при желании остатки можно удалить ватной палочкой с небольшим количеством изопропилового спирта. Formula 331 имеет флюсовый сердечник, который работает с большим количеством материалов, чем 245. Однако при использовании 331 нужно вскоре после пайки промыть плату водой, иначе остатки флюса станут липкими и, возможно, будут мешать работе схемы. Многие дистрибьюторы продают припой Kester; например, Kester 24-6337-8802 (25-gauge проволока припоя Formula 245 в катушке 1 фунт) имеет номер детали Digi-Key KE1410-ND. Припой, продаваемый в большинстве магазинов Radio Shack, тоже вполне хорош для пайки, хотя он обычно оставляет липкий чёрный налёт и требует очистки органическими растворителями.

Припой также может поставляться в виде пасты, где крошечные шарики припоя взвешены во флюсовой матрице. Паяльная паста может быть очень полезна при установке компонентов поверхностного монтажа с малым шагом. (Дополнительные сведения см. в приложении B, «Приёмы пайки».)

Если паяное соединение упорно не формируется, флюс - панацея. Всегда держите под рукой немного флюса. Когда соединение образуется неправильно, небольшая капля флюса, нанесённая прямо на соединение, обычно исправляет проблему. Флюсы также выпускаются во множестве паст и жидкостей, каждая из которых требует своего метода очистки. Удобное решение для нанесения флюса - флюс-ручка, например Kester 83-1000-0951, no-clean флюс-ручка Formula 951. Эту флюс-ручку можно купить в Digi-Key под номером детали KE1804-ND всего за несколько долларов. Radio Shack также продаёт флюсовую пасту в тюбике, но эта паста грязная и требует очистки.

Наконец, оплётки для выпайки полезны для очистки любых огрехов или ошибок пайки, которые вы можете допустить. Оплётка для выпайки - это тонкая плетёная медная проволока, обычно пропитанная сухим флюсом. Чтобы использовать её, поместите её между паяльником и соединением, которое хотите очистить; когда оплётка нагреется, лишний припой на соединении втянется в капилляры оплётки. Несмотря на то что оплётка может быть уже пропитана флюсом, нанесение капли флюса на оплётку перед использованием всё равно помогает процессу. Chemtronics выпускает хорошую линейку оплёток для выпайки; пример детали - Chemtronics 60-3-5 «No-Clean Solder-Wick» (номер детали Digi-Key 60-3-5-ND).

Базовую технику пайки я обсуждаю в начале главы 2, где вы научитесь устанавливать синий LED на переднюю панель Xbox.

Инструменты для тестирования и диагностики

Электронное измерительное оборудование существует в таком же множестве форм, как и электронные продукты. Для начинающего базовый инструмент из категории «обязательно иметь» - цифровой мультиметр. За последние несколько лет цифровые мультиметры (DMM) стали очень функциональными и доступными; типичный прибор сможет измерять сопротивление, напряжение, ток, ёмкость, полярность диода и целостность цепи по цене около пятидесяти долларов. Radio Shack и Jameco (www.jameco.com) предлагают достаточно разумный выбор мультиметров начального уровня. (В приложении A, «Где купить хакерское оборудование», есть рекомендация по мультиметру начального уровня.)

Для базовых проектов по модификации и сборке наборов DMM полезны при проверке соединений на короткое замыкание, а также при проверке базового состояния схемы до и после подачи питания. Режим прозвонки в DMM может помочь, когда вам кажется, что вы испортили паяное соединение.

В режиме прозвонки DMM издаёт тональный сигнал всякий раз, когда между измерительными щупами существует низкоомный путь. Поэтому функция прозвонки полезна как для проверки целостности паяного соединения, так и для проверки коротких замыканий с соседними соединениями. Не следует использовать режим прозвонки для проверки коротких замыканий в цепях питания, потому что на некоторых платах между питанием и землёй вполне нормально может быть достаточно низкое сопротивление (примерно десять ом), чтобы сработал звуковой сигнал прозвонки. Поэтому перед подачей питания на любую только что модифицированную или собранную плату используйте режим измерения сопротивления, чтобы проверить и убедиться, что на линиях питания нет полного короткого замыкания (нулевого сопротивления).

Для обратной разработки и более продвинутых проектов базовые инструменты, которые вам понадобятся, - осциллограф и иногда логический анализатор. Осциллографы полезны для захвата подробной формы электрических сигналов. С помощью осциллографа можно диагностировать проблемы синхронизации, шума и помех.

Базовые определяющие характеристики осциллографа - количество каналов или форм сигналов, которые он может отображать одновременно, и его максимальная электрическая полоса пропускания. Качественные осциллографы обычно имеют четыре канала и полосу свыше 500 МГц; дешёвые или бывшие в употреблении осциллографы часто имеют только два канала и где-то от 20 до 100 МГц полезной полосы. Главное ограничение всех осциллографов состоит в том, что они могут отображать только короткий фрагмент электрического сигнала.

Логические анализаторы полезны для захвата больших объёмов цифровых данных. Они обменивают способность захватывать форму сигнала на широкие возможности анализа и регистрации данных. Логические анализаторы полезны для диагностики сложных цифровых шин и схем. Базовые определяющие характеристики логического анализатора - количество цифровых каналов, которые он может дискретизировать, максимальная частота дискретизации и максимальная глубина выборки. Типичный современный логический анализатор может иметь несколько десятков каналов, частоту дискретизации в сотни мегагерц и глубину выборки в пару мегабайт. Среди других возможностей логических анализаторов - программируемые алгоритмы запуска и способность обнаруживать глитчи или runt-импульсы.

К сожалению, средняя цена нового осциллографа или логического анализатора составляет от тысяч до десятков тысяч долларов. Хорошая новость состоит в том, что большинству проектов не потребуются самые новые и лучшие измерительные технологии, поэтому можно обойтись подержанным оборудованием. Радиолюбительские барахолки - отличные места, где можно дёшево подобрать старый осциллограф или анализатор; на eBay тоже время от времени встречаются хорошие предложения. Если вам приходится выбирать между покупкой осциллографа и

логического анализатора, я бы рекомендовал сначала купить осциллограф: логический анализатор далеко не так универсален, как осциллограф, и обычно дороже. Осциллограф можно уговорить захватить ограниченный объём логических данных, тогда как логический анализатор никогда нельзя использовать для измерения аналоговой формы сигнала. Кроме того, построить самодельный логический анализатор с использованием FPGA и специальных плат проще, чем построить осциллограф сопоставимого качества. Самодельные логические анализаторы можно относительно дёшево строить для высококлассных высокоскоростных применений. (В главе 8 описано, как я построил самодельный логический анализатор для прослушивания критической высокоскоростной шины в Xbox.)

В крайнем случае очень простое устройство захвата цифровых трасс можно построить примерно из деталей Radio Shack на пятьдесят долларов. Однажды мне пришлось захватить данные на порту клавиатуры PS/2, но у меня не было никакого измерительного оборудования, а данные нужно было захватить немедленно. Макетная плата с несколькими светодиодными шкалами, подключёнными к набору 8-битных регистров (деталь 74HCT574), сдвигающих данные, решила задачу - все компоненты я купил в Radio Shack. Реальная конструкция довольно проста, но поскольку её применение очень ограничено, я избавлю вас от подробностей. Суть в том, что вы можете строить собственные устройства для захвата цифровых данных; об этом стоит подумать, прежде чем выкладывать несколько тысяч долларов за логический анализатор.

Инструменты для проектирования

Последний набор инструментов, необходимый для завершения коллекции любого хакера, - это набор электронных инструментов проектирования для PC-плат и FPGA. Тема проектирования PC-плат и FPGA обсуждается в приложениях, но здесь стоит упомянуть, что качественные версии таких инструментов можно получить почти бесплатно. В результате можно спроектировать и построить полноценную печатную плату со сложными реконфигурируемыми аппаратными компонентами дешевле стоимости Xbox - включая стоимость инструментов проектирования и изготовления.

Проектирование PC-плат раньше было очень дорогим предприятием: инструменты стоили тысячи долларов, а простой производственный запуск платы стоил несколько сотен долларов. Сегодня новичок может изготовить простую плату на общую сумму меньше семидесяти долларов. Для проектирования PC-плат Altium, ранее называвшаяся Protel, продаёт инструмент CircuitMaker2000. Хотя я не использовал CircuitMaker2000 широко, моё первое впечатление таково: он очень похож на теперь снятый с производства Protel 99SE от Altium. Загрузите бесплатную 30-дневную демонстрационную версию или их бесплатную студенческую версию (с ограничениями), которая идеально подходит для первого проекта платы, с http://www.circuitmaker.com. После того как вы спроектируете свою первую плату с помощью бесплатного инструмента, её можно изготовить у поставщика вроде Sierra Proto Express (http://www.sierraprotoexpress.com) примерно за 30 долларов за плату на момент написания, при минимальном заказе в две платы. Как видите, цена больше не является серьёзным барьером, и я призываю вас попробовать построить один-два проекта на собственных печатных платах.

FPGA - Field Programmable Gate Arrays, программируемые пользователем вентильные матрицы - являются решением для недорогого прототипирования кремния. FPGA состоит из большого массива вентилей и элементов хранения с программируемыми соединениями. В результате FPGA могут реализовывать всевозможные цифровые устройства, ограниченные только ёмкостью вентилей и проводников самой FPGA. Более крупные FPGA, ёмкостью в несколько миллионов вентилей, могут содержать целые системы вместе с микропроцессорами и периферией. FPGA также очень доступны: Xilinx Spartan II FPGA на 100 000 вентилей стоит около 20 долларов при

покупке поштучно. И что ещё лучше, для FPGA Xilinx можно бесплатно получить неограниченные среды проектирования и синтеза! У Xilinx есть бесплатный продукт под названием «ISE WebPack», доступный на их сайте (www.xilinx.com), который включает такие возможности, как синтез Verilog и VHDL, генерация HDL testbench и программное обеспечение для анализа энергопотребления. Verilog - C-подобный язык аппаратного проектирования; его можно представить как строго типизированный, многопоточный C. Это отличная новость для программных хакеров, которые хотели бы попробовать себя в аппаратуре. Существуют даже open-source сообщества аппаратного проектирования, такие как www.opencores.org, где можно скачать код микропроцессоров и других интересных цифровых компонентов, опять же бесплатно.

Статическое электричество: убийца схем. Статическое электричество, также известное как Electro-Static Discharge (ESD), - бич интегральных схем. Современные IC особенно чувствительны к ESD; нескольких вольт достаточно, чтобы уничтожить открытый транзистор. Поскольку вы не чувствуете разряды статического электричества до диапазона сотен или тысяч вольт, вы можете уничтожать такие устройства, даже не зная об этом.

Хорошая новость состоит в том, что большинство микросхем строятся со специальными структурами, помогающими сделать их более устойчивыми к ESD. Тем не менее лучше добровольно не участвовать в их проверке. Чтобы нейтрализовать статическое электричество на своём теле, всегда касайтесь заземлённого металлического предмета перед тем, как прикасаться к печатной плате или микросхеме. Открытый металл корпуса компьютера, подключённого к правильно смонтированной бытовой розетке, - хорошая отправная точка.

Ношение антистатического браслета, доступного почти в любом компьютерном магазине, сведёт к минимуму риск повредить Xbox разрядом ESD. Чтобы браслет был эффективен, он должен быть подключён к заземлённому объекту.

Если вам нравится жить на грани, работа босиком на не покрытом ковром бетонном полу также будет держать вас заземлённым. Голые бетонные полы удивительно проводящие, вплоть до того, что от длительного контакта с электронным оборудованием, подключённым к неправильно смонтированным розеткам, можно получить удар током или ожог. Линолеум и деревянные полы тоже могут быть эффективными точками заземления, в зависимости от типа плитки или воска, использованного на полу. Чтобы гарантировать достаточную проводимость пола, можно наносить специальные проводящие воски или спреи.

Разборка Xbox

Теперь, когда мы обсудили некоторые инструменты, которые понадобятся вам для хакинга, давайте займёмся хакингом. Первый шаг в хакинге Xbox - открыть коробку. Вот инструменты, которые понадобятся, чтобы снять крышку:

• Биты Torx T10 и T20 (биты в форме шестиконечной звезды).
• Рукоятка-отвёртка для бит.
• Антистатические средства безопасности (см. «Статическое электричество: убийца схем»).
• Маленькая плоская отвёртка (полезна, но не обязательна).

NB. Перед тем как начать разбирать Xbox, имейте в виду несколько вещей: во-первых, при разборке всегда есть некоторый риск необратимого повреждения, а разборка Xbox аннулирует вашу гарантию. Во-вторых, обязательно прочитайте весь раздел целиком, прежде чем продолжать. И в-третьих, получайте удовольствие.

Шаг 1. Сначала безопасность

Отключите Xbox от сети. Если оставить Xbox подключённой, вы подвергнетесь опасному, возможно смертельному, напряжению.

Шаг 2. Выкрутите винты корпуса

Переверните Xbox вверх дном и осмотрите нижнюю сторону. Верхнюю и нижнюю половины внешней оболочки удерживают шесть винтов, и все они скрыты под наклейками или резиновыми ножками. На рисунке 1-2 показано положение всех мест расположения винтов.

Резиновые ножки приклеены к Xbox сильным клеем. Для снятия ножек обычно потребуется небольшая помощь плоской отвёртки. На рисунке 1-3 показана эта процедура. После того как вы подденете край резиновой ножки, отогните её назад как можно ровнее, чтобы сохранить клейкую подложку на ножке. Если действовать аккуратно, позже вы сможете прикрепить ножку обратно, хотя после пары циклов снятия клей потеряет липкость. В качестве замены можно купить резиновые ножки в любом хозяйственном магазине и прикрепить их, если вы используете Xbox на поверхности, которая скользкая или чувствительна к царапинам.

Используйте биту Torx T20, чтобы выкрутить четыре винта, которые находились под резиновыми ножками. Винты довольно длинные, но резьба у них короткая, поэтому выкручивание должно пройти быстро.

Последние два винта скрыты под наклейкой с серийным номером и наклейкой сертификации продукта. На рисунке 1-4 показано расположение этих винтов.

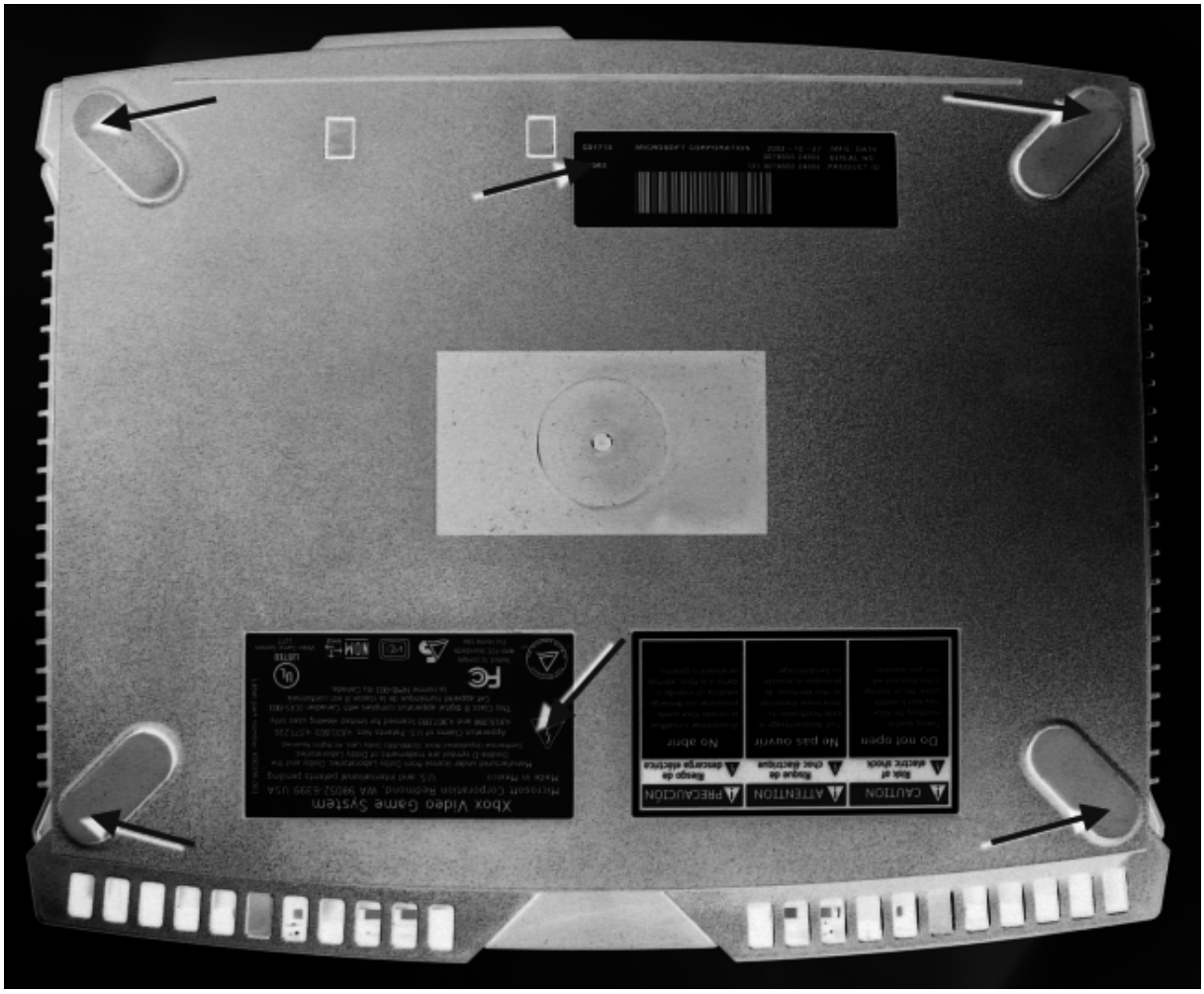


Рисунок 1-2. Расположение винтов корпуса Xbox. Это вид нижней стороны Xbox.

Чтобы найти их, сильно проведите пальцем по общей области, где должны находиться винты. Над отверстиями винтов наклейка слегка продавится. Проколите наклейку битой и сдвигайте или поворачивайте биту, пока она не попадёт в отверстие винта, затем выкрутите винт. Если вам важна косметическая целостность наклеек, удерживайте наклейку на месте, пока выкручиваете винт, иначе она отогнётся или порвётся.

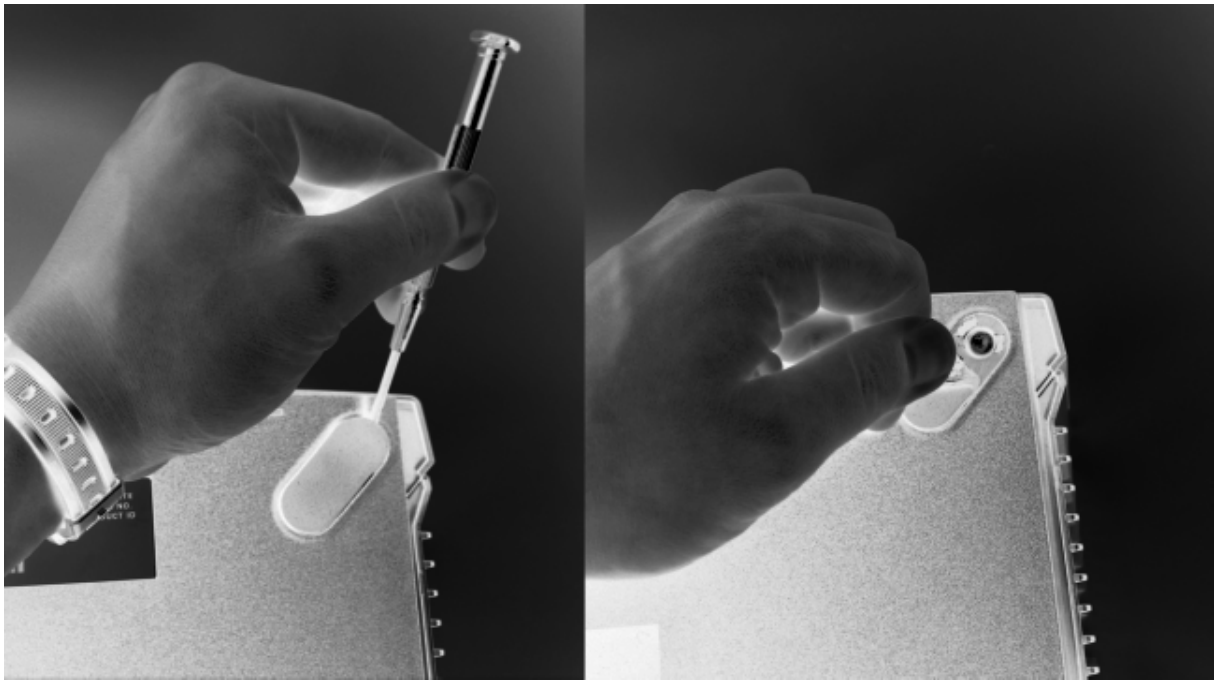


Рисунок 1-3. Используйте маленькую плоскую отвёртку, чтобы поддеть край резиновых ножек Xbox, затем аккуратно отогните их.

Рисунок 1-4. Расположение винтов, скрытых наклейками с серийным номером и сертификацией продукта.

Совет. Держите под рукой небольшой лоток или пластиковый пакет для хранения винтов, чтобы не потерять их. Винты, удерживающие Xbox, довольно специфичны, и вам может оказаться трудно купить подходящую замену в местном хозяйственном магазине.

Шаг 3. Снимите верхнюю крышку

К этому моменту у вас должны быть выкручены шесть одинаковых длинных винтов. Поверните Xbox правильной стороной вверх, осторожно возьмитесь за коробку с боков открытыми ладонями и попробуйте снять крышку лёгким покачиванием. Если этим способом крышка не снимается, возможно, вам нужно «сдвинуть» крышку, поддевая корпус пальцами сзади. В некоторых редких случаях вам также придётся поддеть её отвёрткой спереди, но делайте это осторожно и мягко. На рисунке 1-5 показаны некоторые точки, которые можно использовать, чтобы помочь снять корпус.

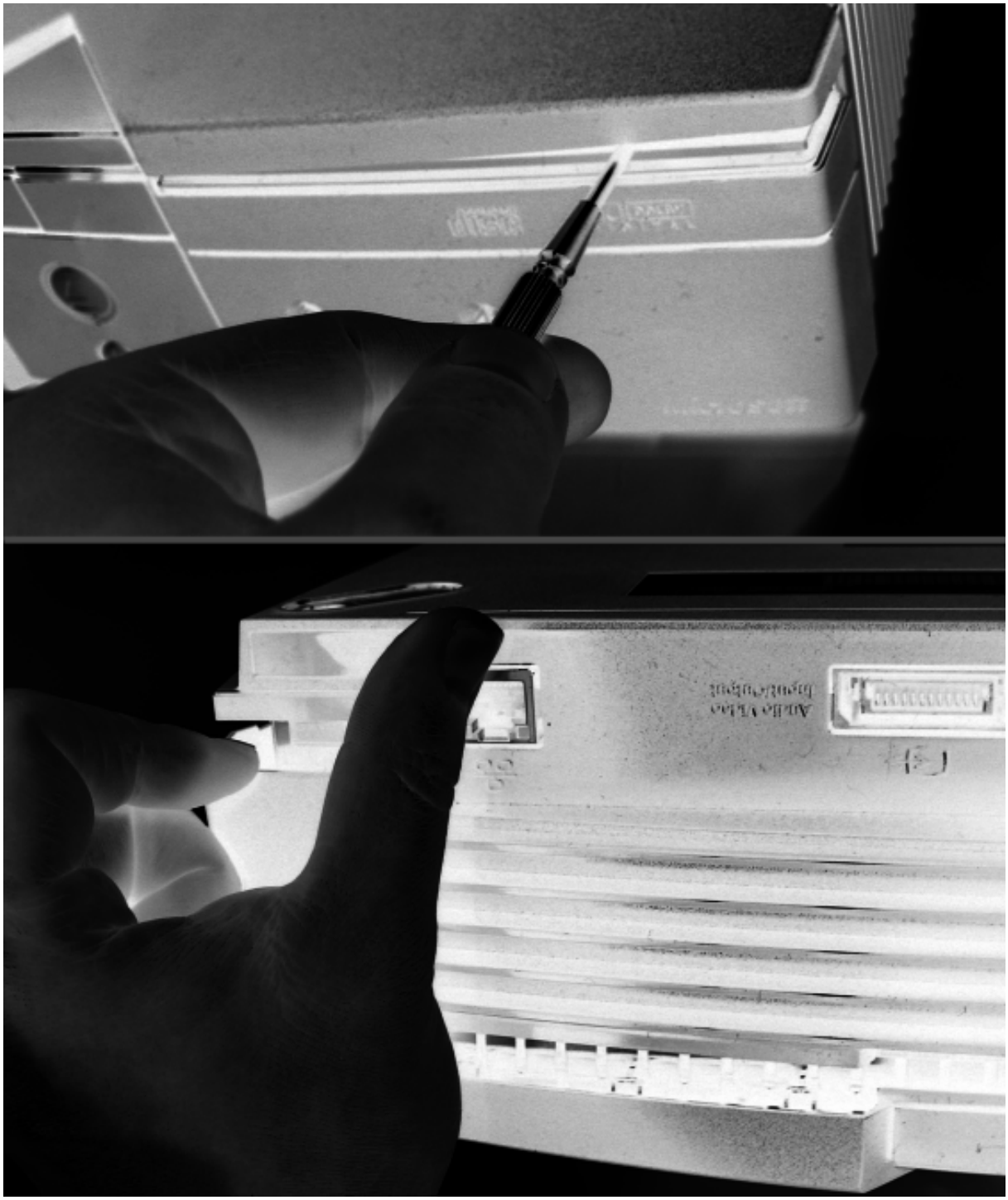


Рисунок 1-5. Некоторые места, где можно поддевать корпус, чтобы открыть упрямую крышку.

Не срывайте крышку корпуса силой. Если описанные выше способы поддевания не дают никакого прогресса, возможно, после публикации этой книги был добавлен дополнительный винт. Попробуйте найти винт, ощупывая наклейки на задней стороне Xbox.

Шаг 4. Переместите дисковые приводы

Теперь, когда вы внутри, вы должны увидеть два привода, установленные на чёрных пластиковых салазках. Чтобы получить доступ к материнской плате, вам нужно переместить (не обязательно снять) дисковые приводы. Отключать кабели приводов не нужно, но нужно выкрутить винты, удерживающие салазки приводов.

Салазки удерживаются тремя винтами Torx T10. На рисунке 1-6 показано расположение этих винтов. Один спрятан под серым ленточным IDE-кабелем около задней части корпуса, а два утоплены примерно на дюйм ниже поверхности приводов около передней части корпуса. Чтобы увидеть утопленные винты, вам может понадобиться фонарик или прямое верхнее освещение.

Утопленные винты могут оказаться немного сложными, если у вашей отвёртки нет магнитного держателя бит: бита будет стремиться соскользнуть, пока вы позиционируете её над винтом. Бита также достаточно мала, чтобы попасть в пространство между винтом и пластиковыми салазками. Если салазки приводов остаются неподвижными, хотя вам кажется, что винты выкручены, перепроверьте, действительно ли они выкручены. Вы должны быть способны слегка приподнять салазки без чрезмерного усилия.

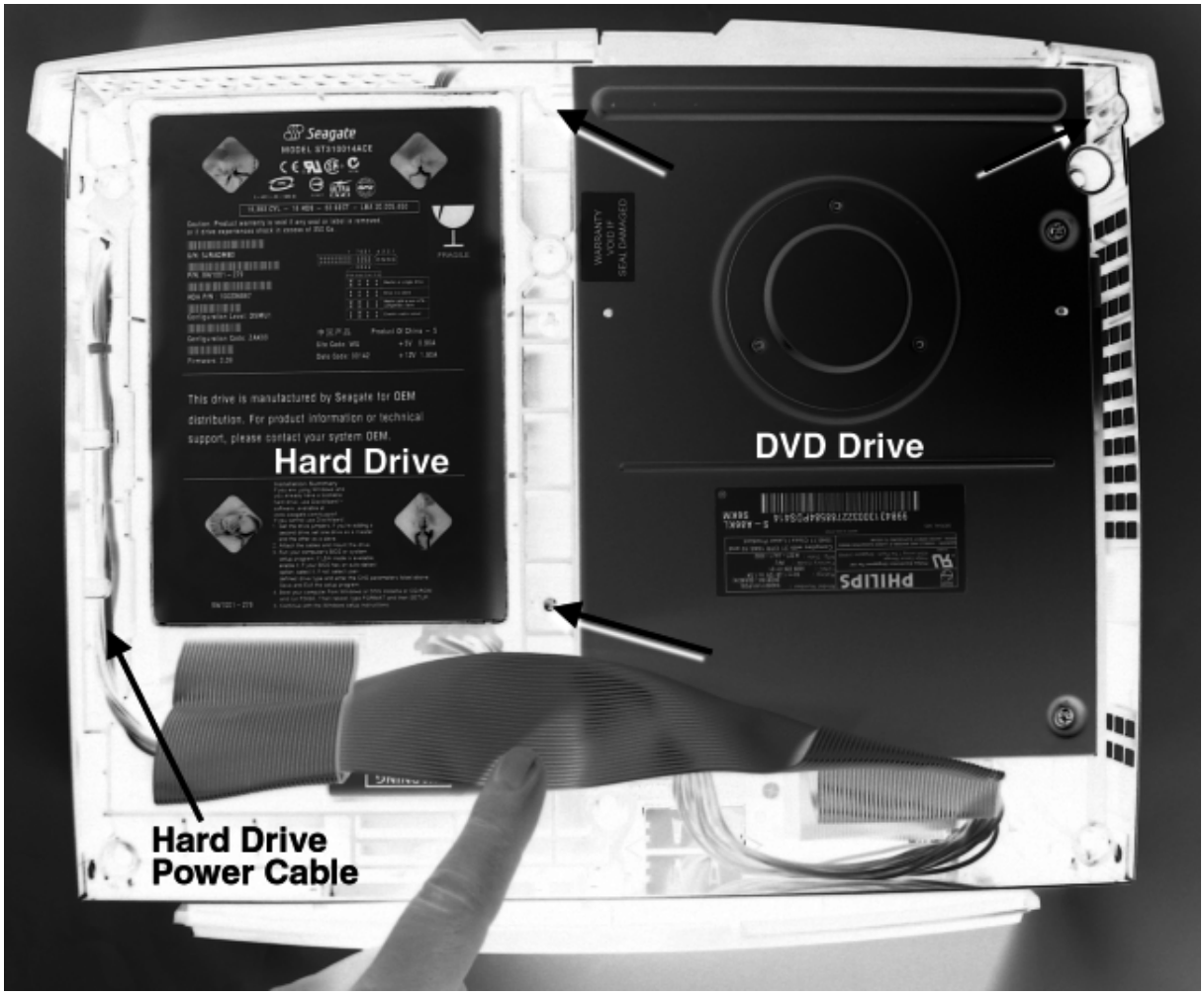


Рисунок 1-6. Расположение трёх винтов салазок приводов. Обратите внимание, что серый ленточный IDE-кабель приподнят для фотографии. Коробка слева - жёсткий диск, коробка справа - DVD-привод.

После снятия винтов нужно освободить кабель питания жёсткого диска, иначе он будет мешать снятию салазок приводов. Кабель питания - это чёрно-жёлто-красный жгут проводов с внешней стороны жёсткого диска. Жёсткий диск - устройство слева на рисунке 1-6. Кабель удерживается в выемке вдоль края салазок. Осторожно освободите кабель из выемки так, чтобы на нём появилось пару дюймов свободного хода. На рисунке 1-7 показано примерно, сколько свободного хода должно быть после завершения.

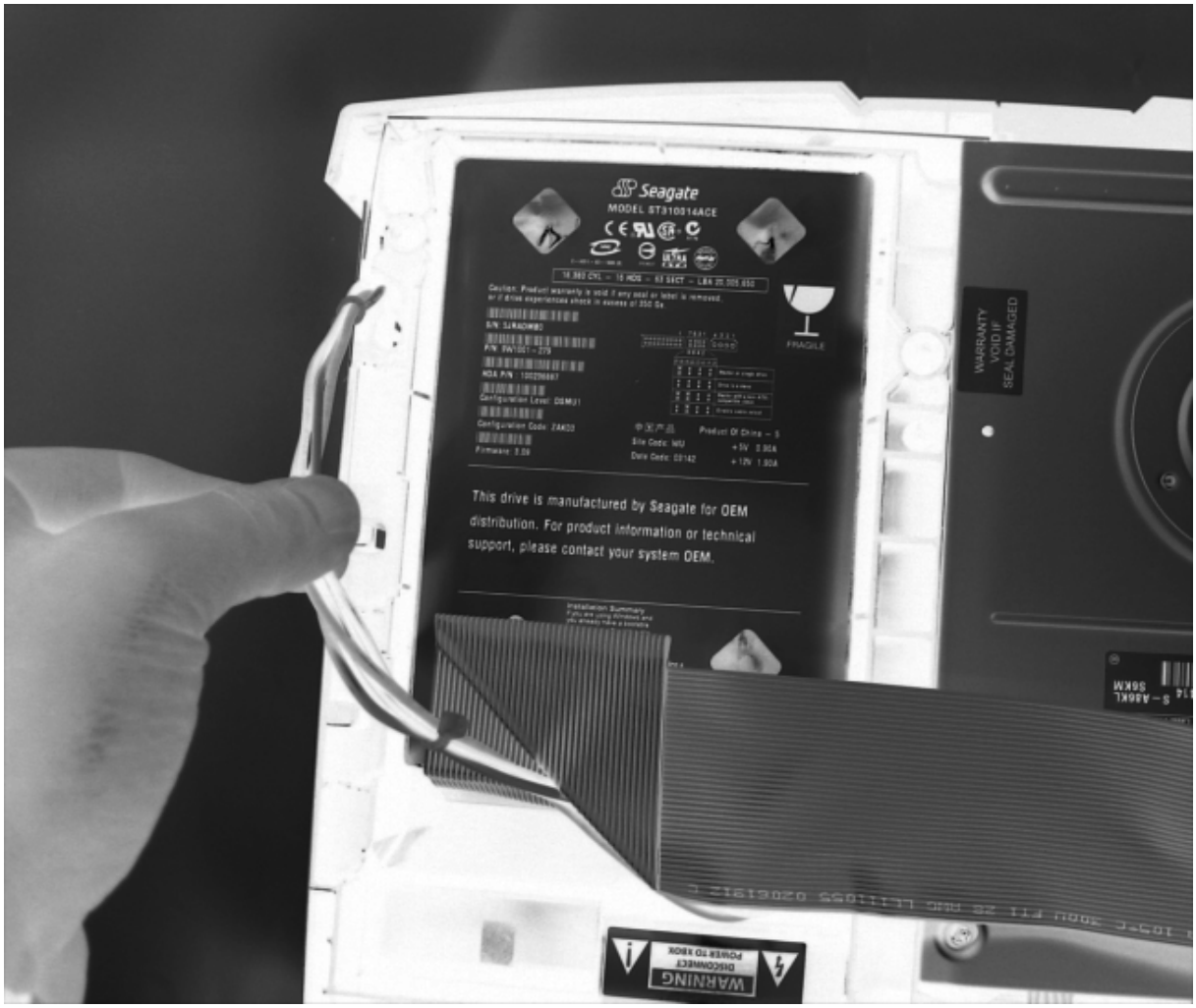


Рисунок 1-7. Освободите кабель жёсткого диска из удерживающей выемки. Когда он будет свободен, у вас должно получиться пару дюймов свободного хода кабеля.

Когда кабель освобождён, вы сможете поднять жёсткий диск из его посадочного места. Поднимите жёсткий диск и положите его поверх DVD-привода. Обеими руками поднимите DVD-привод и жёсткий диск из корпуса и откиньте их наружу так, чтобы они свисали сбоку, как показано на рисунке 1-8. Кабели, соединяющие приводы, должны легко перегибаться без сопротивления. (Следите за жёлтым кабелем, выходящим из задней части DVD-привода: если вы не будете осторожны, его легко выдернуть из разъёма.)

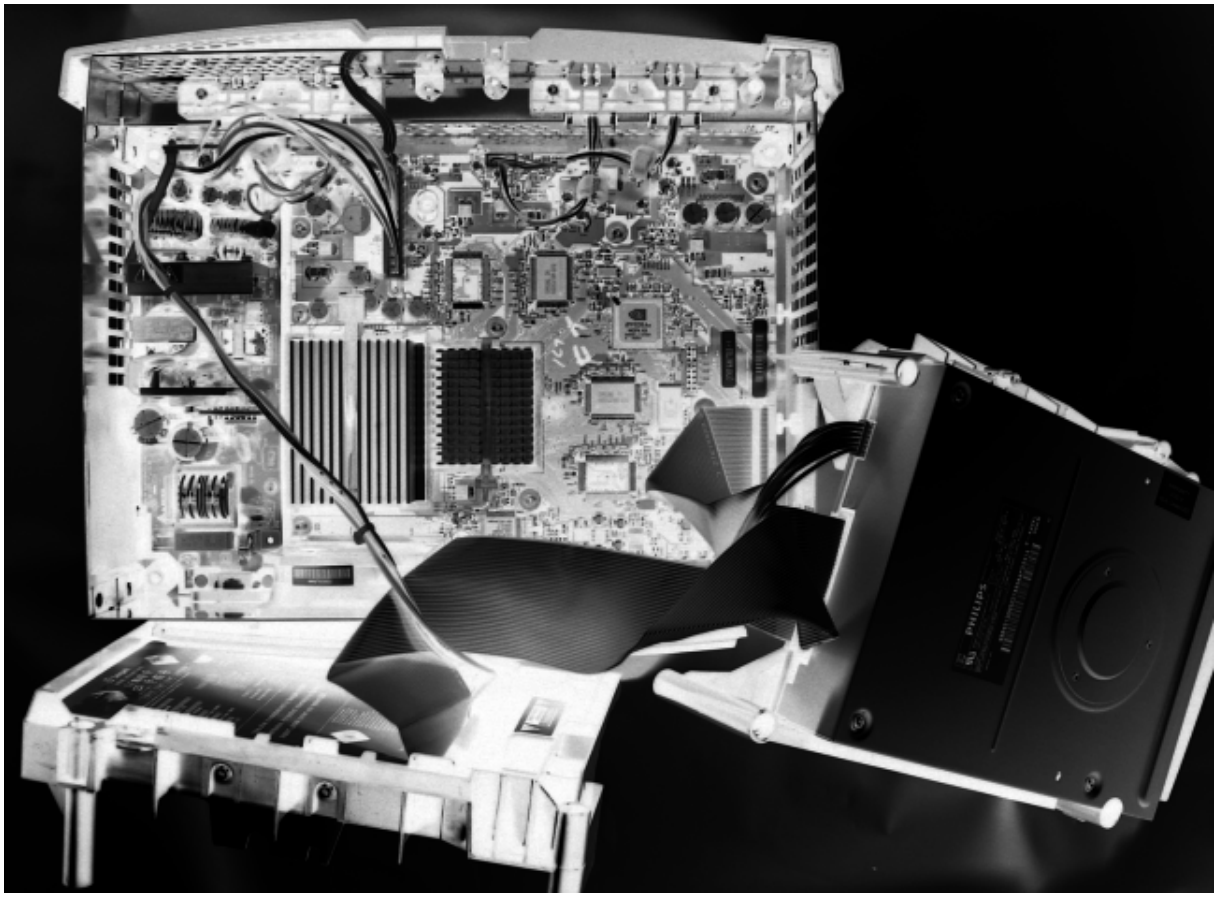


Рисунок 1-8. Положение дисковых приводов с сохранением электрических соединений для тестирования и экспериментов.

Теперь перед вами полный вид материнской платы Xbox и блока питания, при этом ни один из приводов не отключён. Такая компоновка будет полезна при тестировании Xbox после любых аппаратных модификаций.

Всегда помните о блоке питания: на нём есть напряжения, которые могут травмировать или убить вас при прикосновении. Помните, что он находится «под напряжением» всё время, пока Xbox подключена к сети, даже если сама Xbox выключена. Также обратите внимание, что нижняя часть салазок жёсткого диска и DVD-привод имеют вторичное назначение: они образуют воздушные каналы внутри Xbox. Длительная работа Xbox с приводами, откинутыми в стороны, может привести к перегреву Xbox. Точно так же следите за большими алюминиевыми радиаторами на CPU и GPU. Во время работы Xbox они могут становиться неприятно горячими - потенциально достаточно горячими, чтобы обжечь вас.

Для проекта в следующей главе (замена LED на передней панели Xbox) вам не нужно отключать жёсткие диски, хотя сделать это предпочтительно. Это не даст вам создавать чрезмерные нагрузки на кабели при манипуляциях с корпусом Xbox.

Шаг 5. Снимите дисковые приводы (необязательно)

Во многих случаях вы обнаружите, что полностью снять дисковые приводы удобнее и безопаснее для вашего оборудования.

Чтобы сделать это, сначала отключите серый ленточный IDE-кабель от материнской платы Xbox. Затем отключите от материнской платы жёлтый кабель из отдельных проводов, подключённый к DVD-приводу. Этот жёлтый кабель подаёт питание на DVD и передаёт материнской плате

информацию о состоянии лотка DVD-привода. Не тяните за отдельный провод в жёлтом кабеле из отдельных проводов, иначе вы можете выдернуть провод из головки кабеля. Предпочтительный способ снять кабель - взяться за белый разъём и потянуть; однако если ваши пальцы недостаточно малы, чтобы поместиться в тесном пространстве, можно взяться за весь жгут проводов и осторожно потянуть, чтобы снять разъём. Отложите DVD-привод в сторону.

Затем отключите разъём питания жёсткого диска. Вы обнаружите, что разъём очень плотно сидит в жёстком диске. Если просто тянуть разъём напрямую, вы рискуете пораниться об острые края корпуса, когда разъём освободится от диска. Чтобы избежать травмы, используйте маленькую плоскую отвёртку и аккуратно подденьте разъём от корпуса жёсткого диска, как показано на рисунке 1-9.

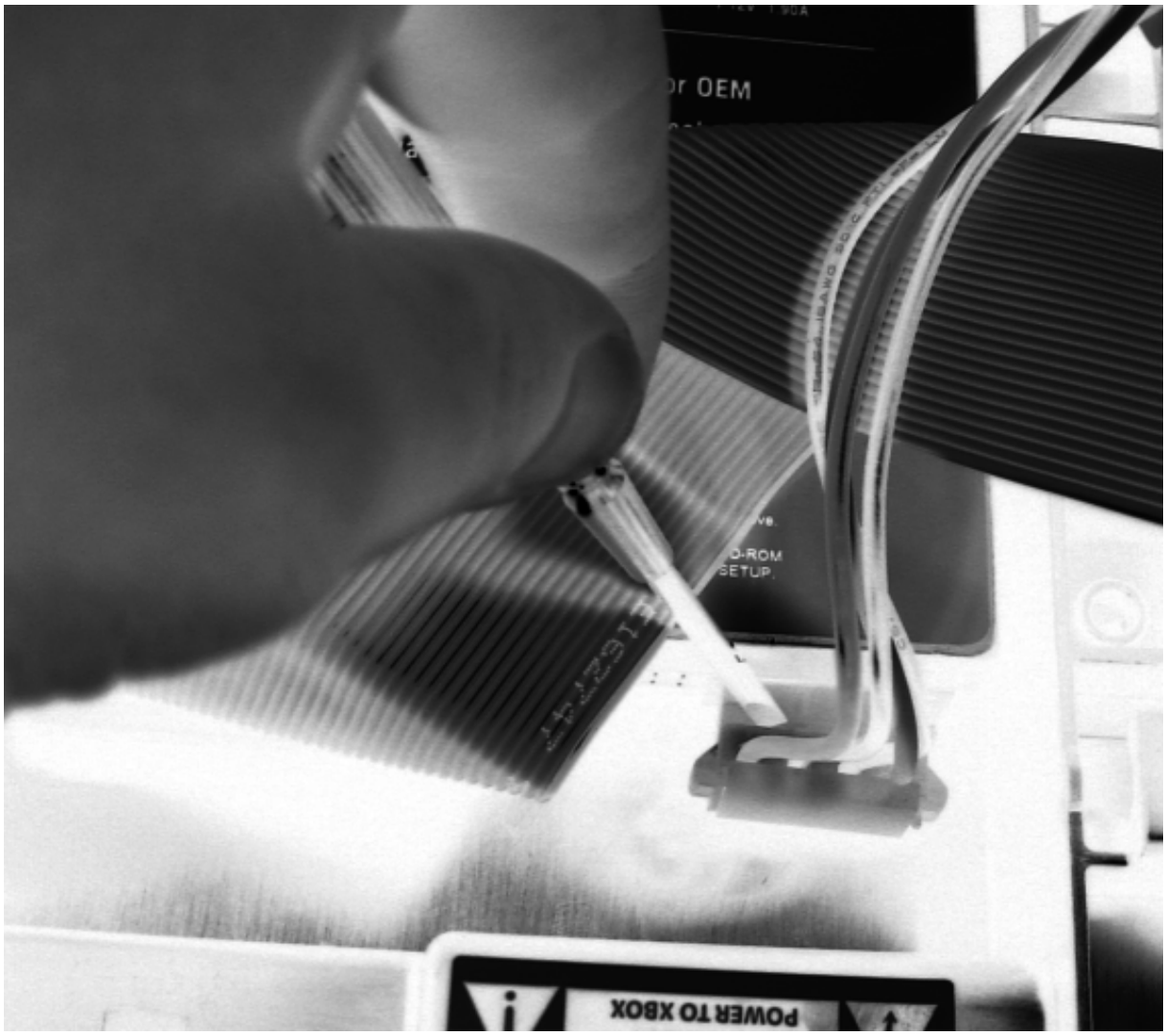


Рисунок 1-9. Поддевание разъёма питания жёсткого диска плоской отвёрткой.

Теперь можно полностью снять сборку дисковых приводов. Серый ленточный IDE-кабель всё ещё будет соединять отдельные блоки приводов; при желании вы можете снять и его, но запомните его ориентацию, чтобы позже подключить приводы к Xbox обратно.

Обратная сборка Xbox

Теперь, когда вы разобрали Xbox, переходите к следующим разделам, где есть несколько интересных проектов, которые можно попробовать. Когда закончите, прочитайте этот раздел: здесь приведены замечания о том, как собрать Xbox обратно.

Перед тем как что-либо прикреплять к Xbox, переверните её вверх дном и осторожно встряхните, чтобы убедиться, что внутри нет свободных винтов или деталей, которые вы могли случайно уронить в Xbox. Свободный винт станет концом вашей игровой консоли и представляет потенциальную пожарную опасность, поэтому эту проверку стоит выполнить, если у вас есть хоть малейшие сомнения.

Первый шаг при обратной сборке Xbox - снова подключить дисковые приводы. Если вы следовали процедуре из предыдущего раздела, DVD-привод и жёсткий диск уже должны быть соединены серым ленточным IDE-кабелем. Если это не так, подключите конец кабеля с наименьшим количеством сгибов к жёсткому диску, а затем подключите средний разъём кабеля к DVD-приводу. Кабели подключаются только в одном направлении; обратите внимание на выступ на разъёме и положение выемки на разъёмах приводов.

Подключите оставшийся свободный конец серого ленточного IDE-кабеля к материнской плате Xbox. Кабель подключается к материнской плате только в одном направлении; обратите внимание на выступ посередине IDE-разъёма и выемку на гнезде материнской платы. Теперь подключите жёлтый DVD-кабель к материнской плате Xbox. Этот кабель также вставляется в разъём материнской платы только в одной ориентации, и у него тоже есть выступы на кабеле и соответствующие выемки на гнезде материнской платы. Теперь установите DVD-привод в Xbox. Проверьте, что он стоит заподлицо и ровно, наблюдая, как лоток DVD-привода с логотипом Xbox расположен относительно края корпуса. Скорее всего, вам придётся попробовать установить привод пару раз, прежде чем он сядет правильно. Если вы запутались в ориентации разных частей, используйте сгибы на сером ленточном IDE-кабеле как подсказку.

Вы почти закончили. Опустите жёсткий диск на место рядом с DVD-приводом. Этот привод, вероятно, тоже придётся немного покачать, чтобы он встал на место. Привод должен быть на одном уровне с DVD-приводом и заподлицо с краями металлического EMI-экрана вокруг корпуса Xbox. Проденьте кабель питания жёсткого диска через его удерживающую выемку в лотке привода и подключите его к жёсткому диску. Чтобы создать надёжное соединение, вам потребуется приложить к разъёму довольно заметное усилие; когда разъём полностью войдёт, вы должны почувствовать лёгкий щелчок. Наконец, проденьте серый ленточный IDE-кабель через его первоначальный удерживающий крючок на салазках приводов.

На этом этапе полезно подключить Xbox к сети, соединить её с телевизором и проверить, что Xbox запускается правильно. Так можно бесконечно долго запускать Xbox со снятой крышкой, поскольку все охлаждающие каналы, образованные нижней частью салазок приводов и DVD-приводом, находятся на месте. Если приводы подключены к Xbox неправильно, консоль всё равно загрузится, но выведет сообщение о том, что ваша консоль требует обслуживания. Если появится это сообщение, внимательно проверьте соединения.

Теперь пора закрепить приводы винтами. К этому моменту у вас должно быть девять винтов: три коротких винта T10 для салазок приводов и шесть длинных винтов T20 для крышки корпуса. Не паникуйте, если у вас не хватает пары винтов или если пара оказалась лишней. Xbox всё равно будет держаться, даже если вам недостаёт одного-двух винтов. Установите винты и крышку корпуса в порядке, обратном их снятию. (Если нужно напомнить, куда они идут, см. фотографии ранее в этом разделе.)

После установки крышки корпуса Xbox снова готова к использованию!

Глава 2. Мышление внутри коробки

Обратную разработку можно представить как очень сложную, но очень вознаграждающую игру. Чтобы победить, вам понадобится немного мастерства и немного удачи. И, как в любой игре, чтобы развить навыки, нужно просто играть, играть и играть.

Первый шаг в развитии ваших навыков хакера - выработать интуицию относительно материала. В случае аппаратуры хороший способ почувствовать вещи - снимать крышки со всего подряд и пытаться понять, что представляют собой все компоненты и что они могут делать. Также полезно заказать бумажный каталог у поставщика деталей, такого как Digi-Key, Jameco или Newark Electronics, и просто листать страницы в свободное время. Сначала чтение каталога деталей может ощущаться как чтение словаря, но по мере того как вы будете смотреть на всё большее количество печатных плат, вы постепенно обнаружите, что всё начинает обретать смысл.

Следующий по силе инструмент специалиста по обратной разработке - сопоставление с шаблонами. Все инженеры аппаратуры ограничены одними и теми же законами природы, и все инженеры аппаратуры используют одни и те же виды строительных блоков. Инженеры также любят модульность и повторное использование существующих конструкций. В результате один мотив конструкции можно найти во многих разработках. Распознавание проектных мотивов позволит вам определять функцию схем даже в том случае, если вы не узнаёте ни одного номера детали. Точно так же в обратной разработке можно продвинуться довольно далеко без формального электротехнического образования.

Последний инструмент специалиста по обратной разработке - экспериментирование. Когда интуиция и сопоставление с шаблонами не раскрывают секреты схемы, приходится прибегать к прозвону и возмущению системы и пытаться вывести функцию на основании наблюдаемых реакций. Хотя экспериментирование может привести к отказу аппаратуры, можно утешиться тем, что большая часть потребительской аппаратуры рассчитана на прозвон и тестирование как обязательную часть производства. Кроме того, в случае Xbox можно немного успокоиться тем, что новая Xbox стоит относительно недорого. Покупка двух коробок заранее и отношение к одной из них как к «жертвенной» помогает снять психологический барьер, который иначе мог бы возникнуть перед агрессивными экспериментами с аппаратурой.

Эта глава познакомит вас с основами обратной разработки, уделяя основное внимание базовым приёмам, таким как чтение печатных плат для развития интуиции, а также частично охватит промежуточные приёмы, например сопоставление с шаблонами и распознавание базовых проектных мотивов.

Чтение печатной платы

Первое, что вы видите, когда снимаете крышку с типичного электронного устройства, - это печатная плата. Обычно окрашенный в зелёный или желтовато-коричневый цвет, этот многослойный сэндвич из меди, стекловолокна и эпоксидной смолы содержит в своих дорожках точный список соединений схемы. Иными словами, следуя по дорожкам, можно точно определить, как подключён каждый компонент. Размещение компонентов и разводка дорожек также содержат подсказки, способные дать понимание хода мысли разработчика.

Основы печатных плат

Типичная печатная плата состоит из нескольких слоёв меди с рисунком, разделённых тонкими листами стеклоткани, пропитанной эпоксидной смолой. Цвет необработанной печатной платы беловатый или желтовато-коричневый, с медными дорожками; однако почти все печатные платы покрываются тонким полимером, называемым паяльной маской, который придаёт платам привычный зелёный цвет. Расплавленный припой не пристаёт к паяльной маске, поэтому во время производства лишний припой не прилипает к плате и не вызывает коротких замыканий. В паяльной маске есть отверстия для соединений с компонентами. Эти отверстия обычно имеют серебристый цвет из-за тонкого покрытия оловом или припоем, нанесённого для предотвращения окисления меди и улучшения паяемости.

Поверх паяльной маски обычно находится слой белых надписей, называемый шелкографией. Каждый компонент на печатной плате имеет контур и уникальное позиционное обозначение на слое шелкографии. Обозначение позволяет людям быстро связать компонент на печатной плате с компонентом на схеме. Вы можете использовать обозначение, чтобы по схеме именования компонентов предположить функцию компонента. В таблице 2-1 кратко приведена схема именования компонентов, используемая в Xbox.

> Совет. Материнская плата Xbox содержит удобную координатную систему, напечатанную по краям платы на слое шелкографии. На стороне компонентов координаты идут от A до G по бокам и от 1 до 8 вдоль верхнего и нижнего краёв. Обратная сторона платы имеет координаты от M до V по бокам. Обратите внимание, что буквы I, O, Q и S пропущены, потому что их можно спутать с цифрами 0, 1 и 5. Позиционные обозначения компонентов на материнской плате Xbox закодированы с использованием этой координатной системы; таким образом, J7D1, отладочный порт LPC, можно найти на верхней стороне в координатах 7D. В этой книге эта координатная система вместе с позиционными обозначениями будет часто использоваться для ссылки на конкретные компоненты.

Таблица 2-1. Схема именования компонентов Xbox.

Обозначение	Тип компонента
C	Конденсатор
R	Резистор
U	Интегральная схема или транзистор
L	Индуктор
RP	Резисторная сборка
Q	Транзистор
CR	Диод
J	Разъём или перемычка
RT	Самовосстанавливающийся предохранитель
Y	Кварцевый резонатор

Соединения между слоями проводников выполняются заполненными медью отверстиями, называемыми переходными отверстиями, или vias. Поскольку стоимость печатной платы растёт с количеством слоёв, большинство потребительских электронных устройств проектируются так, чтобы количество слоёв было минимальным. Радиоприёмники и аудиоусилители обычно используют односторонние платы, тогда как новейшие материнские платы PC могут иметь до шести или восьми слоёв. Материнская плата Xbox имеет четыре слоя. Два верхних слоя предназначены для передачи информации между микросхемами, а два внутренних слоя

предназначены для подачи питания. На первый взгляд материнская плата Xbox будет казаться непрозрачной, потому что внутренние силовые слои по сути представляют собой сплошные листы меди. Хорошая новость для обратной разработки состоит в том, что мы можем проследить каждое соединение на материнской плате Xbox обычным визуальным осмотром, поскольку все сигнальные слои находятся снаружи непрозрачных силовых слоёв. Конструкция Xbox отличается от материнских плат, которые прячут два или четыре сигнальных слоя внутри силовых слоёв. Скрытые сигнальные слои могут затруднять трассировку сигналов. (Обратите внимание, что решение спрятать сигналы внутри силовых слоёв обычно продиктовано не безопасностью, а физикой взаимодействия электрических сигналов между слоями печатной платы.)

Довольно легко проследить сигнал. Начиная от соединения исходного компонента с платой, следуйте по медной дорожке. Если дорожка пересекается с кругом, есть хороший шанс, что сигнал продолжается на противоположной стороне платы. Если дорожка заканчивается и соединения с другой стороной платы нет, есть хороший шанс, что дорожка подключена к одной из силовых плоскостей.

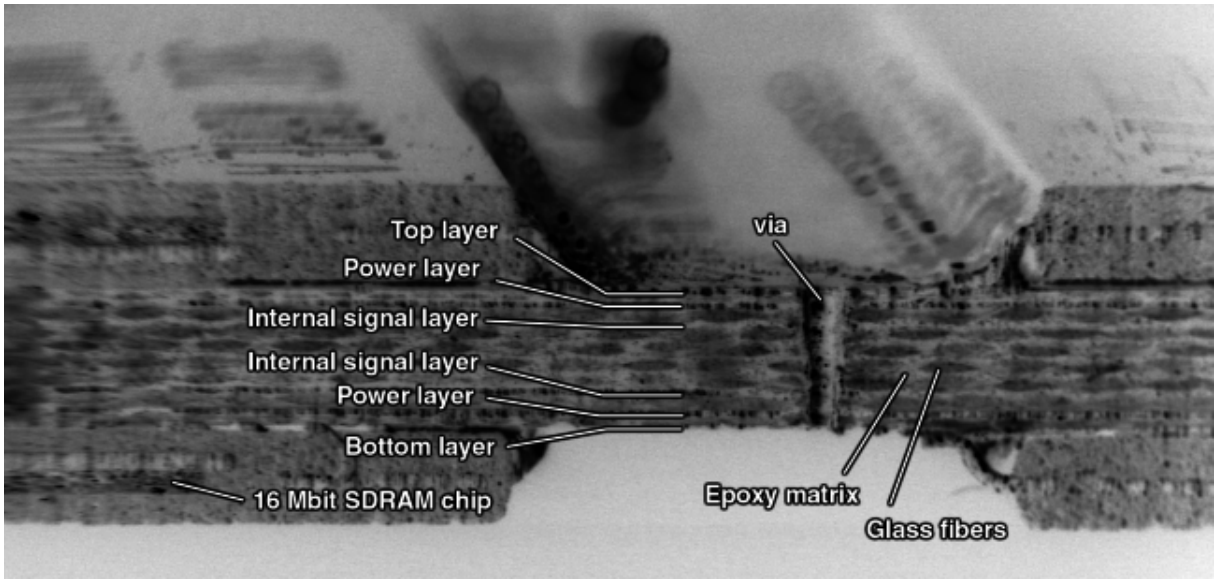


Рисунок 2-1. Поперечное сечение типичной печатной платы.

> Попробуйте. Попробуйте проследить несколько сигналов на материнской плате Xbox. На материнской плате Xbox взгляните на разъём J8C1, 40-контактный IDE-разъём в секторе 8C. Почти все сигналы от IDE-разъёма идут к одной микросхеме, MCPX, на материнской плате через несколько резисторных сборок. Какой вывод вы могли бы сделать об этой микросхеме? Обратите внимание, что некоторые дорожки, выходящие из IDE-разъёма, петляют туда-сюда. Это приём, используемый для попытки гарантировать, что все проводники имеют одинаковую длину. Дополнительное объяснение см. во врезке «Почему дорожки печатной платы петляют повсюду?».

Компоненты

Теперь, когда у вас есть небольшой опыт трассировки сигнала назад, пора узнать, как выглядят некоторые базовые компоненты.

> **Почему дорожки печатной платы петляют повсюду?.** После изучения нескольких печатных плат вы, вероятно, начнёте замечать, что дорожки на печатной плате часто

> петляют повсюду, иногда несколько раз уходя туда и обратно перед подключением к месту назначения. Это кажется бессмысленным, когда прямая дорожка справилась бы с задачей. Однако на печатной плате редко встретится структура, размещённая по легкомысленной прихоти разработчика. Оказывается, скорость сигналов в большинстве высококлассных электронных устройств, около 1/4 скорости света, мала по сравнению со временем, требуемым сигналу, чтобы прибыть в пункт назначения. Например, за один такт процессора 1 ГГц сигнал пройдёт по печатной плате всего 3 дюйма (один такт на 1 ГГц длится одну миллиардную секунды, или одну наносекунду). Поэтому два сигнала, выходящие из одной микросхемы, прибудут к месту назначения в довольно разное время, если длины дорожек сильно отличаются. Чтобы бороться с этим, разработчики добавляют дополнительные изгибы в более короткую дорожку, чтобы эффективная длина дорожки стала такой же, как у более длинной.

Компоненты классифицируются как пассивные и активные. Грубо говоря, пассивные компоненты не могут усиливать сигнал, поэтому обычно имеют только два вывода. Иногда несколько пассивных компонентов упакованы вместе, поэтому один корпус пассивных компонентов будет иметь несколько выводов. К пассивным компонентам относятся конденсаторы, резисторы и индукторы. Самые распространённые пассивные компоненты на материнской плате Xbox - конденсаторы. Конденсаторы хранят энергию в виде электрического заряда; в Xbox они в основном используются для сглаживания локальных колебаний питания, возникающих при переключении цифровой CMOS-логики, и для подавления высокочастотного шума.

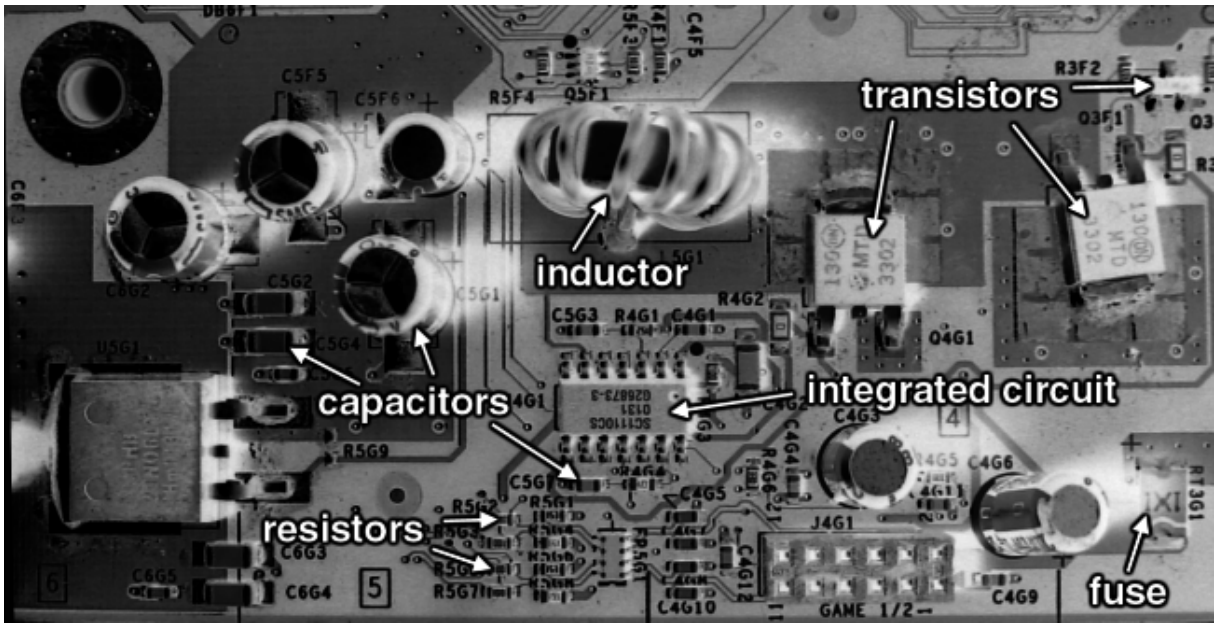


Рисунок 2-2. Типичные пассивные компоненты в Xbox.

Другие крупные пассивные компоненты, встречающиеся на материнской плате Xbox, включают индукторы и резисторы. Крупные проволочные тороидальные (кольцевые) индукторы, встречающиеся на материнской плате Xbox, все являются частью подсистемы питания. Индукторы хранят энергию в виде магнитного потока. Электрические свойства индуктора дополняют свойства конденсатора. Комбинации индукторов и конденсаторов с транзисторными ключами между ними используются для построения очень эффективных регуляторов питания. Большинство резисторов на материнской плате Xbox используется либо для поглощения избыточной энергии на оконечных участках сигнальных дорожек, либо для смещения провода к определённому логическому уровню.

Есть два способа опознать пассивное устройство на материнской плате Xbox. Первый - по форме корпуса. Распознавание формы корпуса осуществимо, потому что базовых разновидностей пассивных деталей очень мало. На рисунке 2-2 есть фотографии конденсаторов, индукторов и

резисторов, которые можно увидеть на материнской плате Xbox. Второй метод - прочитать надпись рядом с деталью на материнской плате и вывести функцию детали по позиционному обозначению, используя таблицу 2-1 как руководство.

> **Что делают все эти резисторы и конденсаторы на цифровой печатной плате?.** Мотив, который вы заметите на многих печатных платах, - преобладание резисторов и конденсаторов. Конденсаторы повсюду, потому что они помогают свести шум к минимуму и стабилизировать напряжения питания. Они требуются потому, что медные плоскости, используемые для распределения питания, имеют небольшое сопротивление и индуктивность. Эти небольшие паразитные параметры могут вызвать большие проблемы, когда через цепь питания переключается большой ток. Точное размещение и выбор конденсаторов считается в некоторой степени чёрной магией. Если во время работы с печатной платой вы случайно сшибёте один из крошечных конденсаторов размером с песчинку, есть шанс, что вам удастся обойтись без его замены. Однако при таком дефекте наиболее вероятная проблема, с которой вы столкнётесь, - периодические проблемы надёжности.
>
> Пока конденсаторы повсюду обеспечивают локальное хранение энергии для всех компонентов, резисторы удаляют избыточную энергию. Быстрые сигналы на материнской плате несут много энергии, и если эта энергия не рассеивается у приёмника контролируемым образом с помощью резистора, энергия сигнала отразится обратно к передатчику и вызовет проблемы. Это явление похоже на звук в спортивном зале. Когда вы говорите в пустом спортивном зале, возникает эхо. Если говорить слишком быстро, люди не смогут вас понять, потому что эхо начнёт мешать вашей речи. Однако если покрыть стены спортивного зала пеной, эхо будет поглощаться пеной, и вы сможете говорить без помех от собственного эха.
>
> Резисторы похожи на акустическую пену, которую вы поместили бы на стены, чтобы заглушить эхо и позволить схемам разговаривать друг с другом на высоких скоростях. В отличие от большинства конденсаторов, если во время экспериментов вы случайно сшибёте один из таких резисторов, его придётся заменить, чтобы схема работала правильно. Такие «терминирующие резисторы» часто упакованы по четыре или восемь штук в корпус, поэтому почти похожи на небольшие интегральные схемы. Резисторные сборки можно отличить от других компонентов по тому, что они блестящие, слегка бугристые, имеют белую рамку, а рядом с ними будет позиционное обозначение с префиксом «RP». При трассировке сигнала через резисторную сборку довольно безопасно предполагать, что сигналы проходят прямо насквозь, так что соединение на одной стороне идёт прямо к выводу непосредственно на другой стороне.

Активные компоненты могут усиливать сигналы и имеют три или более выводов. Самый простой активный компонент - транзистор, с тремя, а иногда и четырьмя выводами (иногда дискретные транзисторы «MOSFET» имеют явный четвёртый вывод «body»). Самые сложные активные компоненты - интегральные схемы, такие как CPU и микросхемы памяти, с сотнями, иногда тысячами выводов. Интегральные схемы выпускаются в широком разнообразии корпусов, и иногда соединения скрыты под корпусом, как в случае корпуса Ball Grid Array (BGA). Графический чип, MCPX и CPU на материнской плате Xbox используют корпуса BGA. Рисунок 2-3 показывает поперечное сечение BGA-устройства, раскрывая скрытые соединения под ним.

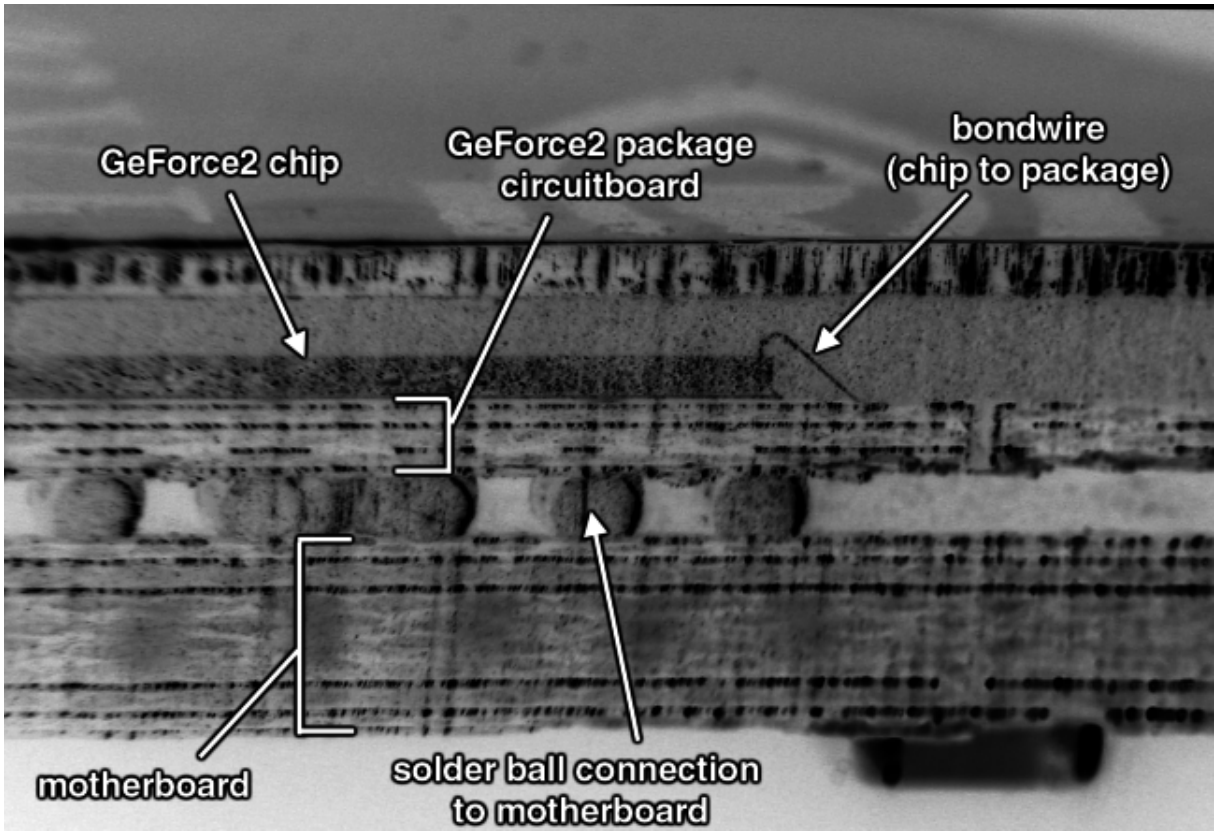


Рисунок 2-3. Вид в поперечном сечении детали в корпусе BGA (GeForce2), установленной на материнской плате.

Определить функцию конкретной интегральной схемы сложнее, чем определить функцию пассивного устройства. Функционально одинаковый кремний можно купить во множестве корпусов, которые могут выглядеть очень по-разному. В некоторых случаях можно угадать функцию устройства, наблюдая, к чему оно подключено или как оно выглядит, но самый надёжный метод - прочитать номер детали на микросхеме и найти его в вебе. (Обычно у деталей есть какой-то логотип или префикс номера детали, идентифицирующий производителя; его можно использовать, чтобы найти больше данных об устройстве на сайте производителя.) Если вы не узнаёте логотип или префикс номера детали, перечисленные ниже сервисы помогут вам найти функции деталей.

1. www.findchips.com может принять номер детали или фрагменты номера детали и поискать совпадения в запасах многих дистрибьюторов. Большинство распространённых деталей появится в FindChips, а предоставленные ссылки часто приведут вас не только к краткому описанию детали, но и к информации о цене и заказе.

2. www.google.com индексирует всё в вебе, и номера деталей не являются исключением. Google также можно использовать, чтобы помочь найти сайты производителей, если искать буквы в логотипе плюс описательный термин вроде «semiconductors». На сайте производителя вам, вероятно, понадобится найти специализированный поиск деталей, спрятанный внутри сайта, или перейти на подстраницу semiconductor products, чтобы выполнить поиск по номеру детали. Функция поиска на главной странице сайта компании иногда находит номера деталей, но чаще индексирует только бесполезные корпоративные и маркетинговые страницы.

3. Если ни один из этих сервисов не приводит вас к цели, попробуйте отбросить некоторые префиксы и суффиксы в номере детали. В нашем примере M29F080A запрос только по номеру детали 29F080 приведёт вас на веб-страницы нескольких производителей, выпускающих детали, функционально совместимые с деталью STMicroelectronics.

```
M29F080A        Номер детали
70N1            Класс скорости
5881K           Код партии
0141            Код даты
SINGAPORE       Страна происхождения
ST              Логотип производителя (STMicroelectronics)
Pin 1 marker    Метка вывода 1
```

Рисунок 2-4. Анатомия типичного номера детали IC. Схема представляет собой условное изображение микросхемы в позиции U7D1 на материнской плате Xbox.

> Попробуйте. Давайте попробуем найти номер детали Xbox. Найдите U7D1 на материнской плате Xbox. Рисунок 2-4 показывает, что вы можете обнаружить. Номер детали обычно является самым длинным номером на микросхеме, и он часто начинается с одного или двух буквенных символов. Микросхемы памяти и процессоры также часто имеют после номера детали суффикс класса скорости или качества. Кроме того, почти все микросхемы имеют код даты. Коды даты обычно представляют собой четырёхзначное число формата YY-WW, где YY - год изготовления микросхемы, а WW - рабочая неделя. В нашем примере деталь M29F080A была изготовлена на 41-й неделе 2001 года в Сингапуре и имеет класс скорости 70N1. Оставшееся число, 5881K, - код партии, значение которого зависит от производителя, но в общем случае связывает микросхему с конкретной кремниевой пластиной или номером отслеживания партии кремниевых пластин на фабрике. Логотип «ST» указывает, что производитель этой микросхемы - STMicroelectronics, и, к счастью, сайт этого производителя можно быстро найти через Google или просто угадать, поскольку URL компании - www.st.com. Ввод номера детали M29F080A в поле поиска на главной странице приводит прямо к результатам поиска, включающим подробные datasheet и описания этой детали - 8 Mbit Uniform Block Single Supply FLASH ROM.

Контрольные точки

Почти все печатные платы в потребительской электронике имеют структуры, предназначенные для ускорения тестирования готовой платы на заводе. Эти «контрольные точки» существуют, чтобы справляться с неприятной реальностью производственных дефектов. Xbox не является исключением в отношении контрольных точек и производственных дефектов. Нижняя сторона материнской платы Xbox усеяна сотнями контрольных точек - крошечных серебристых кружков, - которые позволяют контактному щупу получить доступ почти к каждому интересному сигналу внутри Xbox. Эти контрольные точки - желанный подарок специалистам по обратной разработке и людям, желающим модифицировать свою аппаратуру, потому что они обеспечивают лёгкий доступ к сигналам, которые иначе могли бы потребовать микроскопа и твёрдой руки.

Набор контрольных точек прозванивается на производственной линии весь сразу с помощью оборудования, называемого «bed of nails tester». Это название удачно: тестер типа bed of nails состоит из сотен подпружиненных структур «pogo pin». Материнская плата совмещается с испытательным стендом и прижимается механическими плунжерами или вакуумным зажимом. Аналогично вы можете использовать pogo-pins для собственных модификаций материнской платы Xbox без пайки, используя контрольные точки. Вам потребуется изготовить собственные печатные платы (см. приложение), но результатом будет плата, которую можно установить, просто прикрутив её винтами, - пайка не требуется!

Архитектура Xbox

Прежде чем погружаться в примеры сопоставления с шаблонами, нам понадобится эталон шаблона. Воспользуемся этой возможностью, чтобы изучить внутреннюю архитектуру Xbox как эталон шаблона, а затем сравнить архитектуру Xbox с PC и другой игровой консолью.

Общая организация

Xbox имеет CPU класса Pentium-III, работающий на частоте 733 МГц. Номер «S-Spec» на CPU ближе всего к Mobile Celeron. CPU подключён через стандартную Front Side Bus (FSB) P6 133 МГц к графическому процессору (GPU) и комбинированной микросхеме northbridge под названием NV2A от nVidia. Её ближайший родственник в мире PC - микросхема nForce IGP от nVidia. Поскольку логика northbridge и GPU объединены в одной микросхеме, CPU и графический процессор могут совместно использовать общий банк памяти. Это называется «унифицированной архитектурой памяти» (UMA).

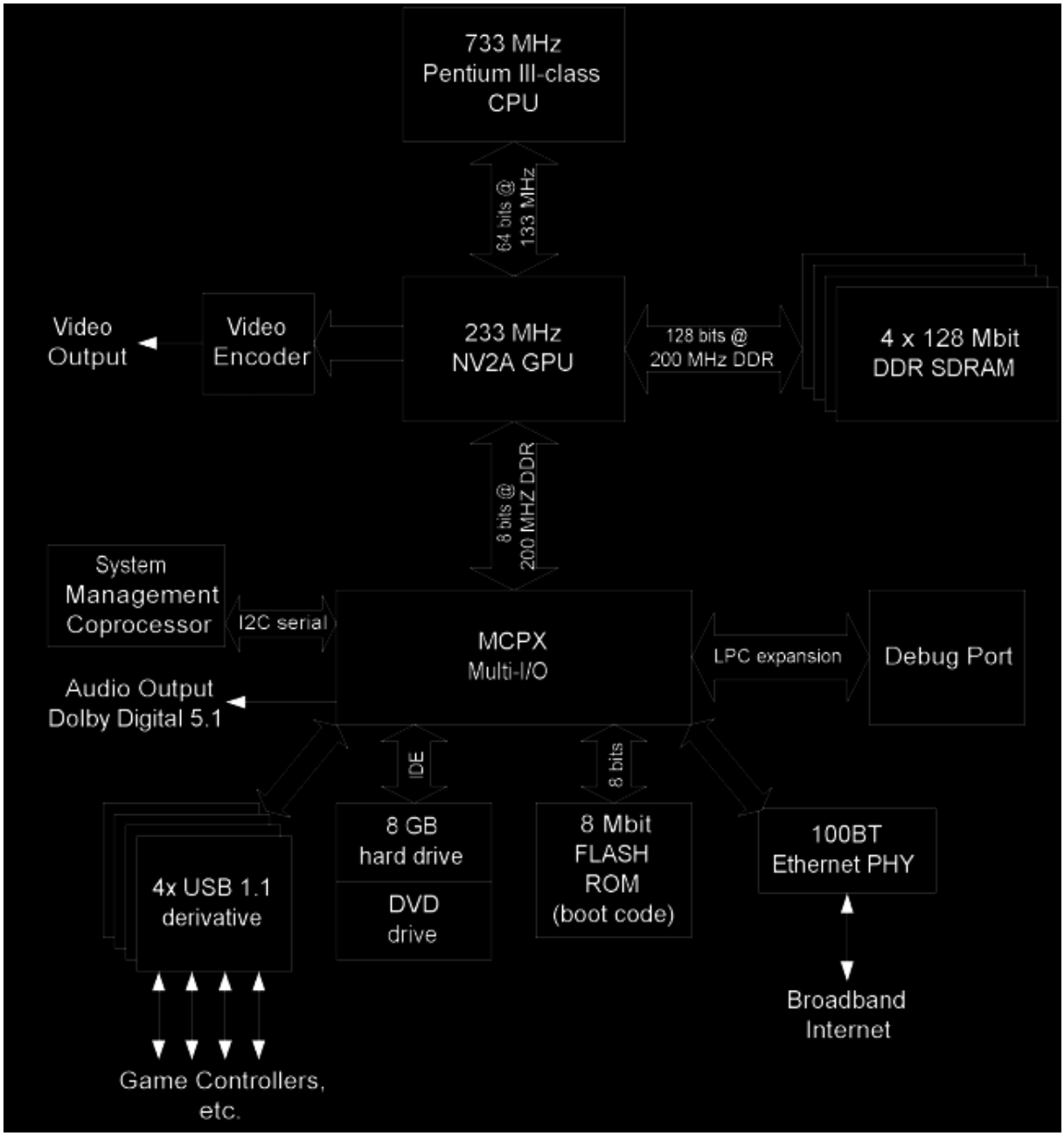


Рисунок 2-5. Архитектурный вид Xbox на верхнем уровне.

По сравнению с традиционной архитектурой с раздельной видеопамятью и основной памятью UMA дешевле в производстве, потому что устраняет выделенную видеопамять. Однако в некоторых ситуациях UMA имеет более низкую производительность, поскольку вводит конкуренцию за доступ к памяти между главным процессором и графическим процессором. Чтобы частично ослабить эту конкуренцию, системную память часто делят на несколько банков. Например, nForce IGP делит память на два банка, к которым и GPU, и CPU могут независимо обращаться через коммутирующую сеть.

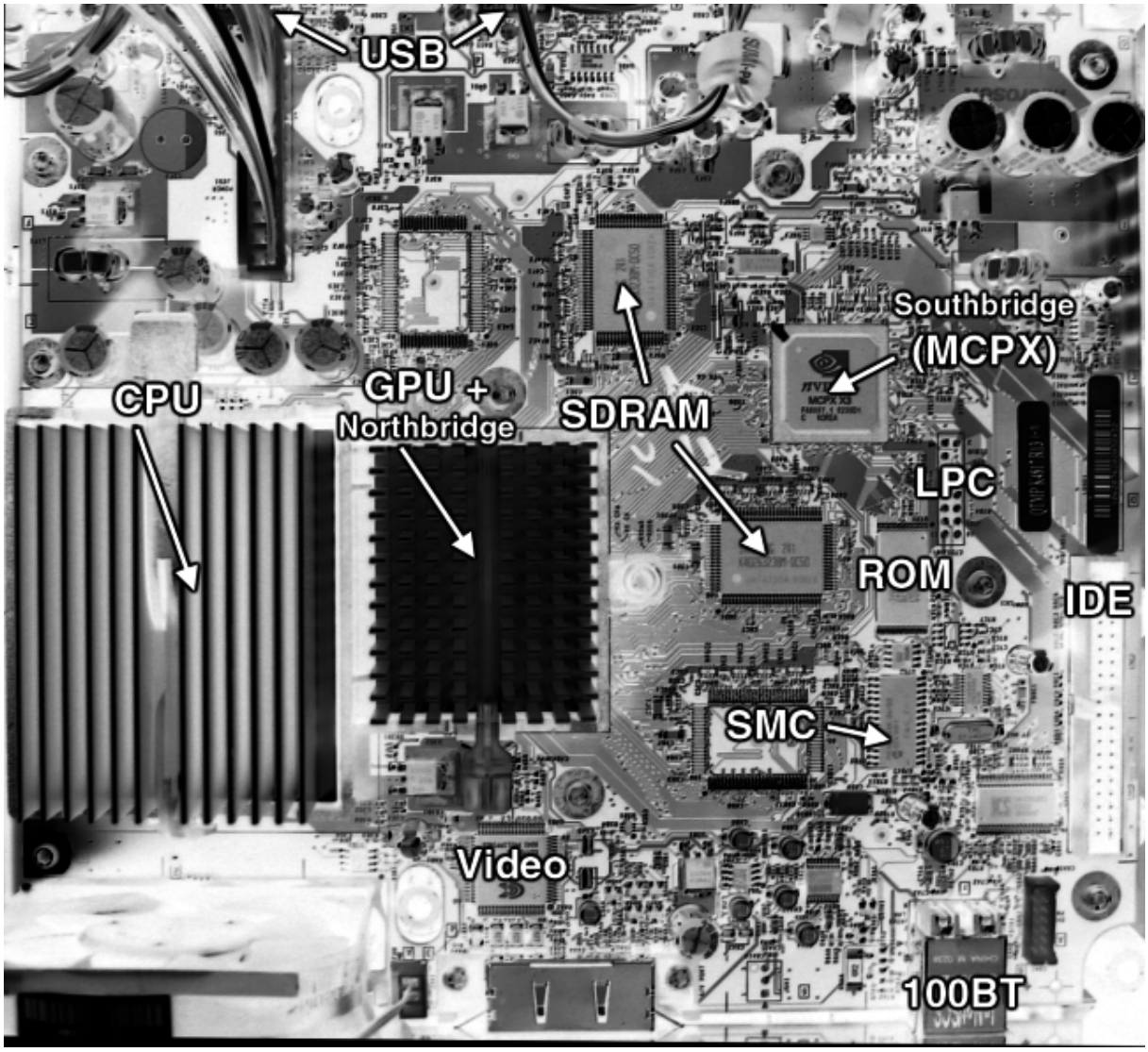


Рисунок 2-6. Фотография материнской платы Xbox с подписанными основными компонентами.

GPU подключён к микросхеме «всё-в-одном» под названием «MCPX» через быструю узкую шину HyperTransport. MCPX объединяет микросхему southbridge и почти все периферийные устройства Xbox, включая USB-контроллеры, устаревший интерфейс boot ROM, аудиопроцессор Dolby digital, IDE-контроллер массового хранения, ethernet-контроллер и интерфейсы к функциям управления системой.

Связность всех основных блоков Xbox показана на рисунке 2-5, а рисунок 2-6 показывает расположение этих блоков на реальной материнской плате Xbox.

Функциональные подробности

В следующих разделах представлен краткий обзор частей, составляющих архитектуру Xbox. Мы уделяем особое внимание деталям, необходимым для понимания того, как выполнять обратную разработку механизмов безопасности Xbox.

CPU

CPU (Central Processing Unit, центральный процессор) - вычислительное сердце обычного компьютера. Тема архитектуры CPU сама по себе заслуживает отдельной книги, поэтому мы рассмотрим только материал, необходимый для понимания обратной разработки Xbox. В частности, мы исследуем, как получить контроль над CPU Xbox.

CPU читает последовательности инструкций, хранящихся в памяти, - программы, - которые указывают CPU выполнять различные вычисления или принимать решения на основании доступных данных. Инструкции хранятся в памяти как числа, называемые opcodes. Opcodes принимают operands как аргументы. Программисты используют буквенные мнемоники при написании низкоуровневого машинного кода, чтобы не помнить сотни чисел opcode. Например, один вид инструкции вычитания шириной в байт имеет opcode 0010.1000 (двоичный) или 0x28 (шестнадцатеричный) и мнемонику «SUB». Требуемый opcode вычитания меняется в зависимости от источника и ширины данных вычитания. Отслеживать все правила соответствия opcode и operand трудно, поэтому процесс перевода мнемоник и operands в номера инструкций выполняется программой, называемой assembler. Аналогично процесс перевода номеров инструкций обратно в мнемоники выполняется disassembler. Существенно, что большинство программ не пишется на языке assembly; обычно используется язык более высокого уровня, например C. Эти высокоуровневые языки переводятся в машинные инструкции с помощью compilers. Автоматическая decompilation машинных инструкций обратно в язык высокого уровня может быть трудной, потому что процесс compilation - особенно optimized compilation - отбрасывает большую часть высокоуровневой структурной информации, содержащейся в исходном коде.

Процессор отслеживает, какая инструкция выполняется, с помощью instruction pointer (IP). В некоторых контекстах IP также называют program counter (PC). IP обычно продвигается по программе на одну инструкцию за раз, если только не встретится branch instruction. Branch instruction даёт программе возможность принять решение, проверяя данные внутри CPU и переходя к новому месту в зависимости от результата проверки. Понимание движения instruction pointer - центральная часть обратной разработки Xbox. Возможность манипулировать IP равнозначна контролю над тем, что Xbox может и не может делать. Меры безопасности, реализованные в программной архитектуре Xbox, пытаются гарантировать, что IP всегда исполняет только код, одобренный Microsoft, всегда криптографически проверяя фрагмент кода на подлинность перед его запуском.

> **Двоичные и шестнадцатеричные числа.** Цифровые схемы используют 1 и 0 для представления чисел. Эта двоичная, или «base-2», запись отражает способ, которым электрические сигналы используются для представления чисел: два диапазона уровней напряжения используются для определения одного логического состояния или другого. Можно построить электрические системы, представляющие информацию с помощью более чем двух уровней напряжения, но только ценой мощности и сложности. Современные модемы, например, используют несколько уровней напряжения и фазовую информацию, чтобы представлять несколько битов данных за одну единицу времени.
>
> Составление чисел и арифметика в двоичной системе следуют тем же правилам, что и привычное десятичное («base-10») представление. В десятичной системе нули используются как заполнители, чтобы помнить, когда цифра переполнилась. Например, 1 больше 9 приводит к переполнению, потому что нет одной цифры больше 9. Поэтому число 10 фиксирует, что у нас было одно переполнение самой правой десятичной позиции. Аналогично в двоичной системе 1 больше 1 равно 10, поскольку самая большая одиночная цифра в двоичной системе - 1.

Таким образом, в десятичной системе значение четырёхзначного десятичного числа `d4d3d2d1` можно разложить так:

```
d4 * 10^3 + d3 * 10^2 + d2 * 10^1 + d1 * 10^0
= d4 * 1000 + d3 * 100 + d2 * 10 + d1 * 1
```

Аналогично четырёхзначное двоичное число `b4b3b2b1` можно разложить так:

```
b4 * 2^3 + b3 * 2^2 + b2 * 2^1 + b1 * 2^0
= b4 * 8 + b3 * 4 + b2 * 2 + b1 * 1
```

Например, число 1010 = 1*8* + *0*4 + 1*2* + *0*1 = 10 в десятичной системе.

Следить за числами в прямой двоичной записи быстро становится неудобно; например, для представления десятичного 968 нужно десять двоичных цифр. Чтобы экономить место на экране, двоичные числа переводятся в восьмеричную или шестнадцатеричную запись. Восьмеричный формат, или «base-8», был популярен в ранние дни компьютеров, но с тех пор стал редкостью. Шестнадцатеричная, или «base-16», запись является фактической системой нумерации. В шестнадцатеричной системе 16 цифр, поэтому hex-цифры, соответствующие десятичным числам от 10 до 15, представлены буквами от A до F. Таблица 2-2 кратко показывает преобразование между двоичной, десятичной и шестнадцатеричной системами для первых 16 положительных целых чисел.

Чтобы отличать шестнадцатеричные числа от десятичных, многие используют соглашение языка C, где `0x[number]` представляет шестнадцатеричное число, а `[number]` неявно является десятичным числом. Для двоичных чисел в C нет похожего стандарта, поэтому некоторые используют стандарт Verilog, `[digits]'b[number]`, где `[digits]` - количество цифр в двоичном числе. Суффикс «b» после строки из 1 и 0, например `1010.1100.1110b`, также используется для обозначения двоичного числа. Обратите внимание, как «.» использована для группировки двоичных цифр в наборы по четыре; это помогает мысленно переводить двоичное число в шестнадцатеричное: `0xACE`.

Двоичные и шестнадцатеричные числа (продолжение).

Bin	Dec	Hex	Bin	Dec	Hex
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	10	A
0011	3	3	1011	11	B
0100	4	4	1100	12	C
0101	5	5	1101	13	D
0110	6	6	1110	14	E
0111	7	7	1111	15	F

Таблица 2-2. Таблица преобразования между двоичной, десятичной и шестнадцатеричной системами.

Сердце CPU - крошечная, но очень быстрая память, называемая register file. Несколько фрагментов данных могут записываться в register file и считываться из него каждый такт процессора. Данные из register file подаются в исполнительный блок, называемый arithmetic logic unit (ALU). Функция, вычисляемая ALU, управляется инструкциями, извлекаемыми из памяти. После обработки данных ALU они могут быть либо записаны обратно в register file, либо сохранены в память.

Важная функция производительности почти каждого современного CPU - ускоритель доступа к памяти, называемый cache. Caches - это небольшие быстрые памяти, хранящие копии данных и фрагментов инструкций, которые, вероятно, будут использованы ядром CPU в ближайшем будущем. Caches медленнее register files, но быстрее основной памяти; аналогично caches хранят больше данных, чем register file, но меньше данных, чем основная память.

Одна важная особенность cache CPU Xbox, о которой следует знать, состоит в том, что это writeback cache. Writeback caches позволяют копиям данных, хранящимся внутри CPU, расходиться с тем, что существует в основной памяти. Эта временная разница может усложнить попытки трассировать выполнение CPU, наблюдая только внешний трафик памяти. Cache memory также может использоваться security routines, чтобы скрывать промежуточные результаты вычислений от того, кто наблюдает шину памяти.

Северные и южные мосты

Термины Northbridge и Southbridge - разговорные термины, специфичные для архитектуры PC. Они относятся к двум базовым вспомогательным микросхемам, встречающимся практически в каждом PC. Микросхема Northbridge соединяет CPU с основной памятью, а также с любыми высокопроизводительными шинами расширения, такими как AGP и PCI. Микросхема Southbridge висит за микросхемой Northbridge и содержит всю дополнительную периферию, встречающуюся в типичном PC: parallel, serial, USB, mouse, keyboard, IDE controllers, audio codecs и многое другое. Разделение архитектуры PC на эти три основных модуля - CPU, Northbridge и Southbridge - позволяет разработчикам PC смешивать и сочетать разные виды архитектур памяти с разнообразным выбором процессоров и периферии.

Соединение между наборами микросхем Northbridge и Southbridge отличается от chipset к chipset. В случае Xbox в качестве соединения между функциональным эквивалентом микросхем Northbridge и Southbridge используется высокопроизводительная узкая параллельная шина HyperTransport. Шина имеет ширину всего 8 бит в каждом из двух направлений, но тактируется на 200 МГц, а данные считываются на каждом фронте тактового сигнала, поэтому эффективная пиковая скорость передачи составляет 400 Мбайт/с в каждом направлении. Микросхема Northbridge соединена с CPU через шину Front Side Bus (FSB). В случае Xbox FSB - это 64-битная шина 133 МГц, использующая логические уровни AGTL+.

Знание и понимание видов соединений между микросхемами критически важно в обратной разработке, потому что вид соединения определяет, насколько трудно перехватывать данные, идущие между разными компонентами. Подробности относительно более простой для подключения шины, HyperTransport, обсуждаются в главе 8, «Обратная разработка безопасности Xbox».

В Xbox Southbridge - это микросхема, разработанная nVidia, под названием MCPX; она является производной от nVidia nForce MCP Multimedia and Communications Processor. Микросхема Northbridge также была разработана nVidia и называется NV2A GPU. И Northbridge, и Southbridge были изготовлены TSMC (Taiwan Semiconductor Manufacturing Corporation). NV2A объединяет GPU (Graphics Processing Unit) и традиционные контроллеры памяти и шины расширения, встречающиеся в большинстве микросхем Northbridge. Как объяснялось ранее, объединение графического процессора и Northbridge позволяет системным разработчикам объединить графическую память с основной памятью, ценой некоторой потери производительности.

RAM

Материнская плата Xbox использует 64 MB DDR SDRAM в качестве основной памяти. DDR SDRAM означает Double Data-Rate Synchronous Dynamic Random Access Memory. Благодаря объединению синхронизации и методов DDR совокупная пропускная способность основной памяти Xbox достигает 6,4 Гбайт/с.

RAM в основе представляет собой таблицу информации, индексируемую CPU. Каждая ячейка в RAM имеет уникальный индексный номер, называемый её address, и, как подразумевает название «random access», в RAM нет ограничений на порядок доступа к данным. [1]

Термин «dynamic» применяется к RAM, которую нужно постоянно обновлять, чтобы сохранить целостность данных. Например, RAM, используемая в Xbox, должна считывать и записывать обратно каждую ячейку примерно тридцать раз в секунду. Потеря производительности не так плоха, как звучит, поскольку в современные микросхемы DRAM встроено специальное оборудование, помогающее оптимизировать процесс.

Префикс «synchronous» означает, что внутри DRAM процедура доступа к данным разбита на серию шагов. Каждый из этих шагов независим и может происходить параллельно, так что несколько запросов данных могут одновременно находиться в обработке. Внешний времязадающий сигнал, известный как clock, используется для синхронизации перемещения запросов доступа к данным через различные шаги внутри DRAM. В результате запросы доступа к данным проходят через каждый шаг, как вода по трубе, и этот приём также известен как pipelining. Synchronous DRAM имеют более высокую пропускную способность, чем их предшественники, потому что pipelining позволяет обрабатывать несколько запросов одновременно. Однако время от первого выдавания запроса к SDRAM до появления данных на выходе - access latency - не улучшается pipelining.

Термин «Double Data Rate» относится к тому, как синхронные данные передаются относительно синхронизирующего clock. Форма clock состоит из повторяющегося шаблона высоких и низких сигналов. В традиционных системах данные передаются только на переходе формы clock от низкого уровня к высокому. В системе DDR данные передаются и на переходе от низкого к высокому, и на переходе от высокого к низкому. Поэтому при той же частоте clock можно передать вдвое больше данных. Мнемоника производительности, указываемая поставщиками DDR SDRAM, например DDR266, относится к скорости передачи, так что фактическая частота clock равна половине мнемоники производительности, то есть 133 МГц в этом случае.

ROM

Каждому компьютеру нужна какая-то постоянная, или non-volatile, память для хранения программы запуска, или boot program. DDR SDRAM, обсуждённая выше, не подходит для этого применения, потому что все данные в DDR SDRAM теряются при снятии питания. Текущие версии Xbox вместо этого используют FLASH ROM для хранения данных, которые должны сохраняться даже при выключенном питании. ROM означает Read-Only Memory, а FLASH относится к конкретному стилю элемента хранения, который электронно перепрограммируется. FLASH-память удобна в PC, потому что конечный пользователь может перепрограммировать её для исправления ошибок в boot code. Однако в Xbox программирование FLASH ROM конечным пользователем намеренно отключено. Сигнал записи, необходимый для программирования, отключён путём отсутствия перемычки, расположенной на обратной стороне материнской платы Xbox в позиции компонента R7R4 (дополнительные сведения см. во врезке «Включение аппаратного программирования FLASH ROM»). В случае Xbox перепрограммируемость FLASH в основном используется Microsoft как удобство во время разработки и производства. Вполне вероятно, что через несколько месяцев Xbox будет использовать более дешёвые жёстко прошитые «mask ROMs», как только Microsoft решит, что готова высечь свою boot program и kernel в камне (или в кремнии, в зависимости от обстоятельств).

Boot ROM имеет ключевое значение в обратной разработке любого компьютера, потому что содержит критический код, отвечающий за инициализацию всей системы. В случае Xbox boot FLASH ROM играет ещё более важную роль, потому что частично отвечает за реализацию строгой программной системы безопасности. Точная роль FLASH ROM в системе безопасности будет объяснена позже, но сейчас важно запомнить, что FLASH ROM управляет инициализацией аппаратуры Xbox, а также содержит начальный образ kernel операционной системы.

> **Включение аппаратного программирования FLASH ROM.** Восстановление сигнала, который Microsoft отключила, чтобы предотвратить внутрисистемное программирование FLASH ROM, - довольно простая процедура. Сигнал записи FLASH ROM был отключён отсутствием одного резистора, компонента R7R4, расположенного на нижней стороне материнской платы Xbox в секторе 7R. Можно припаять кусочек провода между двумя серебристыми площадками резистора или даже просто замкнуть площадки большим количеством припоя. Обратите внимание: даже если программирование FLASH ROM включено в аппаратуре этим исправлением, у вас всё ещё нет программы, которая действительно выполняет перепрограммирование. Запуск такой программы является гораздо большей трудностью из-за криптографической программной системы безопасности, установленной Microsoft.

Разное

Xbox имеет небольшой 8-битный сопроцессор, называемый System Management Controller (SMC). SMC - это полноценный миниатюрный компьютер с RAM, ROM и процессором в одном корпусе. Процессор внутри SMC использует архитектуру PIC (Peripheral Interface Controller), первоначально разработанную в Гарвардском университете примерно в 1975 году и адаптированную General Instruments для коммерческой продажи. Arizona Microchip Technology (теперь называющаяся Microchip Technology, www.microchip.com) приобрела продуктовую линейку PIC в 1985 году и с тех пор продаёт её. SMC можно найти в секторе 7B на Xbox, а его позиционное обозначение - U7B2.

SMC следит за кнопкой питания на передней панели Xbox, поэтому SMC должен работать даже тогда, когда CPU выключен. В результате блок питания Xbox имеет маломощную линию 3.3V «standby», которая всегда активна, когда Xbox подключена к сети. SMC также отвечает за

управление огнями вокруг кнопки питания на Xbox и управляет механизмом извлечения DVD. Наконец, SMC имеет функцию, которая следит за состоянием CPU и перезагружает CPU в случае сбоя. Функцию мониторинга SMC необходимо отключить, если вы хотите запустить на Xbox собственную операционную систему. SMC общается с CPU через MCPX по 1-битному последовательному интерфейсу, известному как I2C.

Ещё одна важная особенность Xbox - отладочный порт LPC. Отладочный порт LPC - это шина шириной 4 бита, работающая на частоте 33 МГц. LPC означает «Low Pin Count», и изначально он был придуман как метод подключения большого количества медленных устаревших устройств, таких как клавиатуры, serial ports, parallel ports и boot ROMs, к микросхеме Southbridge через простую промежуточную микросхему преобразования. Отладочный порт на Xbox, предположительно, предоставлен для производственного тестирования аппаратным подрядчиком Microsoft. Когда Xbox приближается к финальным стадиям производства, отладочный порт LPC используется для загрузки boot program, которая выполняет тесты, диагностику и burn-in материнской платы Xbox. Отладочный порт LPC подробнее обсуждается в главе 11, но сейчас важно знать, что можно заставить Xbox читать свой начальный образ boot ROM через отладочный порт LPC, подключив LPC-совместимое ROM-устройство и замкнув один из выводов данных (D0) на FLASH ROM на определённое напряжение (ноль вольт). Это, возможно, самый простой способ заставить Xbox загружать ваш собственный код - при условии, что вы знаете, как обойти секретный boot code, защищающий Xbox.

Сопоставление с шаблонами

Теперь, когда мы знакомы с архитектурой Xbox, у нас есть точка отсчёта для, возможно, одного из самых мощных инструментов обратной разработки - сопоставления с шаблонами. Способность делать обоснованные предположения о функции разных деталей, просто наблюдая их соединения, размещение и форму, - первый шаг к тому, чтобы стать первоклассным специалистом по обратной разработке. Чтобы продемонстрировать силу сопоставления с шаблонами, мы сравним материнскую плату Xbox с материнской платой PC и материнской платой Nintendo Gamecube.

Лучший способ стать хорошим распознавателем шаблонов - изучить много шаблонов. Я разбираю каждую единицу оборудования, которую покупаю, и внимательно изучаю печатные платы, пытаясь узнать, что знают другие разработчики, «читая» печатную плату. Каждая печатная плата рассказывает историю своего процесса проектирования; редко встретится странная особенность схемы, у которой нет какого-то предполагаемого назначения.

> Важно. При разборке любого электронного оборудования обязательно сначала отключите его от сети и подождите минуту, чтобы заряд на крупных конденсаторах в блоке питания рассеялся. Также обязательно используйте надлежащие меры контроля статического электричества, описанные в главе 1!

Сравнение: Xbox и PC

Сходство Xbox с PC - благо для хакеров, поскольку платформа PC очень хорошо документирована. Каждая деталь в Xbox имеет аналог в типичном PC, поэтому почти на любой высокоуровневый вопрос можно ответить, просто прочитав о похожей детали PC. Поэтому полезно внимательнее рассмотреть сходства между материнской платой Xbox и стандартной материнской платой PC. Ещё одно преимущество состоит в том, что значительная часть информации в этой книге напрямую применима к PC, так что вы сможете легко применять то, чему научитесь при хакинге Xbox, во множестве ситуаций.

Ближайшие родственники Xbox - системы на базе chipset, использующих унифицированную архитектуру памяти, такие как nForce от nVidia или ProSavageDDR от Via Technology. Архитектурная схема, представленная в предыдущем разделе, была выведена чтением опубликованных спецификаций Xbox и материалов, доступных на сайте nVidia о chipset nForce. В этом разделе мы сравним Xbox с материнской платой P4M266 на базе Via Technology ProSavageDDR. Здесь Xbox сравнивается с материнской платой на chipset не от nVidia, чтобы подчеркнуть широкое сходство Xbox с PC.

На рисунке 2-7 показана фотография материнской платы PC, Via P4M266. Хотя chipset изготовлен другим поставщиком, сходства между P4M266 и Xbox поразительны. Почти весь материал, рассмотренный в предыдущем разделе, применим к этой материнской плате PC. Основные различия - несколько разных портов и разъёмов, а также наличие высокопроизводительных портов расширения PCI и AGP. Via P4M266 также не имеет явного отладочного разъёма LPC, поскольку вся устаревшая периферия напрямую реализована LPC-микросхемой multi-I/O.

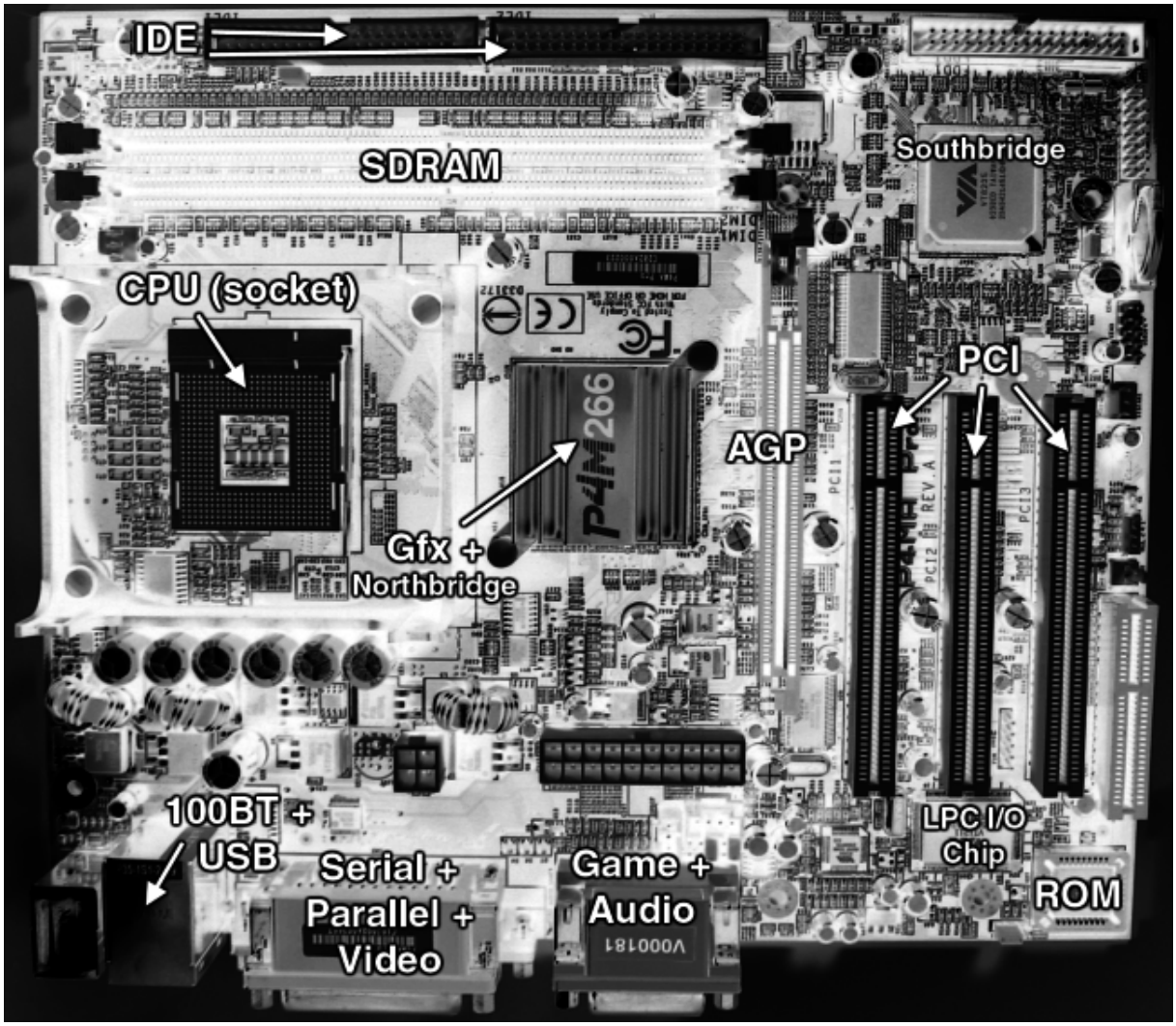


Рисунок 2-7. Материнская плата Via P4M266 со встроенной графикой.

Контраст: Xbox и Gamecube

Nintendo Gamecube интересна в сравнении с Xbox. Gamecube - машина, спроектированная для той же цели, что и Xbox, - игр, - но с учётом совсем другой философии проектирования. И Xbox, и Gamecube используют одну и ту же общую архитектуру: CPU, графический сопроцессор, немного памяти и несколько вспомогательных микросхем, но на этом сходство заканчивается. Конструкция Gamecube демонстрирует пристальное внимание к деталям и стоимости. Материнская плата Gamecube мала и проста, количество компонентов сведено к минимуму, а радиаторы и тепловая конструкция очень просты. Чистая, прямая разводка большинства дорожек PCB на материнской плате Gamecube отражает тот факт, что почти каждая IC специально спроектирована для Gamecube. В результате Gamecube является гораздо более экономичной платформой для производства, чем Xbox.

Общую организацию Gamecube можно распознать, выводя функцию каждой микросхемы из базовой маркетинговой информации, предоставляемой Nintendo. Дополнительные подробности об архитектуре Gamecube вывести трудно, потому что она использует так много заказных компонентов, не имеющих аналога в стандартном PC. По рисунку дорожек на материнской плате можно было бы предположить, что крупная микросхема в центре платы, чип «Flipper», является эквивалентом интегрированного графического Northbridge в PC. Это почти верно. Ключевое

отличие состоит в том, что хотя Flipper объединяет и контроллер памяти, и графический контроллер в одном корпусе, графическая функция всё равно имеет собственную выделенную память, встроенную в ту же микросхему. Такая организация позволяет графическому engine использовать память с очень высокой производительностью, ценой того, что память оказывается немного меньше, чем если бы использовалась память вне микросхемы. Меньший размер встроенной памяти отчасти компенсируется использованием чрезвычайно быстрой внешней памяти.

Gamecube не использует DDR SDRAM, как Xbox; вместо этого она использует то, что называется 1-T SRAM. 1-T SRAM - это DRAM-память, имитирующая очень быстрый тип памяти, известный как Static RAM (SRAM). SRAM имеет гораздо меньшие задержки random access, чем DRAM, и ей также не требуется обновлять каждую ячейку памяти 30 раз в секунду, как DRAM. Реальная магия того, как DRAM может притворяться быстрой SRAM, довольно сложна и выходит за рамки этой книги, но больше информации можно найти на сайте производителя 1-T SRAM, www.mosys.com.

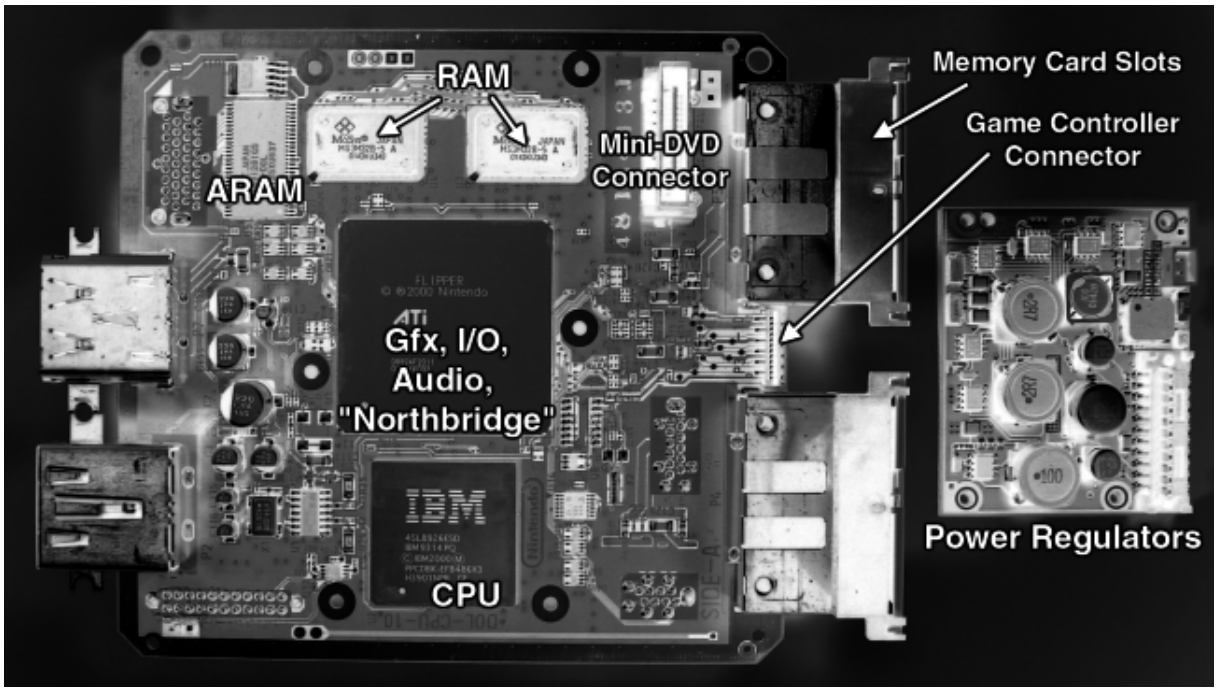


Рисунок 2-8. Материнская плата Gamecube вместе с платой регулятора питания. Материнская плата примерно вдвое меньше материнской платы Xbox.

Gamecube также имеет ещё один вид памяти, известный как ARAM, который медленнее памяти 1-T SRAM и используется для хранения вещей вроде аудиосэмплов, не требующих высокополосного доступа. Разнородная архитектура памяти означает, что Gamecube может выжимать более стабильный объём производительности из каждой подсистемы, что важно для сведения задержек кадров к минимуму. Однако компромисс состоит в том, что Gamecube может быть сложнее программировать, а неправильное управление несколькими фрагментами памяти может привести к проблемам производительности.

Ещё одно важное различие между Gamecube и Xbox состоит в том, что Gamecube потребляет гораздо меньше энергии, чем Xbox. Потребление энергии на первый взгляд может казаться неважным, поскольку обе консоли рассчитаны на подключение к розетке, но более низкий энергетический профиль Gamecube требует меньшего количества компонентов теплопередачи и меньших блоков питания, экономя стоимость. На рисунке 2-8 для справки приведена фотография регулятора питания Gamecube; регулятор питания занимает долю объёма блока питания Xbox плюс локальных импульсных регуляторов на материнской плате Xbox.

Справедливости ради отметим, что Gamecube действительно имеет небольшой внешний преобразователь AC to DC, тогда как Xbox принимает сетевое питание прямо внутрь консоли. Кроме того, электронные компоненты деградируют гораздо быстрее при повышенных температурах, как описывает правило Аррениуса. Например, повышение рабочей температуры на 10 градусов Цельсия примерно удваивает частоту отказов компонента. Поэтому Gamecube должна быть надёжнее Xbox с годами, поскольку Gamecube выделяет меньше тепла, а её система управления теплом не хуже, а возможно и лучше, чем у Xbox.

Наконец, интересно отметить, что Gamecube повсюду использует проприетарные I/O-интерфейсы. Формат игрового диска - mini-DVD, а DVD reader подключается к материнской плате через проприетарный разъём. Использование DVD-носителя меньшего размера позволяет Nintendo снизить задержку поиска данных, что означает более короткое время загрузки игр. Игровые контроллеры и карты памяти также используют проприетарный формат сигналов. Всё в Gamecube в некоторой степени похоже на знакомый нам PC, но ничто не было напрямую включено в конструкцию без изменений.

Помимо оптимизации технологичности и стоимости Gamecube, использование в основном проприетарных микросхем и стандартов делает консоль намного более трудной для обратной разработки, чем Xbox. Например, обратите внимание, что на рисунке 2-8 в Gamecube нет очевидной микросхемы ROM. Поэтому, чтобы хотя бы начать смотреть на код Gamecube, нужно разыскать и извлечь ROM, скрытый где-то в одной из микросхем на материнской плате! Это один из редких случаев, когда безопасность через неясность работает. Даже если бы на Gamecube вообще не было безопасности, стоимость и усилия, необходимые для записи собственного кода на заказной DVD-формат Nintendo, просто не стоят того для отдельного энтузиаста.

Сноски

[1] [назад] На самом деле SDRAM могут иметь некоторые ограничения на шаблоны доступа к памяти (такие как page modes и burst modes) по причинам производительности. Прозвище «random» предназначено для отличия RAM от памятей стиля First-In, First-Out (FIFO) и Last-In, First-Out (LIFO), где доступ к данным выполняется по строгому набору правил упорядочивания.

Глава 3. Установка синего LED

Теперь, когда вы сняли крышку со своей Xbox, пора выполнить пару начальных проектов. Следующие главы проведут вас через несколько элементарных модификаций и ремонтов, которые можно выполнить на Xbox. Эти проекты спроектированы и написаны для читателей, у которых мало опыта аппаратного хакинга или его нет совсем. Более продвинутые темы, связанные с Xbox, можно найти в следующих главах.

В этой главе вы узнаете, как заменить обычно зелёный LED на передней панели Xbox синим LED. Этот проект требует минимальной пайки; большая часть усилий приходится на снятие передней панели и сборки схемы LED. Начнём!

> NB. Стандартная Xbox использует комбинированный зелёно-красный LED, но красный LED применяется только для индикации условий ошибки. Процедура, описанная в этой главе, превратит индикатор передней панели Xbox в LED только синего цвета. Правильной заменой был бы сине-красный комбинированный LED (T-1, линза диаметром 3 мм). Однако их трудно найти, поэтому приведённые здесь инструкции их не используют. Инструкции дадут необходимые базовые знания, чтобы вы могли импровизировать и внедрить собственное LED-решение, если захотите.

Что понадобится

Ниже приведён список оборудования, которое понадобится для завершения этого проекта:

• Низковаттный паяльник с тонким жалом.
• Припой.
• Флюс и очиститель жала паяльника (необязательно).
• Маленькая плоская отвёртка.
• Отвёртка Torx с битой T-10.
• Два низковольтных (3 вольта) синих LED в корпусе T-1 (3 мм).
• Малярная лента для удерживания деталей на месте во время пайки.

Вы можете заменить LED на любой цвет, который вам нравится, но он должен включаться при напряжении около 3 вольт и иметь корпус стиля T-1. Будьте внимательны при покупке LED, потому что многие синие и белые LED рассчитаны на работу только при 5 вольтах. Здесь используется LED Lumex SSL-LX3044USBC, который можно купить через Digi-Key (www.digikey.com); номер детали 67-1747-ND. (Для тех, кто бережёт бюджет, Digi-Key может отправить LED через first class mail Почтовой службы США. Обратите внимание, что за заказы в Digi-Key меньше 25 долларов взимается сбор за обработку 5 долларов.)

Если ограничение Digi-Key на минимальный заказ является для вас проблемой, Mouser Electronics (www.mouser.com) также имеет линейку синих LED, и у них нет минимального заказа. Пример детали - синий LED Kingbright в корпусе T-1 с прозрачной водяной линзой; складской номер Mouser - 604-L7104PBC/H для более яркой версии с немного более высоким напряжением или 604-L7104QB/D для версии, работающей при более низком напряжении, но с меньшей заявленной яркостью.

Вы также можете использовать двухцветный LED, если хотите сохранить функциональность LED для индикации условий ошибки. Xbox требует двухцветный LED с общим катодом в корпусе T-1 с тремя выводами. К сожалению, двухцветные LED с синим элементом в меньшем корпусе T-1 очень трудно найти.

Снятие передней панели Xbox

Передняя панель Xbox - это формованная деталь из ABS-пластика, удерживаемая четырьмя винтами torx T-10 и тремя формованными фрикционными замками. Электроника в передней панели подключается к материнской плате Xbox через один девятипроводный разъём, который проходит через отверстие в металлическом экране электромагнитных помех.

Откройте Xbox, как указано в главе 1. Поднимите и переместите жёсткий диск и DVD-привод вверх и к задней части коробки ровно настолько, чтобы открыть передний край материнской платы Xbox. Вам не потребуется отсоединять какие-либо кабели дисковых приводов. Рисунок 3-1 показывает, как должна выглядеть Xbox после этих шагов.

Рисунок 3-1. Расположите дисковые приводы так, чтобы передний край материнской платы был открыт.

> NB. В старых моделях Xbox рядом с передней частью Xbox будет вертикально установленная PC-плата. Эту PC-плату можно снять, взявшись за плату и вытянув её из разъёма. Вы можете обнаружить, что снятие вертикально установленной PC-платы помогает при попытке освободить средний фрикционный замок передней панели. Не забудьте вернуть PC-плату на место, когда закончите!

Выкрутите четыре винта, удерживающих сборку передней панели на месте (см. Рисунок 3-2).

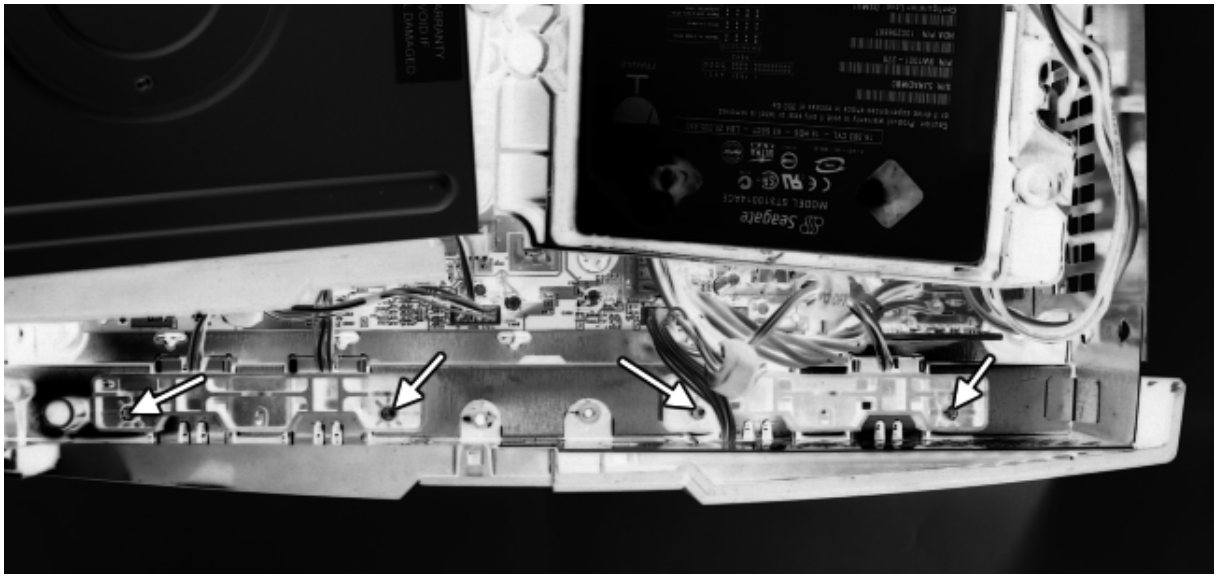


Рисунок 3-2. Расположение четырёх удерживающих винтов на сборке передней панели.

Отсоедините проводной разъём передней панели от материнской платы Xbox, как показано на рисунке 3-3. Разъёму должно требоваться только уверенное, равномерное усилие. (Не выдёргивайте разъём рывком, потому что можете повредить провода.)

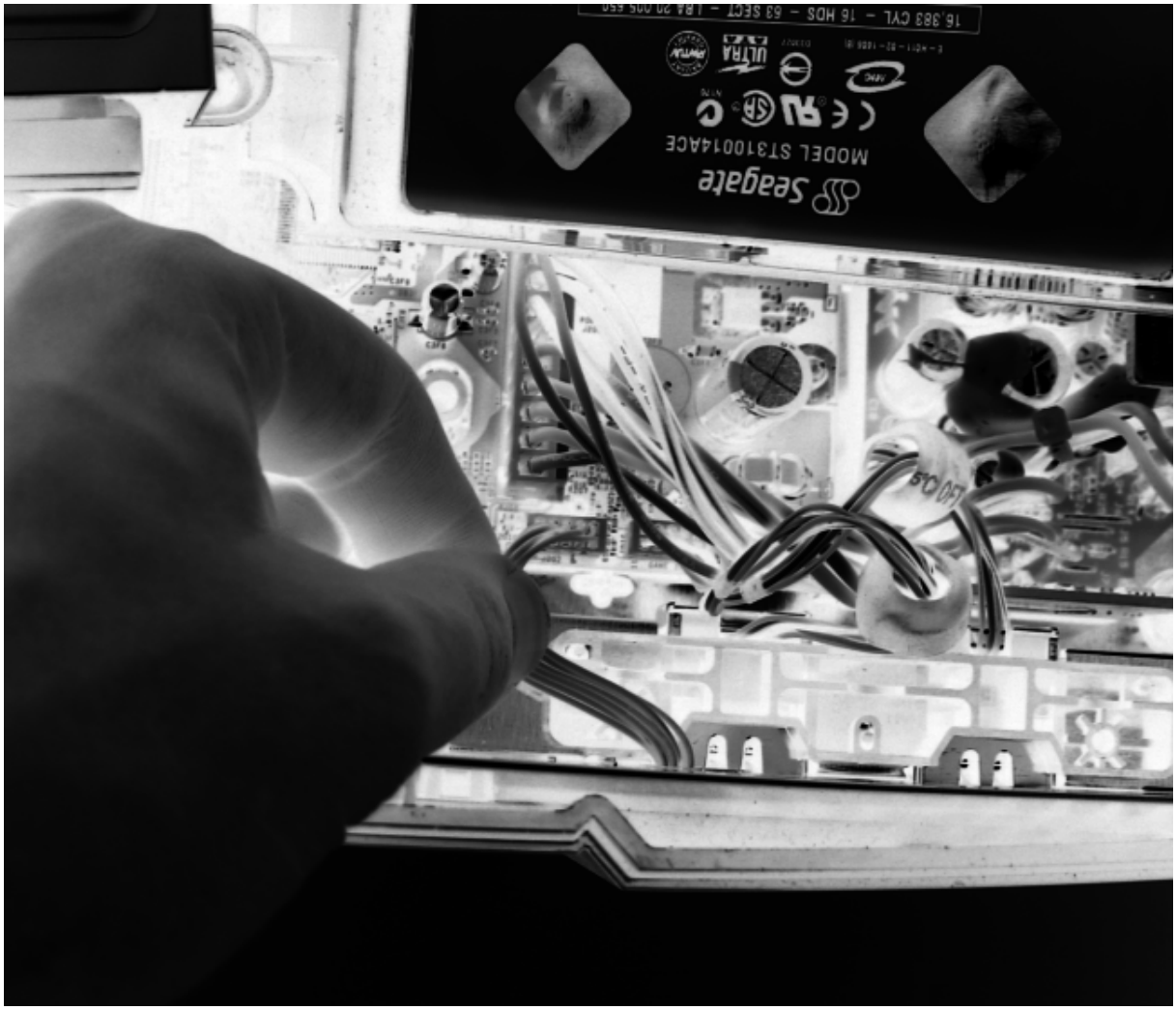


Рисунок 3-3. Отсоедините проводной разъём передней панели от материнской платы Xbox.

Теперь сложная часть: фрикционные замки. Фрикционный замок - это крючок из пластика, удерживающий детали вместе. Крючок сформирован так, что его легко вставить, но трудно извлечь. Освобождение фрикционного замка обычно требует какого-то изгиба или нажатия на пластик.

Переднюю панель удерживают три фрикционных замка: по одному на каждом краю передней панели и один посередине, проходящий через металлический экран электромагнитных помех. Сначала ослабьте фрикционные замки на краях с помощью плоской отвёртки с тонким жалом, как показано на рисунке 3-4. Эти замки очень тугие, и вам, возможно, придётся освобождать их по частям, начиная с верхней части. Вставьте кончик отвёртки в пространство сбоку между панелью и основным корпусом и поддевайте, пока не почувствуете лёгкое податливое движение. Уберите отвёртку и повторите процесс около нижней части корпуса. Может потребоваться несколько попыток, прежде чем замок освободится.

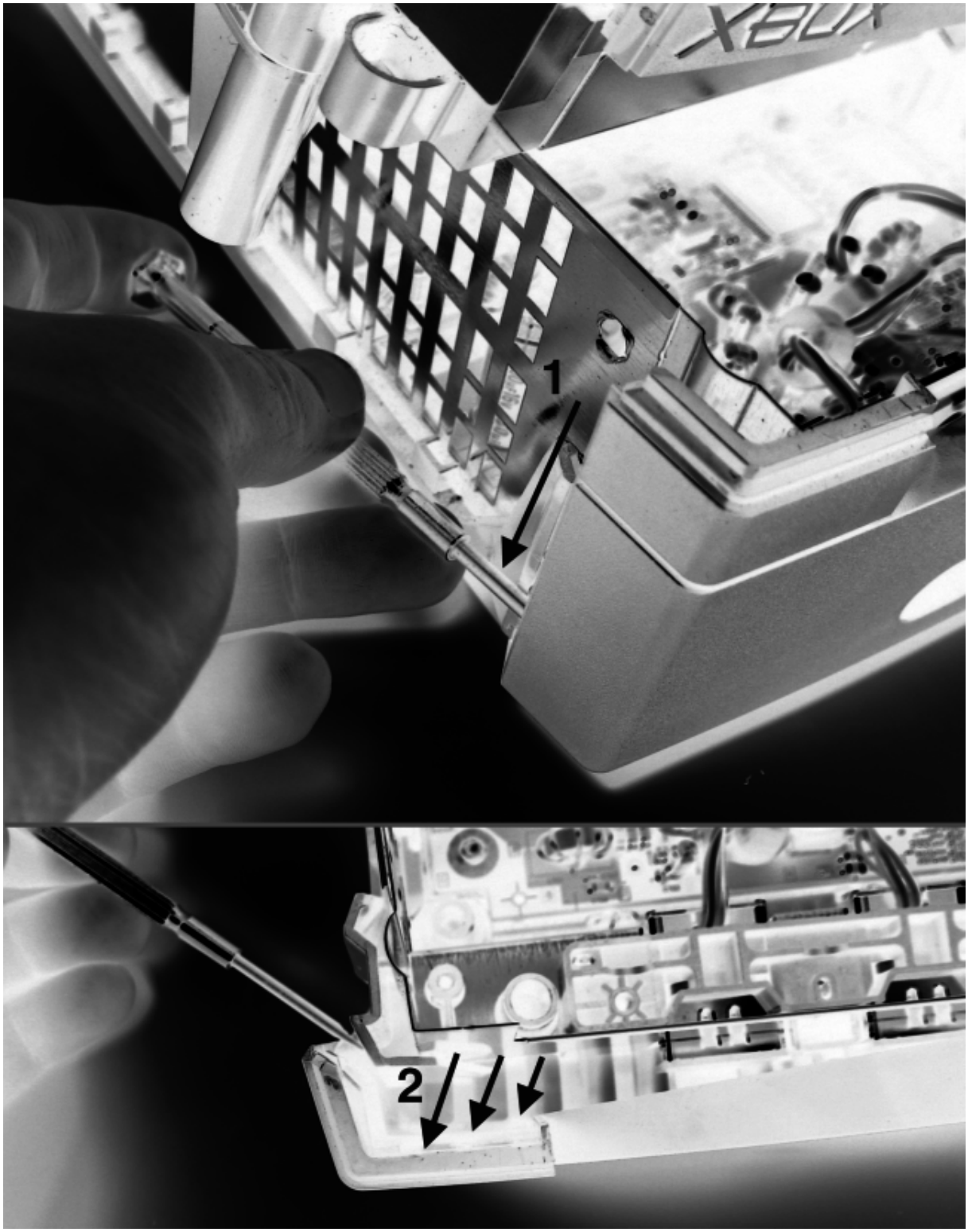


Рисунок 3-4. Ослабьте крайние фрикционные замки плоской отвёрткой. (1) Начните работать сверху и двигайтесь вниз; (2) когда панель освободится, она должна отогнуться наружу от корпуса.

Не прикладывайте к корпусу чрезмерную силу, потому что можете треснуть или надрезать пластик. Когда край передней панели освобождён, вы сможете отогнуть его от корпуса. Повторите этот процесс для обоих краёв.

Когда оба края освобождены, потяните вверх средний фрикционный замок (показан на рисунке 3-5), и передняя панель должна отщёлкнуться.

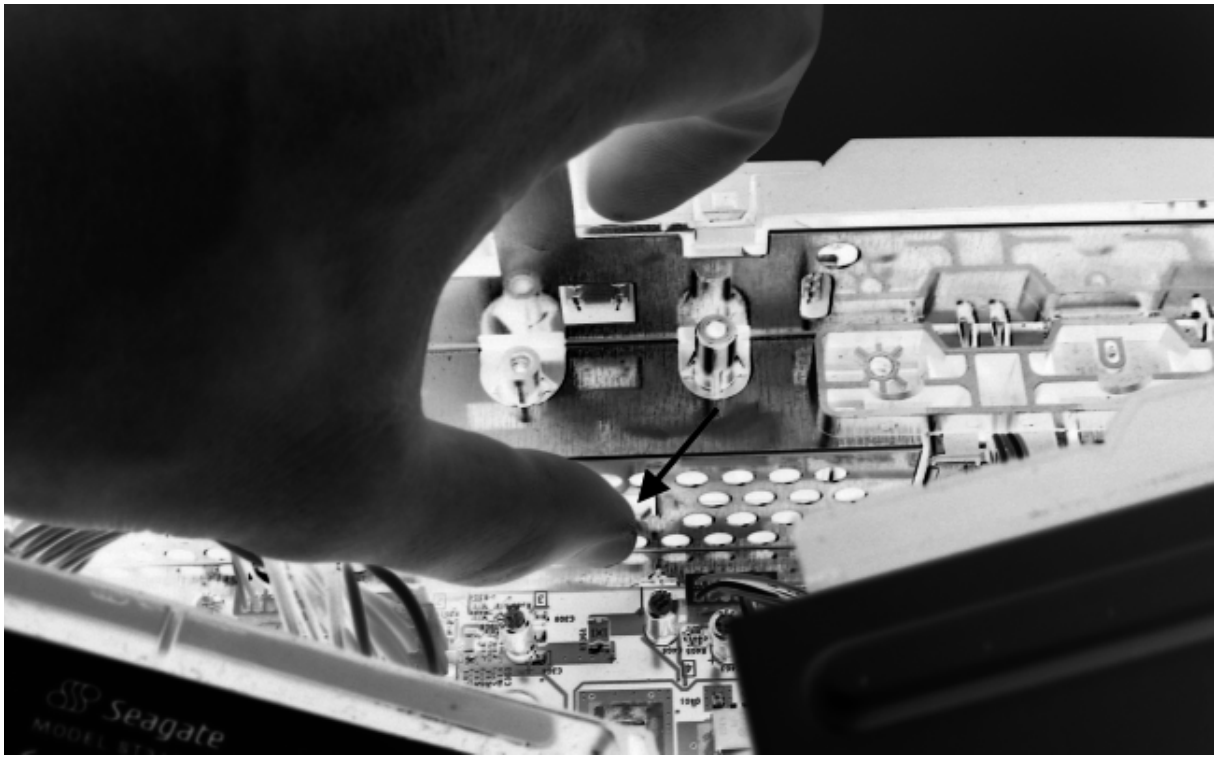


Рисунок 3-5. Большой палец нажимает на средний фрикционный замок.

После освобождения передней панели проденьте проводной разъём передней панели через отверстие в металлическом экране электромагнитных помех и положите панель плашмя на стол внешней стороной вниз.

Снятие печатной платы передней панели

Сборка передней панели Xbox содержит небольшую печатную плату, удерживаемую на месте одним удерживающим зажимом фрикционного замка, как показано на рисунке 3-6. Пальцем или отвёрткой нажмите на фрикционный замок и вытяните сборку печатной платы передней панели из её ложемента.

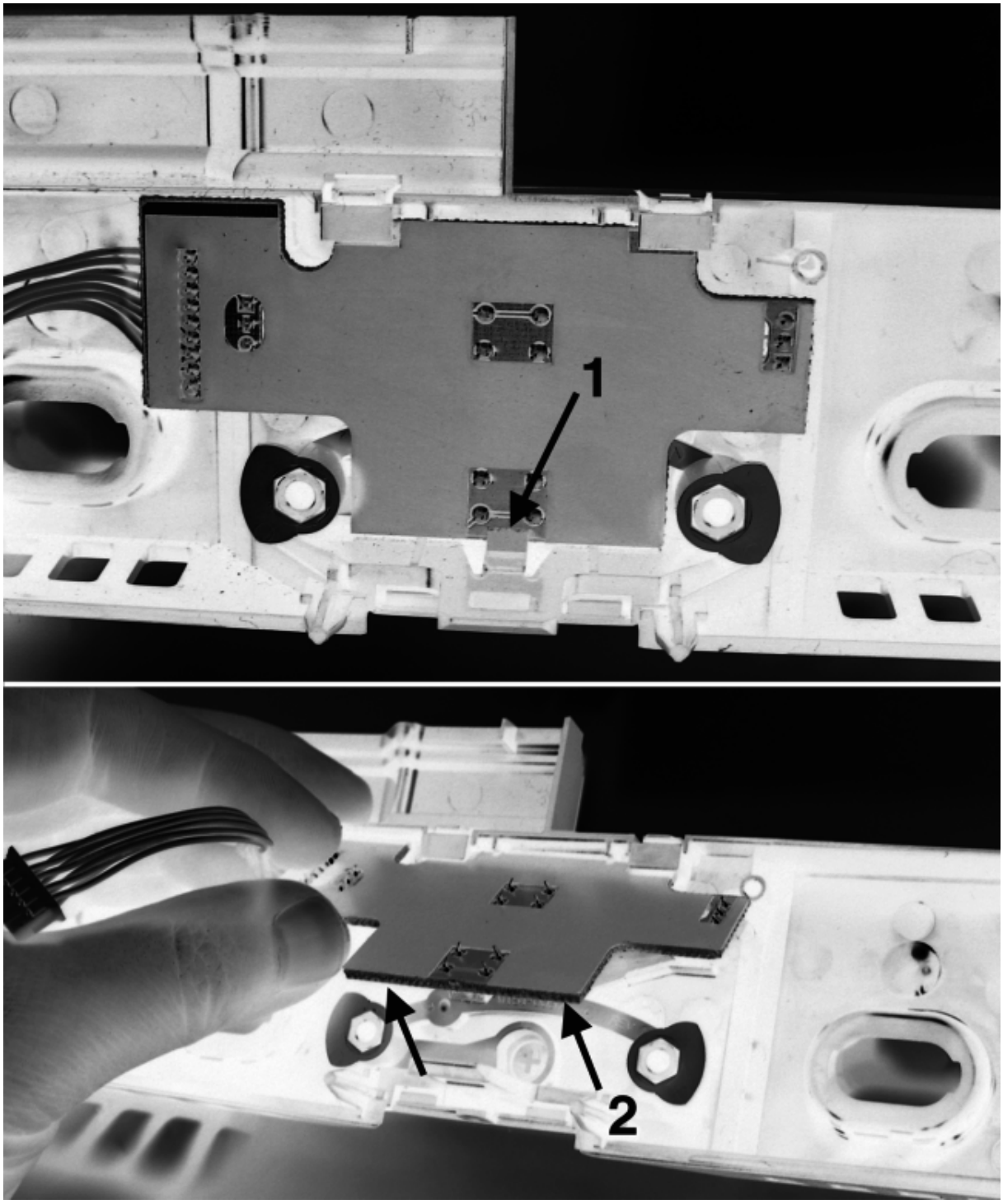


Рисунок 3-6. Поднимите сборку печатной платы из передней панели. (1) Нажмите на язычок фрикционного замка, при необходимости используя отвёртку, (2) затем поднимите сборку из передней панели.

Положите сборку печатной платы на стол зелёной стороной вниз. Вы должны увидеть два прозрачных LED и два плоских кнопочных переключателя.

Теперь, когда сборка печатной платы снята, пора установить синий LED. Для этого потребуется немного пайки, поэтому включите паяльник и дайте ему нагреться. Помните, что важно использовать паяльник с тонким жалом и очиститель жала паяльника, чтобы подготовить жало перед использованием. См. приложение A, «Где купить хакерское оборудование», если вам нужны какие-либо из этих предметов; вы сможете укомплектоваться чуть дороже стоимости видеоигры.

Установка синего LED

Удалите оба существующих LED кусачками заподлицо. Сохраните как можно больше металлических ножек, выходящих из LED, потому что позже вы используете их для крепления синего LED. Рисунок 3-7 показывает, как должна выглядеть печатная плата после завершения.

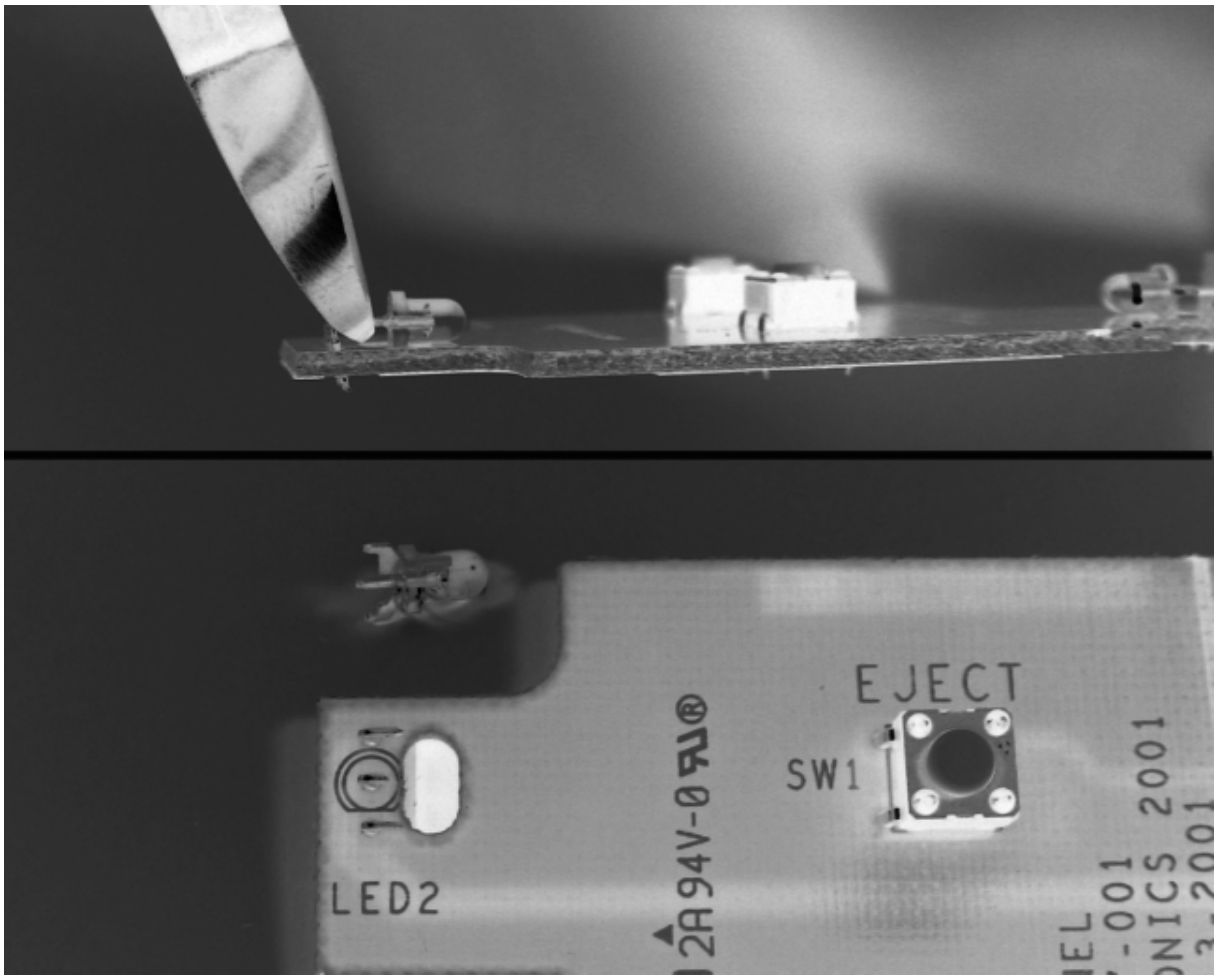


Рисунок 3-7. Срежьте существующие LED со сборки печатной платы. (Вверху) срежьте LED как можно ближе к корпусу; (внизу) LED удалён. Используйте эту процедуру для удаления обоих LED.

Чтобы помочь пайке, приклейте печатную плату к плоской поверхности кусочком малярной ленты, чтобы она не двигалась. Расположите синие LED так, чтобы их ножки касались металлических остатков старых LED на сборке печатной платы. (Рисунок 3-8 показывает, как определить полярность LED и их правильную ориентацию на сборке печатной платы. Если вы не уверены, как определить полярность LED, см. врезку «Анатомия LED».)

Приклейте линзовую часть LED малярной лентой, чтобы они не перекатывались, пока вы их паяете. Вам потребуется согнуть или обрезать ножки LED, который будет установлен на правой стороне платы, потому что жёлтый проводной разъём будет мешать.

> Важно. Внимательно следите за полярностью LED. Если установить LED задом наперёд, свет излучаться не будет. Если вы не уверены, как определить полярность LED, см. врезку «Анатомия LED».

Рисунок 3-8 также показывает полярность и функцию штатных LED Xbox. Смелым читателям предлагается импровизировать и установить несколько LED или корпуса LED поверхностного монтажа, чтобы попытаться получить больше цветов и функциональности. Можно установить LED, немного более крупные, чем корпус T-1, используемый в Xbox, если сначала сточить края LED.

```
EJECT / POWER

LED2: green / red
LED1: red / green

Расположите синие LED поверх соединений "green" LED,
короткой ножкой к середине.
Красный провод LED не должен касаться других проводов.
```

Рисунок 3-8. Размещение синих LED на сборке печатной платы передней панели. Обратите внимание на антисимметрию цветов LED на каждой стороне печатной платы.

После того как вы дважды проверили полярность LED и убедились, что короткая ножка обоих LED упирается в остаток центральной ножки исходного LED, припаяйте LED на место. Рисунок 3-10 показывает пайку LED на месте. (Если вы никогда раньше не паяли, перед продолжением может быть полезно прочитать приложение B, «Приёмы пайки».)

Перед использованием паяльника расплавьте немного проволочного припоя, чтобы убедиться, что жало достаточно горячее. Проволочный припой должен плавиться мгновенно, если жало достаточно горячее. Если паяльник слишком холодный, вы не сможете сформировать хорошее соединение и рискуете повредить печатную плату.

Держите горячее жало паяльника у ножки синего LED и прижимайте ножку к металлическому остатку на сборке печатной платы. Пока ножка нагрета, нанесите немного проволочного припоя в точку, где ножка синего LED встречается с металлическим остатком. Поверхностное натяжение расплавленного припоя должно заставить припой смочить ножки синего LED и металлический остаток на сборке печатной платы. Если этого не происходит, уберите паяльник, нанесите на соединение немного флюса и попробуйте снова.

> **Анатомия LED.** LED, или light emitting diodes, - поляризованные устройства, которые пропускают ток только в одном направлении. Это означает, что они не будут работать, если поставить их задом наперёд. Рисунок 3-9 показывает анатомию LED. Более короткая ножка со стороны корпуса LED, где есть небольшая плоская грань, называется cathode. Чтобы LED работал, cathode должен быть подключён к потенциалу более отрицательному, чем другая ножка, anode.

```
flat mark (most cases) -> плоская метка (в большинстве корпусов)
shorter lead           -> более короткая ножка
cathode (negative)     -> катод (отрицательный)
```

> **Рисунок 3-9.** Анатомия LED.

> Разные LED обычно требуют разного forward voltage для включения. Красные LED обычно требуют 1,7 вольта, зелёные LED - около 2,1 вольта, а синие LED - 3,5 вольта и выше. Ранние синие LED требовали почти 5 вольт forward voltage, но прогресс технологии снизил их напряжение, упростив интеграцию в питаемую от батарей и низковольтную электронику. Покупая LED для этого проекта, помните о требуемом forward voltage. Если установить синий LED на 5 вольт, его световой выход будет очень тусклым, поскольку максимальное forward voltage, создаваемое драйверами Xbox, составляет около 3 вольт.

Не держите жало паяльника у металлического остатка длительное время, иначе оно расплавит припой, удерживающий остаток на месте. Вы поймёте, что паяное соединение остатка с платой расплавилось, когда остаток начнёт свободно качаться. Если это случится, держите жало

паяльника у платы и медленно отводите жало в сторону. Перетаскивание жала предотвратит вытягивание остатка из платы вместе с паяльником. Подождите, пока остаток остынет, и согните его обратно в нужное положение.

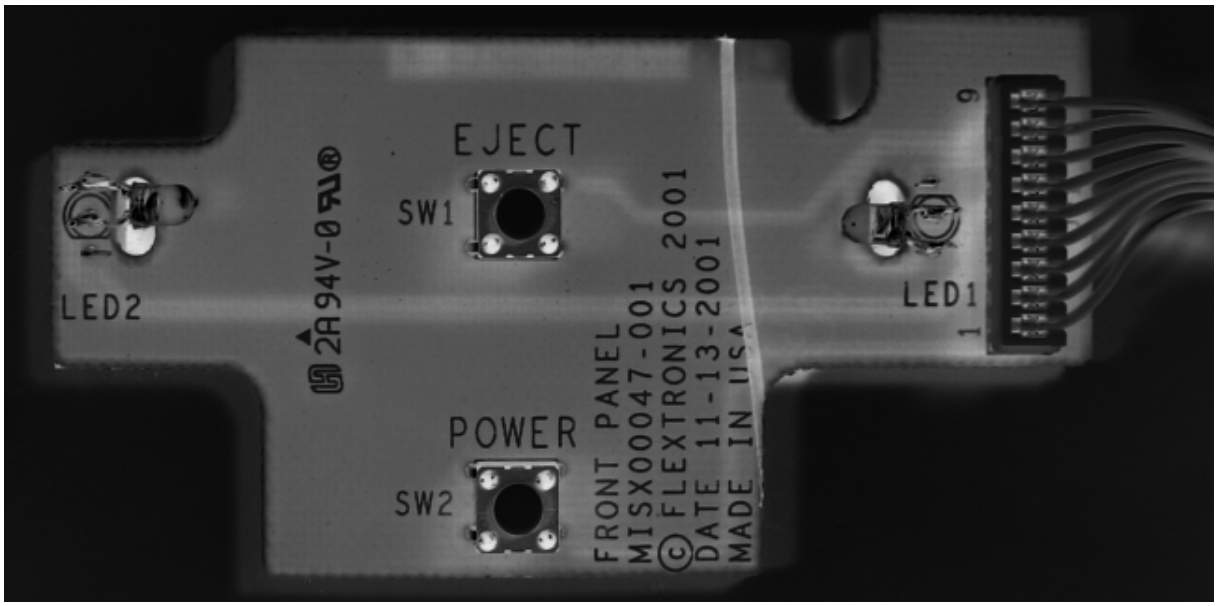


Рисунок 3-10. Пайка LED на месте. Обратите внимание, как плата и LED закреплены малярной лентой.

После того как вы припаяли все четыре соединения на двух LED, обрежьте лишние ножки LED настолько коротко, насколько сможете, не отрезая металлические остатки. Готовая плата должна выглядеть похоже на рисунке 3-11.

Рисунок 3-11. Готовая сборка платы.

Обратная сборка передней панели

Защёлкните сборку печатной платы обратно в переднюю панель, совместив верхние края под удерживающими зажимами и вдавив плату во фрикционный замок.

Теперь возьмите всю сборку передней панели и состыкуйте её с Xbox. Сначала пропустите проводной разъём через исходное овальное отверстие в экране электромагнитных помех. Затем вдавите переднюю панель в Xbox, и все три фрикционных замка должны защёлкнуться на месте.

Снова подключите проводной разъём передней панели к материнской плате Xbox. Разъём имеет такую форму, что вставить его можно только в одном направлении, но пластик, использованный при формовке разъёма, мягкий, и вы можете вставить его задом наперёд, если нажмёте достаточно сильно, возможно нанеся ему необратимое повреждение. Проверьте правильную ориентацию, совместив отсутствующий контакт на гнезде платы с отсутствующим проводом на разъёме, как показано на рисунке 3-12.

Рисунок 3-12. Признаки ориентации проводного разъёма передней панели. Стрелки указывают положение пустого поляризующего контакта.

Если у вас старая Xbox, установите вертикальную сборку печатной платы обратно, вдавив её в разъёмы. Эта печатная плата является интерфейсной картой игровых контроллеров Xbox, поэтому вам точно нужно, чтобы она была установлена правильно.

Теперь вы готовы проверить недавно модифицированную сборку передней панели. Верните дисковые приводы в их отсеки и убедитесь, что их кабель питания и ленточный кабель надёжно подключены. Подключите Xbox к сети, и вы должны увидеть синее свечение, исходящее от передней панели Xbox. (Если вы не видите ожидаемого, не паникуйте. Следующий раздел, «Отладка», предлагает решения некоторых возможных проблем.)

Когда вы удовлетворены модификацией Xbox, выключите Xbox и верните четыре удерживающих винта на сборку передней панели. (Вам потребуется снова снять дисковые приводы, чтобы получить доступ к отверстиям винтов.) Верните дисковые приводы, включите Xbox ещё раз, чтобы проверить, что всё работает нормально, а затем соберите остальную часть Xbox, как описано в главе 1.

Отладка

Иногда что-то идёт не так. По моему опыту, что-то идёт не так чаще, чем наоборот. Самое важное, что нужно помнить, если что-то не работает: не паникуйте! Сохраняйте ясность ума, наблюдайте, что именно идёт не так, и попытайтесь выдвинуть гипотезу о причине неисправности. Для справки таблица 3-1 содержит список распространённых проблем и их возможных причин. Приложение E, «Отладка: подсказки и советы», содержит более глубокое обсуждение методов отладки.

Таблица 3-1. Руководство по отладке установки синего LED.

Проблема	Возможная причина
Xbox не включается.	Проводной разъём передней панели неправильно вставлен.
Xbox не включается.	Xbox не подключена к сети.
Xbox не включается.	Проводной разъём передней панели повреждён или повреждена сборка печатной платы передней панели.
Xbox включается и работает правильно, но из LED не выходит свет или светится только половина светового круга вокруг кнопки eject.	Один или несколько LED установлены задом наперёд.
Xbox включается и работает правильно, но из LED не выходит свет или светится только половина светового круга вокруг кнопки eject.	Один или несколько LED установлены на металлические остатки красного LED вместо металлических остатков зелёного LED. Проверьте это, включив Xbox без подключённого видеокабеля. Это заставит Xbox отправить мигающий сигнал и на красные, и на зелёные LED.
Xbox проходит начальную анимацию, но консоль указывает, что требует обслуживания.	Разъёмы жёсткого диска или DVD ослабли. Проверьте, что серый ленточный разъём полностью вставлен в каждый привод и что разъём питания каждого привода полностью вставлен.
Xbox проходит начальную анимацию, но консоль указывает, что требует обслуживания.	На старых моделях Xbox проверьте, что вертикальная интерфейсная плата игровых контроллеров была правильно установлена обратно.
Xbox включается, но игровые контроллеры не отвечают.	На старых моделях Xbox проверьте, что вертикальная интерфейсная плата игровых контроллеров была правильно установлена обратно.
Xbox включается, но игровые контроллеры не отвечают.	Проверьте, что проводной разъём между передней панелью и материнской платой Xbox правильно вставлен.
Xbox включается, но DVD не извлекается.	Проверьте, что разъём питания DVD правильно вставлен.

На этом руководство по установке синего LED заканчивается.

Глава 4. Сборка USB-адаптера

Сборка кабелей - обязательная часть аппаратного хакинга. Многие модификации и эксперименты, выполняемые с аппаратурой, потребуют специального кабеля, чтобы адаптировать имеющиеся разъёмы к тому, что вам нужно.

В этой главе вы узнаете, как собрать USB-кабель-адаптер для Xbox, вещь, недоступную в большинстве розничных магазинов (хотя отдельные онлайн-продавцы, например Lik-Sang (www.lik-sang.com), действительно имеют этот товар). USB-адаптер позволяет подключать к Xbox стандартные USB-хабы, клавиатуры и мыши для запуска Linux. (Эта глава в форме учебного примера представляет основы надёжной сборки кабелей; опытные аппаратные хакеры могут свободно просмотреть её бегло или пропустить.)

Исходные материалы

Для этого проекта потребуются следующие материалы:

- Один запасной разрывной кабель игрового контроллера Xbox или удлинитель игрового контроллера. (См. Рисунок 4-1.)
- Один удлинительный кабель USB type A или розетка USB type A female. (Фотография розетки USB type A female показана на рисунке 4-2.)
- Паяльник, припой и флюс.
- Диагональные кусачки и инструмент для зачистки проводов.
- Изолента.
- (Необязательно) термоусадочная трубка 3/8 дюйма и термоклей.
- (Необязательно) держатель «третья рука» для пайки.

Пошаговое описание в этой главе использует запасной разрывной кабель (доступный в любом магазине видеоигр) и USB-удлинитель. Если вы хотите использовать что-то другое, см. приложение F, «Справочник по аппаратуре Xbox», где приведены распиновки различных разъёмов, применяемых в Xbox.

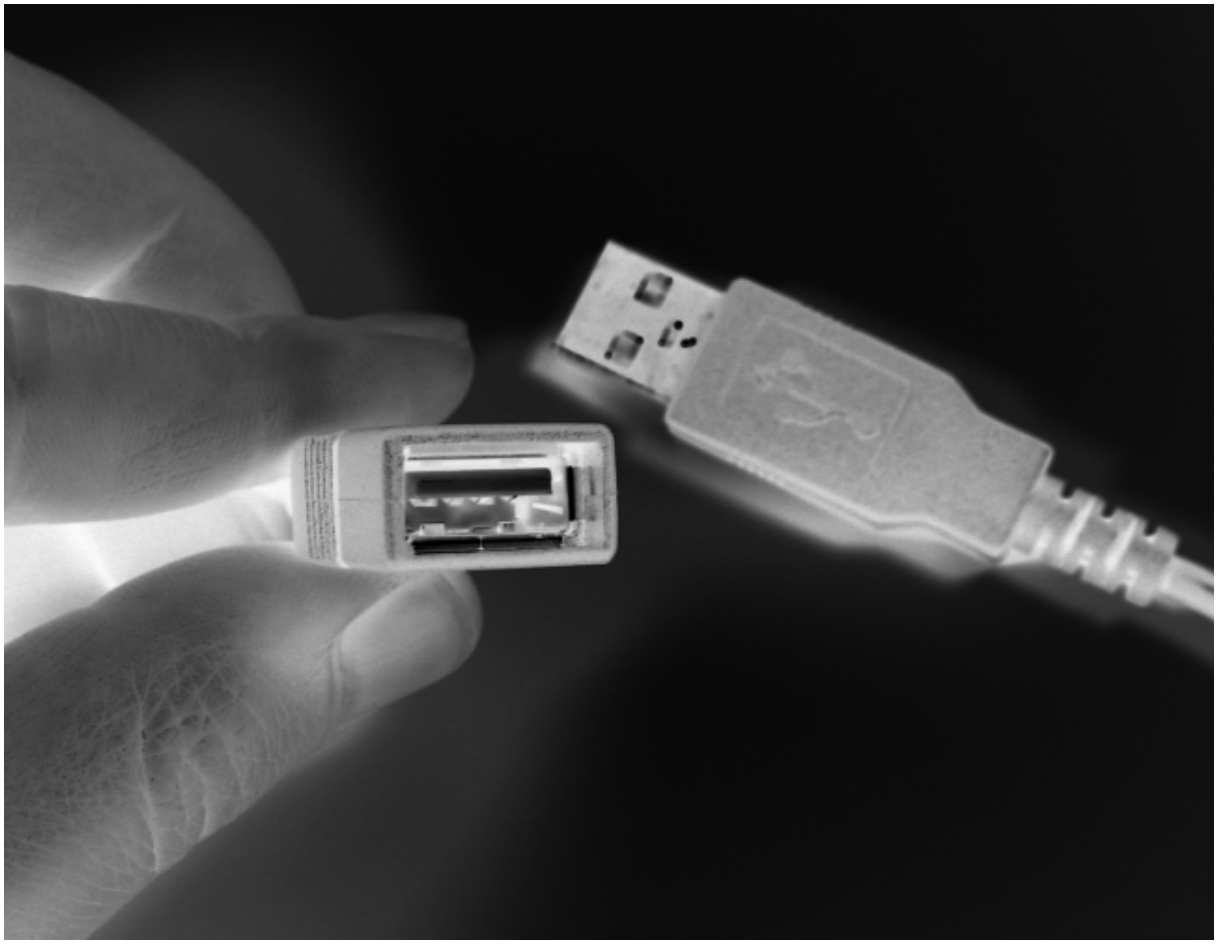


Рисунок 4-1. (Слева) удлинительный кабель игрового контроллера Xbox. (Справа) разрывной кабель Xbox.

Рисунок 4-2. Разъём USB type A female.

Стратегия

При сборке USB-кабеля-адаптера Xbox основная идея состоит в том, чтобы разрезать пополам разрывной кабель Xbox и USB-удлинитель и соединить правильные концы двух кабелей. К счастью, для USB-совместимых кабелей существует стандартная схема проводов. Красный - питание +5V, чёрный - Ground, белый - Data (-), а зелёный - Data (+). Чтобы соединить кабели, просто соедините провода одинаковых цветов. (Обратите внимание, что кабель Xbox будет иметь дополнительный жёлтый провод, несущий копию composite video sync signal для использования в игровых интерфейсах типа light-gun. Этот дополнительный жёлтый провод можно безопасно игнорировать.) Эта глава пошагово проведёт вас через соединение проводов и герметизацию кабеля.

> Важно. Некоторые производители USB-кабелей не соблюдают стандартную схему цветов USB. Самое распространённое отклонение - перестановка белого и зелёного проводов, хотя в редких случаях цветовой код игнорируется полностью. Всегда полезно использовать измеритель прозвонки, чтобы проверить, что ваш USB-удлинитель действительно соответствует цветовому коду USB.

Реализация

Сначала разрежьте разрывной кабель Xbox рядом с концом разъёма Xbox (примерно в 2 дюймах от него) и разрежьте USB-удлинитель рядом с концом female-разъёма. Выбросьте male-половину USB-удлинителя и половину кабеля Xbox с меньшим разъёмом.

Затем диагональными кусачками сделайте надрезы длиной 1/2 дюйма в изоляции каждого кабеля, как показано в кадре 1 на рисунке 4-3. Отогните изоляцию назад, чтобы открыть провода, которые будут защищены плетёным металлическим экраном и металлической фольгой. Отогните металлическую оплётку и фольгу назад и срежьте лишнюю изоляцию и экран. Зачистите концы красного, зелёного, белого и чёрного проводов так, чтобы было видно около 1/8 дюйма голого проводника. (Не зачищайте жёлтый провод в разрывном кабеле Xbox.) Окуните голые концы проводников в немного паяльного флюса. (См. Рисунок 4-3.)

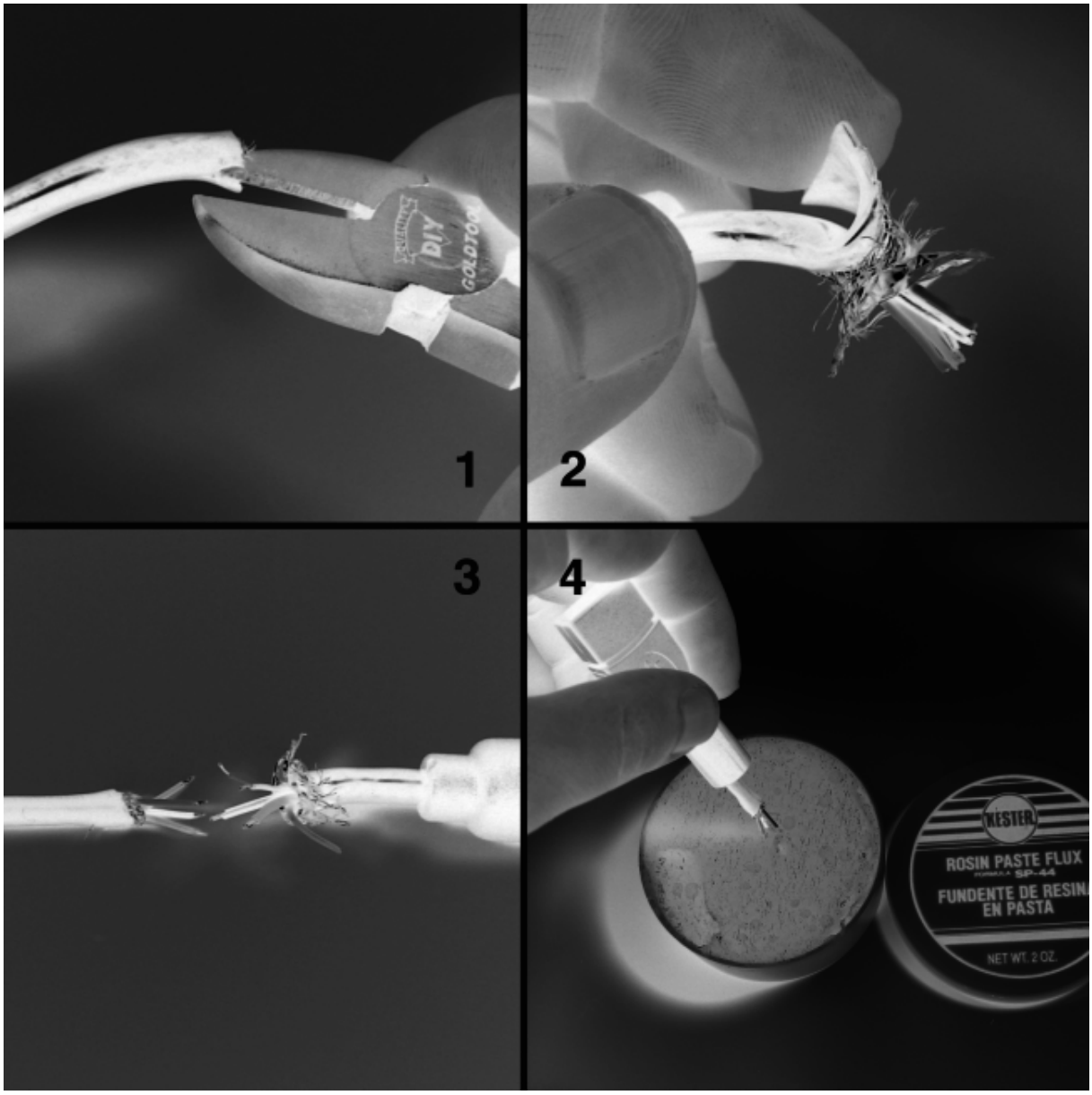


Рисунок 4-3. (1) Сделайте надрезы в изоляции кабеля и (2) отогните изоляцию, чтобы открыть провода и экран внутри. (3) Срежьте лишний экран и изоляцию и зачистите 1/8 дюйма с концов проводов. (Обратите внимание, что жёлтый провод в кабеле разъёма Xbox справа не зачищен.) (4) Наконец, окуните зачищенные концы проводов в паяльный флюс.

Если вас беспокоит прочность или безопасность сборки кабеля-адаптера, наденьте на один из кабелей отрезок термоусадочной трубки длиной 1-1/4 дюйма. (Позже эта трубка будет надета поверх голых паяных соединений и заполнена термоклеем, чтобы сделать соединение прочным.) Экранирование не является обязательным, но без усиления термоусадкой кабель будет не таким прочным; он будет подвержен поломке при повторяющихся изгибах или натяжении.

Продолжайте процесс сборки кабеля, спаивая вместе провода одинаковых цветов, как показано на рисунке 4-4. Пусть друг поможет удерживать кабели на месте, пока вы их паяете, или используйте инструмент «третья рука» (номер заказа Jameco 26690) с зажимами «крокодил», чтобы удерживать кабель на месте. Все паяные соединения должны выглядеть гладкими и блестящими. Сильно потяните за каждое соединение, чтобы проверить, что пайка хорошая. Оберните небольшой кусочек изоленты вокруг открытых соединений, чтобы предотвратить короткое замыкание между оголёнными соединениями. Проверьте кабель, прежде чем переходить к следующему шагу, где соединения будут окончательно заключены в оболочку.

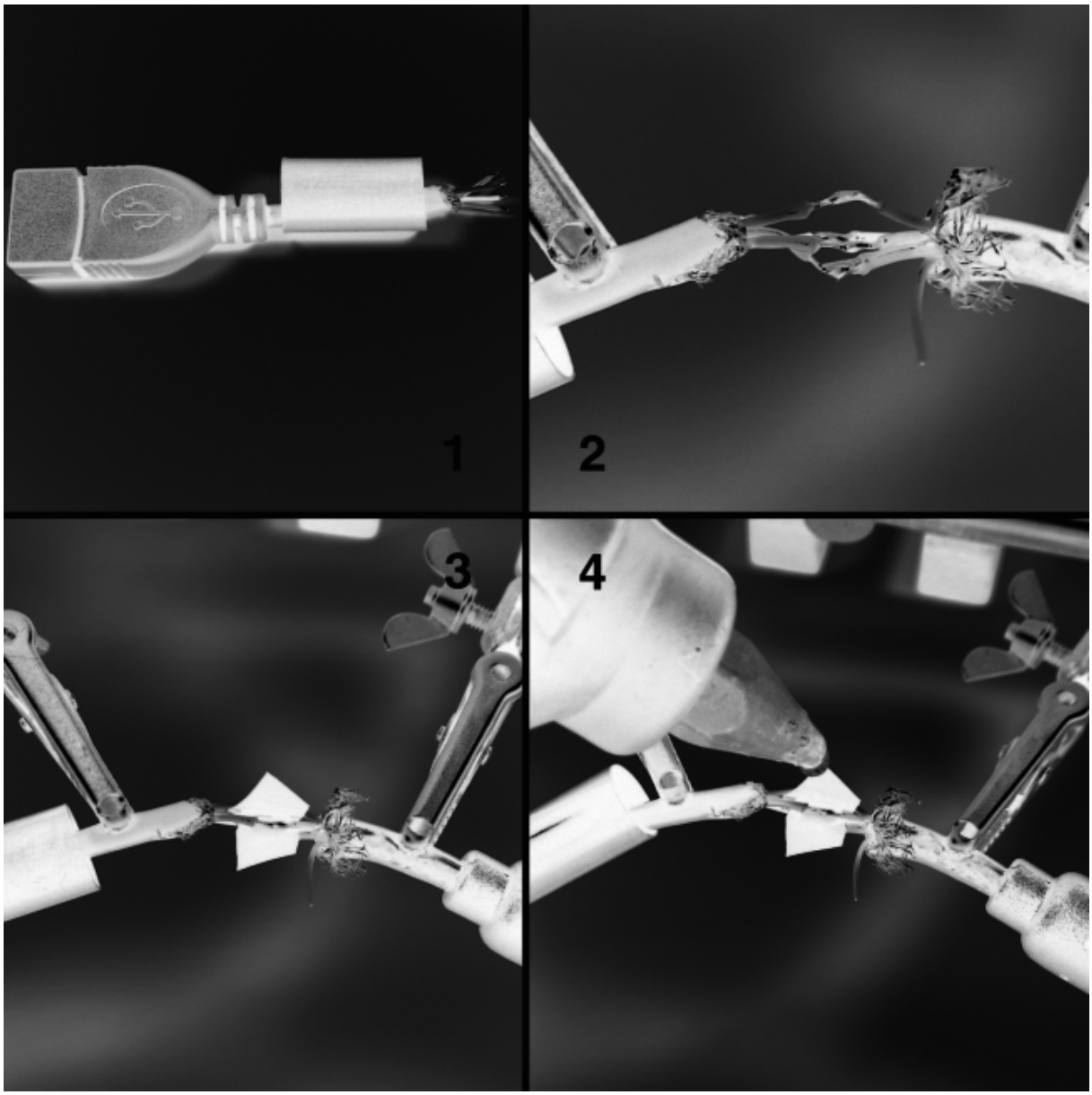


Рисунок 4-4. (1) Перед пайкой наденьте на кабель кусок термоусадочной трубки. (2) Спаяйте провода вместе, цвет к цвету. (3) Оберните провода изолентой, чтобы предотвратить короткое замыкание. Проверьте кабель. (4) Нанесите небольшие капли термоклея на проверенный кабель, чтобы удержать изоленту и провода на месте для следующего шага.

После того как кабель проверен и подтверждён как исправный, пора поместить вокруг паяных соединений прочный кожух для механического усиления, как показано на рисунке 4-5. Нанесите пару небольших капель термоклея поверх открытых паяных соединений, чтобы удержать их на месте и не дать им замкнуться друг с другом, сдвиньте термоусадочную трубку поверх паяных соединений, затем заполните обе стороны трубки термоклеем. Трубка должна усесть от тепла клея и сформировать твёрдый постоянный облегающий кожух поверх соединения. (Термоклей также действует как разгрузка натяжения для соединения, поэтому кабель будет прочным в большинстве обычных условий эксплуатации.)

Хотя кабель полностью работоспособен без обработки термоклеем и термоусадочной трубкой, вы всё равно можете улучшить стабильность соединения, если этих предметов под рукой нет. Изоленту можно аккуратно обернуть вокруг отдельных соединений как импровизированное механическое усиление.

Теперь, когда вы закончили адаптер USB - игровой порт Xbox, главы 11 и 12 описывают некоторые шаги, необходимые для установки Linux на Xbox, чтобы вы могли использовать свой новый адаптер.

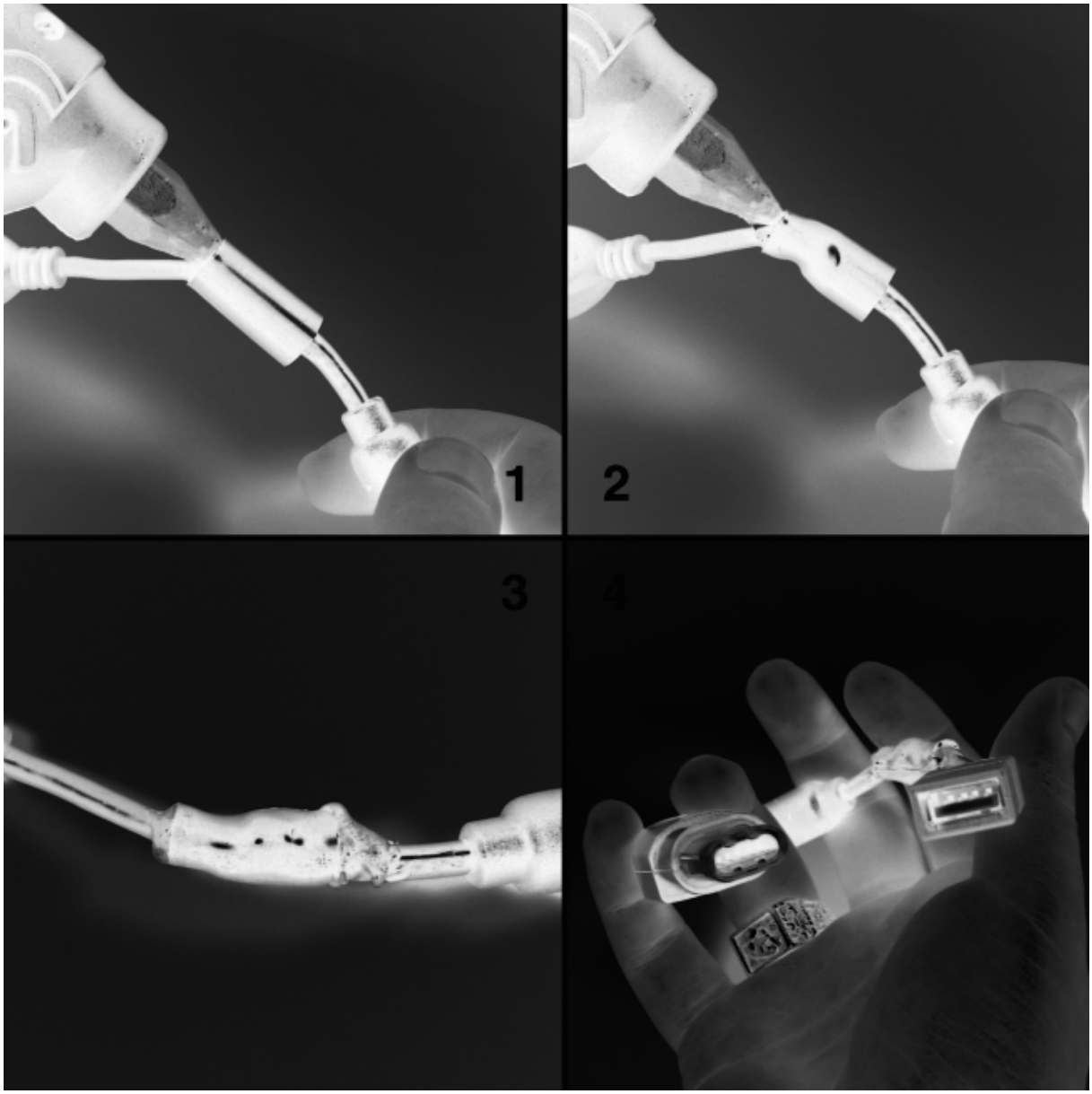


Рисунок 4-5. (1) Наденьте термоусадочную трубку поверх соединения и начните заполнять трубку термоклеем. (2) Термоклей заставит термоусадочную трубку сжаться. (3) Конечный продукт - постоянный облегающий кожух, который выдержит большую часть нагрузок. (4) Фотография конечного продукта, показывающая концы кабелей Xbox и USB.

Глава 5. Замена сломанного блока питания

В несчастливом - и на удивление частом - случае, когда Xbox ломается после истечения его трехмесячной гарантии, единственный официальный способ починить его - заплатить Microsoft за ремонт. Даже самые простые исправления могут стоить больше ста долларов, то есть примерно половину первоначальной цены консоли. В результате я получил множество писем от людей с вопросами о том, как чинить сломанные блоки питания и жесткие диски.

К сожалению, замена сломанного жесткого диска требует обхода системы безопасности Xbox, поскольку уникальный ключ используется для привязки материнских плат Xbox к конкретному жесткому диску. Установка нового жесткого диска потребовала бы modchip, который может перепрограммировать или обойти блокировку жесткого диска. Кроме того, требуется копия установленного на заводе программного обеспечения Xbox, а распространять или копировать ее незаконно даже для целей ремонта. Поэтому тема замены жесткого диска Xbox слишком щекотлива, чтобы обсуждать ее в этом тексте. Читателям рекомендуется поискать в сети любой из многочисленных FAQ по замене жестких дисков.

С другой стороны, блок питания, используемый в Xbox, очень похож на тот, который используется в стандартном PC. Есть много веб-сайтов, где можно купить точные заменители блоков питания Xbox, например Llama.com (www.llama.com/xbox/Repairs/repairs.htm), XboxRepair.com (www.xboxrepair.com) и Firefly-HK (www.firefly-hk.com), или можно попытаться собрать такой блок самостоятельно из стандартного PC-блока питания!

Учитывая частоту отказов блоков питания, я покажу вам, как приспособить стандартный PC ATX-блок питания для Xbox. Подход, описанный в этом разделе, не требует пайки, ценой необходимости переключать дополнительный выключатель, чтобы включить Xbox. Приложение C, «Getting Into PCB Layout», описывает простой проект, который можно собрать, чтобы получить удобство без дополнительных выключателей питания. Конечно, легче купить точный заменитель блока питания для Xbox и использовать части процедуры, приведенной в этой главе, чтобы помочь себе с установкой. Однако в прямой замене меньше образовательной ценности, чем в адаптации. Если вы решите адаптировать ATX-блок питания для использования с Xbox, вы научитесь делать обжимные кабели, а также немного узнаете об электронной теории и о том, как работает Xbox.

Еще одна причина адаптировать стандартный PC ATX-блок питания к Xbox - обеспечить консоли дополнительную мощность. OEM (original equipment manufacturer, производитель оригинального оборудования), или «штатный», блок питания Xbox выдает ровно столько мощности, сколько едва хватает для потребностей Xbox. Подключение дополнительных приводов или вентиляторов к в остальном немодифицированному Xbox может перегрузить OEM-блок питания Xbox и привести к его выгоранию.

Обратите внимание, что на момент написания этой книги была выпущена новая аппаратная ревизия Xbox (известная в сообществе взлома Xbox как «v1.2»), у которой, по-видимому, установлен стандартный разъем ATX-блока питания вместо фирменного разъема блока питания Xbox, описанного далее в этой главе. Прежде чем продолжать процедуру адаптации, убедитесь, что разъем блока питания вашего Xbox соответствует разъему, описанному в этой главе. Разъем блока питания Xbox, предполагаемый в этой главе, имеет двенадцать контактов в один ряд, тогда как более новый разъем блока питания Xbox имеет 20 контактов, расположенных в два ряда по десять. Кроме того, даже несмотря на то что последняя аппаратная ревизия Xbox имеет ATX-подобный разъем, он не обязательно электрически совместим со стандартным ATX-блоком питания. Было бы разумно измерить напряжения на разъеме питания и сравнить их со спецификацией ATX (www.formfactors.org/developer/specs/atx/atx2_1.pdf), прежде чем пытаться соединить стандартный ATX-блок питания с Xbox, имеющим новый ATX-подобный разъем.

Важно. Замена блока питания может подвергнуть вас воздействию опасных напряжений. Перед снятием блока питания всегда отключайте Xbox от настенной розетки и подождите минуту, чтобы накопленный заряд рассеялся. Кроме того, неправильная установка сменного блока питания может привести к постоянному, даже взрывному, повреждению консоли. Выполняйте эту процедуру только в том случае, если вы уверены, что блок питания разряжен и выключен, и если вы готовы принять риск дальнейшего повреждения вашей консоли.

Диагностика сломанного блока питания

Если ваш Xbox испытывает проблемы с включением, сначала необходимо диагностировать проблему и найти источник отказа. Нет смысла заменять блок питания, когда неисправность на самом деле находится внутри консоли или в настенной розетке. Выполните следующие диагностические шаги, чтобы проверить, что виновником действительно является блок питания Xbox, а не что-то другое.

1. Проверьте, что электрическая розетка работает, подключив к ней лампу. Используйте лампу минимум на 100 ватт, чтобы точно имитировать нагрузку Xbox.
2. Визуально осмотрите шнур питания на перегибы и порезы.
3. Проверьте, что вилка шнура питания плотно вставлена в гнездо питания Xbox, и что Xbox все равно не включается, несмотря на эти проверки.
4. Визуально осмотрите внутренность Xbox на следы обугливания или разорванные конденсаторы. Если следы обугливания видны на материнской плате, возможно, вам придется заменить материнскую плату Xbox (то есть купить новый Xbox). Если следы обугливания есть на блоке питания, скорее всего, блок питания был поврежден и можно начинать его замену. (Помните, что отказ блока питания также может повредить материнскую плату, поэтому все еще есть вероятность, что Xbox не заработает после замены блока питания.)
5. Проверьте, что основной разъем блока питания и разъем узла платы передней панели плотно посажены. (Расположение разъема узла платы передней панели показано на рисунке 3-3 в главе 3, «Installing a Blue LED».) Выключатель питания Xbox соединен с материнской платой через разъем узла платы передней панели.
6. Пока Xbox выключен, но все еще подключен к сети, с помощью вольтметра проверьте, что дежурное напряжение 3.3V (3.3VSB) находится в пределах спецификации. Измерьте 3.3VSB, касаясь щупом шестого провода в разъеме блока питания, считая от конца, ближайшего к передней панели, и любого из черных проводов на разъеме питания. Напряжения блока питания можно измерять, вставляя кончики щупов вольтметра в свободное пространство между проводами питания и разъемом питания. (Внутри корпуса разъема питания материнской платы вокруг проводов питания есть металлическая обойма.) Если значение 3.3VSB не находится между 3.14 и 3.47 вольта, возможно, вам нужно заменить блок питания.
7. Нажмите кнопку питания на Xbox, чтобы «включить» его (предположительно, если блок питания сломан, Xbox мало что сделает). Если блок питания шумит или дымит, отключите коробку от сети и переходите к замене блока питания. Если коробка кажется мертвой, измерьте каждое из основных напряжений, поступающих от блока питания. На желтом проводе должно быть напряжение между 11.4 и 12.6 вольта; на красном проводе - между 4.8 и 5.25 вольта; на оранжевом проводе - между 3.14 и 3.47 вольта. (Все эти напряжения отсчитываются относительно черного провода.) Также проверьте, что напряжение на сигнале Power OK (расположен на контакте 12, самом дальнем от передней панели контакте) выше 3.1 вольта.

Если все пункты списка на предыдущей странице проходят проверку, весьма маловероятно, что проблема в вашем блоке питания. Дальше стоит проверить электрическую и механическую целостность выключателя питания (см. главу 3, где объясняется, как снять плату с выключателем питания) и работоспособность материнской платы. Однако если вы действительно наблюдали признаки отказавшего блока питания, читайте дальше.

Замена блока питания

Общая стратегия замены блока питания Xbox состоит в том, чтобы адаптировать стандартный PC ATX-блок питания для использования в Xbox. Вот список оборудования, которое вам понадобится:

- Стандартный ATX-блок питания. Блок питания 1-U поместится в пределах проекции корпуса Xbox, но, вероятно, будет немного слишком высоким, чтобы закрыть корпус.
- (Необязательно) Удлинитель кабеля питания ATX-материнской платы. Удлинители силового кабеля можно купить у многочисленных продавцов, включая PC Power and Cooling (www.pcpowerandcooling.com). Модификация удлинительного кабеля для Xbox вместо кабеля ATX-блока питания позволяет вам снова использовать блок питания в стандартном PC, когда вы будете готовы выбросить свой Xbox.
- Обжимной инструмент. Универсальный обжимной инструмент Molex (артикул Digi-Key WM9999-ND) настоятельно рекомендуется, но он немного дорогой (около $35). Более дешевый обжимной инструмент, например Jameco 159265, можно купить примерно за треть цены, но им труднее пользоваться, и вам, возможно, придется пропаивать обжимы, чтобы добиться нужной прочности соединения.
- Один 12-позиционный корпус разъема с шагом 0.156" (то есть артикул Digi-Key WM2313-ND) или два стыкуемых 6-позиционных корпуса разъема с шагом 0.156" (то есть артикул Jameco 104731). Этот корпус используется как замена разъема питания Xbox.
- Тринадцать обжимных контактов для силового разъема с шагом 0.156" (то есть артикул Digi-Key WM2313-ND или артикул Jameco 78318).
- Два кремниевых выпрямительных диода 1N4001 или лучше в корпусе DO-41 (то есть артикул Digi-Key 1N4001DICT-ND или артикул Jameco 35975).
- Инструмент для снятия изоляции. Подойдет любой стриппер, который может работать с проводом 18 gauge.
- Кусачки. Подойдут любые диагональные кусачки.
- Изолента.

Стратегия

Интерфейс стандартного ATX-блока питания очень похож на интерфейс блока питания Xbox. Xbox требует +3.3V, +5V, +12V, дежурное питание +3.3V, а также два управляющих сигнала: «power OK» и «power on». Сигнал power OK показывает, что выходное питание от блока питания стабильно и правильно отрегулировано, а сигнал Power On является управляющим сигналом от Xbox, который включает и выключает блок питания. Типичный ATX-блок питания имеет выходы +3.3V, +5V и +12V с достаточной силой, чтобы питать Xbox, а также имеет сигнал power OK, совместимый с материнской платой Xbox. Однако ATX-блок питания формирует дежурное напряжение +5V вместо дежурного напряжения +3.3V, а сигнал Power On имеет полярность, обратную той, что у Xbox. Оба этих несовместимых места можно устранить способом, который не требует пайки.

Два последовательно соединенных диода используются, чтобы снизить дежурное напряжение +5V до напряжения чуть меньше +3.6V. Сигнал Power On к ATX-блоку питания по умолчанию находится в состоянии «включено», поэтому он останется неподключенным, делая блок питания всегда включенным, даже если консоль выключена. Это не создает проблем для электроники консоли, но может эстетически смущать. Приложение C, «Getting Into PCB Layout», описывает примерный проект, который можно реализовать, чтобы обойти эти несовместимости более изящным образом. Однако проект, изложенный в приложении, потребует от вас вложить некоторые усилия в виде пайки и разработки платы.

Использование диодов для понижения напряжений

Xbox требует дежурное напряжение питания +3.3V, но ATX-блок питания выдает только дежурное напряжение питания +5V. «Правильным» решением этой проблемы было бы использовать стабилизатор напряжения, который точно преобразует +5V в +3.3V, но цель этого хака - заменить блок питания с минимальным количеством пайки.

Альтернативное решение - использовать два диода, чтобы уменьшить питание +5 вольт до «достаточно близкого» питания +3.6 вольта. Мы можем сделать это, потому что напряжение на проводящем в прямом направлении диоде логарифмически пропорционально току через диод. Другими словами, для большинства токов напряжение на диоде почти постоянно. Оказывается, кремниевые диоды почти единообразно имеют прямое падение напряжения около 0.7 вольта, так что два таких диода последовательно сбросят 1.4 вольта.

Диоды, используемые в этом хаке, 1N4001, способны проводить только 1 ампер тока, поэтому не используйте этот прием в других приложениях, которым требуется большой ток. К счастью, дежурное питание для Xbox должно потреблять лишь крошечный ток, поэтому выгорание диодов не является предметом беспокойства.

В качестве заключительного замечания: напряжение, сбрасываемое диодом, немного колеблется в зависимости от величины тока через него, поэтому не используйте этот прием в приложениях, которым требуются точно стабилизированные напряжения. В применении для Xbox мы подаем напряжение немного выше номинала, но, к счастью, цифровая логика, питающаяся от этого источника, способна переносить это условие.

Процедура

Процедура замены блока питания Xbox состоит из двух частей:

1. Модификация стандартного ATX-кабеля питания в кабель питания Xbox.
2. Удаление старого блока питания и установка нового.

Сборка кабеля питания Xbox

Начните с отрезания существующего разъема материнской платы от кабеля ATX-блока питания, как показано на рисунке 5-1. Вы можете предпочесть выполнить эту модификацию с использованием удлинительного кабеля питания ATX-материнской платы, чтобы сохранить ATX-разъем питания на блоке питания для будущего использования. Процедура идентична для обоих вариантов, но фотографии в этой главе сделаны с использованием удлинительного кабеля ATX-материнской платы.

Рисунок 5-1. Отрежьте разъем от кабеля ATX-блока питания.

Теперь прикрепите обжимные контакты к следующим десяти проводам ATX-кабеля, как показано на рисунке 5-2:

• Один желтый провод
• Три красных провода
• Один оранжевый провод
• Четыре черных провода
• Один серый провод

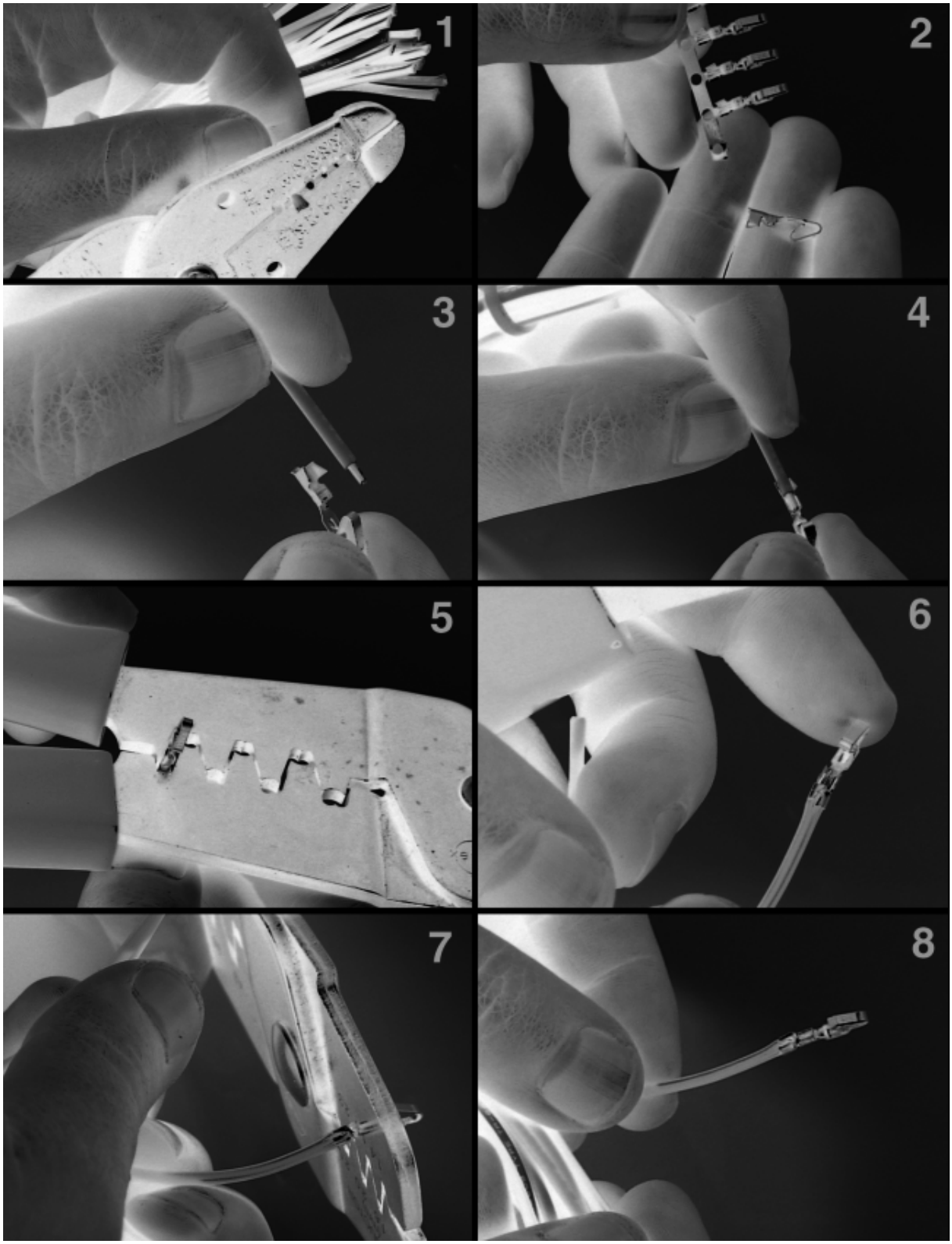


Рисунок 5-2. Прикрепление обжимного контакта к концу провода. (1) Снимите примерно 1/8" изоляции с конца провода. (2) При необходимости отделите новый обжимной контакт от удерживающей полосы. (3 и 4) Вставьте провод в обжимной контакт так, чтобы 1/16" изоляции находилась между более длинной парой обжимных лапок. (5) Обожмите изоляционную часть (более длинную пару) обжимных лапок. (6) Показан провод, у которого обжата только изоляционная часть. (7) Обожмите проводниковую часть обжимных лапок. (8) Готовый обжимной контакт. Проводниковые обжимные лапки должны быть плотно загнуты на оголенный провод для хорошего контакта. Проверьте обжимное соединение, твердо потянув за конец контакта.

Если вы используете более дешевый обжимной инструмент, у вас могут возникнуть трудности с выполнением достаточно прочного обжимного соединения. В этом случае завершите соединение, припаяв обжимной контакт к проводу. При пайке используйте обильное количество тепла, иначе припой не полностью проникнет в провод и обжимной контакт. Паяльник должен находиться в контакте с соединением примерно пять секунд до и после нанесения припоя.

На фиолетовом проводе (дежурный провод +5V) прикрепите два диода последовательно между концом провода и обжимным контактом. Процедура, показанная на рисунке 5-3, использует части обжимных контактов для соединения диодов, поэтому пайка не требуется. (Обратите внимание, что диоды являются полярными устройствами: они не будут проводить электричество, если установлены задом наперед. Диоды должны быть установлены катодами (концом с нанесенной полосой) в сторону материнской платы.)

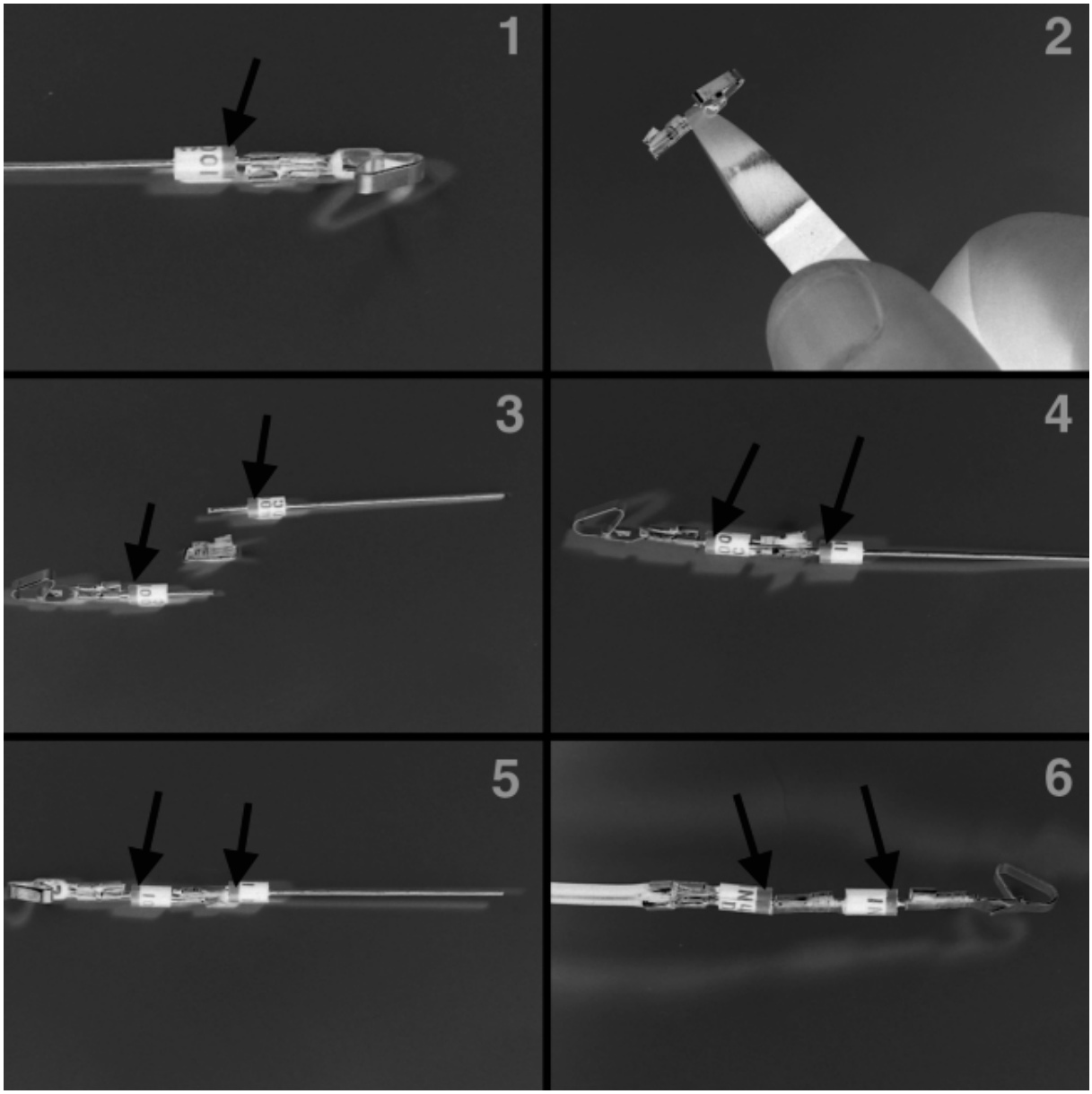


Рисунок 5-3. Прикрепите два диода последовательно между концом фиолетового провода и обжимным контактом, который входит в разъем питания. Стрелки показывают правильную ориентацию поляризующей линии, нанесенной на конец диода. Эта процедура израсходует в общей сложности три обжимных контакта. (1) Прикрепите один диод к обжимному контакту, расположив поляризующую полосу рядом с обжимным контактом. (2) Разрежьте обжимной контакт пополам, чтобы удалить пластинчатый контакт. (3 и 4) Разместите диоды внутри обжимной части разрезанного контакта, обращая внимание на полярность диодов. (5) Диоды показаны после обжима. (6) Прикрепите диоды к концу фиолетового провода, используя ту же процедуру со вторым разрезанным обжимным контактом.

Затем обмотайте все неиспользуемые провода на ATX-кабеле изолентой, чтобы предотвратить любые случайные короткие замыкания, которые повредят блок питания и, возможно, Xbox. Обязательно также обмотайте диоды изолентой. (См. Рисунок 5-4.)

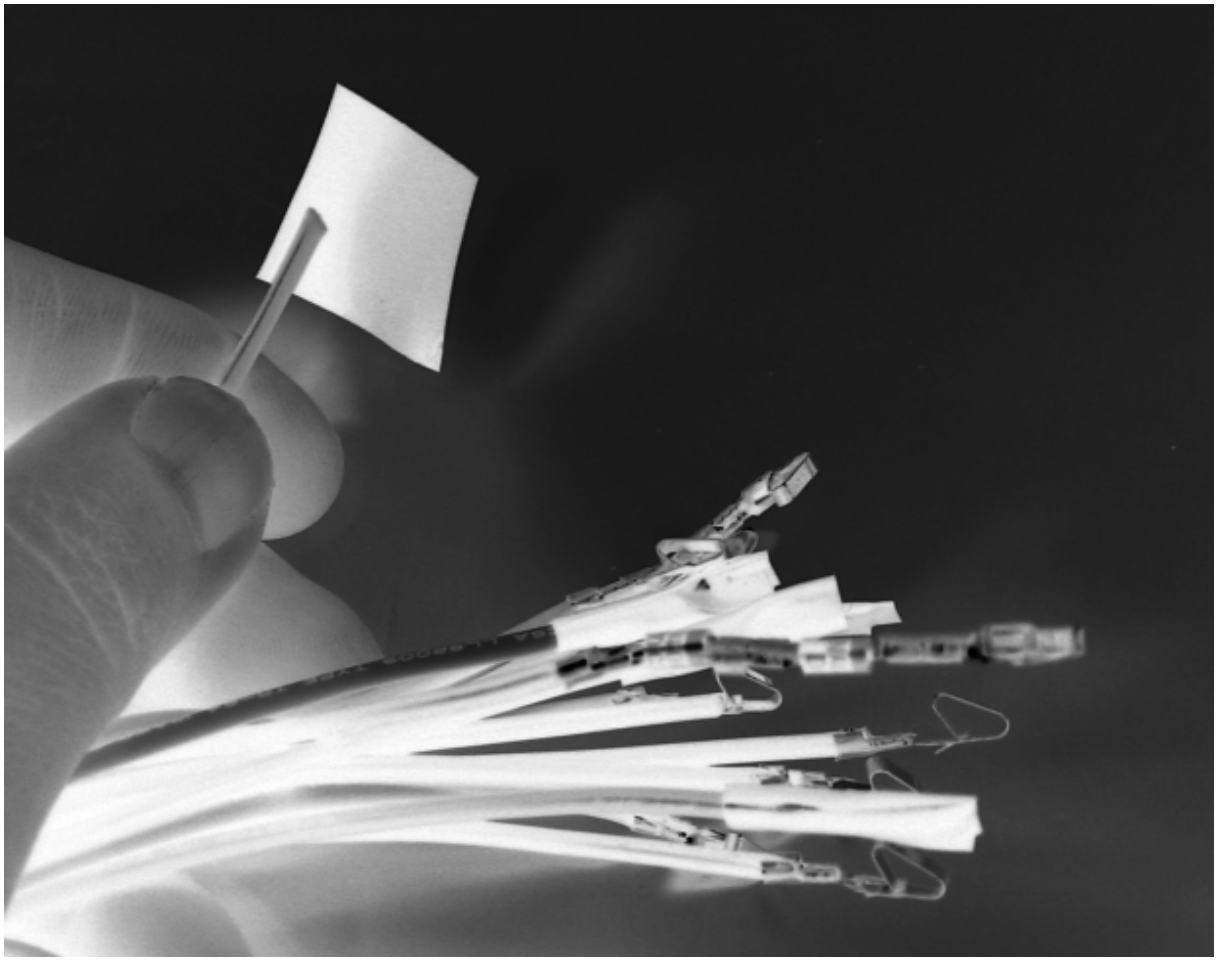


Рисунок 5-4. Обмотайте неиспользуемые провода и диоды изолентой, чтобы предотвратить случайное короткое замыкание.

Наконец, вставьте готовые обжимные контакты в корпус разъема 0.156". Обжимные контакты зафиксируются внутри корпуса, когда будут полностью и правильно вставлены. (См. Рисунок 5-5.) Вставляйте провода в порядке, указанном в рисунке 5-1.

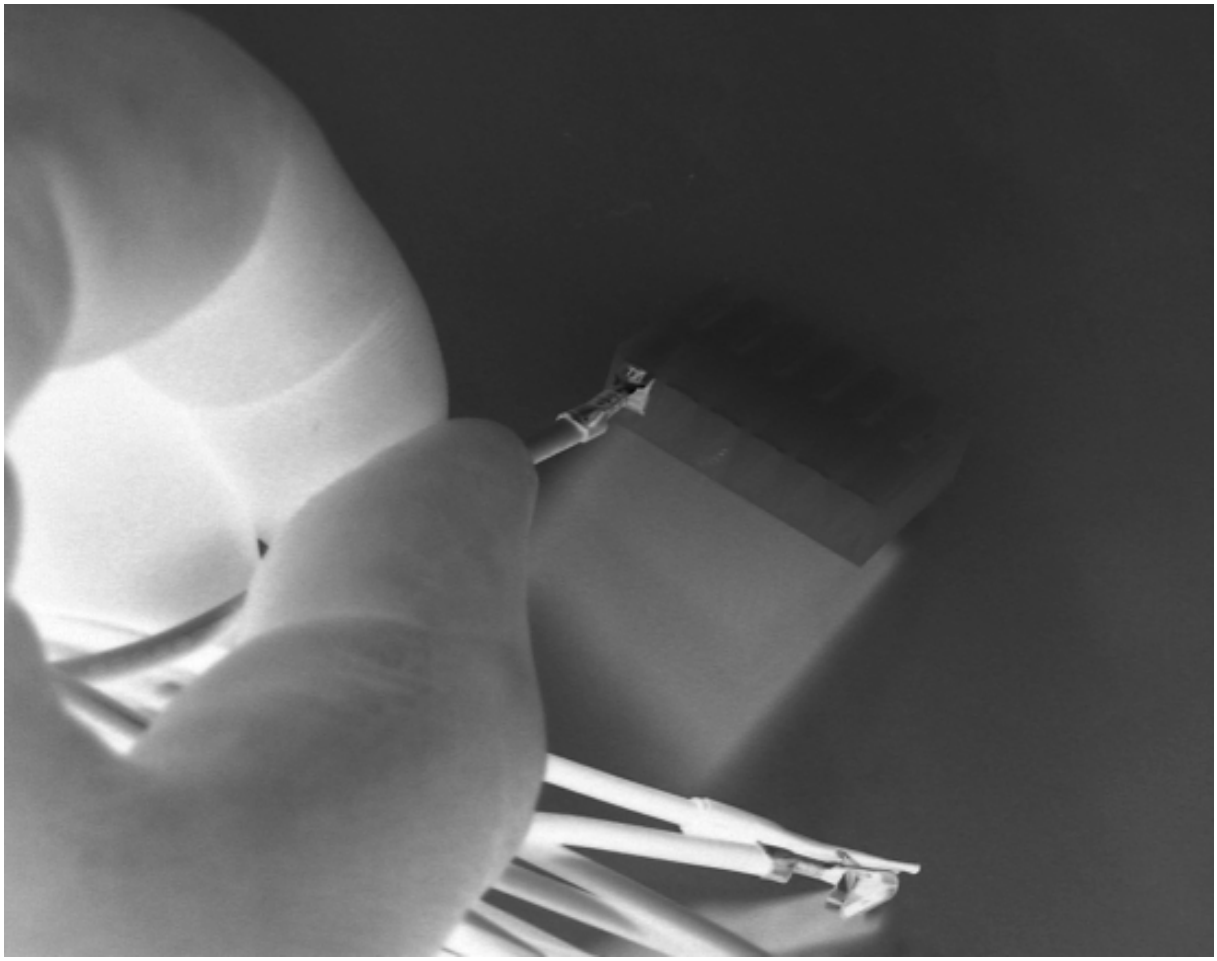


Рисунок 5-5. Вставка обжимного контакта в корпус разъема.

Некоторые продавцы не продают более крупный 12-позиционный корпус разъема. В этом случае используйте два 6-позиционных корпуса разъема и уделите особое внимание порядку, который вы выберете для стыковки разъемов. Также обратите внимание на расположение контакта 1 относительно поляризующей кромки разъема. Контакт 1 расположен в верхней части разъема, когда поляризующая кромка находится слева и вы смотрите на разъем со стороны, с которой вставляются провода (вид сверху). Поскольку очень легко перевернуть разъем и тем или иным способом все это обратить, используйте существующий разъем Xbox как образец. Желтые, красные, оранжевые и черные провода должны совпадать при сравнении друг с другом.

Таблица 5-1. Таблица разводки для подключения кабеля ATX-блока питания к разъему питания Xbox (вид со стороны входа проводов, с поляризующим ребром слева).

Контакт	Цвет
1	Yellow
2	Red
3	Red
4	Red
5	Orange
6	Diodes + Violet
7	Black
8	Black
9	Black

Контакт	Цвет
10	Black
11	Empty
12	Gray

Поляризующее ребро находится слева.

Дважды проверьте свою работу после завершения сборки кабеля блока питания, поскольку любая ошибка может привести к постоянному, непоправимому повреждению консоли. Рисунок 5-6 показывает, как должен выглядеть готовый узел разъема. При наличии вы можете использовать кабельную стяжку, чтобы собрать неиспользуемые провода в пучок, чтобы они не мешали и случайно не закоротили или не были повреждены.

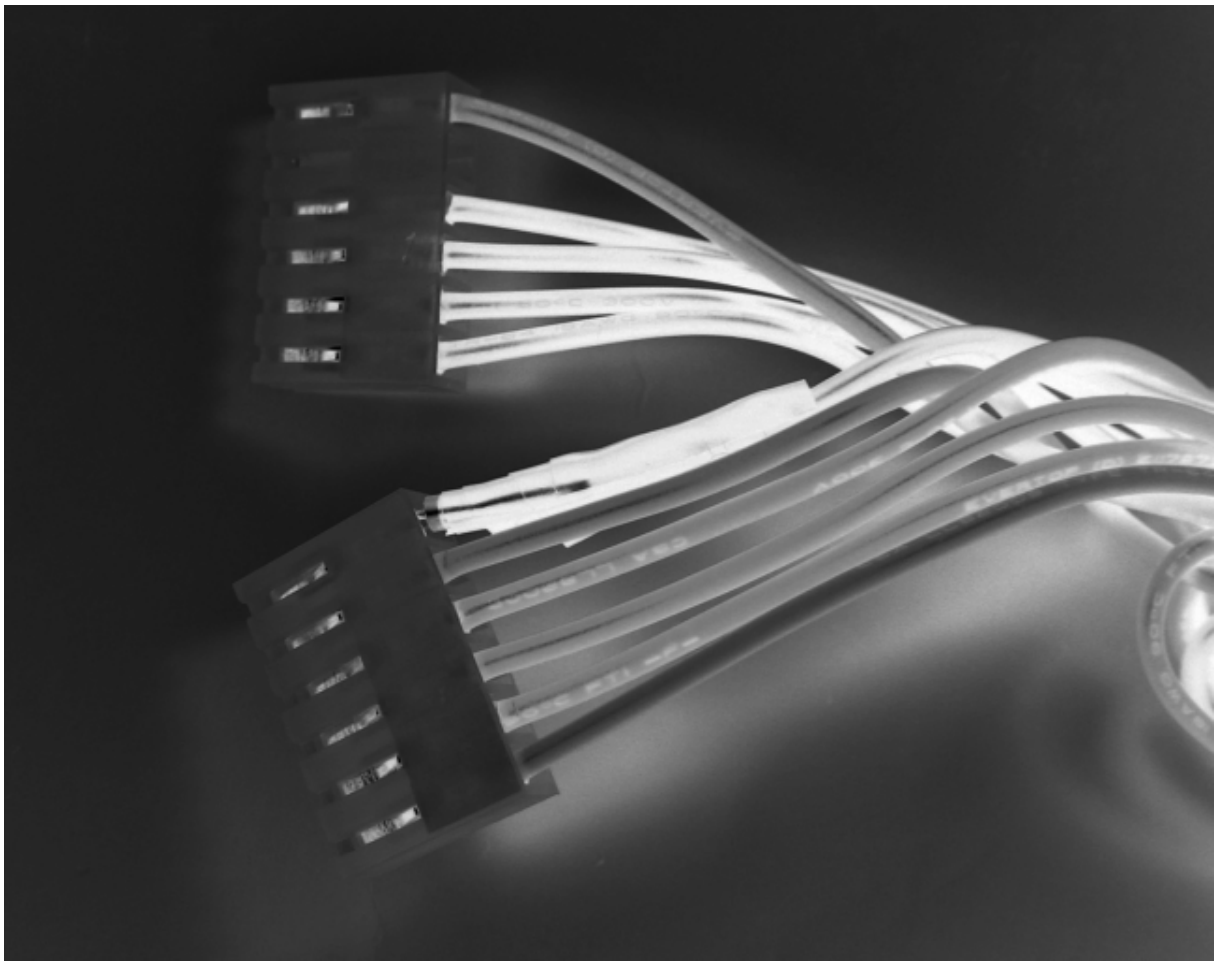


Рисунок 5-6. Готовый узел кабеля.

Установка сменного блока питания

Теперь, когда мы подготовили сменный блок питания, нужно заменить им старый, сломанный. Сначала снимите верх корпуса Xbox, как описано в главе 1, «Voiding the Warranty», затем отсоедините разъем питания жесткого диска и поднимите жесткий диск из корпуса. Вам не нужно отсоединять серый ленточный IDE-кабель, подключенный к жесткому диску.

NB. На этом этапе проверьте, что Xbox отключен от сети и что у него была возможность постоять как минимум минуту, чтобы рассеять любой сохраненный заряд в блоке питания. Работа с Xbox, когда он подключен к сети или вскоре после того, как он был отключен, чрезвычайно опасна. Xbox нанесет неприятный, возможно смертельный, удар током, если вы коснетесь любой части блока питания голыми руками до того, как он рассеет сохраненный заряд.

1. Отключите разъем блока питания Xbox, взявшись за весь пучок силовых проводов и твердо потянув за кабель, удерживая коробку другой рукой. При снятии кабеля следите за острыми металлическими краями корпуса и радиаторов.

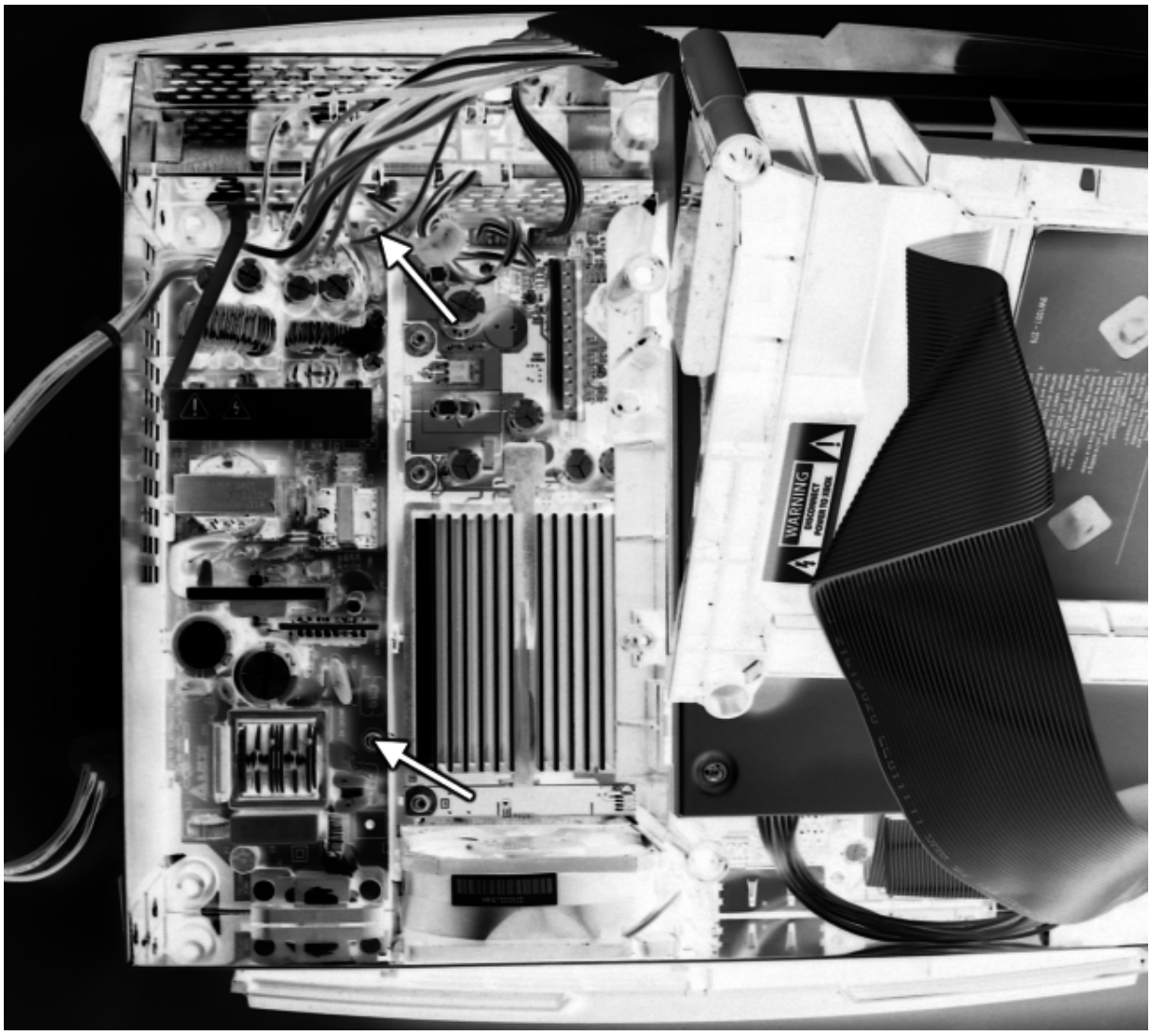


Рисунок 5-7. Расположение двух крепежных винтов блока питания.

2. Выкрутите два винта T-10 torx, которые удерживают блок питания Xbox на месте. (См. Рисунок 5-7.)
3. Выньте блок питания из корпуса Xbox, сначала приподняв тот конец блока питания, который ближе к передней части Xbox.
4. Сравните созданный вами кабель с кабелем блока питания Xbox, обязательно совмещая поляризующие ребра. Красные, желтые, оранжевые и черные провода должны совпадать между двумя разъемами. (Эта проверка поможет убедиться, что вы не повредите Xbox из-за ошибки разводки на кабеле.)
5. Подключите ваш узел кабеля ATX-блока питания к разъему питания Xbox. Желтый провод должен совпадать с контактом, ближайшим к передней части Xbox. Помните, что ничто не

мешает вам вставить кабель питания со смещением на один или два контакта. Если вы видите оголенный контакт на разъеме питания, вы вставили кабель со смещением на один контакт. Дважды проверьте это состояние, потому что оно может навсегда повредить аппаратное обеспечение Xbox и/или блок питания.

6. Подключите жесткий диск Xbox к одному из разъемов питания дисковых приводов ATX-блока питания. Жесткий диск использует разъем, идентичный стандартному разъему питания дискового привода PC, поэтому никакой модификации не требуется.
7. Проверьте, что никакие провода не замкнуты на корпус и не попали в лопасти охлаждающих вентиляторов. Теперь вы готовы включить Xbox.

Работа со сменным блоком питания

Большинство ATX-блоков питания поставляются с выключателем питания на них. Переведите этот выключатель в положение off, затем подключите ATX-блок питания к сети, а затем переведите выключатель питания в положение on. В этот момент ATX-блок питания подаст питание на Xbox, даже несмотря на то что системный контроллер Xbox считает коробку выключенной. В результате охлаждающие вентиляторы внутри Xbox должны вращаться. Теперь нажмите выключатель питания на передней панели Xbox. В этот момент Xbox должен включиться нормально. Если так, поздравляю!

> Важно. В некоторых версиях Xbox отсутствует вентилятор радиатора для GPU. Если на материнской плате вашего Xbox нет вентилятора, установленного непосредственно на одном из чипов, значит, у вас именно такой Xbox. Xbox без вентилятора радиатора для GPU склонны к перегреву при определенных условиях, и для надежной долговременной работы они должны эксплуатироваться с дисковыми приводами, установленными над материнской платой. Нижние стороны дисковых приводов образуют систему воздушных каналов, которая направляет воздух от основного вентилятора корпуса Xbox поверх радиатора GPU. (Ваш кабель ATX-блока питания не позволит дисковым приводам установиться вровень с корпусом, но это не является серьезной причиной для беспокойства с точки зрения направления воздушного потока.)

Рисунок 5-8. Низкопрофильный 1U ATX-блок питания, подключенный к Xbox с использованием модифицированного удлинительного кабеля ATX-блока питания.

Когда вы будете готовы выключить Xbox, вы можете просто щелкнуть выключателем питания на ATX-блоке питания. Или вы можете сначала использовать выключатель питания на передней панели Xbox, чтобы выключить питание Xbox, а затем выключить ATX-блок питания.

Советы по отладке

Если после замены блока питания Xbox не включился правильно, проверьте кабель блока питания, используя контрольный список в разделе в начале этой главы, который называется «Диагностика сломанного блока питания». Самая вероятная проблема, с которой вы столкнетесь, - плохое обжимное соединение или плохо либо неправильно прикрепленные диоды. Плохое обжимное соединение также может привести к прерывистой работе, при которой Xbox включается, но часто зависает.

Если Xbox включается, но по какой-то причине останавливается во время последовательности включения, см. Рисунок 3-1 в разделе «Debugging» в конце главы 3 для списка возможных проблем и их причин. Приложение E, «Debugging: Hints and Tips», содержит более глубокое обсуждение приемов и методологии отладки.

Если Xbox работает правильно, но иногда зависает, возможно, у вас Xbox без вентилятора радиатора над GPU. См. примечание Caution на противоположной странице, описывающее эту проблему. Чтобы исправить эту проблему, вам, возможно, понадобится добавить дополнительный вентилятор или улучшить существующую систему воздуховодов с помощью листа бумаги и клейкой ленты.

Глава 6. Лучшая игра Xbox: взлом безопасности

Следующий шаг за пределами модификации и тонкой настройки аппаратуры Xbox - это получение контроля над аппаратурой Xbox. К сожалению, получить контроль над аппаратурой не так легко, как можно подумать. Проектировщики Xbox вложили много мысли в защиту аппаратуры от изощренных программных атак, а также от большинства простых аппаратных атак. Механизмы безопасности Xbox являются артефактом ее архитектуры управления цифровыми правами.

> NB. В принципе, применение аппаратуры для целей «fair-use», например для запуска собственных homebrew-программ, не должно быть незаконным. Однако связь между fair use, защищенной аппаратурой и относительно новыми законами об обходе средств контроля авторских прав все еще неясна. Глава 12, «Caveat Hacker», обсуждает юридические вопросы взлома подробнее.

Есть много способов обойти меры безопасности Xbox. В этой главе и в главе 8, «Reverse Engineering Xbox Security», я рассказываю историю своих приключений по картированию системы безопасности Xbox. Я пишу не только об успехах, но и о неудачах, с которыми столкнулся, чтобы вы могли учиться на моем опыте. Глава 9, «Sneaking in the Backdoor», объясняет некоторые подходы, использованные другими для обхода мер безопасности Xbox. Глава 7, «A Brief Primer on Security», дает базу, необходимую, чтобы оценить главы 8 и 9.

Первые встречи с параноидальной конструкцией

Когда Xbox была анонсирована весной 2000 года, по сообществу энтузиастов аппаратуры прокатилась волна возбуждения. Причиной этого возбуждения был не только игровой потенциал Xbox, но и ее потенциал как высокопроизводительного сетевого PC архитектуры x86 по доступной цене $300. Снижения цены через несколько месяцев после ее появления с тех пор опустили стоимость Xbox ниже $200. Сходство Xbox с x86 PC означало, что огромная база существующих приложений и знаний могла, в теории, быть легко перенесена на консоль.

Впервые я заглянул внутрь Xbox в конце ноября 2001 года, когда моя девушка (теперь невеста) подарила мне ее как ранний рождественский подарок. Я немедленно взялся за дело. Чтобы получить контроль над аппаратурой Xbox, первая задача - извлечь boot ROM и проанализировать его содержимое. Вспомните из обсуждения архитектуры Xbox в главе 2, что boot ROM Xbox содержит весь код для установления операционной среды Xbox.

Слямзить ROM

Тип ROM, используемый в Xbox, - это электрически стираемая и программируемая разновидность, известная как FLASH ROM. FLASH ROM обычно выпускается в одном из нескольких типов корпусов, и Xbox использует один из самых популярных корпусов, TSOP (Thin Small Outline Package). Он расположен в секторе U7 на верхней стороне материнской платы Xbox, а позиционное обозначение детали - U7D1. Корпус TSOP очень легко узнать, потому что это один из немногих корпусов микросхем, который имеет прямоугольную форму и выводы только на узких краях корпуса. Большинство других корпусов размещают выводы на длинном крае или на всех краях, чтобы максимизировать связность, но FLASH ROM имеет относительно низкие требования к I/O на площадь кремния. Быстрая проверка базового номера детали, 29F080, через поисковую систему Web подтверждает, что эта деталь действительно является 8 Mbit FLASH ROM.

Есть несколько приемов, которые можно использовать, чтобы считать (snarf) содержимое FLASH ROM. Подход без пайки - купить тестовый зажим, который защелкивается на FLASH ROM, и считать его содержимое, подавая питание и управляя ROM через тестовый зажим, в то время как остальная часть Xbox выключена. Подходящий тестовый зажим для этой цели можно приобрести у Emulation Technology, www.emulationtechnology.com. (У подхода с перехватом через тестовый зажим есть несколько проблем, самая большая из которых - возможность навсегда повредить микросхемы, подключенные к FLASH ROM, которые не получают питание через тестовый зажим. Однако в случае Xbox это, похоже, не является проблемой, и те, кто пробовал этот подход, добились успеха. [1] Изначально я не выбрал этот подход, потому что не хотел рисковать повреждением материнской платы и потому что не мог позволить себе тестовый зажим за $300, необходимый для этой работы.)

Другой изобретательный подход - припаять провода к тестовым точкам вокруг FLASH ROM, чтобы подслушивать Xbox, когда он считывает содержимое ROM. Подслушивание можно выполнить либо подключением проводов к специальной плате, которая может интерфейсироваться с ROM, либо использованием логического анализатора для захвата данных по мере доступа к ним CPU Xbox. Последний подход также был успешно использован, и фактически некоторые back door в последовательности загрузки Xbox были обнаружены как следствие этой методологии. [2] Я решил не использовать и этот подход, поскольку у меня не было логического анализатора, когда я получил свой первый Xbox, а припаивание всех проводов может быть очень утомительным, сложным и подверженным ошибкам. Мой подход был более традиционным: просто снять FLASH ROM и вставить его в считыватель ROM. Я также поставил на материнскую плату панельку, чтобы будущие снятие и программирование ROM были очень быстрыми и надежными.

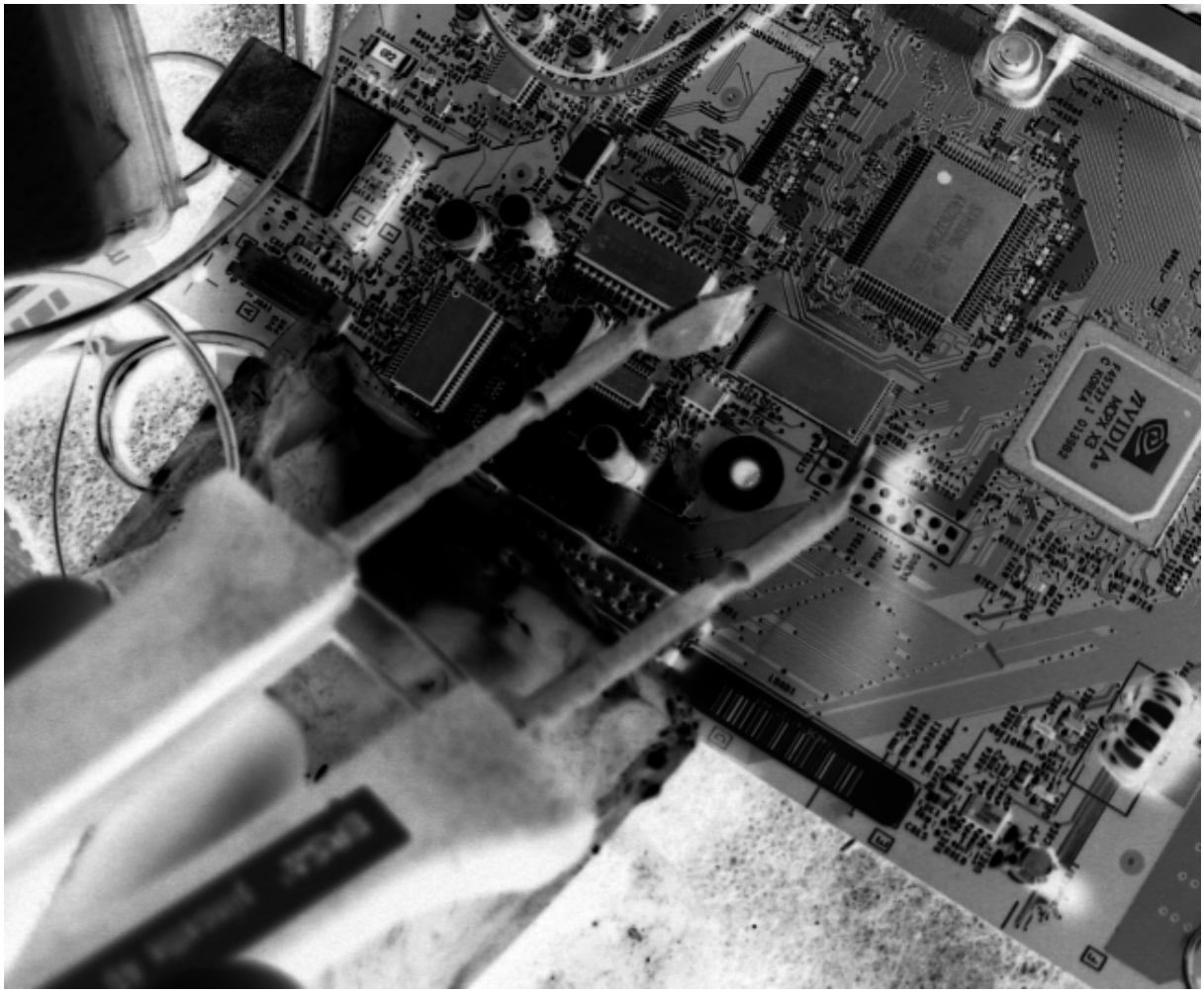


Рисунок 6-1. Снятие Xbox FLASH ROM паяльником-пинцетом.

Снятие FLASH ROM способом, который сохраняет целостность его выводов с мелким шагом, просто, если у вас есть правильные инструменты, и почти невозможно с неправильными инструментами. Ключ в том, чтобы нагреть все выводы FLASH ROM одновременно; как только достигнут равномерный нагрев, FLASH ROM прямо отвалится от материнской платы. Очевидно, стандартный паяльник карандашного типа не сможет нагреть все выводы одновременно. Правильный инструмент для этой работы - паяльник в стиле «щипцов» или «пинцета», как показано на рисунке 6-1 ниже. Эти паяльники имеют два нагревательных элемента, поэтому могут нагревать обе стороны микросхемы одновременно. Кроме того, паяльник должен иметь лопатообразное жало, достаточно широкое, чтобы нагреть длину микросхемы за один раз.

Паяльник с такими возможностями может стоить немало (сотни долларов), но это оправданное вложение, поскольку он оказывается полезным в самых разных ситуациях. Я использую паяльник Ersa SMT Unit 60A, который купил с хорошей скидкой на стенде торговой выставки, и он быстро окупил себя через несколько работ по сборке плат, которые я взял на стороне, пока заканчивал свою степень. Более доступный паяльник от Xytronic можно купить через Jameco (#168410) примерно за $70, но я им не пользовался, поэтому не могу поручиться за его качество. Другой бюджетный подход, очень простой и прямолинейный, - использовать сплав для выпайки, как описано в приложении B, «Soldering Techniques». (Обратите внимание, что подходящая панелька для ROM [3] относительно дешева - меньше $20 - хотя ее установка действительно требует твердой руки и какого-либо оптического увеличительного устройства.)

После того как ROM снят, а его выводы очищены и осмотрены, его содержимое можно считать в считывателе ROM. Конечно, считыватели ROM можно купить, но сборка собственного всегда

является хорошим учебным опытом. Немного почитать о программаторах ROM, которые я собирал, можно на моем веб-сайте, http://www.xenatera.com/bunnie. Мой оригинальный программатор Flashburner [4] - простое устройство, которое легче понять и собрать, чем его вторая ревизия [5], но оно менее мощное.

Однако если ваша цель - считывать ROM как можно быстрее, просто сразу купите считыватель ROM. Хороший считыватель ROM - важный инструмент в наборе любого серьезного аппаратного хакера. Needham's Electronics (http://www.needhams.com) делает отличную линейку программаторов/считывателей ROM, подходящих для широкого диапазона бюджетов.

Встреча с Microsoft

После извлечения содержимого ROM следующий шаг - поделиться его содержимым с другими хакерами для анализа. Или нет? В течение двенадцати часов после публикации содержимого ROM на моем веб-сайте я получил звонок от инженера Microsoft с вежливой просьбой удалить их защищенное авторским правом содержимое с моего веб-сайта. Конечно, я немедленно удалил их содержимое со своего веб-сайта; мне следовало лучше понимать это до того, как я вообще его опубликовал.

Это первое столкновение с Microsoft было отрезвляющим предупреждением, что reverse engineering Xbox не будет похож на любой другой проект reverse engineering бытового прибора. Есть законы, которые защищают аспекты reverse engineering, и огромный массив закона об авторском праве, который защищает владельца интеллектуальной собственности (IP). Совместный reverse engineering Xbox при уважении прав Microsoft - это юридическое минное поле.

С одной стороны, Microsoft должна иметь возможность инвестировать в продукт и рисковать в надежде на прибыль. Однако прибыльность не гарантирована законом. Например, продажа консолей с огромным убытком, как это сделала Microsoft, в надежде продавать программное обеспечение, чтобы компенсировать разницу (как «loss leader»), - рискованное предложение, и закон не гарантирует, что Microsoft в конце концов должна выйти вперед. С другой стороны, мы, как хакеры, имеем право возиться («fair use») с аппаратурой, купленной за наши собственные тяжело заработанные деньги, и если Microsoft хочет в сущности продавать нам PC с огромной скидкой, это нормально. Купим ли мы достаточно игр (около десяти или больше), чтобы компенсировать убытки Microsoft на Xbox, полностью зависит от бизнес- и маркетинговой стратегии Microsoft.

На мой взгляд, большое отношение убытков Microsoft к выручке - некоторая аномалия в этой индустрии. Sony и Nintendo примерно выходят в ноль по стоимости аппаратуры своих консолей. Кроме того, операторы сотовой связи часто продают свои телефоны с убытком, сопоставимым с убытком Xbox, но требуют, чтобы абонент заключил контракт, гарантирующий возврат стоимости телефона; нарушение контракта подразумевает штрафы за расторжение. Возможно, это отражение уверенности Microsoft в бизнес-модели Xbox Live.

Где-то посередине всего этого находится взаимодействие криптографических механизмов защиты авторского права и права на fair use. Оказывается, Xbox широко использует криптографию для обеспечения защиты от копирования, а также политик использования консоли, что приводит нас к Digital Millennium Copyright Act of 1998 (DMCA), относительно новому, непроверенному массиву закона. При малом количестве установившихся судебных прецедентов и обилии серой зоны между буквами закона вы, как хакер, должны оценивать потенциальную ответственность, с которой можете столкнуться. Глава 12, «Caveat Hacker», подробнее исследует юридические вопросы взлома в новом тысячелетии.

Анализ содержимого ROM

Получив отпор от Microsoft, но все еще имея содержимое ROM на руках, я приступил к его анализу. Можно было бы ожидать, что boot ROM содержит процедуру аппаратной инициализации, за которой следуют инструкции, загружающие операционную систему, и, возможно, сам код операционной системы. Но с чего начать?

Программу внутри ROM можно представить как клубок пряжи: как только вы найдете начальную точку нити, дальше дело лишь во времени и упорстве, пока вы не размотаете клубок до сердцевины.

К счастью, начальная точка процессора Pentium в Xbox очень хорошо документирована Intel. При включении питания процессор начинает выполнять код в специальном жестко заданном месте, называемом reset vector. Этот reset vector находится по адресу `0xFFFF.FFF0`, рядом с верхом памяти. Посмотрим на данные, содержащиеся по этому адресу (в шестнадцатеричном виде):

```
0xFFFF.FFF0 EBC6 8BFF 1800 D8FF FFFF 80C2 04B0 02EE
```

Первые два байта, `EBC6`, являются инструкцией перехода к месту `0xFFFF.FFB8`. Первый байт, `EB`, - это конкретный opcode для «jump, short, relative, displacement relative to next instruction»; второй байт, `C6`, - это 8-битное знаковое смещение перехода. Другими словами, первое, что делает процессор, - переходит в другое место, что делает всякая загрузочная программа, поскольку в reset vector у вас есть только 16 байт разбега перед тем, как вы выпадете за верхний конец памяти. Поскольку этот код типичен для reset vector, его допустимо перепечатать здесь в образовательных целях.

Следующий кусок кода - это часть, которая инициализирует состояние GDT (Global Descriptor Table) и IDT (Interrupt Descriptor Table) процессора. GDT и IDT настраивают схему управления памятью процессора и схему обработки прерываний. Вам не нужно понимать точно, что делают эти регистры, но если вам любопытно, Intel «IA-32 Intel Architecture Software Developer's Manual, Volume 3: System Programming Guide» подробно объясняет функцию этих регистров. Это руководство доступно на разработческом веб-сайте Intel, http://developer.intel.com.

После настройки этих регистров процессор выталкивается в protected mode и переходит к `0xFFFF.FE00` - области ровно на 512 байт ниже верха памяти - и именно здесь начинается интересное. После короткого фрагмента кода, который настраивает сегментные регистры, выполняется программа, называемая интерпретатором jam table (в сообществе Xbox также известная как интерпретатор X-Code). Jam table - это отраслевой жаргон для таблицы значений, содержащей opcodes для чтений, записей и простых операций принятия решений, используемых в контексте аппаратных инициализаций. Для инициализации типичного PC требуются сотни операций, и jam tables помогают справляться с этой сложностью, не раздувая базу основного кода инициализации. Использование jam tables также помогает сделать инициализацию более гибкой и способной иметь дело с настраиваемыми пользователем аппаратными параметрами, например типом и объемом установленной памяти. В случае Xbox интерпретатор jam table начинает выбирать opcodes jam table из места рядом с нижней частью FLASH ROM. (Имейте в виду, что opcodes, реализованные интерпретатором jam table, довольно мощны; используя opcodes jam table, можно записывать и читать данные из любого места в Xbox.)

Как только интерпретатор jam table выполняет терминальный opcode, процессор очищает MTRRs (Memory Type Range Registers, используемые для объявления кэшируемости различных областей памяти) и начинает расшифровывать область памяти 16 kB, начинающуюся с `0xFFFF.A000`. Шифр, используемый для расшифровки этой области памяти, выглядит очень похожим на RC-4, с некоторыми тонкими отличиями. Листинг 6-1 показывает шифр, восстановленный reverse engineering в C-код, с помощью инструмента IDA Pro корпорации Data Rescue (подробнее об этом инструменте в следующих нескольких главах).

```
// key initialization routine
unsigned char K[256]; // 0xFFFFC80 in flash
unsigned char S[256]; // 0x10000 in SDRAM
for( i = 0; i < 256; i++ ) {
  S[i] = i;
}
j = 0;
for( i = 0; i < 256; i++ ) {
  // RC-4 would do j = (j + K[i] + S[i]) % 256
  j = (j + K[i] + S[j]) % 256;
  // swap S[i], S[j]
  temp = S[i];
  S[i] = S[j];
  S[j] = temp;
}

// decryption routine
unsigned char cipherText[16384]; // 0xFFFFA000 in FLASH
unsigned char plainText[16384];  // 0x400000   in SDRAM
for( index = 0x4000, i = 0, k = 0; index > 0; index-- ) {
  // xbox version
  t = (S[i] ^ cipherText[k]) % 256;
  plainText[k] = t;
  // swap( S[i], S[t] );
  temp = S[i];
  S[i] = S[t];
  S[t] = temp;
  i = (i + 1) % 256;
  k++;
}
```

Листинг 6-1. Декомпиляция фиктивного шифра, найденного во FLASH ROM.

Данные, расшифрованные этим шифром, на самом деле являются блоком кода, который выполняется в конце процесса расшифровки, но здесь что-то идет очень неправильно. Расшифрованный код - мусор. Он не работает.

Приемы декодирования адресов памяти

Существует ряд приемов, которые можно использовать, чтобы области памяти каким-либо образом выглядели иначе, чем указывало бы их физическое представление. Два приема, относящиеся к анализу последовательности загрузки Xbox, - это aliasing и overlaying.

Адреса памяти являются aliased, когда два адреса указывают на одно и то же место памяти; обычно это достигается игнорированием нескольких битов адреса. Чтобы проиллюстрировать aliasing, рассмотрим систему, использующую 3-битный адрес. В 3-битной системе доступно только 2^3 = 8 уникальных адресуемых мест: 000, 001, 010, 011, 100, 101, 110 и 111. Теперь предположим, что у вас есть память с четырьмя местами; для различения каждого из четырех мест ей нужны только два бита: 00, 01, 10 и 11. Если вы используете нашу 3-битную схему адресации для обращения к этой четырехместной памяти, один из битов адреса должен игнорироваться. Если игнорируется старший бит, то адреса 000 и 100 оба отобразятся на место 00 в памяти. Другими словами, место 00 aliased к адресам 000 и 100.

Memory overlaying - это прием, при котором внеполосная информация используется для выбора между разными банками памяти. Предположим, что мы хотим иметь банк секретной памяти. Для этого мы вставляем селектор между нашими публичной и секретной памятями и CPU. Этот селектор может выбирать, какие данные предъявить CPU - из секретной памяти или из публичной памяти, как указано на рисунке 6-2. В результате программа, управляющая селектором адресов, также управляет тем, кто имеет доступ к секретному блоку. Если компьютер запускается, выполняя код, расположенный в секретном банке памяти, программа в области секретного кода может

использовать этот механизм, чтобы скрыть себя, установив селектор на публичную память перед запуском программ, расположенных в публичной памяти.

Рисунок 6-2. Memory overlaying для скрытия секретных областей.

```
Public Memory  ---> selector ---> Data to CPU
Secret Memory  ---> selector

secret / public selector control from trusted program
Memory address from CPU
```

Более того, opcodes jam table кажутся поврежденными. Это явление было подтверждено другими хакерами, работавшими над проблемой, что исключало ошибку трансляции кода. Очевидно, в Xbox есть больше, чем видно глазу.

Начали появляться теории и слухи, объясняющие это странное поведение. Некоторые популярные теории включали:

- Перемешивание адресных и/или линий данных. Где-то адресные или линии данных инвертировались или переставлялись некоторой функцией отображения 1:1. Функция перемешивания могла быть запрограммирована в чипсете как часть процедуры инициализации, так что начальный загрузочный блок читался бы как plaintext, тогда как остальные данные были бы перемешаны.
- Вторичный криптопроцессор. Другой процессор на Xbox на самом деле выполнял инициализацию Xbox, а загрузочный код в ROM - фиктивный.
- Загрузочный код, содержащийся в процессоре. Процессор на самом деле инициализируется куском кода, находящимся на кристалле процессора, а загрузочный код в ROM - фиктивный.
- Загрузочный код, содержащийся в чипсете. Процессор функционирует идентично стандартному Pentium, но чипсет содержит загрузочный код, который подменяет фиктивный код внутри ROM.

Почти для всех этих теорий единственный способ доказать или опровергнуть их - выполнять эксперименты на аппаратуре. Например, чтобы убедиться, что SMC (System Management Controller, 8-битный самодостаточный процессор, который всегда включен, когда Xbox подключен к сети) не играл роли в защищенной последовательности загрузки машины, хакеры захватили трассы форм сигналов на всех выводах SMC и проанализировали их относительно ожидаемой последовательности событий, если бы SMC играл ключевую роль в инициализации машины.

Ключевое наблюдение от другого хакера состояло в том, что Xbox загружался идеально даже тогда, когда код reset vector по адресу `0xFFFF.FFF0` был изменен. Можно было бы ожидать, что если первая инструкция, выполняемая процессором по адресу `0xFFFF.FFF0`, была повреждена, то машина упадет. Напротив, машина работала безупречно. Это наблюдение было проверено набором экспериментов, где различные части FLASH ROM намеренно повреждались. Результаты показали, что повреждение удивительно больших областей FLASH ROM не влияло на загрузку Xbox. В частности, вся последовательность загрузочной инициализации от `0xFFFF.FE00` до `0xFFFF.FFFF` могла быть занулена, и Xbox загружался совершенно нормально.

Одна эта находка сильно поддерживала теорию фиктивного загрузочного блока во FLASH ROM. Однако оставался вопрос, где хранится настоящий загрузочный код. Было три варианта: во вторичном криптопроцессоре, в процессоре и в чипсете. Теория вторичного криптопроцессора была отброшена на основании того, что на материнской плате не было микросхем, достаточно мощных или достаточно активных во время загрузки, чтобы играть роль криптопроцессора. Хранение загрузочного блока в процессоре также сочли менее вероятным вариантом, чем хранение загрузочного блока в чипсете.

Обоснование этого анализа основано на экономике производства микросхем. Процессор Pentium III очень сложен, с множеством вручную спроектированных блоков, и модификация кремния для включения защищенного загрузочного блока потребовала бы значительных инженерных ресурсов,

а также первоначального вложения около четверти миллиона долларов только на маски, необходимые для производства заказного кремния. Кроме того, ходили слухи, что Microsoft изначально выбрала для Xbox процессор AMD и в последний момент переключилась на Intel. Если бы заказные блоки были интегрированы в ядро процессора, Microsoft не смогла бы так легко переключаться между поставщиками CPU. С другой стороны, чипсеты nVidia проектируются модульно с использованием silicon compilers, так что технически проще добавить такие наросты, как защищенный загрузочный блок. Кроме того, чипсет в Xbox является заказной сборкой nForce, сделанной специально для Microsoft, адаптированной именно под Intel front-side bus (FSB). В результате стоимость добавления защищенного загрузочного блока могла быть включена в инженерные ресурсы и наборы масок, уже выделенные для такого проекта.

Работая в рамках теории, что настоящий загрузочный код расположен в секретном ROM overlay в чипсете, оставшиеся вызовы состояли в том, чтобы определить, в какой микросхеме (Northbridge или Southbridge) хранится код, и как извлечь этот секретный ROM. Возникло несколько стратегий извлечения секретного ROM:

- Использовать функцию JTAG «boundary scan» на Pentium, чтобы попытаться захватить начальный загрузочный код. JTAG - диагностическая шина, которая позволяет читать и устанавливать состояние каждого вывода микросхемы через специальный последовательный порт. Это очень мощный и универсальный инструмент отладки.
- Зондировать FSB (Front Side Bus) процессора, чтобы попытаться захватить загрузочный код, когда он входит в процессор.
- Установить memory sniffer, чтобы попытаться захватить расшифрованный поток данных, когда он записывается в память.
- Использовать микроскопию, чтобы считать содержимое защищенной загрузочной области с поверхности микросхемы.
- Зондировать шину между микросхемами Southbridge и Northbridge, чтобы попытаться захватить загрузочный код, передаваемый процессору чипсетом. Это сработало бы только в том случае, если загрузочные данные хранятся где-то в микросхеме Southbridge.

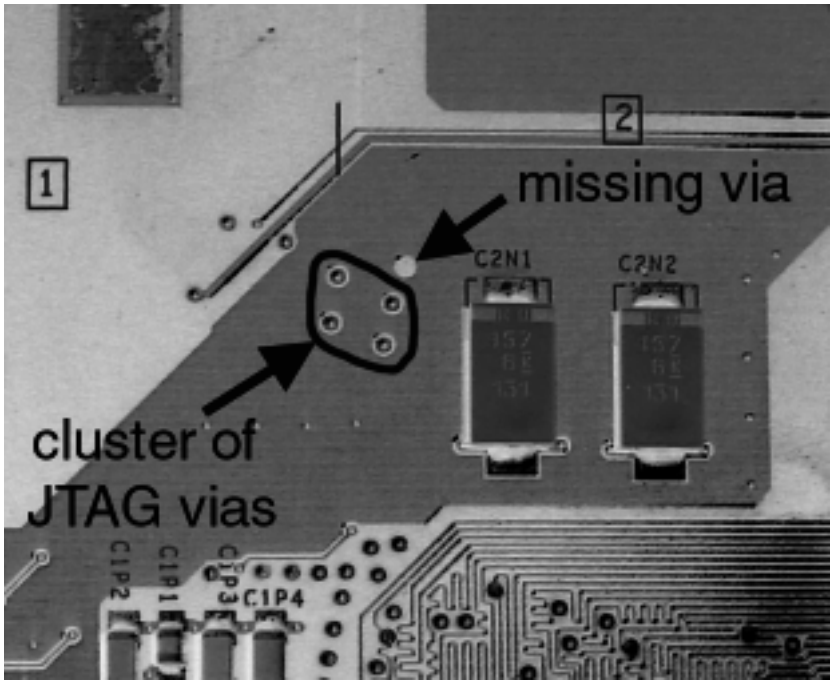


Рисунок 6-3. Отсутствующее переходное отверстие JTAG. Обратите внимание, что в заполненной медной области (более светлая область) есть отверстие там, где раньше было переходное отверстие. Это результат изменения разводки платы в последний момент без перерасчета областей заливки.

Ни одну из этих теорий нельзя было проверить тривиально, поэтому усилия по взлому Xbox медленно зашли в тупик, когда разочарованные хакеры отказались от попыток криптоанализировать образ FLASH ROM. Я был бы одним из сдавшихся (в конце концов, мне нужно было закончить и написать докторскую диссертацию всего за несколько месяцев), если бы не сообщество решительных хакеров, подкармливавших меня поддержкой.

Во время рождественских каникул в декабре 2001 года я поддерживал связь с друзьями-хакерами через IRC-каналы и web fora. Хакеры со всего мира и из всех слоев общества наполняли IRC-канал взлома Xbox, и мне нравилось учиться у них и болтать с ними об их разном опыте, как техническом, так и личном.

Несмотря на то что я был полон решимости провести весь январь за написанием своей PhD-диссертации [6] и избегать взлома Xbox, меня все равно втянула интригующе сложная безопасность, примененная в Xbox. Со временем необходимость в аппаратчике, который присоединился бы к маленькой группе hardcore-хакеров, зависавших в IRC-канале, становилась все яснее. К концу января сообщения, которые я слышал о схеме безопасности Xbox, стали слишком интересными, чтобы их игнорировать.

Я купил второй Xbox и начал снимать все его ключевые детали с помощью термофена. Разбор Xbox до основания служил многим целям. Во-первых, снятие микросхем обнажило все дорожки и соединения на Xbox, так что я мог легко прослеживать соединения между микросхемами, используя режим проверки непрерывности на своем мультиметре. Во-вторых, я смог опустить все интересные микросхемы в горячую кислотную ванну и снять их пластиковую инкапсуляцию для анализа под микроскопом. Наконец, покупка Xbox и полное разрывание его на части дали мне своего рода душевное спокойствие, когда дело дошло до зондирования и модификации работающего Xbox. (Reverse engineering похож на садоводство. Сажать сад гораздо сложнее, если вы пытаетесь сохранить руки и колени чистыми, так что можно сразу смириться и начать валяться в грязи.)

Результаты разборки Xbox раскрыли некоторые меры, которые Microsoft предприняла для защиты коробки от аппаратных хакеров. Например, сначала я проверил JTAG-соединения на Pentium CPU. Все JTAG-сигналы были удобно выведены к набору резисторов рядом с процессором, к которым легко подключиться, кроме одного - сигнала TRST#. TRST# играет критическую роль в инициализации JTAG-интерфейса. Интересно, что TRST# был привязан к внутреннему слою земли в труднодоступной области, навсегда деактивируя механизм JTAG. Дальнейший осмотр материнской платы Xbox выявил намеки на то, что сигнал TRST# был вырезан в последний момент. (Самая большая подсказка отсутствующего via - отверстие в силовой дорожке, идеально подходящее по размеру для via, рядом с группой via, выделенных под JTAG-сигналы, как показано на рисунке 6-3.)

Другой удар по JTAG-подходу для извлечения секретного ROM - тот факт, что JTAG scan codes Intel являются закрытыми. Reverse engineering кодов до уровня, на котором я мог бы использовать их для извлечения секретных загрузочных данных, был крупным проектом сам по себе.

Отказавшись от JTAG-подхода, следующим методом извлечения секретного ROM было снять корпуса с CPU, GPU и MCPX, осмотреть голые кристаллы под микроскопом и поискать любые структуры-кандидаты ROM. Удаление корпуса, или «decapsulation», выполнялось купанием микросхем в дымящей горячей серной кислоте. (Я не рекомендую пробовать этот подход дома; однажды я пролил токсичный, коррозионный раствор на всего себя, и, к счастью, было съедено мое защитное снаряжение, а не кожа. Дымящая серная кислота пожирает органический материал быстрее горящего пламени.) Дымящую азотную кислоту, также очень токсичную и опасную, тоже можно использовать. Хотя сам я ее не пробовал, сообщения указывают, что дымящая азотная кислота более эффективна при удалении эпоксидной инкапсуляции, особенно в ситуациях, где требуется выборочное удаление корпуса.

Подход ручного осмотра с использованием традиционного микроскопа видимого света давал некоторую надежду; однако техника ограничена физикой света. Даже лучшая технология микроскопии видимого света не может разрешить транзистор 150 nm, поскольку самая короткая длина волны света - 450 nm (соответствует синему цвету). Я надеялся, что секретный код будет храниться на микросхемах с использованием традиционной структуры array ROM, с металлическими линиями, задающими 1 или 0, вытравленными в верхних металлических слоях, которые можно идентифицировать оптическим микроскопом. Использование жестко прошитой структуры ROM мотивируется стоимостью: FLASH ROM и PROM на плавких перемычках требуют дополнительных этапов обработки и производства, которые могут значительно добавить к стоимости системы, тогда как использование верхних металлических слоев было бы мотивировано управлением риском со стороны проектировщика. Верхние металлические слои - самые грубые слои (настолько грубые, что оптический микроскоп может их разрешить), а потому самые дешевые слои для изменения, если в ROM-коде есть ошибка. Кроме того, во время первоначального bring-up верхний слой легче всего разрезать и перемкнуть с использованием машины ремонта микросхем, известной как FIB (focused ion beam) machine. К сожалению, быстрый взгляд на микросхему под микроскопом не выявил таких структур.

В этот момент единственной оставшейся возможностью извлечения секретного ROM было зондировать живую аппаратуру Xbox в попытке захватить код во время загрузки в процессор Xbox. Подслушивание кода выше микросхемы Southbridge и FLASH ROM означало зондирование либо Front Side bus, либо шины Northbridge-Southbridge, либо основной шины памяти. Мы обсудим компромиссы выполнения этих подходов зондирования в главе 8, после короткого введения в базовые концепции безопасности в следующей главе.

Сноски

[1] [назад] У Andy Green есть отличная страница, документирующая его опыт с подходом через тестовый зажим, по адресу http://www.warmcat.com/milksop/milksop.html
[2] [назад] Visor описал свой опыт с подходом подслушивания логическим анализатором по адресу http://www.xboxhacker.net/visor/aXventure1.txt
[3] [назад] Emulation Technologies (http://www.emulation.com) делает широкую линейку доступных панелек именно для таких целей. Конкретная модель для Xbox - S-TS-SM-040-A.
[4] [назад] http://www.xenatera.com/bunnie/proj/flashburn/fb.html
[5] [назад] http://www.xenatera.com/bunnie/proj/fb2/
[6] [назад] Для тех, кому интересны архитектура суперкомпьютеров, миграция данных и потоков, отказоустойчивость, высокоскоростные сети с малой задержкой или массивно многопоточные машины, смотрите мою диссертацию по адресу http://www.xenatera.com/bunnie/phdthesis.pdf.

Глава 7. Краткое введение в безопасность

Взлом Xbox требует взлома безопасности в дополнение к аппаратному и firmware-взлому, как вы обнаружили в прошлой главе. Мы рассмотрим некоторые возможные мотивы добавления сложной безопасности к такой обыденной вещи, как игровая машина, а затем погрузимся в базовые принципы и алгоритмы, необходимые для понимания и оценки механизмов безопасности Xbox.

Кому вообще нужна безопасность?

Игровая консоль для большинства людей - игрушка: недорогая потребительская электроника. Почему Microsoft прошла через такие мучения, чтобы защитить свою систему? В игре взлома безопасности понимание мотива защищающей стороны довольно часто помогает найти слабости, которые можно использовать.

Криптография - это не безопасность. Криптография - средство для достижения безопасности, но настоящая безопасность включает всю системную архитектуру, включая конечных пользователей. Как сказал Kevin Mitnick в недавнем интервью Slashdot, «. . . security is not a product that can be purchased off the shelf, but consists of policies, people, processes, and technology.» [1] Я считаю, что безопасность фундаментально является социальным понятием. На практике вы можете открыть окна и оставить входную дверь запертой, и люди не станут просто залезать через ваше окно или взламывать ваш дверной замок, хотя обе задачи относительно легки. Запертые двери и открытые окна работают потому, что запертая дверь в основном является символической мерой; она заставляет нарушителя совершить сознательный акт нарушения, чтобы войти в дом, и одного этого достаточно, чтобы отделить преступников от добропорядочных людей. Консоль Sony Playstation имеет хороший пример безопасности типа замка на входной двери. Механизм, используемый для защиты их игр от копирования, прост, не включает криптографию и легко обходится с помощью легко устанавливаемых, недорогих аппаратных модификаций. Несмотря на это, цифры продаж указывают, что покупка игр Playstation не вышла из моды; замок на входной двери работает.

Microsoft использует вариацию безопасности в стиле замка на входной двери. Видеоигры для Xbox распространяются с использованием (пока что) некопируемого формата DVD-9, одностороннего двухслойного формата носителя. Пользовательски записываемые DVD, с другой стороны, всегда имеют формат DVD-5, односторонний однослойный формат носителя. DVD-9-совместимые пишущие приводы вряд ли скоро станут доступны из-за сложности создания записывающей системы, способной прожигать один слой без воздействия на целостность другого слоя. Таким образом, распределяя данные безопасности между двумя слоями диска DVD-9 и требуя, чтобы исполняемый файл игры поступал с носителя DVD-9, Microsoft имеет довольно эффективный замок на входной двери своих видеоигр. Разумно требуя формат DVD-9, Microsoft в значительной степени вынудила любого потенциального копировщика игр попасть в область, где необходима какая-либо аппаратная модификация.

Почему же тогда Microsoft рискнула вложиться в такую сложную схему безопасности на Xbox? Главная мотивация Microsoft действительно в подавлении пиратства? Вполне возможно, что на самом деле первичная причина остальной части системы безопасности Xbox - защищенных загрузочных секторов, подписанных исполняемых файлов, отношений доверия и зашифрованных/аутентифицированных сетевых протоколов - лежит не в антипиратских мерах.

Один возможный мотив всей этой безопасности - предотвратить использование консоли Xbox для любых целей, кроме игр. Консоль Xbox находится в уникальном положении почти на 100 процентов штатного PC. В отличие от Gamecube и Playstation2, существует огромный массив программного

обеспечения, которое, кажется, должно просто работать на Xbox при правильном программировании BIOS. Хуже того, Microsoft теряет на аппаратуре своей консоли намного больше денег, чем ее конкуренты. Некоторые оценивают, что ее убытки могут достигать $200 на консоль, если считать последнюю розничную цену $199. Поэтому в интересах Microsoft попытаться гарантировать, что она не продает субсидированные коробки GNU/Linux. Однако даже это, вероятно, не главная цель Microsoft. 64 MB основной памяти Xbox, отсутствие клавиатуры или мыши из коробки и довольно медленный по сегодняшним меркам процессор делают ее менее привлекательной, чем, например, PC Microtel за $200, доступный в Walmart в конце 2002 года. Кроме того, у Microsoft глубокие карманы; если бы Xbox получила рыночную тягу и обошла Sony Playstation2 по продажам, Microsoft пришлось бы лишь переварить несколько миллиардов долларов первоначального убытка - относительно небольшую сумму по сравнению с примерно $40 billion денежной горой, на которой она сидит. Таким образом, вполне возможно, что критическая миссия безопасности Xbox - не предотвращение альтернативного использования консоли и не сдерживание пиратства.

Возможно, реальная причина сложной безопасности Xbox - обеспечение успеха Xbox Live, онлайнового игрового сервиса Microsoft. Маркетинговая шумиха Microsoft и PR-заявления указывают, что она делает ставку на успех Xbox Live как двигатель продаж аппаратуры. Более того, Xbox Live - подписочный сервис, и через год после его запуска пользователям придется платить ежемесячную плату. Если Microsoft сможет подсадить своих подписчиков на Xbox Live, тогда бизнес Xbox внезапно выглядит довольно прибыльным, даже если значительная сумма денег теряется заранее на аппаратуре. Фокус, конечно, в том, чтобы подсадить пользователей Xbox на Xbox Live. Описываемая как «Disneyland of on-line gaming», цель Xbox Live - предоставить хорошо исполненный и честный игровой опыт. Центрально для ценностного предложения Xbox Live то, что там нет читеров. Чтобы гарантировать, что никто не читерит, пользователей нужно заставить аутентифицироваться через реестр, поддерживаемый Xbox Live, а их игровое состояние должно оставаться защищенным и неизменяемым. Кроме того, игровое программное обеспечение должно быть непатченым. Еще важнее тот факт, что нужно всего несколько читеров, чтобы испортить игровой опыт всей пользовательской базы. Внезапно защиты входной двери, предлагаемые форматом DVD-9, выглядят недостаточными. Шансы против вас, если вы ставите успех бизнеса на мораль и честь пользовательской базы из миллионов двадцатилетних hardcore-геймеров мужского пола с разумным объемом компьютерной грамотности, распределенных повсюду. Аппаратура должна быть заслуживающей доверия, сетевые соединения защищенными, а исполняемые файлы подписанными и запечатанными.

Утверждение, что аппаратура должна быть заслуживающей доверия, стоит повторить. При пользовательской базе, которой нельзя доверять, единственный способ установить отношение доверия с клиентами - если семя доверия существует в каждом экземпляре аппаратуры. Следовательно, Microsoft должна включить в каждого клиента кусок tamper-proof аппаратуры, который включает какой-либо вид attestation. Attestation - это способность доказать, что некоторый фрагмент данных, например личность игрока или игровое состояние, действительно сгенерирован незапятнанными программным обеспечением и аппаратурой. Tamper-proof аппаратура не обязана напрямую реализовывать функцию attestation, но она как минимум должна гарантировать, что система находится в заслуживающем доверия состоянии перед attestation.

Есть много способов гарантировать, что аппаратура заслуживает доверия. Метод грубой силы - сделать весь фрагмент аппаратуры физически защищенным. Автоматические банковские автоматы являются первоклассными примерами аппаратуры, которая физически защищена. Запечатанные в толстый листовой металл и покрытые датчиками вторжения, они трудны для физического проникновения и модификации аппаратуры ATM. Но хотя это эффективно, это непрактичное и дорогое решение для игровой консоли.

Более экономичное решение - использовать небольшой кусок доверенной tamper-proof аппаратуры, который может делать «измерения» на остальной системе. Такого рода измерения обычно выполняются с помощью криптографической хэш-функции. Если все эти измерения доверия соответствуют ожидаемым значениям, тогда можно было бы заключить, что вся система заслуживает доверия.

Я говорю «можно было бы», потому что эта схема все еще уязвима к man-in-the-middle-атакам, где хакер отправляет поддельные корректные данные в ответ на запрос измерения. Man-in-the-middle-атаки относятся к общему классу атак, где противник может свободно изменять и контролировать информацию, передаваемую между двумя сторонами. Из-за слабости man-in-the-middle нет смысла использовать чрезвычайно сложный tamper-proof-модуль для выполнения системных измерений. Одной корпусированной кремниевой микросхемы, вероятно, достаточно, поскольку обычно легче перехватить и подделать данные измерений, проходящие по печатной плате, чем проникнуть в эпоксидный корпус микросхемы и модифицировать схему микросхемы.

Система измерения доверия может быть реализована с использованием подхода measure-once. Начиная с последовательности холодной загрузки процессора, каждый фрагмент кода измеряется на доверие перед выполнением. Если процессор никогда не выполняет недоверенный код, тогда чему тут не доверять? Эта схема требует очень простого tamper-proof аппаратного модуля - tamper-proof ROM, который хранит код холодной загрузки, «семя» доверия. Тип криптографии, используемой для процесса измерения и проверки, обычно является комбинацией хэшей и криптографии с открытым ключом. Криптография с открытым ключом предпочтительна для этого приложения, потому что закрытый ключ, требуемый для генерации корректного сегмента кода, является секретом, хранимым только поставщиком аппаратуры. Опять же, эта схема уязвима ко многим видам man-in-the-middle-атак, а также к чисто криптографическим атакам и атакам на реализацию системы.

Краткое введение в криптографию

ci-pher (n): 1 a: ZERO b: one that has no weight, worth, or influence : NONENTITY. 2 a: a method of transforming text in order to conceal its meaning - compare to CODE [2]

Шифры сами по себе не дают безопасности. Точнее, шифры дают безопасность только если ключ защищен, если алгоритм силен и если в систему нет back doors. Если кто-то вручит вам CD-ROM, зашифрованный сильным шифром, и запрет вас в мягкой комнате с суперкомпьютером, солнце, вероятно, успеет стать сверхновой, прежде чем вы сможете расшифровать CD-ROM. С другой стороны, если бы вы могли наблюдать и зондировать машину, пока она работает над шифрованием CD-ROM, шифрование потеряло бы смысл. Вы могли бы получить ключ шифрования, подслушав клавиатуру. Или вы могли бы выгрузить содержимое памяти компьютера и получить plaintext, не зная ключа.

Ситуация с Xbox похожа на последнюю. В конечном счете Xbox должна обращаться к программам, представленным ей на корректных дисках, и запускать их. Более того, Pentium CPU, используемый в Xbox, не может отличить авторизованную инструкцию от неавторизованной инструкции. Наконец, пользователь имеет полный доступ для зондирования и модификации аппаратуры Xbox. Таким образом, даже если Xbox использует сильные шифры, безопасность ключей сомнительна, и в системе могут быть back doors.

Этот раздел кратко опишет виды криптографических алгоритмов, используемых в Xbox. Мы сосредоточимся на практических последствиях и вопросах реализации этих алгоритмов. Вам нужно понять эти алгоритмы, чтобы оценить доступные атаки на систему безопасности Xbox.

NB. Я не притворяюсь, что рассматриваю теоретические аспекты криптографии; они за пределами моих возможностей и за пределами объема книги. Вместо этого я отсылаю заинтересованных читателей к превосходной книге Bruce Schneier, Applied Cryptography (John Wiley & Sons); большая часть моих знаний по криптографии взята из этой книги. Читатели, уже знакомые с криптографией, смогут бегло просмотреть или пропустить остаток этой главы.

Классы криптографических алгоритмов

Есть несколько важных классов криптографических алгоритмов, используемых в Xbox. Это:

• Хэши
• Симметричные шифры
• Шифры с открытым ключом

Хэши бывают нескольких разновидностей. Хэши криптографической разновидности используются для суммирования или «digest» большого объема данных. Сводка является числом фиксированной длины, обычно примерно от 100 до 200 бит, тогда как исходные данные могут быть почти любого размера. Самое важное свойство хэша - это то, что он является однонаправленным вычислением. Другими словами, хэш легко вычислить, но очень трудно (см. боковую вставку «Very Difficult Problems», чтобы понять, что именно это означает) вывести последовательности данных, чей hash digest идентичен, или определить что-либо об исходных данных из хэша.

Сила хэша против поиска двух последовательностей данных, которые хэшируются в одно и то же значение, называется его collision resistance, и в общем хороший n-битный хэш требует примерно 2^(n/2) случайных последовательностей данных, которые нужно хэшировать и сравнить, чтобы вызвать collision. Поскольку хэши проектируются так, чтобы быть очень легкими для вычисления и очень устойчивыми к collision, они часто используются для обнаружения, был ли изменен бит в больших областях защищенных данных. Для многих приложений достаточно включить только зашифрованный хэш сообщения вместо траты вычислительных усилий на шифрование всего сообщения.

Симметричные шифры - это алгоритмы, у которых ключи шифрования и расшифровки могут быть легко получены друг из друга. В большинстве случаев ключи шифрования и расшифровки одинаковы. Симметричные шифры используют mixing function, чтобы объединить key schedule с данными, обработанными некоторой криптографической функцией. Это смешивание может повторяться несколько раз над одним блоком данных, как в block cipher, или может происходить один раз, как в stream cipher. Все базовые функции в симметричном шифре вычислительно просты, поэтому симметричные шифры являются предпочтительным методом шифрования массовых данных.

Типичные примеры mixing functions - XOR, модульные сложения и модульные умножения. Самая простая функция, XOR, имеет свойство, что любое число XOR само себя равно нулю. Операция XOR часто обозначается символом . Операция XOR также имеет все обычные свойства арифметики (коммутативность, ассоциативность, дистрибутивность и т. д.), поэтому:

```
(A  B)  B = A  (B  B) = A  0 = A
```

Таким образом, если бы A было сообщением, а B было ключом, `(A  B)` было бы ciphertext, а plaintext можно было бы восстановить, просто выполнив XOR с B еще раз.

Key schedule - это алгоритм, который берет относительно короткий ключ и расширяет его информацию на длинную последовательность битов. Key schedules используются, чтобы помочь рассеять данные ключа по большему блоку данных, чтобы связь между ciphertext и ключом была скрыта.

Очень трудные проблемы

Криптографические функции все основаны на математических алгоритмах, результаты которых легко вычислить при наличии всех операндов, но операнды которых очень трудно вычислить, имея только результат. Безопасность криптографической функции - это именно трудность вычисления этих операндов при наличии только результатов. Давайте на мгновение исследуем, что значит «очень трудно».

Рассмотрим симметричный шифр AES. Он использует 128-битный ключ, и до сих пор он силен против всех известных аналитических криптографических атак, например дифференциального и линейного криптоанализа. Когда я говорю, что он силен против анализа X, я имею в виду, что для восстановления ключа или plaintext с использованием brute-force search потребуется как минимум столько же операций, сколько с использованием анализа X. Brute-force search - это когда я беру очень быстрый компьютер и пробую каждый из 2^128 возможных ключей, чтобы восстановить исходные данные.

Большинство криптографических алгоритмов, широко используемых сегодня, сильны против всех известных техник криптоанализа, поэтому важное число для понимания - сила brute-force-атаки.

Как оказывается, более старые алгоритмы, такие как DES, 56-битный шифр, не являются очень трудной проблемой. Довольно легко построить машину с использованием FPGAs (Field Programmable Gate Arrays), которая может взламывать ключи при экономике примерно 2^22 ключей/секунда/доллар (2^22 - примерно четыре миллиона). Обратите внимание, что это число растет со временем со скоростью, эквивалентной закону Moore. Сегодня, если вы готовы ждать примерно неделю для каждого ключа, вы можете восстановить их за цену хорошей машины. Будем надеяться, что банки не используют DES для шифрования данных счетов!

Преемник DES, AES, - это шифр, который может использовать 128-, 192- или 256-битные ключи. Эти ключи достаточно велики, чтобы считаться непроницаемыми для brute-force-атак (то есть очень трудной проблемой). Согласно AES Q&A, опубликованному NIST (http://csrc.nist.gov/encryption/aes/aesfact.html), машине, достаточно мощной, чтобы восстанавливать один DES-ключ в секунду brute force (пробуя в среднем 2^55 ключей в секунду), все равно потребовалось бы 149 триллионов лет, чтобы восстановить 128-битный AES-ключ. Мой любимый анализ силы 256-битных ключей против brute-force-атак взят из Applied Cryptography Bruce Schneier. В своей книге он использует аргумент, основанный на количестве энергии, требуемой для взлома 256-битного ключа. Оказывается, даже с термодинамически идеальным компьютером потребовалось бы более 32 годовых энергетических выходов нашего солнца, чтобы просто досчитать до 2^192, не говоря уже о том, чтобы сделать что-либо полезное с этим счетом. (Я должен подчеркнуть, что все это предполагает, что самая эффективная атака - brute force. Кто

знает, может быть, кто-то обнаружит слабость в алгоритме, которую можно использовать для гораздо более эффективной атаки. Новые техники анализа постоянно изобретаются и медленно подтачивают силу шифра.)

Шифры с открытым ключом, с другой стороны, основаны на широком разнообразии труднообратимых математических операций, например умножении простых чисел и модульном возведении в степень. В результате пространство ключей для многих шифров с открытым ключом разрежено, поэтому для эквивалентной безопасности симметричного шифра требуется больше битов ключа. Например, длины ключей в шифре с открытым ключом RSA обычно составляют несколько тысяч бит.

Точная корреляция между безопасностью длин открытых ключей RSA и длинами ключей симметричного шифра неизвестна. Считается, что безопасность RSA - это трудность факторизации произведений больших простых чисел; однако могут существовать и другие атаки на алгоритм, еще не обнаруженные. Даже так, эффективная трудность факторизации произведения больших простых уменьшается не только прогрессом вычислительной технологии (закон Moore), но и прогрессом теории чисел, например изобретением и совершенствованием Quadratic Sieve и General Number Field Sieve.

В августе 1999 года группа исследователей использовала Number Field Sieve, чтобы факторизовать 512-битное простое число за 7.4 календарных месяца, включая время, требовавшееся для настройки запуска факторизации. [3] Кроме того, новые технологии, такие как квантовые вычисления, обещают сделать возможной факторизацию простых чисел за полиномиальное время. Однако я бы не стал задерживать дыхание; все еще идут споры о том, возможно ли построить квантовый компьютер, достаточно большой для факторизации интересного простого числа.

На сегодняшний день RSA Security, Inc. рекомендует длины ключей 1024 бита для большинства корпоративных применений и 2048 бит для «extremely valuable keys». [4] Bruce Schneier оценивает во втором издании Applied Cryptography, что длина открытого ключа 2304 бита дает безопасность, эквивалентную 128-битному симметричному ключу, и что длина открытого ключа 1792 бита соответствует примерно 112-битному симметричному ключу.

Когда вы читаете о схеме безопасности Xbox, держите в уме эти базовые ориентиры о том, насколько трудно может быть взломать эти схемы безопасности brute-force-методами. Снова и снова на хакерских форумах и досках объявлений появляются сообщения с вопросом: «почему бы нам не начать распределенное усилие по поиску этих ключей?» Теперь вы знаете ответ.

Типичная криптографическая функция, используемая в симметричном block cipher, состоит из набора тщательно спроектированных подстановок, перестановок, сжатий и расширений. Эти функции служат, чтобы confusion и diffusion plaintext. Тонкие изменения в любой части криптографической функции обычно оказывают глубокое влияние на безопасность шифра.

Тот факт, что ключи шифрования и расшифровки тесно связаны в симметричном шифре, делает их трудными для использования в некоторых приложениях безопасности. Например, если я хочу распространить зашифрованный документ в список рассылки, каждый в списке рассылки также фактически должен знать мой ключ шифрования, если он может прочитать документ. Кроме того, начать контакт с удаленной стороной трудно, потому что в какой-то момент мне нужно передать ей ключ. Кто-то, наблюдающий среду передачи, мог бы украсть ключ и читать, подделывать и изменять все последующие сообщения.

Шифры с открытым ключом - это алгоритмы, использующие разные ключи для шифрования и расшифровки. По этой причине их также называют асимметричными шифрами. Большое преимущество шифров с открытым ключом в том, что один из ключей можно держать в секрете. Это позволяет обмениваться данными с недоверенными пользователями, не давая недоверенному пользователю возможности подделывать или читать другой защищенный контент. Обратная

сторона алгоритмов с открытым ключом в том, что они обычно требуют более сложных вычислений и поэтому медленнее симметричных шифров. Шифры с открытым ключом также имеют тенденцию требовать более длинных ключей для эквивалентной безопасности. В результате, если нужно обменяться большим объемом данных, шифры с открытым ключом часто используются для шифрования ключа для симметричного шифра, который используется для шифрования основной массы данных. Этот ключ симметричного шифра может быть уникальным для каждой транзакции, и поэтому его часто называют «session key».

Хэш SHA-1

SHA-1 - это Secure Hash Algorithm, рекомендованный федеральным правительством в публикации FIPS 180-1 (http://www.itl.nist.gov/fipspubs/fip180-1.htm). Придуманный NSA и основанный на алгоритме message digest MD4 Ronald L. Rivest, SHA-1 работает с сообщениями любой длины меньше 2^64 бит и выдает 160-битный результат.

Алгоритм хэша SHA-1 начинается с детерминированного 160-битного начального состояния; это состояние смешивается с блоком 512 бит данных сообщения в течение четырех раундов. Каждый раунд состоит из серии нелинейных функций, rotations, shifts и XOR. Результат раунда используется как seed для вычисления следующего раунда. В общем случае нужно сгенерировать, хэшировать и «одновременно» сравнить 2^80 случайных сообщений, чтобы найти два сообщения, имеющих одно и то же значение хэша (то есть hash collision). Поиск двух случайных сообщений, имеющих один и тот же хэш, известен как «birthday attack», названная по вероятностному явлению «birthday paradox»: вероятность того, что два человека имеют один и тот же день рождения в комнате из 23 человек, больше 50%. С другой стороны, нужно сгенерировать, хэшировать и сравнить 2^160 случайных сообщений, чтобы найти сообщение, которое хэшируется в то же значение, что и конкретное сообщение. Таким образом, сила хэш-функции сильно зависит от того, каким образом она используется.

TEA

TEA, или tiny encryption algorithm, был разработан David Wheeler и Roger Needham в Computer Laboratory Кембриджского университета. (У разработчиков есть веб-страница TEA по адресу http://vader.brad.ac.uk/tea/tea.shtml; большая часть материала, представленного здесь, почерпнута с этой страницы.)

Как подразумевает его имя, TEA - компактный, быстрый алгоритм шифрования, пригодный для шифрования потоков данных реального времени и встроенных приложений, где производительность процессора и место хранения ограничены.

```
void encipher(unsigned long *const v,unsigned long *const w,
              const unsigned long *const k)
{
  register unsigned long
    y=v[0], z=v[1], sum=0, delta=0x9E3779B9,
    a=k[0], b=k[1], c=k[2], d=k[3], n=32;
  while(n-->0) {
    sum += delta;
    y += (z << 4)+a ^ z+sum ^ (z >> 5)+b;
    z += (y << 4)+c ^ y+sum ^ (y >> 5)+d;
  }
  w[0]=y; w[1]=z;
}
```

Листинг 7-1. Алгоритм TEA на ANSI C.

Источник: см. примечание [5].

Рисунок 7-1. Сценарии использования шифра TEA.

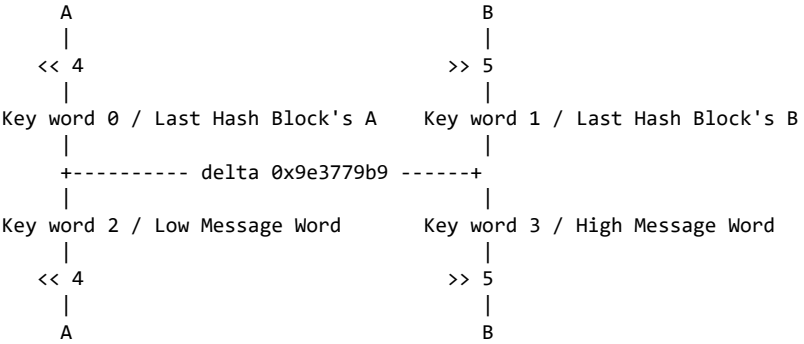
```
TEA в применении как шифр:

128-bit key in
64 bits plaintext in
TEA cipher
64 bits ciphertext out

TEA, используемый как хэш-функция:

64 bit magic seed number + 64-bit key data[1], data[0] -> TEA cipher
previous hash block + data[3], data[2] -> TEA cipher
previous hash block + data[n], data[n-1] -> TEA cipher
hash value out
```

Рисунок 7-2. Внутренняя структура TEA. Эта диаграмма изображает один раунд TEA, который повторяется 32 раза для полного шифра. Key schedule описан в блоках справа для использования и как шифр, и как хэш-функция.



TEA имеет 128-битный ключ и работает с 64 битами данных за раз, а каждый из его 32 раундов использует только shifts, XOR и additions. (Алгоритм, приведенный в листинге 7-1 и рисунок 7-2, оптимизирован для реализации на 32-битных процессорах общего назначения.)

Считается, что маленький алгоритм TEA довольно защищен при использовании для шифрования и расшифровки данных. Однако TEA не используется для шифрования в Xbox; на самом деле он используется как хэш-функция при работе шифра в модифицированном режиме Davies-Meyer. Область, которую нужно хэшировать, делится на 64-битные блоки. Эти блоки исходных данных используются как половина входа ключа для TEA. Другая половина входа ключа берется из результата предыдущей операции TEA, а первая операция TEA использует magic number как свой вход.

Результатом является 64-битная хэш-функция, как изображено на рисунке 7-1. Этот хэш слаб против birthday attacks, особенно учитывая вычислительную эффективность TEA, поскольку в среднем нужно проверить только 2^32 пар сообщений, чтобы найти collision. Даже несмотря на то что birthday attack не применяется в сценарии использования Xbox, Xbox выполняет хэш дважды, каждый раз с другим seed magic number, и сцепляет результаты, чтобы сгенерировать одно 128-битное значение хэша - вероятно, в попытке сорвать brute-force-атаки.

К сожалению, у TEA есть слабость в key schedule: каждый ключ TEA имеет четыре связанных ключа. Другими словами, для каждого ключа можно сгенерировать три других ключа, которые дают тот же результат ciphertext при тех же входных данных. Генерация related-key так же проста, как complementing pairs of key bits (bits 31 and 63 is one pair, bits 95 and 127 are the other pair). Это делает TEA непригодным для использования как хэш-функции, и эта слабость хорошо документирована в статье «Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and triple-DES» John Kelsey,

Bruce Schneier и David Wagner, представленной много лет назад на CRYPTO 1996. Позже эта слабость была использована командой под руководством Andy Green, чтобы взломать вторую версию схемы безопасности Xbox.

RC-4

RC-4 (Ron's Code или Rivest Cipher 4) - stream cipher с переменной длиной ключа от Ron Rivest. Сердце RC-4 - генератор keystream. Его можно представить как криптографический псевдослучайный генератор чисел (CPRNG). Выход CPRNG XOR'ится по одному байту с потоком plaintext, чтобы сгенерировать ciphertext. Расшифровка выполняется похожим образом. Грубо говоря, генератор «seeded» значением (ключом) длиной до 256 байт (2048 бит). Если ключ короче 256 байт, он повторяется, чтобы заполнить 256 байт перед использованием как seed; это позволяет ключи переменной длины. В Xbox ключ имеет длину 16 байт (128 бит), и поэтому шифр назван RC-4/128.

```
typedef struct rc4_key {
     unsigned char state[256];
     unsigned char x;
     unsigned char y;
} rc4_key;

void prepare_key(unsigned char *key_data_ptr, int key_data_len,
                 rc4_key *key) {
   unsigned char swapByte, index1, index2;
   unsigned char* state;
   short counter;
   state = &key->state[0];
   for(counter = 0; counter < 256; counter++)
     state[counter] = counter;
     key->x = 0;     key->y = 0;
     index1 = 0;     index2 = 0;
   for(counter = 0; counter < 256; counter++) {
     index2 = (key_data_ptr[index1] + state[counter] +
               index2) % 256;
     swap_byte(&state[counter], &state[index2]);
     index1 = (index1 + 1) % key_data_len;
   }
}

void rc4(unsigned char *buffer_ptr, int buffer_len, rc4_key
         *key) {
   unsigned char x, y, xorIndex;
   unsigned char* state;
   short counter;
   x = key->x;     y = key->y;
   state = &key->state[0];
   for(counter = 0; counter < buffer_len; counter ++) {
        x = (x + 1) % 256;
        y = (state[x] + y) % 256;
        swap_byte(&state[x], &state[y]);
        xorIndex = state[x] + (state[y]) % 256;
        buffer_ptr[counter] ^= state[xorIndex];
    }
    key->x = x;     key->y = y;
}
```

Листинг 7-2. Код RC-4 на C, из оригинальной публикации в Usenet.

Источник: см. примечание [6].

Считается, что RC-4 - сильный шифр, хотя есть несколько известных слабостей в алгоритме key scheduling, которые можно использовать в плохо спроектированных криптосистемах, таких как

WEP. Scott Fluhrer, Itsik Mantin и Adi Shamir документируют эти слабости в статье «Weaknesses in the Key Scheduling Algorithm of RC4», представленной на Eighth Annual Workshop on Selected Areas in Cryptography (August 2001). Ни одна из этих слабостей не может быть применена против реализации RC-4 в Xbox.

Однако есть потенциальная проблема в том, как RC-4 используется в первой версии безопасности Xbox. RC-4 используется на Xbox, чтобы зашифровать поток x86-кода, и над расшифрованным кодом не выполняется значимой проверки, гарантирующей целостность plaintext. Это означает, что изменения в ciphertext приведут к изменениям в коде, который выполняет Xbox. Фокус в том, чтобы вычислить изменение в ciphertext, которое приводит к осмысленной модификации кода. Поскольку RC-4 шифрует по одному байту за раз, а x86 opcodes могут быть длиной всего один байт, требуется не больше 2^8 = 256 итераций, чтобы «brute force» инструкцию в одном известном месте путем мутации ciphertext.

Определить, какое место brute force'ить, может быть хитро, но я подозреваю, что много информации можно было бы получить, мутируя биты ciphertext и наблюдая, что происходит с шаблоном выборок инструкций, даже при включенных кэшах. Цель состояла бы в том, чтобы попытаться идентифицировать место операндов opcode перехода и изменить назначение перехода так, чтобы защищенная программа прыгнула в незащищенную область памяти. Процесс был бы похож на игру в классическую настольную игру «Battleship». Имейте в виду, что атака настолько легка, что угадывание через килобайт кода требует максимум 2^18 итераций. Процесс угадывания можно было бы автоматизировать, интегрировав логический анализатор с эмулятором ROM через управляющий скрипт, выполняющийся на host-компьютере.

История RC-4 на самом деле довольно интересна. RC-4 был изобретен в 1987 году Ron Rivest и хранился как коммерческая тайна RSA Security, Inc. до тех пор, пока не был опубликован в 1994 году анонимным сообщением в списке рассылки cypherpunks (см. листинг 7-2). Благодаря достоинствам RC-4 - простоте и устойчивости - он попал во множество приложений, включая WEP, SSL, SQL и CDPD. Хотя исходный код RC-4 широко распространяется и хорошо известен, шифр все еще является интеллектуальной собственностью RSA Security. Я бы не рекомендовал интегрировать его в коммерческий продукт, предварительно не получив лицензию у RSA Security.

Рисунок 7-3. Использование RSA с session keys.

```
SENDER:
plaintext + session key -> AES (fast) -> ciphertext
session key + public key -> RSA (slow) -> key

untrusted transmission medium

RECEIVER:
key + private key -> RSA (slow) -> session key
ciphertext + session key -> AES (fast) -> plaintext
```

Рисунок 7-4. RSA, используемый для реализации digital signatures.

```
SENDER:
message -> hash
hash + private key -> RSA -> signature
message + signature -> untrusted transmission medium

RECEIVER:
message -> hash
signature + public key -> RSA -> hash
compare
if equal, message is likely to be authentic and unaltered
```

RSA

RSA - это алгоритм с открытым ключом, придуманный Ron Rivest, Adi Shamir и Leonard Adleman в 1977 году. В алгоритме с открытым ключом используются два разных ключа, public key и private key. Как подразумевают их названия, private key должен храниться в секрете, тогда как public key можно свободно распространять. Математика, стоящая за RSA, кратко описана во вставке «The RSA Algorithm». Вам не нужно понимать детали математики RSA, чтобы уловить, как RSA используется в контексте Xbox.

Brute-force-атаки в настоящее время считаются неосуществимыми против RSA с длинами ключей более примерно тысячи бит. Также обратите внимание, что нельзя слишком легкомысленно относиться к тому, как RSA интегрируется в криптосистему. Есть некоторые атаки против протоколов, использующих RSA, например обман владельца private key, заставляющий его подписывать тщательно сконструированные сообщения, которые затем можно использовать для вывода private key подписывающего.

Шифрование сообщения с помощью RSA так же просто, как применение RSA к сообщению. Однако RSA-шифрование работает с блоками сообщений, которые слишком коротки, а процесс шифрования слишком медленный, чтобы быть практичным для большинства сообщений. Поэтому RSA обычно используется для шифрования одноразового случайного ключа, называемого session key, для быстрого симметричного шифра, такого как AES, который затем используется для шифрования основного сообщения. Этот процесс показан на рисунке 7-3.

В дополнение к шифрованию RSA включает digital signatures. Digital signature позволяет сторонам, обменивающимся сообщениями через небезопасную среду, гарантировать, что сообщения не подделаны и не изменены. Сообщение не обязано быть зашифровано. Типичный протокол digital signature работает следующим образом: отправитель вычисляет хэш отправляемого сообщения. Затем этот хэш шифруется private key отправителя и включается вместе с plaintext сообщения. Получатель расшифровывает зашифрованный хэш сообщения, используя public key отправителя, и сравнивает этот хэш с локально вычисленным хэшем полученного сообщения. Если расшифрованный хэш, отправленный с сообщением, и локально вычисленный хэш совпадают, тогда получатель может заключить, что сообщение является подлинным и неизмененным. Этот процесс обрисован на рисунке 7-4.

Если этот протокол звучит для вас сложным, так и есть. Есть много мест, где что-то может пойти не так. У получателя может быть ложная копия public key отправителя. Private key отправителя мог быть скомпрометирован. У хэша могут быть слабости. Применение digital signatures в состязательной среде требует внимания к деталям на всех уровнях системного проектирования.

В Xbox digital signatures используются, чтобы контролировать распространение и продажу программ для консоли. Microsoft фактически контролирует и отправителя, и получателя сообщений. Получатели - консоли Xbox - запрограммированы запускать только программы, digitally signed Microsoft. В идеальном мире это гарантирует, что Microsoft имеет последнее слово в том, кто может или не может запускать программы на консоли, а хакеры не могут модифицировать игры, чтобы вставить вирусы, Trojan horses или back doors. Saved games также запечатываются с использованием шифрования, и в результате номинально невозможно взломать игру и читерить путем патчинга executable или накручивания характеристик вашего персонажа.

Алгоритм RSA

Алгоритм RSA был запатентован Massachusetts Institute of Technology и эксклюзивно лицензирован RSA Data Security, Inc в 1983 году. Патент на алгоритм RSA с тех пор истек в сентябре 2000 года. Таким образом, сегодня RSA свободен для использования в любом приложении. Много отличных учебников и образовательных примеров с RSA теперь можно найти в Internet. Выполните поиск Google по ключевым словам «RSA algorithm», чтобы найти некоторые из этих примеров.

Алгоритм RSA следующий (адаптировано из http://world.std.com/~franl/crypto/rsa-guts.html):

1. Найдите два больших (тысячи бит длиной) простых числа, «P» и «Q».
2. Выберите «E» так, чтобы E > 1, E < PQ, и E было взаимно простым с (P-1)(Q-1). E не обязано быть простым, но должно быть нечетным. Пара E и PQ является public key.
3. Вычислите «D» так, чтобы (DE - 1) делилось без остатка на (P-1)(Q-1). Это можно выполнить, найдя целое X, которое делает D = (X(P-1)(Q-1) + 1)/E целым числом. D является private key.
4. Plaintext «T» шифруется с использованием функции:

```
C = (T^E) mod PQ
```

5. Ciphertext «C» расшифровывается с использованием функции:

```
T = (C^D) mod PQ
```

Обратите внимание, что T < PQ. Сообщения больше PQ должны быть разбиты на последовательность меньших сообщений, а очень короткие сообщения должны быть дополнены тщательно выбранными значениями, чтобы сорвать dictionary attacks, среди прочего.

Очевидно, ключевой вопрос во взломе консоли Xbox - их реализация системы digital signature. Xbox использует SHA-1 hash с 2048-битными RSA-ключами, что делает шанс успешной brute force-атаки очень, очень малым. Конечно, вероятность равна нулю, если вы никогда не пробуете, но шансы сложены против вас (см. вставку «Very Difficult Problems»). Вам больше повезет, если вы попытаетесь выиграть в лотерею. Это не ошибка; открытие private key сделало бы копирование игр тривиальным, а разработчикам не нужно было бы платить Microsoft royalties (юридически они могут быть обязаны, но нет технической причины, мешающей им). Учитывая, что этот ключ, вероятно, стоит для Microsoft несколько миллиардов долларов, весьма вероятно, что ни один человек не знает весь ключ целиком, поскольку техники rubber-hose (избиения) и green-paper (подкуп) криптоанализа имеют тенденцию быть довольно эффективными на людях. (Не сбрасывайте со счетов настоящую «brute force» как возможность, если вы пытаетесь защитить чрезвычайно ценный секрет!) Такие продукты, как система управления центром сертификации BBN SignAssure, обеспечивают физическую безопасность высокоценных ключей и реализуют схемы secret-sharing, требующие нескольких доверенных пользователей для активации машины.

Как упоминалось ранее, существует несколько известных жизнеспособных атак против RSA, но не все из них применимы в сценарии Xbox, поскольку они опираются на группы пользователей или требуют chosen-ciphertext. Кроме того, список слабостей широко известен, и большинство реализаций digital signatures реализуют правильные контрмеры для защиты от таких атак.

Остальная картина

Эффективная система безопасности требует хорошего управления ключами, сильных протоколов и, в случае Xbox, физической безопасности в дополнение к сильным шифрам и хэшам.

Управление ключами, возможно, одна из самых трудных задач системной реализации, с которыми сталкивается любой архитектор безопасности. В конечном счете ключи расшифровки должны

попасть в руки пользователя. Пользовательский интерфейс должен быть спроектирован так, чтобы средний пользователь с минимальным обучением случайно не утекал ключевую информацию. По мере того как шифры становятся сильнее, самый легкий путь атаки все чаще проходит через пользователя. Подслушивание через видеонаблюдение, social engineering или даже анализ шаблона звуков, издаваемых клавиатурой при наборе пароля, вероятно, даст больше информации на единицу усилия о passphrase, чем криптоанализ. Криптография с открытым ключом частично помогает решить проблему распространения ключей, но fingerprints открытых ключей следует сравнивать лично, чтобы исключить возможность man-in-the-middle-атак. Криптография с открытым ключом также не мешает кому-то с физическим доступом к клиентской машине подслушивать расшифрованный выход.

Кроме того, protocol attacks находят слабости в том, как манипулируются ключи и данные, или в том, как используются сильные шифры. Атака WEP на RC-4 и атака Mike Bond и Ross Anderson на IBM 4758 Cryptoprocessor - оба примеры protocol attacks. Красные флаги для потенциальных protocol attacks - системы, реализующие меры обратной совместимости, и системы, реализованные инженерами, чья основная работа не является crypto-security.

Наконец, в системе вроде Xbox, где одна из целей - установить trustable client, back doors и buffer-overrun-атаки также являются жизнеспособными атаками на trust state машины. Никакие широко используемые коммерческие процессоры не встраивают привилегии выполнения в потоки инструкций или теги данных. Процессоры слепо выполняют любой кусок кода, к которому им велено перейти, независимо от того, был ли переход вызван transient hardware failure или вредоносно размещенным кодом. Периодические хэши состояния машины можно использовать для противодействия этому недостатку, но даже тогда проверки состояния можно подделать.

Как обсуждалось в начале этой главы, установление trust state клиента также требует куска tamper-resistant аппаратуры, чтобы нести семя доверия. Объем физической безопасности должен быть достаточным, чтобы сделать экономически невыгодным однократное преодоление безопасности, и достаточно robust, чтобы один экземпляр сломанной безопасности не открывал тривиальные атаки на остальные консоли. Некоторые компромиссы при проектировании физической безопасности, а также решения, принятые Microsoft для этой цели, обсуждаются в следующей главе.

Мораль этой главы в том, что безопасность требует хорошо спроектированной системы. Хотя шифры стали достаточно сильными, чтобы сделать brute-force-атаки бессмысленными, системы выросли в сложности. Эта сложность повышает вероятность жизнеспособной protocol- или back door-атаки, но мало что делает, чтобы спасти пользователей от более традиционных атак подслушивания, rubber-hose и user-error.

Сноски

[1] [назад] Интервью Slashdot с Kevin Mitnick.
[2] [назад] Merriam-Webster OnLine Dictionary (www.webster.com).
[3] [назад] http://www.rsasecurity.com/rsalabs/challenges/factoring/rsa155.html
[4] [назад] http://www.rsasecurity.com/rsalabs/faq/3-1-5.html
[5] [назад] Код взят с http://vader.brad.ac.uk/tea/source.shtml#ansi
[6] [назад] Код взят с http://www.cc.jyu.fi/~paasivir/crypt/rciv/rc4article.txt. Внесены небольшие изменения пробелов, чтобы все поместилось на одной странице. Определение функции swap byte также не включено, но вы можете догадаться, что она делает, по ее имени.

Глава 8. Обратная разработка безопасности Xbox

В этой главе я опишу, как я победил первоначальную производственную версию системы безопасности Xbox, впервые встреченную в главе 6. Система безопасности была обнаружена после анализа FLASH ROM и понимания, что истинная последовательность аппаратной инициализации и расшифровки загрузочного образа каким-то образом скрыта вне FLASH ROM. Глава 7 ввела некоторые базовые криптографические понятия, которые будут полезны для понимания содержимого этой главы.

Извлечение секретов из аппаратуры

Скрытый загрузочный код в Xbox, как было заключено в главе 6, можно восстановить, подслушивая одну из следующих шин: (1) FSB, (2) основную шину памяти или (3) соединение Northbridge-Southbridge.

Формат Front Side Bus (FSB) процессора Pentium, используемого в Xbox, документирован в datasheets процессора Pentium III, доступных на Intel Developer Website. FSB - двунаправленная 64-битная шина данных примерно с пятьюдесятью адресными и управляющими сигналами, все они работают на 133 MHz. Шина использует сигнальную конвенцию, известную как AGTL+. Подслушивание этой шины - дорогое и трудное предприятие из-за большого числа сигналов и сложного физического форм-фактора. Жизнеспособные подходы включают: (a) установку процессора в специальную emulator break-out socket, которая стоит многие тысячи долларов, или (b) reverse engineering значения каждой дорожки FSB на материнской плате Xbox и припаивание короткого провода-щупа к каждому из почти ста сигналов. Кроме того, требуется логический анализатор, поддерживающий AGTL+-сигнализацию. Совокупность всех этих факторов заставила меня искать другую отправную точку для подслушивания.

Наш следующий кандидат на подслушивание, основная шина памяти, - это 128-битная шина данных плюс адресные и управляющие сигналы, работающая на 200 MHz с тактированием double data rate (DDR). Шина памяти использует сигнальную конвенцию, известную как SSTL-2. (Детали этой шины можно вывести, прочитав datasheet для детали памяти Samsung K4D263238M, доступный на веб-сайте Samsung Electronics.) Несмотря на более высокие скорости, подслушивание основной шины памяти, вероятно, легче, чем подслушивание FSB процессора, из-за пустых (запасных) посадочных мест памяти, спроектированных на материнской плате Xbox.

Относительно недорогой стандартный 100-выводной TQFP-адаптер (Thin Quad Flat Pack, прямоугольный корпус микросхемы со 100 выводами формы gull-wing) можно было бы припаять к пустым посадочным местам памяти. Эти адаптеры дали бы удобные точки зондирования для подключения логического анализатора. Проблема этого подхода в том, что можно захватить только данные, записываемые в основную память. Ключи расшифровки обычно являются read-only-данными, а read-only-данные пойдут прямо из скрытого boot ROM в кэш процессора, никогда не сохраняясь в основную память. Как только процессор закончит с cache line, содержащей ключ, она будет перезаписана, так что ключ никогда не должен покидать физический периметр процессора.

Подробнее о высокоскоростной передаче информации

Подслушивание и модификация данных на компьютерных шинах - мощная техника, которой трудно противостоять. Чтобы понять, как подслушивать, вам понадобится немного фона о том, как цифровая информация передается внутри компьютера.

Есть две основные категории сигнальных стандартов: single-ended и differential. Передача цифровой информации по проводу требует перевода в физические величины, такие как напряжение и ток. Классически сигналы определялись в терминах напряжений, измеряемых относительно общего опорного потенциала, называемого «ground». Такой вид сигнализации известен как single-ended или unbalanced signaling. К сожалению, идея опорной точки ground работает только тогда, когда сигналы меняются медленно относительно времени их распространения. В реальности каждое изменение потенциала сопровождается потоком тока. Законы природы требуют сохранения тока, то есть для каждого потока тока в одном направлении должен быть поток тока в обратном направлении. В single-ended signaling обратный ток, также известный как return current, должен найти путь обратно через «ground». На очень высоких скоростях пути возврата тока не обязательно следуют тому же пути, что и сигнальный ток. Этот дисбаланс приводит к искаженному сигналу.

Differential signaling борется с этой проблемой, используя два провода для передачи сигнала: один провод используется для сигнального тока, а другой - для явного пути return current. Дифференциальный подход позволяет расположить пути сигнала и возврата так, чтобы они отслеживали друг друга, гарантируя сбалансированность потока тока. Результат - более надежная система передачи сигнала ценой удвоения числа проводов.

Конкретный стандарт интерпретации напряжений как логических значений называется signaling convention. Почтенные сигнальные конвенции TTL и 3.3V CMOS были изобретены в эпоху, когда транзисторы работали настолько плохо, что требовались большие размахи сигнала. В последнее время набирает популярность множество новых и даже старых сигнальных конвенций, таких как SSTL (series stub terminated logic), GTL (gunning transceiver logic), LVDS (low voltage differential signaling) и PECL (pseudo emitter coupled logic). Эти высокоскоростные сигнальные конвенции учитывают тот факт, что электрические волны движутся медленно относительно скорости передачи данных. Они также учитывают тот факт, что электрические волны несут энергию, которую нужно рассеять при завершении ее пути, иначе энергия отразится и вызовет помехи с входящими волнами.

В высокоскоростных приложениях провода часто называют «transmission lines», чтобы подчеркнуть тот факт, что эти волны движутся медленно по сравнению со временем перехода сигнала (временем, требуемым сигналу для перехода между состоянием «1» и «0»). (Обратите внимание, что сравнение скорости делается относительно transition time сигнала, а не его общей signaling frequency.) Распространенная ошибка - думать, что effects transmission line можно игнорировать, потому что тактовая частота сигнала низкая. Даже если есть только один переход в год, проблемы все равно могут возникнуть, если длительность этого перехода занимает всего пикосекунду (одну триллионную секунды).

Хорошая новость для новичков в том, что последние FPGAs от поставщиков вроде Xilinx поставляются со встроенной поддержкой почти каждого широко развернутого сигнального стандарта. Другая хорошая новость в том, что сигнальные стандарты становятся все лучше документированы. Например, datasheets Xilinx FPGA иллюстрируют ожидаемые положение и значение терминирующих резисторов для каждого поддерживаемого сигнального стандарта. Следуя рекомендованным практикам в datasheet и application notes, можно использовать FPGA для подслушивания широкого диапазона сигналов. Просто помните, что ваши ответвления для подслушивания должны быть как можно короче, и вы не ошибетесь.

Третий потенциальный кандидат на подслушивание, соединение Northbridge-Southbridge, - это пара однонаправленных 8-битных дифференциальных шин, каждая всего с одним управляющим сигналом и одним тактовым сигналом. Шина использует сигнальную конвенцию HyperTransport и работает на 200 MHz с DDR-тактированием. Сигнальная конвенция шины была выведена из публично доступной информации на веб-сайте nVidia о nForce, чипсете, тесно родственном чипсету Xbox. Несколько измерений осциллографом, сверенных с открытыми спецификациями HyperTransport, доступными на веб-сайте консорциума HyperTransport, были использованы для проверки предположения, что сигнальная конвенция HyperTransport действительно используется.

Шина HyperTransport реализована на материнской плате Xbox со всеми сигналами параллельными и равномерно расположенными, решение, вероятно, продиктованное высокой рабочей скоростью шины. Это делает шину идеальной целью для подслушивания, за исключением того факта, что она работает на такой высокой скорости передачи данных. Подслушивание шины, работающей на этой скорости, требует особого внимания к stub length дорожек подслушивания (чтобы сохранить целостность сигналов), а также требует довольно дорогого логического анализатора или специальной схемы анализатора.

В конечном счете соединение Northbridge-Southbridge было выбрано первой шиной для подслушивания, потому что у него гораздо меньше проводов, чем у остальных, и поэтому оно требует наименьшего количества пайки. Соединение Northbrige-Southbridge имеет только десять уникальных сигналов, тогда как и FSB, и основная память имеют примерно по сто сигналов каждая. Пайка большого числа соединений не только потребляет много времени, но и сильно увеличивает риск аппаратных отказов из-за solder bridges или поврежденных дорожек. Таким образом, минимизация числа паяных соединений минимизирует риск сопутствующего повреждения материнской платы.

Подслушивание высокоскоростной шины

Я принял решение о подходе подслушивания HyperTransport в конце января 2002 года. Значительные технические вопросы этого подхода были:

- Подключение к высокоскоростной дифференциальной шине без нарушения signal integrity
- Поиск или сборка logging tool, который мог бы поспевать за скоростями данных 400 MB/s на шине HyperTransport
- Определение полярности и порядка битов дифференциальных дорожек шины HyperTransport на материнской плате

Подключение к шине при ограниченном бюджете

Первые два вопроса тесно связаны. Инструменты анализа и логирования высокоскоростных шин обычно имеют фирменные интерфейсы, которые потребовали бы специального адаптера к материнской плате Xbox. Последний вопрос, определение полярности и порядка битов, требует только большого объема post-processing и data massaging после того, как data logger подключен и функционирует.

HyperTransport - открытый стандарт, получивший признание в индустрии, что означает, что для шины доступны готовые protocol analyzers и logging tools. Один такой пример - HyperTransport protocol analyzer от FuturePlus. К сожалению, этот protocol analyzer стоил больше $25,000 на момент выполнения работы. Кроме того, protocol analyzer требует, чтобы целевая плата была специально спроектирована для размещения bus interface pod анализатора протокола.

Вместо покупки protocol analyzer и вложения времени и усилий в его адаптацию для использования с Xbox я построил собственный упрощенный. Эта задача осуществима, потому что протокол HyperTransport довольно прост. Реализация HyperTransport в Xbox использует две 8-битные однонаправленные шины, одну для передачи и одну для приема. Каждая шина имеет связанный с ней clock и strobe line. Сигнальный стандарт требует, чтобы корректные данные были представлены на каждом фронте clock. Начало нового packet обозначается выходом линий данных из idle state. Strobe line различает command и data packets. Все sideband signals, типичные для других шин, такие как address, read/write control, chip select и interrupt lines, обрабатываются в HyperTransport с использованием in-band command packets. Следовательно, всего десять дифференциальных сигналов (двадцать проводов) - это все, что нужно для подслушивания шины, отличная новость для хакеров.

Протокол HyperTransport достаточно прост, но как насчет поиска чего-то, что может и физически интерфейсироваться с шиной Xbox, и поспевать за скоростями 400 MB/s? Идеальным инструментом для построения этого HyperTransport bus tap была бы FPGA. Однако в то время не было доступно FPGA, которая могла бы поспевать за высокими скоростями данных, и, что важнее, не было FPGA, сертифицированной поставщиком для использования с HyperTransport. Теоретически FPGA Xilinx Virtex-II подошла бы для этого приложения, но продукт только что был запущен, и устройства были чрезвычайно дорогими и труднодоступными (сегодня low-end Virtex-II FPGA можно купить существенно дешевле ста долларов). Лучшая FPGA, которая у меня была под рукой в то время, была Xilinx Virtex-E FPGA, которую я ранее встроил в прототип платы сетевого маршрутизатора суперкомпьютера как часть своей диссертации. Плата сетевого маршрутизатора использовала сигнализацию CTT (Center Tap Terminated) для своих сетевых интерфейсов, а также имела на борту процессор Intel StrongArm для configuration, control и debugging purposes.

Поэтому вызов свелся к выяснению, как интерфейсировать сигналы HyperTransport с сигналами CTT, и как выжать производительность 400 MB/s из FPGA, которая не предназначалась для работы на таких скоростях.

Сигнальная конвенция HyperTransport, как оказалось, является близкой родственницей более распространенной конвенции LVDS (low voltage differential signaling), определенной стандартом TIA/EIA-644. Драйверы HyperTransport создают сигнал с дифференциальным размахом обычно 600 mV, центрированным вокруг common mode voltage 600 mV. LVDS-приемники, с другой стороны, могут понимать данные, имеющие дифференциальный размах больше 100 mV и common mode voltage где угодно между 50 mV и 2.35 V. Так что LVDS-приемники напрямую совместимы с драйверами HyperTransport! (Хотя Virtex-E поддерживает прямой интерфейс к LVDS-сигналам, я не мог этим воспользоваться, потому что имевшиеся у меня детали Virtex-E уже были спроектированы в систему, жестко разведенную под CTT-сигналы.) Если вы проектируете собственную tap board, лучшим подходом было бы использовать native LVDS-возможности FPGA вместо хака, описанного здесь. Кроме того, LVDS receiver должен быть расположен очень близко к материнской плате Xbox, чтобы не испортить целевые сигналы. Длинный кабель рассеял бы энергию из проводов и внес бы шум и отражения, которые могли бы заставить систему перестать функционировать.

А как насчет подачи сигналов на HyperTransport?

Приложение подслушивания, описанное в этой главе, требует только HyperTransport receiver. Приложения вроде «man-in-the-middle»-атак требуют устройство, которое может переопределять сигналы HyperTransport и вставлять ложный бит или два. Такое устройство осуществимо, потому что HyperTransport, как и LVDS, использует current-mode drivers. Другими словами, драйверы спроектированы подавать только измеренное количество тока в провод, независимо от напряжения, которое он создает. В нормальной ситуации это прекрасно работает, потому что импеданс провода превращает ток в напряжение в соответствии с законом Ohm. Однако токи могут суммироваться и взаимно уничтожаться. Антагонистический дифференциальный драйвер, который прикладывает overdrive current, уничтожающий предполагаемый сигнал, может быть подключен к линии HyperTransport. Такой overdrive можно выполнить с использованием гибкого программируемого I/O, предоставляемого в FPGAs вроде Xilinx Virtex-E и Virtex II.

Самым простым применением такого устройства переопределения шины было бы устройство, которое изменяет destination reset vector, когда он передается CPU, позволяя вам получить контроль над Xbox. Destination reset vector закодирован в одном байте, который следует за opcode «jump», расположенным по адресу `0xFFFF.FFF0`. Reset vector, вероятно, передается через детерминированное число тактов после деактивации reset, так что timing element для этой атаки может состоять просто из timer, тактируемого clock шины HyperTransport и синхронизированного с reset-сигналом. Такая «man-in-the-middle»-атака победит даже криптографически защищенную реализацию загрузочного блока с открытым ключом.

Решение проблемы доставки сигналов HyperTransport в FPGA - использовать микросхему преобразования сигналов. LVDS - популярный стандарт для интерфейсов LCD panel и backplanes, используемых в телекоммуникационных системах, поэтому доступно множество недорогих LVDS-to-CMOS converters. Конечно, желаемая сигнальная конвенция - CTT, но более пристальный взгляд показывает, что интерфейсирование CMOS drivers с CTT receivers на самом деле не является проблемой. CTT - current-mode signaling convention, которая подает +8 mA или -8 mA в transmission line 50 ohm, терминированную на 1.5 volts. Receiver - дифференциальный усилитель, который сравнивает reference termination voltage с voltage transmission line. В Virtex-E усилитель CTT receiver специфицирован как работающий, пока принимаемое напряжение отклоняется более чем на 200 mV вверх или вниз от reference voltage. Большинство CMOS transmitters, управляющих CTT-терминированной линией, без проблем смогут sourcing или sinking 8 mA тока в нагрузку 50 ohm. Также CMOS transmitters не должны иметь проблем с управлением проводом, терминированным в фиксированное напряжение. Таким образом, стандартную LVDS-to-CMOS converter chip можно использовать, чтобы взять сигналы HyperTransport материнской платы Xbox и подать их в плату, которую я ранее построил для своей диссертации. Микросхема, которую я выбрал, была Texas Instruments SN65LVDS386, а datasheets для этой микросхемы можно найти на веб-сайте Texas Instruments.

Прикрепление LVDS-to-CMOS converter chip к плате сделано восхитительно простым чистой разводкой, использованной для шины HyperTransport на материнской плате Xbox. Рисунок 8-1 - фотография того, как выглядят дорожки шины HyperTransport. Обратите внимание, что все провода идут параллельно и равномерно расположены. Некоторые провода, такие как clock (`TX CK/TX CX*` и `RX CK/RX CX*`) и strobe line (`TXD8/TXD8*` и `RXD8/RXD8*`), даже подписаны для нас с отметками полярности! Эта простая разводка позволяет использовать tap board, которую легко спроектировать.

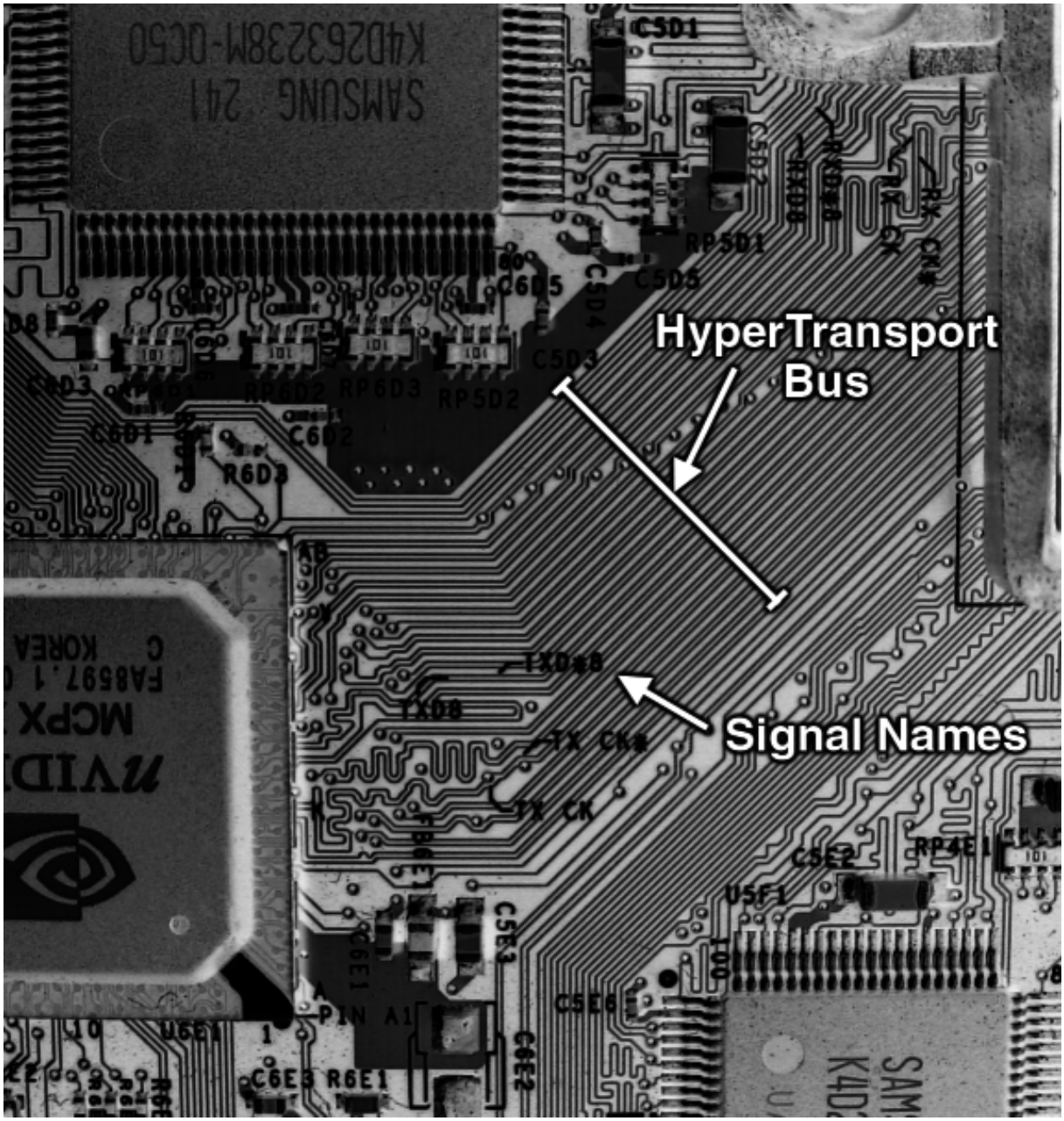


Рисунок 8-1. Дорожки шины HyperTransport, как они разведены на материнской плате Xbox.

Tap board содержит только LVDS-to-CMOS converter chip, немного circuitry power conditioning и набор дорожек, выведенных прямо к краю платы, которые имеют точно такой же шаг, как шина HyperTransport на материнской плате Xbox. Для идентичного шага, легкого выравнивания и монтажа я измерил размеры этих дорожек с помощью digital caliper tool. Рисунок 8-2 иллюстрирует размеры дорожек шины HyperTransport.

Рисунок 8-2. Размеры дорожек шины HyperTransport на материнской плате Xbox.
«mil» - это 1/1000 дюйма, или 25.4 микрона.

```
12 mil   13 mil   12 mil
| trace | space  | trace |

differential signal pair
6 mil trace
```

Измерения было немного хитро сделать. Мой подход состоял в том, чтобы измерить общую ширину шины и разделить ширину на число дорожек и промежутков, чтобы получить средний ожидаемый шаг и ширину дорожки. Затем я развел эти дорожки в PCB CAD program и распечатал layout на

бумаге в масштабе 1:1. Я сравнил напечатанные дорожки с дорожками платы и сделал несколько ручных корректировок. (Обратите внимание, что многие принтеры имеют некоторую небольшую scaling error, поэтому, если вы пробуете это, откалибруйте себя, распечатав несколько длинных линий известной длины и измерив их. Принтеры могут иметь разные ошибки масштабирования по горизонтальной и вертикальной осям, так что обязательно печатайте линии в обоих направлениях.)

Проектировать собственные платы довольно легко с правильным программным обеспечением. Больше узнать о том, как делать собственные платы, можно, прочитав приложение C, «Getting Into PCB Layout».

После завершения процесса выбора компонентов design и layout HyperTransport tap and signal conversion board заняли всего еще несколько часов. Схему дизайна платы можно увидеть на рисунке 8-6. Затем плата была изготовлена по заказу, размещенному через Internet. Многие board houses предлагают доступные, быстрые услуги изготовления плат, принимающие designs плат в формате Gerber file через email или ftp upload. В этом случае я получил две копии платы за пять дней по цене $33 за плату (см. приложение C, «Getting Into PCB Layout», для дополнительной информации о том, как строить собственные платы). Эта цена включает только цену вырезания платы в квадратный кусок. Однако мне нужно было, чтобы сторона платы с HyperTransport tap имела специальную форму, облегчающую монтаж платы без вмешательства в существующие компоненты на материнской плате Xbox. Мне также нужно было, чтобы сопрягаемый край платы был скошен так, чтобы плата устанавливалась под небольшим углом, для упрощения задачи припаивания tap board к материнской плате. Я использовал belt sander, чтобы вручную вылепить край в форму, описанную на рисунке 8-3. При формовке плата должна была быть ориентирована так, чтобы abrasive belt belt sander сначала контактировала со стороной платы с дорожками, чтобы предотвратить срывание медных дорожек с платы. Будьте осторожны при использовании belt sander для формовки маленьких плат вроде tap board - belt sander с тем же успехом может случайно отшлифовать ваши пальцы.

Рисунок 8-3. Формирование края HyperTransport tap board.

```
front view:
  clearance for motherboard components
  bare traces brought out to board edge
  motherboard

side view:
  beveled edge for angled mounting
  soldering iron tip
```

После формирования скошенного края все детали были припаяны к плате. (См. приложение B, «Soldering Techniques».)

Готовую tap board теперь нужно было прикрепить к материнской плате Xbox. Этот критический шаг был, возможно, самым трудным. Сначала материнская плата Xbox была подготовлена с использованием мелкозернистой наждачной бумаги, чтобы снять зеленую soldermask, открыв яркую голую медь целевых дорожек. Затем на эти дорожки был нанесен flux, и тонкий слой припоя был нанесен с использованием горячего жала паяльника.

Процедура, которую я использовал для прикрепления tap board к материнской плате, показана на рисунке 8-4. Подготовленная tap board была прихвачена к материнской плате в приблизительном месте и под приблизительным углом с использованием тонкого (30 AWG) провода, припаянного между дорожкой на tap board и материнской платой. Прихватывающий провод служит только временной помощью для удержания платы на месте и будет удален, поэтому не имеет значения, если провод замыкает несколько дорожек. После прикрепления провода я осторожно отрегулировал положение tap board на материнской плате, нагревая провод, чтобы освободить его связь и избежать подъема медных дорожек. (Я использовал микроскоп, чтобы помочь определить оптимальное выравнивание.) Когда я был удовлетворен положением платы, я нанес прочную

эпоксидку на joint платы, чтобы удержать все на месте. Эпоксидка должна отвердеть и сформировать жесткий, stiff joint. (Обратите внимание, что некоторые эпоксидки при неправильном применении отверждаются в гель; это неприемлемо, поскольку вся механическая целостность joint должна исходить от эпоксидки, а не от solder joints.) Я использовал Miller-Stephenson Epoxy formula 907, и она схватывается с достаточной прочностью, чтобы я мог поднять Xbox за tap board и не нарушить tap connection.

Рисунок 8-4. Процедура пайки tap board.

```
1. 30-AWG wire soldered across bare traces to tack board at desired angle
2. Apply epoxy and let cure
3. Remove tacking wire; clean up bridges
4. Solder all connections
```

После отверждения эпоксидки я удалил временный прихватывающий провод, который использовался для удержания tap board на месте, и очистил голые сопряженные дорожки небольшим количеством solderwick и flux. Последний шаг пайки дорожек tap board к голым дорожкам материнской платы теперь ничем не отличался от пайки любого surface mount component на плату; большинство стандартных техник, описанных в приложении B, напрямую применялись к этой ситуации. Рисунок 8-5 показывает, как выглядит готовая сборка.

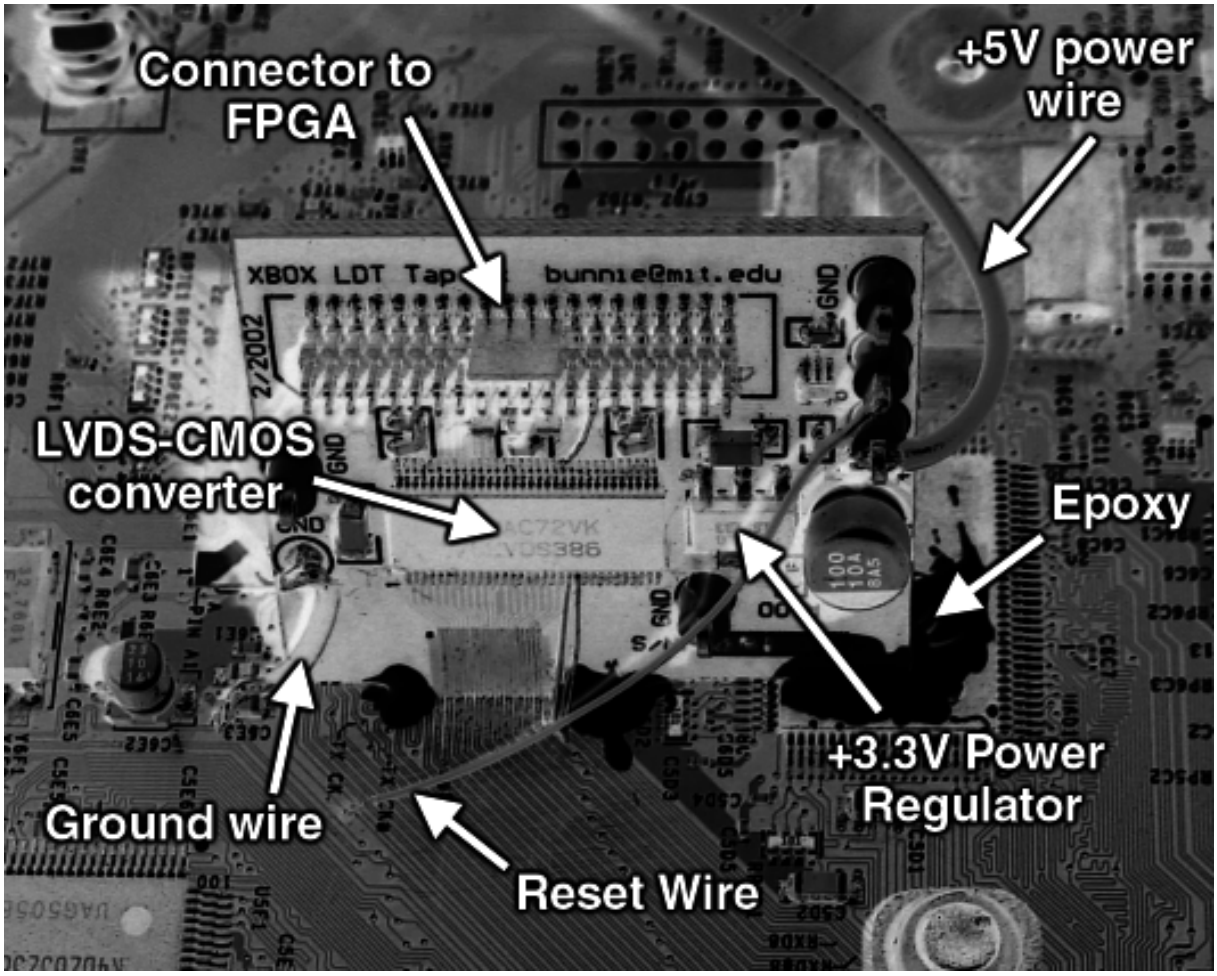


Рисунок 8-5. HyperTransport tap board, установленная на материнской плате Xbox.

Сборка регистратора данных

Второй вызов подслушивания шины HyperTransport - приобретение или сборка logging device, который может поспевать за скоростью данных 400 MB/s шины. Учитывая мой бюджет, я решил, что мой единственный вариант - построить logger, поскольку покупка любых инструментов с достаточной производительностью для этой работы была далеко за пределами моего бюджета.

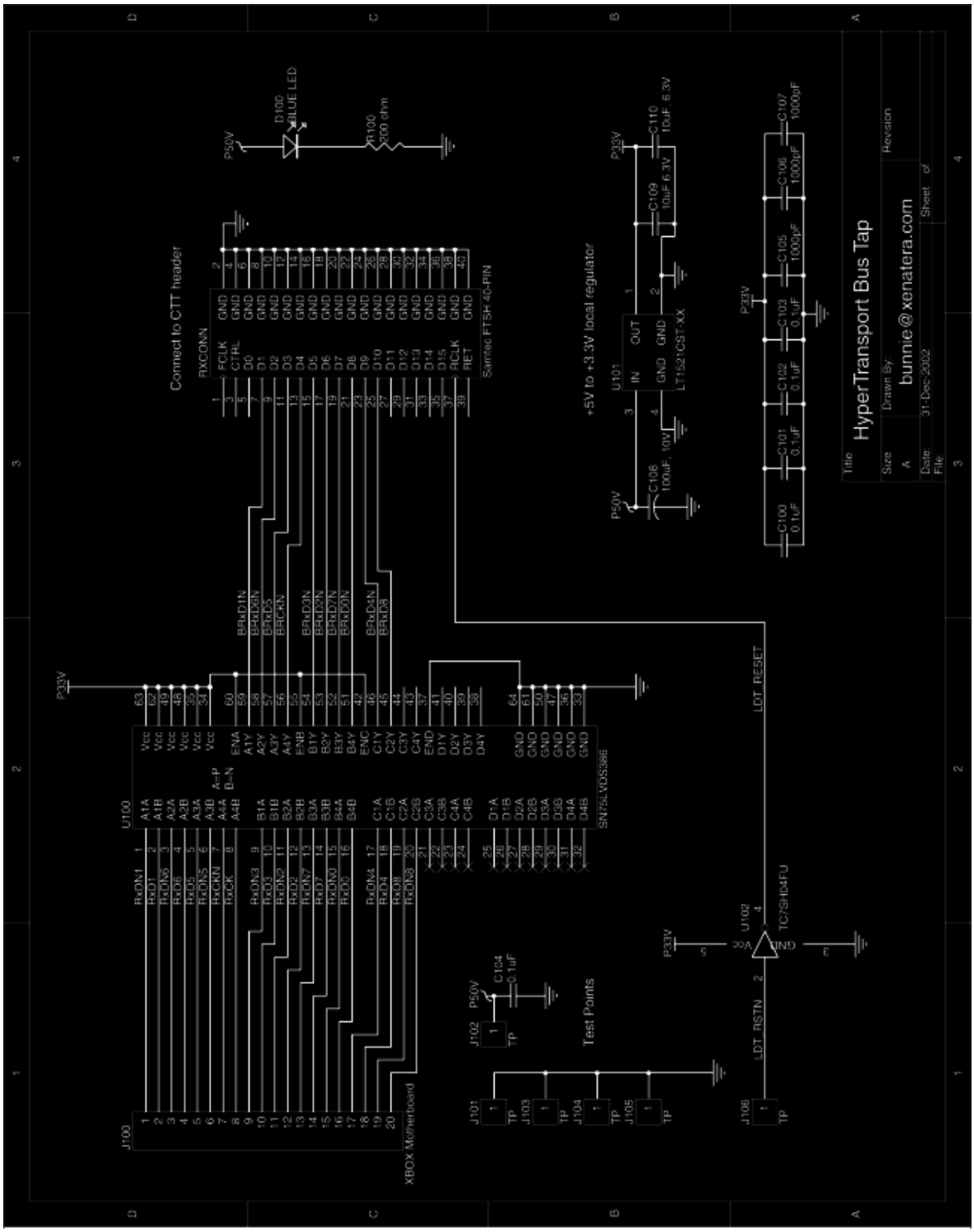


Рисунок 8-6. Схема HyperTransport tap board.

При сборке logging device я остановился на использовании Virtex-E FPGA, интегрированной в плату, которую я построил ранее. Однако единственная проблема использования Virtex-E FPGA в том, что производительность FPGA (как специфицировано в databook) недостаточна, чтобы поспевать за шиной HyperTransport. К счастью, FPGAs хорошо overclock, потому что их производственный запас очень консервативен, а производительность FPGA в значительной мере ограничена задержками распространения сигналов в configurable wiring fabric. В результате некоторые ключевые performance-limiting paths можно вручную идентифицировать и компенсировать с использованием soft delay lines и selectively inverted clocks. Самые чувствительные к производительности blocks можно hand-place для оптимизации задержек, в то время как compiler и automated place-and-route tool обрабатывают некритические части схемы. Рисунок 8-7 показывает общий design, который использовался для захвата данных на шине HyperTransport.

Рисунок 8-7. Блок-схема data logger, построенного в Xilinx Virtex-E FPGA.

```
HyperTransport Data (200 MHz DDR) + HyperTransport Clock
  -> Dual-edge triggered data demux
  -> Quad-phase data demux
  -> Align vs Clock Phase
  -> 2 kB x 36 deep FIFO memory
  -> logged data

RESET / HyperTransport bus reset line
  -> 32-bit up counter
  -> compare and trigger
  -> data sequence count / alignment / read strobe

Clocking:
  2x200 MHz SDR
  4x100 MHz SDR

Adjust clocking phase on a bit by bit basis to compensate for FPGA delays.
```

Design довольно прост по идее: взять high speed data с шины HyperTransport и тактировать их в четыре фазы clock четвертной скорости, создавая поток данных, который в четыре раза медленнее, но в четыре раза шире. Это ограничивает весь hand-placing и tweaking только первыми несколькими входными flip flops. Затем данные заново выравниваются с использованием набора delays и rotators и сохраняются по одному фрагменту в first in, first out (FIFO) memory. Сигнал, запускающий начало FIFO capture, генерируется timer-comparator, который начинает считать вверх с первого reset. Длинные окна данных можно захватывать, concatenating результаты нескольких запусков, каждый с capture trigger point, задержанным относительно предыдущего. Более поздняя оптимизация, примененная к trigger circuit, - функция «do not store zeros» (DNSZ). В режиме DNSZ данные, состоящие из одних 0, не сохраняются в FIFO. Это помогает отсеивать все idle data на шине HyperTransport. Получающиеся data traces являются time-stamped серией 32-битных words.

Самой трудной частью дизайна FPGA data logger была калибровка delays на input paths. Delay calibration была выполнена с использованием осциллографа для зондирования маленького окна данных на шине HyperTransport. Wire delays и byte-wide rotations подстраивались до тех пор, пока зондированные данные не совпали с log data. Этому процессу помогал тот факт, что во время idle times распространенная последовательность команд повторялась на шине каждые несколько сотен микросекунд, что служило calibration reference.

Определение порядка и полярности шины

Финальный вызов после логирования данных - выяснить порядок сигналов на шине HyperTransport и их полярности. Обратите внимание, что хотя два самых важных сигнала шины HyperTransport на материнской плате Xbox подписаны для нас, оставшиеся восемь линий данных имеют неоднозначную полярность и порядок битов.

Правильная полярность восьми сигналов данных была определена наблюдением шаблона битов idle bus data. Шина HyperTransport проводит большую часть времени в idle state, так что это нетрудно. Если idle pattern должен быть all 0s, тогда любая bit position, показывающаяся как 1, имеет инвертированную полярность. Это было исправлено в hardware вставкой inversion term в FPGA на соответствующем проводе.

Определение правильного порядка битов, однако, намного труднее. Работая в предположении, что данные, приходящие через шину HyperTransport, должны в значительной части приходить из FLASH ROM, был выполнен 1's count на byte by byte basis. Теория состоит в том, что порядок шины является pure permutation, то есть число двоичных 1 в байте сохраняется между данными FLASH ROM и данными, захваченными logger. Шаблоны 1's counts были выровнены друг с другом, чтобы определить candidate regions of correspondence между FLASH ROM и logged data. К счастью, первые несколько words, приходящих через шину HyperTransport, являются некоторыми chipset-specific initializations, расположенными рядом с нижней частью FLASH memory, поэтому поиск набора patterns, которые правильно выравнивались, не занял слишком много времени. Набор bytes из каждого ROM и logger был сведён в таблицу, и с помощью короткой C-программы столбцы битов транспонировались, пока не был найден порядок, при котором все row values совпали.

Осмысление захваченных данных

Теперь, когда корректные data traces извлечены, остается проблема расшифровки смысла всего этого. Прежде чем делать это, давайте резюмируем, что мы знаем о данных, собранных до сих пор.

- Temporal correlation. Залогированные данные в макроскопическом масштабе должны иметь сильную временную корреляцию с ожидаемой последовательностью событий инициализации: jam table initialization, затем decryption step, затем execution from RAM. Области log traces, соответствующие каждому из этих событий, можно определить, просто наблюдая, когда происходят большие bursts of activity, за которыми следуют области тишины.
- Transaction lengths. Поскольку процессор Pentium имеет и data cache, и instruction cache, все fetches на шине HyperTransport к FLASH ROM или скрытому boot ROM должны приходить в bursts of traffic четной длины.
- Guaranteed ordering. Собранные данные time stamped и хронологически корректны, поэтому если первую инструкцию, fetched в reset vector, можно идентифицировать в data logs, положение и структура оставшихся инструкций могут быть выведены.

Изначально я пренебрег проверкой макроскопической организации данных, приходящих через шину HyperTransport, и это создало мне некоторые проблемы. Упрощенная block diagram logging machine на рисунке 8-7 предусматривала бы reset log FIFO каждый раз, когда reset выполняется на шине HyperTransport. Это кажется хорошей идеей, однако изначально я неправильно предположил, что reset шины HyperTransport выполняется только один раз при подаче питания. В реальности reset шины HyperTransport выполняется второй раз после шага jam table initialization. Поэтому, когда я впервые начал смотреть traces, все, что я видел, было encrypted data плюс россыпь кода, ничего из чего нельзя было реально выровнять каким-либо логическим образом с boot vector.

Представьте, как это было разочаровывающе! Я сделал шаг назад и наблюдал события шины HyperTransport на осциллографе с разверткой в миллисекундах на деление. Я увидел, что был более ранний reset pulse, и после настройки trigger mechanism на захват только первого pulse boot instruction было легко идентифицировать. Шестнадцать байтов по адресу `0xFFFF.FFF0` в secret ROM оказались идентичны тем же шестнадцати байтам во FLASH ROM. С этого момента я отслеживал текущее значение program counter, выполняя много грязного tracing и disassembling с bookkeeping, чтобы я мог поместить каждый instruction block в правильное место в памяти. Каждый cache line fetch состоял из 16 или 32 последовательных bytes памяти, что давало характерный time stamp pattern data logger, помогавший процессу reverse engineering.

Еще инструменты ремесла: инструменты анализа программного обеспечения

В какой-то момент ваших хакерских опытов вы неизбежно столкнетесь с необходимостью дизассемблировать код assembly language. Меня познакомили с отличным инструментом для этой работы некоторые друзья software hackers в январе 2002 года, когда я выполнял reverse engineering безопасности Xbox. Инструмент называется «IDA Pro» от Ilfak Guilfanov, продается DataRescue Corporation (http://www.datarescue.com/idabase/). IDA Pro способен дизассемблировать не только x86 code, но и огромное разнообразие кода embedded processors. Качество вывода IDA Pro также очень высокое: code segments автоматически аннотируются и организуются для читаемости. IDA Pro также имеет широкий набор полезных и забавных tools. Некоторые из моих любимых включают способность автоматически pattern match code library signatures с function calls и способность переходить по jumps нажатием клавиши.

Другим инструментом, который был весьма удобен во время code analysis, был HackMan. HackMan - freeware от TechnoLogismiki Corporation (http://www.technologismiki.com/hackman/). Номинально это «hex editor», то есть file editor, позволяющий напрямую манипулировать binary data, но у него есть множество уникальных возможностей, выходящих далеко за пределы простого editing. Например, HackMan имеет встроенный disassembler. Disassembler не такой мощный, как IDA Pro, но он interactive с hex editor. Это позволило мне быстро тестировать candidate cache lines на valid code, пока я tracing through data logs, одновременно собирая финальный binary image secret ROM.

После нескольких часов просеивания traces в поисках cache lines у меня было достаточно кода, чтобы подать его в disassembler. (См. боковую вставку об инструментах анализа программного обеспечения для дополнительной информации о disassembler, который я использовал.)

После некоторого data massaging и существенной помощи от нескольких on-line hacker friends мы определили, что используемый шифр - RC-4/128. RC-4 - symmetric cipher, и ключ должен был храниться где-то в Xbox, но у меня были трудности с попыткой идентифицировать ключ в data stream. Ключ, казалось, занимал cache line fetches, которые были shared с кусками кода, которые в то время я не мог mapped to a definitive location.

Поскольку ночь затягивалась, а я уставал смотреть на hex digits, я решил попробовать кое-что, что никогда не должно было сработать. Я адаптировал программу расшифровки RC-4, чтобы расшифровывать target image во FLASH ROM с использованием ключа, derived from sliding window внутри data log. Это довольно brute-force-подход, поскольку он требует десятки тысяч decryptions (по одной для каждого byte в log) для поиска по всему data stream. Я автоматизировал процесс, направив output RC-4 decryption в histogram routine. Если ключ не совпадал, output должен был быть статистически «white». Другими словами, histogram output должен был показать, что все values примерно equally probable для несовпадающего ключа. Однако если ключ был правильным, histogram должен был быть biased, с некоторыми values значительно более популярными, чем все остальные values.

В конце концов я закончил программу `trykeys` для выполнения этого brute-force search около 5 AM. С мутными глазами и усталый, я решил дать программе test run перед тем, как бросить все на ночь. Представьте ошеломленное выражение моего лица, когда я смотрел на output программы, пока она перемалывала candidate data stream:

```
$ ./trykeys.exe ms4.bin binout.full
.........................................................
.....................found possible key combo: avg 96, min 5,
offset 8745.............................................
.........................................................
```

Образ FLASH ROM называется `ms4.bin`, a binary data logger trace называется `binout.full`. Программа `trykeys` идентифицировала статистически отличающуюся histogram (со средним значением 96 и минимальной высотой bucket 5) для расшифровки ROM image с использованием, как test key, данных, начинающихся со смещения 8745. Затем я изолировал candidate key из data stream и проанализировал расшифрованный output с использованием candidate key. Output выглядел как реальный, корректный код. Я нашел ключ в скрытом boot sector, хранящемся в Southbridge chip! Через несколько дней, после того как я немного поспал и нагнал школьную работу, я закончил корректный анализ data stream и собрал image всего secret boot sector.

Имея RC-4 key секретного boot code на руках, я получил способность генерировать FLASH ROM images, которые могли быть приняты любым Xbox того времени. Следствие в том, что весь trust mechanism Xbox можно было нарушить, просто переопределив или заменив ROM на материнской плате Xbox. Это достигается использованием test structures, предоставленных Microsoft для переопределения FLASH ROM во время производства в целях тестирования и диагностики. Xbox должны сходить с производственной линии со скоростью один каждые пару секунд, поэтому Microsoft спроектировала набор quick-connect test points, позволяющих FLASH ROM override. Способность загрузиться в alternate ROM image ценна для запуска production test programs с использованием native Xbox CPU. Физическая структура реализации интерфейса Xbox LPC позволяет пользователям, а также контрактному производителю Microsoft, устанавливать правильно спроектированное FLASH ROM override device без какой-либо пайки.

Юридические вызовы взлома

Оглядываясь назад, взлом Xbox был менее труден технически, чем социально и юридически. После извлечения секретного ключа из Southbridge chip я встретился со своим научным руководителем, Prof. Tom Knight, в MIT Artificial Intelligence Laboratory, чтобы обсудить свои результаты. Мой руководитель указал, что моя работа, возможно, нарушает DMCA, поэтому до публикации мы связались с юридическим отделом MIT за консультацией. MIT Legal в конце концов ответил, что DMCA делает дело слишком рискованным и что я должен публиковаться как частное лицо, несмотря на то что моя работа была выполнена в MIT как часть моего исследования computer architecture. Я отчаялся, думая, что никогда не смогу позволить себе юриста и никогда не смогу опубликовать свои результаты, но затем Prof. Hal Abelson связал меня с Electronic Frontier Foundation (EFF). В результате Lee Tien и Joe Liu из EFF и Boston College были назначены помогать мне публиковать мою работу. Последовали месяцы обсуждений и позиционирования. Это была битва на два фронта: мы должны были убедить MIT принять работу, одновременно пытаясь умиротворить Microsoft. Через четыре месяца MIT капитулировал после ободряющего review моей работы Microsoft и подавляющей поддержки моих коллег и профессоров по лаборатории. MIT решил, что я могу публиковать свою работу как студент MIT, вместо того чтобы делать это как независимая сущность. Результатом пяти месяцев юридического stalemate стал technical memorandum AI Laboratory, за которым последовала академическая презентация работы на конференции Cryptographic Hardware in Embedded Systems (CHES) в августе 2002 года.

Хотя конец этой истории может быть счастливым, все могло быть совсем иначе, если бы не поддержка моего руководителя, моей лаборатории и талантливых юристов EFF. DMCA проводит нечеткую линию между rogue hacker и legitimate researcher; возможно, без одобрения MIT я не смог бы удовлетворить research exemption DMCA, и мое исследование никогда не было бы опубликовано, или оно могло бы быть опубликовано и оспорено Microsoft. Свобода слова относится ко всем, а не только к тем, кому повезло сидеть в ivory towers уважаемых академических учреждений. Есть бесчисленное множество других, кто также работал над Xbox с отличными результатами, но их голоса навсегда останутся молчать за занавесом DMCA.

Очевидно, способность переопределить trust mechanism, используемый в Xbox, имеет липкие юридические последствия. Хотя моим намерением было в основном удовлетворить свое любопытство и во вторую очередь запускать собственный код на Xbox в рамках моих fair-use rights, другие люди имеют желание копировать игры, а также изменять и перераспространять защищенный авторским правом kernel code Microsoft. Поскольку шифр слеп к своему применению, извлечение RC-4 key включает все приложения одинаково. В результате я связался с Electronic Frontier Foundation (EFF), чтобы они помогли мне разобраться с юридическими вопросами. Юридический процесс медленный и тяжеловесный. Я извлек ключ в феврале 2002 года, и потребовалось почти до июня, прежде чем мне разрешили опубликовать результаты моего исследования на подходящем академическом форуме.

Никогда я не испытывал столько шума из-за 128 bits. Digital Millennium Copyright Act (DMCA) of 1998 навсегда изменил ландшафт аппаратного взлома. Reverse engineering раньше был защищенным действием, считавшимся частью того, что делает рынок здоровым и конкурентным. Теперь возня с криптографической системой безопасности и ее обход для осуществления ваших fair-use rights в приватности собственного дома может принести вам тысячи долларов штрафов и lawsuits. Я настоятельно рекомендую прочитать главу 12, «Caveat Hacker», чтобы вы понимали свои юридические права и обязанности.

Безопасность через неясность

Техника, использованная Microsoft в первой версии безопасности Xbox, является отличным примером security through obscurity. Сильный шифр, RC-4/128, был использован для шифрования ROM image, чтобы предотвратить анализ содержимого ROM или создание собственных ROM. Однако RC-4/128 - symmetric cipher, что означает, что Xbox должен содержать decryption key, также пригодный как encryption key. Этот decryption/encryption key - важный фрагмент информации, buried inside secret boot ROM. Сокрытие этого ключа - security through obscurity: как только ключ найден, шифр теряет смысл и вся безопасность потеряна.

Настоящая безопасность требовала бы, чтобы пользователь имел доступ к каждому отдельному фрагменту Xbox и все равно не мог зашифровать собственный корректный FLASH ROM image. Это подразумевает, что какой-то секрет должен храниться вне Xbox. Криптография с открытым ключом была изобретена именно для такого сценария. Если бы Microsoft использовала шифр с открытым ключом для шифрования или подписи boot code Xbox, тогда знание всего содержимого secure boot ROM было бы бесполезно, поскольку главный секрет, private key Microsoft, остается безопасно вне нашей досягаемости в каком-нибудь хранилище в Redmond, Washington.

Однако есть и положительная сторона. Следующая глава представляет находки моих коллег, многие из которых включают обнаружение back doors в последовательности инициализации Xbox. Эти backdoors позволяют вам запускать собственный код на Xbox без включения доступа к защищенным авторским правом works Microsoft и без включения копирования игр. Следующая глава также представит Xbox security version 1.1, которая была взломана всего за несколько дней Andy Green в UK.

Глава 9. Пробраться через черный ход

Полный диапазон жизнеспособных атак на Xbox слишком многочислен, чтобы описывать его в этой книге. Xbox основана на архитектуре PC, сложной, эволюционировавшей архитектуре, изначально спроектированной без мысли о безопасности. Многие классические аппаратные дыры безопасности, используемые хакерами smart card, такие как modulation power supply, sideband attacks и clock glitching, насколько мне известно, на Xbox даже не затрагивались. (Больше информации об этих слабостях безопасности можно найти в материалах конференции Cryptographic Hardware in Embedded Systems (CHES), в серии Lecture Notes in Computer Science (Springer-Verlag).)

К сожалению, производители консолей и защищенных PC не обеспокоены аппаратными слабостями безопасности, потому что аппаратные атаки «слишком трудны для среднего потребителя» и поэтому представляют малую угрозу. Хотя верно, что исследование атаки требует умелого хакера с правильными инструментами, реализация атаки может быть очень дешевой и легкой. Мне вспоминается притча, где механик, вызванный починить важный кусок сломанного оборудования, тратит час на изучение ситуации и ремонтирует машину, постучав по ней точно в правильном месте. Получив счет на $1,000, владельцы машины требуют объяснить, почему постукивание стоит так дорого. Механик отвечает: «Постукивание стоит десять центов. Знание, где постучать, стоит $999.90.» Следствие этой притчи в том, что выполнить постукивание для ремонта машины мог бы кто угодно, имея конкретные инструкции.

Атаки безопасности часто такие же: трудно выяснить, легко поделиться и реализовать. Производителям защищенной аппаратуры также следует беспокоиться об основном принятии реактивных политик к хакерским вторжениям. Многие хакеры работают тайно и держат свои методы и результаты в тишине, чтобы поставщики не могли разработать правильные контрмеры. Эти хакеры также поддерживают библиотеку известных атак и back doors, раскрывая только по одной за раз, так что поставщики с реактивными политиками аппаратной безопасности всегда играют в догонялки.

Предыдущая глава описала мою eavesdropping-атаку на механизм безопасности Xbox, которая в конце концов дала RC-4 key, скрытый в блоке secret code. Эта глава описывает несколько других атак, доступных на Xbox, которые были придуманы моими коллегами, а также атаку, выполненную на пересмотренной схеме безопасности Xbox, далее называемой security version 1.1.

Черные ходы и дыры безопасности

Класс back door-атак на Xbox использует фундаментальную слабость в том, как аппаратура инициализируется secret boot code.

Комментарий о соглашениях именования

Хакерские сообщества часто изобретают собственную терминологию для важных понятий, которая может отличаться от сообщества к сообществу и от отраслевой стандартной терминологии. Ниже приведен список терминов, принятых сообществом Xbox-Linux. Любые отклонения от терминологии, которую я использую в этой книге, отмечены.

- X-code: opcodes jam table; opcodes, используемые secret Southbridge (MCPX) boot ROM для инициализации аппаратуры Xbox.

• 2BL: Second boot loader. Это код, который расшифровывается secret boot ROM. Он называется second boot loader, потому что основная ответственность этого кода - расшифровать и распаковать kernel image.
• Flash Boot Loader: В security version 1.1 это промежуточный boot loader между secret boot ROM и 2BL. FBL проверяется lightweight hash против hard-coded value внутри secret boot ROM. В результате FBL нельзя изменить без изменения silicon MCPX. FBL отвечает за проверку digital signature на всех критических частях FLASH ROM.
• Kernel: код kernel Xbox. Он хранится в FLASH ROM сжатым и зашифрованным.
• Version 1.0 security: оригинальная система безопасности Xbox, использующая RC-4 encryption на 2BL.
• Version 1.1 security: вторая система безопасности Xbox, использующая TEA hash для проверки областей FLASH ROM. Самая ранняя manufacturing date, замеченная на коробках с version 1.1 security, примерно август 2002 года.

Эта слабость проистекает из того факта, что аппаратная инициализация выполняется через мощный interpreter opcodes jam table, который хранит свои команды в непроверенном cleartext.

Рисунок 9-1. Opcodes jam table по отношению к остальной FLASH ROM.

```
FLASH ROM contents (simplified):
  jam table opcodes (unencrypted)
    opcode argument 1 argument 2
    opcode argument 1 argument 2
    ...
    opcode == memory read/write, PCI read/write, IO read/write, etc.

  2BL (encrypted)
  kernel (compressed, encrypted)
  decoy code

secret code (in Southbridge)
```

Атаки Visor Jam Table

Один класс атак на Xbox включает модификацию последовательности аппаратной инициализации. Вспомните, что аппаратная инициализация Xbox выполняется посредством interpreter opcodes, который получает свои команды из незашифрованной части FLASH ROM, называемой «jam table». Связь записей jam table с остальной FLASH ROM иллюстрирована на рисунке 9-1. Записи jam table хранятся как тройки `opcode, arg1, arg2` около самых низких адресов FLASH ROM. Доступные opcodes включают функции memory read и write ко всем address spaces в архитектуре x86. Поскольку jam tables хранятся незашифрованными и никогда не проверяются на модификацию, opcodes можно вставить в jam table, которые могут «seed» память Xbox rogue instructions или modified hardware state перед RC-4 decryption 2BL из FLASH ROM.

Одно применение модификации jam table - восстановить plaintext kernel без знания RC-4 key. Хакер, известный только как Visor, первым описал мне этот подход. Вот краткое изложение подхода Visor:

1. Загрузите Xbox обычным образом. Часть обычного процесса загрузки поместит расшифрованный kernel image в основную память.
2. Сохраняя питание Xbox, переключите содержимое jam table FLASH ROM на альтернативную таблицу, которая копирует области основной памяти в место, которое легко наблюдать, например на шину FLASH ROM.
3. Выполните soft-reset CPU Xbox. Это заставляет аппаратную re-initialization без стирания основной памяти.

4. Запишите содержимое основной памяти по мере выполнения модифицированной программы jam table.

Dynamic switchover содержимого FLASH ROM, требуемый на шаге 2, можно выполнить посредством ROM emulator или используя oversized ROM с лишними address bits, подключенными к банку переключателей.

Visor также описал, как jam table можно использовать как часть сложного хака для получения контроля над instruction pointer (IP) Xbox. Чтобы лучше понять этот хак, мы дальше исследуем, как secret boot code обрабатывает случай invalid FLASH ROM image.

После расшифровки 2BL в FLASH ROM image secret boot code проверяет magic number в месте около конца 2BL. Для invalid FLASH ROM image это число не совпадает и заставляет CPU перейти к короткой последовательности инструкций, расположенной по адресу `0xFFFF.FFFA`. Этот набор инструкций продолжается до самого последнего адресуемого места в физической памяти, location `0xFFFF.FFFF`. Как только CPU выполняет эту последнюю инструкцию invalid ROM image, он должен упасть и остановить выполнение из-за code segment boundary error, когда IP rollover from `0xFFFF.FFFF` to `0x0000.0000`. Однако этого не происходит; напротив, CPU счастливо пытается выполнить любую инструкцию, корректную или некорректную, расположенную по адресу `0x0000.0000`. Номинально эта инструкция некорректна, и CPU все равно останавливается из-за instruction fault. Однако корректную инструкцию можно поместить туда во время последовательности инициализации jam table Xbox, используя opcode memory write jam table. Таким образом, повреждая или стирая encrypted FLASH ROM image и модифицируя jam tables для вставки jump instruction к вашему собственному незашифрованному коду во FLASH ROM, можно получить контроль над CPU IP Xbox, никогда не касаясь шифра или любой похожей technological measure, effectively controls access to a copyrighted work. Следовательно, этот хак, возможно, легален по DMCA. Я говорю «возможно», потому что DMCA часто является расплывчатым законодательным актом, и судебных прецедентов для прояснения неоднозначностей мало. Аргумент в пользу законности этого подхода состоит в том, что никакой существенный copyrighted code Microsoft никогда не расшифровывается и не выполняется. Единственное исключение - часть secret boot ROM, которая должна выполняться, потому что она hard-wired в silicon Southbridge. См. главу 12, «Caveat Hacker», для более глубокого обсуждения юридических вопросов, с которыми сегодня сталкивается хакерское сообщество.

Атака MIST Premature Unmap

Источник: презентация Andy Green на 19th Annual Chaos Communication Congress о взломе безопасности Xbox.

Чтобы защитить secret boot code на случай, если хакер получит контроль над Xbox, secret boot code в Southbridge chip unmaps itself незадолго до выхода. Другими словами, он навсегда скрывает себя от системы, когда заканчивает выполнение. Поэтому пользовательская программа, пытающаяся обратиться к любому из верхних 512 bytes памяти, увидит decoy block во FLASH memory вместо secret boot code. Michael Steil, руководитель проекта Xbox-Linux, обнаружил способ использовать эту возможность.

Процесс unmapping выполняется записью в `0x8000.8008`, hardware register в PCI configuration space. Базовая стратегия - включить opcode jam table, который пишет в `0x8000.8008` и unmaps secret boot code до завершения последовательности инициализации. Поскольку кэши в это время выключены, процессор начнет fetching и executing instructions из decoy block. К счастью, decoy block можно свободно модифицировать, поскольку он является частью FLASH ROM. Однако загвоздка в том, что interpreter jam table блокирует записи в location `0x8000.8008`, так что это не должно

работать. Однако bug в decoding PCI configuration space в Southbridge chipset заставляет unmap instruction responding to multiple aliased addresses. В частности, bitfield «function» игнорируется. Поэтому запись в `0x8000.8X08`, где X не равно 0, также делает трюк, и эти записи не блокируются interpreter jam table. Следовательно, чтобы получить контроль над CPU IP с использованием MIST hack, нужно модифицировать decoy block во FLASH, чтобы он содержал ваш код, затем добавить соответствующий opcode jam table, чтобы unmap secret boot ROM во время аппаратной инициализации.

Microsoft наносит ответный удар

Открытие security holes побудило многих предположить, что Microsoft быстро сменит свою схему безопасности. В августе 2002 года Xbox с новой материнской платой тихо начали появляться в Австралии. Первое официальное слово о новой системе безопасности пришло из маловероятного источника: nVidia, производителя чипсетов, используемых в Xbox. После непримечательного второго квартала 2002 года представитель nVidia указал это как последнюю из нескольких причин, почему квартал прошел плохо:

> What we said about Xbox was that we reached a volume discount milestone, further reducing the margins. And that we will be taking an inventory write off in Q2 related to the amount of Xbox MCPs that were made obsolete when MSFT transitioned to a new security code (by way of the MIT hacker) and excess in nForce chipsets that we built in anticipation of higher demand of Athlon-based PCs.
>
> - Derek Perez, PR Director, nVidia

Источник: статья The Inquirer, http://www.theinquirer.net/?article=4735.

Обратная разработка безопасности v1.1

Источник: презентация Andy Green на 19th Annual Chaos Communication Congress о взломе безопасности Xbox.

Конкретика изменений security code не была раскрыта до октября 2002 года, когда хакер по имени Andy Green начал исследовать первые Xbox version 1.1, доступные в United Kingdom. Внешние физические отличия между Xbox version 1.1 и 1.0 на материнской плате были тонкими: GPU поменял свой вентилятор на более крупный heat sink, USB daughtercard была объединена с материнской платой, и отсутствовала микросхема PLL clock synthesizer. Также отсутствовали filter capacitors тут и там, но ничего существенного, казалось, не изменилось. Дальнейшее probing показало, что дыра, использованная MIST attack, была patched, но opcodes jam table были неизменны. LPC bus, ключевой vector для получения доступа к Xbox, также присутствовал и был неизменен.

Andy смог извлечь MCPX ROM за один день, используя процедуру, придуманную другим хакером по имени Asterisk. Процедура использовала нераскрытую комбинацию ранее идентифицированных security holes и back doors. Первоначальный анализ содержимого ROM выявил, что безопасность реализована радикально иначе. Беглый обзор Xbox version 1.1 показал, что старая схема security through obscurity была выброшена и заменена схемой, которая номинально получает свою безопасность от силы public-key ciphers.

Рисунок 9-2. Xbox Security Version 1.1. Области, которые нельзя изменить без замены silicon MCPX, заштрихованы серым.

```
MCPX Southbridge Chip:
  Secret Boot ROM
  Hardcoded hash value

FLASH ROM:
  Flash Boot Loader + Public Key
  Public Key Signed Hash
  2BL (kernel decompress, decrypt)
  Kernel Image
  Jam Tables

1. hash-check intermediate flash boot loader against hardcoded hash value
2. hash-check critical ROM regions against digitally signed hash
3. jump to 2BL if hashes check out
4. decrypt, decompress and execute kernel image
```

Реализация Microsoft новой схемы безопасности была, однако, немного counter-intuitive. Поскольку secret boot ROM внутри MCPX оставался того же размера, 512 bytes, они не могли уместить полный алгоритм public-key digital signature внутри secret boot ROM. Вместо этого они выбрали использовать lightweight hash в secret boot ROM для проверки области FLASH ROM, названной Flash Boot Loader (FBL). FBL содержит код (RSA cipher, SHA-1 hash, public key Microsoft и driver program) для digital signature verification FLASH ROM. FBL выполняется только если hash FBL можно проверить против constant, хранимой внутри secret boot ROM. Таким образом, FBL в теории столь же immutable, как secret boot ROM, даже несмотря на то что он хранится в mutable FLASH ROM.

Хотя эта схема звучала довольно bulletproof, хакерское сообщество не сдалось так легко. Они подробно изучили hash secret boot ROM и обнаружили, что он основан на TEA, Tiny Encryption Algorithm от David Wheeler и Roger Needham из Computer Laboratory Кембриджского университета. Franz Lehner, collaborator in the effort, отправил query в newsgroup sci.crypt относительно слабостей TEA hash.

В пятницу днем, 11 октября, на их query ответили. Статья John Kelsey, Bruce Schneier и David Wagner, представленная на CRYPTO 1996, указывала, что cipher TEA имеет слабость в key scheduler, где каждый key имеет три related keys, которые можно сгенерировать инвертированием определенных пар битов (эта слабость вместе с cipher TEA подробнее обсуждается в главе 7). В субботу Andy Green опубликовал это на XboxHacker.net:

```
Aw, I'm a mere mortal, my feet are definitely made of
clay.

OK, I don't think its giving too much away to say the
first 5 bytes of the region.

ffffd400: E9 83 01 00 00

This is a relative longbranch to 0xffffd588

If I flip b31 of that as a DWORD (and flip its friend at
DWORD address +1 the same way) I branch instead to
0x7fd588.... Hmmmm that's, what, 8M up, where...
where
there's
RAM

Xcodes.. visor ram push method... (looks at MCPX for RAM
Write X-Code)

X-Code opcode 3 ... unrestricted
```

```
Hold on to your hat, boys!  Its testing time!

(mviz, a marvelous and well-timed revelation, I feel
mysterious and invisble forces helping me along, for which
I am grateful!)
```

Источник: posting from www.xboxhacker.net under the Xbox Hacker BBS -> Xbox Hacking (TECHNICAL) -> BIOS/Flash ROM/Firmware -> News from the Xbox Linux Team, MS 'made a hash of it,' guts exposed.

Другими словами, related-key weakness cipher TEA означает, что каждую adjacent pair of double words в FBL можно модифицировать на один бит, самый значимый бит, без какого-либо изменения resulting hash. Эта слабость дала Andy и его команде достаточно breathing room, чтобы изменить target одной jump instruction так, чтобы она указывала на location в основной памяти.

Профиль: Andy Green

Можете рассказать нам немного больше о себе и о том, как вы пришли к хакингу?

Мне 37, я живу в England, рядом с Kettering в East Midlands, с женой, нашими четырьмя детьми и двумя кошками.

Я интересовался компьютерами примерно с 12 лет, когда мой брат купил Commodore Pet. Этот 1MHz 6502 занимал меня месяцы и месяцы, пока я пытался писать сначала BASIC-код, набранный из журналов, потом игры для него; в конце концов я написал фантастическую character-cell Space Invaders thing на machine code. Machine code - это когда вы фактически программируете CPU напрямую в hex; я до сих пор помню распространенные 6502 opcodes in hex. Это было настолько трудное усилие, что я решил, что мой следующий проект будет assembler, написанный на machine code. 1978 год был до дней Internet: я не мог позволить себе коммерческие assemblers, потому что был просто ребенком, и вокруг нас не было людей, которых я знал бы, чтобы warez a copy from.

Это было довольно жалко, если говорить об assemblers, но работало нормально. Я научился из этого ценности правильных инструментов: я мог писать намного быстрее на Assembler, и целые виды ошибок - например, miscomputing relative branches by hand - полностью исчезли. Затем у меня был компьютер BBC Model B, и снова мне было интересно делать tools и games. Мне предложили scholarship в public school, но я отказался и вместо этого ушел из школы в 16 без дальнейшего образования. Я был вполне доволен тем, что учил себя сам всему, что меня интересовало.

Я продал несколько games для этого и другой платформы 6502 под названием Oric, и на эти деньги started up a company, making Assemblers and other development tools. По пути я изучил C и C++, и каждый раз, когда я поднимался на следующий уровень, целые массы bugs и timewasting miseries исчезали. Это как та картинка «Ascent of Man»: от Neanderthal relative branch computation к Homo Erectus с его virtual functions.

Параллельно с этим я начал исследовать digital hardware design, опять обучая себя через опыт. Я обнаружил, что hardware и software - две стороны одной монеты, хотя в образовании они рассматриваются полностью отдельно. На самом деле это detail implementation, выбираете ли вы сделать свою logical function в software или hardware, или в некоторой смеси двух. Наличие ноги в обоих лагерях дает больше insight в природу design: например, можно сказать, что C++ заимствует многие concepts из electronics в смысле важности interfaces.

Незадолго до того, как заинтересоваться Xbox, я работал в принадлежащей US компании с офисом в Oxford, выполняя множество задач, но последняя была designing smartcard silicon. Хотя design был интересен и там в trenches работали отличные люди, я становился все более despondent из-за политики и проблем с management. Не помогало и то, что, несмотря на распределение между несколькими projects, мне платили 2/3 salary сотрудников в San Jose просто потому, что я базировался в UK. И не заставляйте меня начинать о patents, которые они получили от меня без

reward. В декабре 2001 года я обнаружил, что integrity важнее денег, resigned и решил вернуться к работе на себя.

Я был довольно tenderized некоторыми неприятными experiences при уходе из этой компании; пока я их переваривал, меня зацепила огромная разница в мировоззрении между ugly, grabbing, controlling instincts средней компании, вовлеченной в Intellectual Property, и природой GPL projects и людей, участвующих в поощрении смягчения patent and copyright laws. Со временем я все больше стал видеть Microsoft и предыдущую компанию, в которой работал, в одном свете.

После этого я прочитал о hack bunnie на Slashdot. Я читал о методах bunnie с некоторыми tart emotions. Мои главные мысли были, что это было чем-то, что я мог бы сделать, поскольку я использовал FPGAs, которые использовал bunnie, с 1989 года; восхищение лаконичностью атаки; и досада на себя, что я не делал со своим временем чего-то настолько же cool and interesting - и соответствующего моим philosophical predilictions. Вместо этого я сидел там, читал Slashdot, пил кофе и ничего не вносил. (В сторону: думаю, это довольно распространенный опыт для многих читателей Slashdot - немного завидовать и чувствовать вызов, когда они читают о чужих cool hacks. Думаю, это объясняет постоянный background noise там из насмешек и вопросов, зачем кому-то хотеть делать такую вещь.)

В течение следующих недель я собрал столько информации о внутренностях Xbox, сколько мог; Xboxhacker.net был crucial для этого. Там же я встретил Michael Steil, когда начинался проект Xbox Linux. Довольно скоро я смог определить интересные projects, в которые мог внести вклад, например проект Milksop. Опять же из этого, с помощью Surferdude, мне стало возможно собрать самый первый clean ROM, который мог boot and keep up the Xbox without being reset. Позже это стало basis of the crom 1MB Linux и cromwell, clean ROM Xbox Linux. После начальных hacks и designs я решил работать почти полностью в направлении цели Xbox Linux.

Можете рассказать, почему вы хакерите Xbox?

Почему? У каждого разные причины, но для меня это было мое понимание возмутительного antitrust behaviour Microsoft - deny everything, appeal everything, delay everything, а тем временем create and dump (потому что они продаются ниже себестоимости) на рынок миллионы Microsoft-only PCs - Xbox. Поскольку наши представители здесь в Europe и U.S., кажется, не заботятся (возможно, как недавно было в EU, потому что они планируют пойти работать на Microsoft и take their silver pieces), было бы честью быть частью прокалывания этого злого плана bloated monopoly с использованием weapons GPL и Linux. Я знаю, что люди закрывают глаза и думают о своих share options, но decent people - a surely most of the people working there are this - должно быть трудно работать на такого monster.

Мне повезло получить пару contracts в 2002 году, которые позволили мне провести последнюю часть года, работая исключительно над поднятием первого Linux kernel в crom и доведением Cromwell до состояния, где он мог control the main peripherals of the box и boot Linux from HDD or CD. С тех пор моя доля prize money Project A (спасибо donor Michael Robertson) позволит мне продолжать работать full time, по крайней мере следующие несколько месяцев.

Есть ли у вас какой-нибудь совет, которым вы хотели бы поделиться?

Моя последняя мысль - поощрить людей, особенно молодых, слушать свой мозг, когда речь идет о вещах, которые их интересуют. Не бойтесь копаться и пытаться узнать о вещах, которые цепляют ваше внимание. То чувство, которое вы испытываете, когда хотите что-то понять, своего рода longing, - это способ вашего мозга сказать вам, что он думает, что это знание может быть полезным позже. Если вы достаточно прислушиваетесь к нему, у вас хорошие шансы знать правильную вещь в правильное время, чтобы сделать какую-то маленькую разницу.

Это единственное location можно pre-load follow-up jump instruction обратно в любой кусок user code, используя ранее обсужденные jam table codes. Хакерское сообщество Xbox собралось в heroic effort и взломало Xbox security version 1.1 за три дня. Отдельное усилие, не менее доблестное, Xecuter также взломало security за тот же time frame.

Первая мораль этой истории в том, что безопасность настолько сильна, насколько сильно ее самое слабое звено. Хотя мало сомнений в robustness RSA cipher и SHA-1 hash для целей digital signature, они не были единственными элементами системы безопасности. Cipher TEA, использованный для расширения trust sphere secure boot ROM во FLASH ROM, имел flaws, которые позволили хакерам обойти strong digital signature algorithms.

Это приводит нас ко второй морали: complexity breeds weaknesses. Сложные системы трудно проектировать, тестировать и анализировать. Security version 1.1 для Xbox, вероятно, была реализована on a short fuse, поэтому было недостаточно времени для анализа системы на weakneses. Либо так, либо Microsoft знала о TEA weakness и спроектировала этот back door в систему, чтобы снизить риск запереть свой FBL в silicon. Кажется довольно сомнительным, что Microsoft намеренно включила этот back door, поскольку модификация MCPX silicon - очень дорогое предложение (хотя expense ended up on nVidia's books). С другой стороны, сложности трудно избежать. Мой руководитель в MIT, Tom Knight, однажды сказал мне: «There are two kinds of designs in this world: those that are useful, and those that you can formally prove to be correct.» В некоторой степени единственный способ обеспечить безопасность real-world system - сделать ее детали открытыми (никакой security through obscurity!) и подвергнуть систему анализу со всех сторон. В некотором смысле тщательный анализ безопасности Xbox проводится бесплатно для Microsoft, благодаря хакерскому сообществу.

Даже если бы Microsoft использовала более сильную hash function в secret boot ROM, все еще существует ряд жизнеспособных атак на Xbox, которые еще не были опробованы. Можно провести man-in-the-middle attack на шине HyperTransport (см. главу 8), overdriving signals carefully timed pulses. Эта атака довольно проста в реализации, поскольку каждая дорожка шины HyperTransport имеет test point, видимый со стороны компонентов материнской платы. Полное hardware solution включало бы FPGA на плате с test connectors в стиле «pogo pin» bed-of-nails. Эту плату можно прижать к этим test points без какой-либо пайки. Другая атака, предложенная мне Adi Shamir на конференции CHES, состоит в использовании timed glitch в CPU clock или power supply, чтобы нарушить расчет jump target address. Такой тип атаки успешно применялся к процессорам в cryptographic smart cards. Опять же, такой тип атаки можно реализовать довольно легко и дешево как user-installable module. (Имейте в виду, что намного более широкий диапазон атак доступен хакерам, если цель - однократное преодоление безопасности для восстановления, например, secret key или блока critical code.)

Угроза черных ходов

Как продемонстрировала эта глава, поиск back doors - практичный метод атаки на криптографически защищенную аппаратуру. Относительно высокий процент успеха нахождения back doors в Xbox частично объясняется тем, что Xbox представляет первую значительную попытку поставщика криптографически защитить PC. Несмотря на уроки, извлеченные из опыта Xbox, будущие реализации secure PC все еще рискуют иметь аппаратные weakneses безопасности, поскольку наследие PC - открытая и незащищенная аппаратная архитектура.

PC hardware сложна, но хрупка, и построение chain of trust из нее трудно из-за этой brittleness. Фундаментально каждый компонент в PC спроектирован как «trusting» своего physical environment. Спецификации любого коммерческого integrated circuit component ясно заявляют, что IC

гарантированно работает в ограниченном диапазоне temperatures, voltages, frequencies и других conditions. Если эти maximum ratings нарушены, тогда поведение устройства «undefined», и все ставки сняты. Большинство chip engineers даже не рассматривают попытку заставить свои circuits gracefully recover from an out-of-range condition, поскольку и так достаточно трудно заставить chip работать при specified operating conditions. Кроме того, большинство consumer applications очень cost-sensitive, и overhead встраивания robust fault tolerance measures приводит к продукту, который не price-competitive.

Таким образом, chips обычно реализованы без internal error-checking. Если по какой-то причине Arithmetic Logic Unit (ALU, computational «brains» CPU) складывает два числа неправильно, проблема проявится только symptomatically; вы можете наблюдать только эффекты такой ошибки, иногда намного позже error-causing event. Можно думать об атаках, использующих faults, induced by out-of-range conditions, как об аналогии buffer overruns в software world.

Другая проблема архитектуры PC в том, что процессор слишком доверяет своему code environment. Архитектура процессора Pentium не имеет provision in hardware для различения code that is insecure or secure. Если instruction pointer оказывается в insecure code segment через bug или induced failure, процессор счастливо выполнит этот код.

Профиль: Franz Lehner

Franz Lehner, 29 лет, живет в Austria со своей девушкой. Он изучал Electrical Engineering 5 лет. Сейчас он программирует «automated solutions», одновременно управляя ISP. В свободное время он ищет projects, которые fun and educational.

Найдя документ bunnie о взломе Xbox, он встретил команду Xbox-Linux на sourceforge.net. Он присоединился к проекту Xbox-Linux, чтобы узнать о team programming, Linux kernel hacking and debugging, and cryptographic systems.

Он также присоединился к проекту Xbox-Linux, чтобы развить лучшее понимание related systems, such as Palladium.

> NB. Code compartmentalization based on hardware security levels - техника, отличная от sand-boxing. Sand-boxing не предоставляет адекватного решения для ситуаций, где пользовательской программе требуется direction from or interaction with secret or protected code or data. В последнее время были предложены новые processor architectures, которые могут решить эту проблему через использование data tags, embed a sort of security audit log.

Источник: http://www.ai.mit.edu/projects/aries/Documents/Memos/ARIES-15.pdf. «A Minimal Trusted Computing Base for Dynamically Ensuring Secure Information Flow», Tom Knight и Jeremy Brown.

Еще один источник back doors - design bugs, которые существуют в каждом сложном chip. Обычная практика - поставлять chips с множеством known bugs, также известных как errata. Например, процессор Intel i860 XP (впервые выпущенный в 1991 году, не путать с недавно выпущенным chipset i860 для процессора Pentium4) поставлялся с книгой errata, сопоставимой по размеру с datasheet процессора. Другой пример ближе к дому - bug в address space decoder nVidia MCPX, который сделал возможной MIST Premature Unmap attack. Большинство этих errata имеют простые work-arounds или имеют minor implications для functionality chip under nominal conditions. Однако некоторые errata, например dealing with cache coherence, address decoding и memory management, могут приводить к major software security holes.

В случае Xbox business impact аппаратного back door, вероятно, мал. Возможно, Microsoft теряет некоторую малую долю revenue от продаж игр, но потери из-за piracy dwarfed by losses Microsoft takes on hardware sales. Кроме того, Xbox - всего лишь игровая консоль: grandma's bank account не

being tapped dry и credit card numbers не украдены как результат слабостей безопасности Xbox. Однако с trusted PC под риском будут не только game revenues. Если архитектура trusted PC не является фундаментальным изменением относительно legacy PCs, люди будут слепо доверять финансовые secrets и personal data security ненадежной аппаратуре.

Как и в большинстве вещей в жизни, первый шаг - образование. Чем больше мы узнаем об аппаратной безопасности, даже если это включает poking around a game console, тем лучше будут наши системы безопасности завтра. Теперь продолжим урок...

Глава 10. Другие аппаратные проекты

Сходство Xbox с архитектурой PC позволяет хакерам заимствовать технологии и опыт из мира PC при построении аппаратных проектов. В результате PC hardware, мониторы, кабели и периферия - все это было адаптировано для работы с Xbox. Эта глава представляет некоторые из этих аппаратных проектов, обнаруженные, задокументированные и реализованные хакерами по всему миру.

Интерфейс LPC

Версия 1.0 интерфейса LPC (Low Pin Count) была определена Intel в 1997 году. Интерфейс LPC - royalty-free-шина, спроектированная, чтобы позволить системы без явных возможностей ISA или X-bus (ISA-like expansion bus для памяти или generic I/O devices). Потребность в интерфейсе LPC проистекает из большого числа устройств и шин с низкой bit-rate, высоким pin count и несовместимыми интерфейсами, встречающихся в стандартном PC, таких как floppy disk, keyboard, mouse, serial, IrDA, parallel, ISA и boot ROM interfaces. Совокупная bandwidth, потребляемая всеми этими устройствами, мала, но число сигналов, требуемых для поддержки их всех, легко превышает signal count, требуемый более high-bandwidth-шинами, такими как PCI или AGP bus.

Хуже того, не все конфигурации компьютеров требуют всех этих legacy I/O devices, а wasted pins и functions просто съедают profits. Стоимость pin на chip package высока относительно стоимости silicon, требуемого для поддержки этих простых interfaces. (Практическое правило: один package pin стоит penny, тогда как в 0.13 µ silicon примерно десять тысяч gates - достаточно logic для реализации маленького processor - стоят penny silicon area, assuming the design is not bond-pad limited [1].)

Интерфейс LPC противодействует этой проблеме одной low-pin count-шиной (семь обязательных pins против 36 pins, требуемых для ISA bus), работающей на высокой скорости. Все legacy I/O и expansion functions отображаются в эту high-bandwidth-шину, позволяя system designers создавать так называемые «Super I/O» chips, которые, в свою очередь, позволяют Southbridge chips с гораздо меньшим pin count. Кроме того, разделение functions между Super I/O chips и Southbridge chips позволяет designers выбирать combinations Super I/O и Southbridge chip, предоставляющие оптимальный set features для данного application.

Физический интерфейс LPC довольно прост. Interface - это 4-битная bi-directional-шина, работающая на clock rate 33 MHz. Interface также имеет два «sideband» signals: один framing signal, указывающий начало и конец LPC bus cycles, и один reset signal, который принудительно переводит все LPC peripheral devices в известное состояние для initialization purposes. Кроме того, есть пара optional signals для интерфейса LPC, которые предоставляют DMA и interrupt capabilities, а также power management для более sophisticated I/O devices. (Больше информации о LPC bus и его protocol можно найти в Intel Low Pin Count (LPC) Interface Specification, version 1.1. Спецификацию можно найти на корпоративном веб-сайте Intel по адресу http://www.intel.com/design/chipsets/industry/lpc.htm.)

Интерфейс LPC на Xbox

Xbox включает интерфейс LPC на материнской плате. Интерфейс LPC в этом случае используется для реализации debug and test bus. Через этот интерфейс LPC можно подключить keyboard и mouse, а также alternate boot ROM для diagnostic purposes. Интерфейс LPC активируется для загрузки alternate boot code, когда FLASH ROM на Xbox недоступен. Отсутствие устройства FLASH ROM можно симулировать, принудительно удерживая младший bit данных (D0) data bus FLASH ROM на уровне zero volts.

Многие предполагают, что интерфейс LPC является существенной частью production line Xbox из-за способности alternate boot ROM, предоставляемой интерфейсом LPC. Полностью собранные Xbox можно конфигурировать с comprehensive self-test program через интерфейс LPC. Применение CPU как fast test controller позволяет быстро и эффективно изолировать дефектные units на factory floor без стоимости дорогих testing machines.

Для хакеров facility alternate boot ROM, предоставляемая интерфейсом LPC, является идеальным механизмом для доставки кода в Xbox. Корректные LPC-loadable boot ROM images для Xbox может создать кто угодно, поскольку cryptographically secured boot procedure Xbox теперь полностью понята. Фактически некоторые vendors alternate boot ROM devices для Xbox использовали регулярность pinout geometry интерфейса LPC на материнской плате Xbox, чтобы создать ROM devices, которые устанавливаются без какой-либо пайки. Эти devices используют набор spring-loaded «pogo-pins», похожих на те, что используются во время производства для testing Xbox, чтобы контактировать с интерфейсом LPC просто pressure-fit. (Pinout LPC bus, как он реализован на Xbox, можно найти в приложении F, «Xbox Hardware Reference».)

Использование интерфейса LPC

Тот факт, что интерфейс LPC является industry standard, весьма удобен для hardware hackers Xbox. Во-первых, есть изобилие LPC-compatible interface devices, от Super-I/O chips до firmware ROMs со встроенными LPC interfaces. Во-вторых, широкое принятие интерфейса LPC как diagnostic and convenience bus для generic PCs помогает снизить юридический риск использования интерфейса LPC и продажи LPC interface devices. Firmware ROM для интерфейса LPC можно продавать без какого-либо Xbox-specific contents, поскольку end-users могут легко перепрограммировать свои LPC bus devices, используя простой дешевый adapter для своего PC. Дополнительная помощь законности LPC firmware devices в том, что pinout LPC connector Xbox почти идентичен тому, который Intel рекомендует для generic PCs. В результате LPC firmware device, продаваемое для Xbox, очень похоже на LPC firmware device, продаваемое для standard PC.

Одно из первых LPC boot ROM devices было разработано Andy Green. Проект называется «Cheapmod», и это устройство SST 49LF020 (256 kByte FLASH ROM с integrated LPC interface) в socket, соединенной с LPC-compatible header. Согласно веб-странице Andy Cheapmod, «http://warmcat.com/milksop/cheapmod.html», «If you can get ahold of the $2.50 SST 49LF020, you can build an alternative BIOS for $4.» Это устройство можно программировать с использованием его программатора «CheapLPC» (http://warmcat.com/milksop/cheapLPC.html), delightfully simple PC parallel-port based device, которое может (медленно) разговаривать с LPC device и перепрограммировать его. Многие commercially available alternate firmware devices были based off of или inspired by его design, включая design Xodus/Matrix.

Xodus/Matrix - особенно интересный вариант original design Andy, поскольку это было первое Xbox alternate firmware device, реализовавшее полностью solderless installation procedure. Это открыло мир взлома Xbox для software-oriented hackers, которые не были склонны паять wires в свои Xbox. (Фотографию Xodus/Matrix можно увидеть на рисунке 10-1.) Устройство Xodus/Matrix поставляется без какого-либо code, programmed in it; пользователь должен предоставить alternate firmware image.

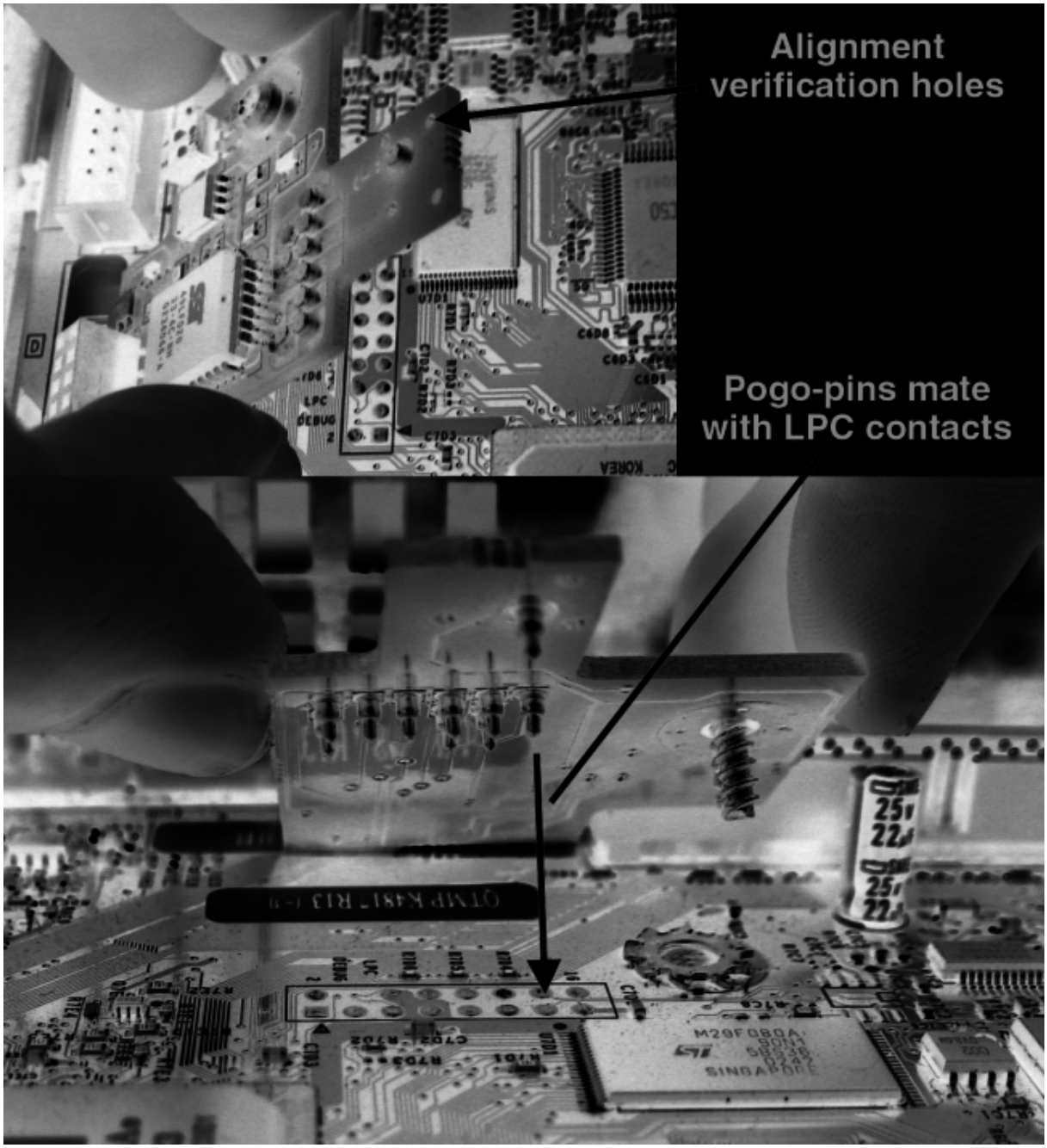


Рисунок 10-1. Solderless Xodus/Matrix alternate firmware device, показывающее spring-loaded «pogo-pin» contacts, которые позволяют solderless connection к LPC connector на материнской плате Xbox.

Есть несколько важных functional considerations при выборе FLASH ROM chip с интерфейсом LPC для использования с Xbox. Самое значительное - то, что native Xbox architecture выделяет 16 MB area для boot ROM. Если physical boot ROM меньше 16 MB, contents boot ROM aliased, чтобы заполнить все 16 MB space. Это дает designers Xbox больше гибкости в выборе размера ROM chip, не вызывая проблем с routines, которые используют и bottom-, и top-relative addressing.

Сделаем понятие bottom- and top-relative addressing более конкретным на примере. Addresses для 16 MB boot ROM area в Xbox простираются от `0xFF00.0000` до `0xFFFF.FFFF`. Programs на Xbox, использующие bottom-relative addressing, будут вычислять addresses с использованием `0xFF00.0000 + offset` (bottom address plus offset), тогда как programs, использующие top-relative addressing, будут использовать `0xFFFF.FFFF - offset` (top address minus offset). Предположим, что в Xbox установлен 1 MB boot ROM. Это означает, что processor увидит 16 идентичных copies этого 1 MB ROM, равномерно распределенных по 16 MB ROM address space. Другими словами, contents boot memory выглядят идентичными для каждого address `A + 0xFF00.0000 + n * 0x0010.0000`, n = 0 through 15, A = 0 through `0x000F.FFFF`. В результате programmers могут упаковывать data в меньший 1 MB boot ROM, используя и top-, и bottom-relative addressing без необходимости менять какой-либо code: корректная copy ROM image появляется рядом и с top-, и с bottom-relative base addresses.

Теперь предположим, что Microsoft решила сэкономить на стоимости и уменьшить свой 1 MB boot ROM до 256 kB boot ROM. Processor теперь видит 64 identical copies этого 256 kB boot ROM, распределенные по 16 MB ROM address space, и весь старый code, использующий bottom- and top-relative addressing, все еще работает. Значительно, CPU в Xbox hard-wired начинать выполнение code при power-up с address, расположенного за 16 bytes от top of memory (его «reset vector»), тогда как hardware initialization routines, wired into Xbox chipsets, используют ROM locations, расположенные рядом с bottom 16 MB FLASH ROM space. В результате hardware Xbox требует LPC ROM implementation, которая либо имеет размер 16 MB, либо aliases contents smaller ROM по всему FLASH ROM address space. (SST 49LF020 - один из немногих LPC FLASH ROMs, который aliases contents ROM по всему address space. Можно утверждать, что эта feature на самом деле bug: игнорируя upper address bits и aliasing contents ROM по всему address space, этот chip занимает space, который мог бы быть allocated to other functions. В результате SST выпустила updated «A-step» детали, называемый 49LF020A, который не aliases contents ROM over memory. Аналогично, A-step silicon не будет работать как alternate firmware device для Xbox.)

Устройства альтернативной прошивки против модчипов

Alternate firmware device - hardware module, который предоставляет method для запуска user-specified firmware на Xbox hardware. Alternate firmware devices отличаются от так называемого «modchip» тем, что alternate firmware device furnished as a blank device и не имеет inherent ability обходить copyright control mechanisms. Blank LPC-interface ROM device, например, является alternate firmware device: вы могли бы прожечь на него copy of the U.S. Bill of Rights, если захотели бы. Любой user-installed FLASH ROM, который comes blank, также является alternate firmware device. Modchip, с другой стороны, colloquially implies device, crafted for playing game backups and otherwise modifying or removing DRM (digital rights management) policy restrictions. Следовательно, термин modchip охватывает certain boot ROM devices, которые have been programmed with code, enabling DRM policy modifications, а также devices вроде «patchers», которые не содержат ROM и работают, dynamically patching a few key Xbox firmware locations as the firmware is loaded for execution.

Другие 64 MB SDRAM

Проницательный наблюдатель заметит, что на верхней стороне материнской платы Xbox отсутствуют две микросхемы, и что эти missing chip spots подозрительно похожи на spots, сейчас занятые memory chips. Переверните плату, и там есть еще два незанятых chip footprints. Эти empty footprints на самом деле предназначены для memory chips. Расположение этих blank spots показано на рисунке 10-2.

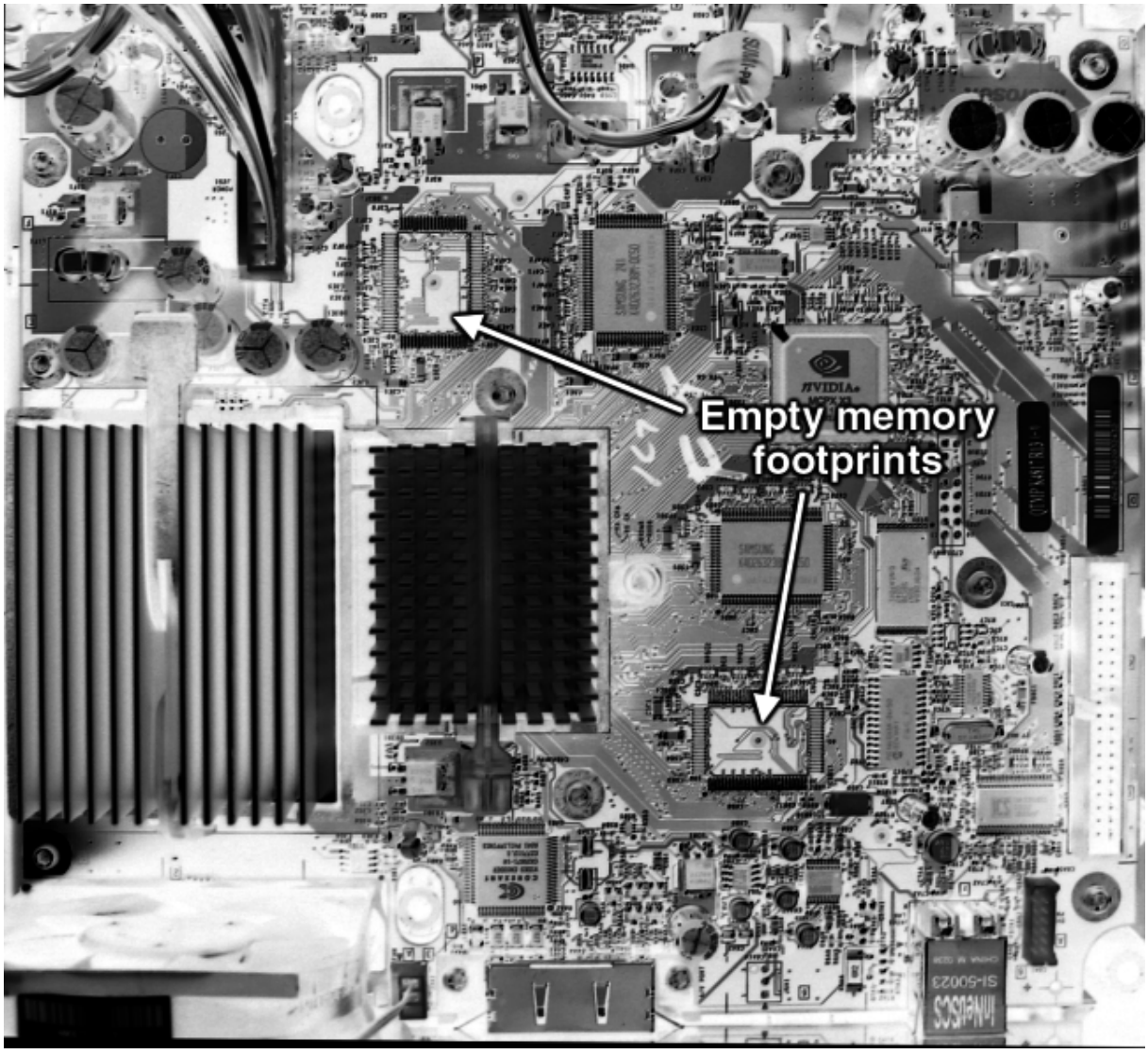


Рисунок 10-2. Незаполненные посадочные места памяти на материнской плате Xbox.

Доверенные лица

Посмотрите на незаполненное место памяти на материнской плате Xbox. Серебряная точка, окруженная темным кольцом внутри этих unpopulated chip footprints, называется fiduciary. Fiduciary patterns используются машинами сборки circuit board как reference points для выравнивания больших chips со множеством pins. Они спроектированы так, чтобы быть легко распознанными machine vision systems, применяемыми в board assembly machines. Specially shaped fiduciaries также можно использовать для автоматической идентификации ориентации и типа circuit board.

Следующий логичный вопрос, конечно: «Можно ли удвоить размер памяти Xbox до 128 MB, припаяв подходящие memory chips в открытые slots на материнской плате Xbox?» Ответ на самом деле да, но initialization code Xbox нужно модифицировать, чтобы chipset распознал и использовал extra memory. Кроме того, extra memory не помогает graphics или gaming performance. Xbox games не спроектированы для использования extra memory, поэтому extra memory обычно будет просто лежать unused. Extra memory spots предоставлены в основном для производства специальных consoles для game developers. Game developers могут использовать extra memory, чтобы облегчить transition игр в относительно tight memory footprint Xbox, а также для удержания debug, performance monitoring и test utilities resident in memory, которые не являются частью game image. Обратите внимание, что extra memory могла бы быть использована home-brew software, но трудность получения и установки memory chips делает Xbox memory expansions скорее интересным упражнением soldering practice, чем практической модификацией.

Xbox VGA

Есть немного путаницы относительно того, что делает Xbox VGA adapter. Многие Xbox VGA adapters на самом деле являются TV-to-VGA converters. Другими словами, они берут low resolution TV output от Xbox и прогоняют его через line doubler, чтобы получить low-quality VGA display. Настоящий Xbox VGA adapter фактически конфигурирует Xbox на output гораздо более high resolution video output, давая display качества лучше TV на VGA monitor.

VGA adapter конфигурирует graphics mode Xbox с использованием designated pins в connector AVIP (Audio Video I/O Port). Главная проблема этого подхода в том, что game должна быть специально написана для поддержки этого higher resolution mode. В результате некоторые games не будут работать с настоящим Xbox VGA adapter, но, к счастью, вернуться к TV resolution так же просто, как подключить стандартный TV adapter cable.

Оригинальный Xbox-VGA adapter был разработан Ken Gasper. Он продает его версию на своем веб-сайте http://xboxvga.xemulation.com. В настоящее время он предлагает Xbox-VGA adapter в форме «bare board», а также в полностью собранной форме. Если вы ищете интересный hardware hacking project для Xbox, который и полезен, и отточит ваши circuit assembly skills, возможно, стоит купить одну из его bare boards и попытаться собрать adapter самостоятельно.

Приложение F содержит pin diagram Xbox AVIP.

Замена массового хранилища

Xbox содержит DVD-ROM drive и hard drive, оба используют PC standard IDE interface для разговора с материнской платой Xbox. DVD-ROM drive также имеет proprietary power and DVD tray state connector. Популярная и иногда необходимая hacking activity для Xbox - замена этих drives.

Пользователи заменяют или tweaking DVD-ROM, потому что native Xbox DVD-ROM drive не способен читать CD-Rs и многие типы CD-RW media. Это может быть особенно раздражающим для тех, кто пытается впервые установить Xbox-Linux, или для пользователей, которые пытаются rip music со своей CD-R collection на hard drive Xbox.

Есть много методов замены и tweaking Xbox DVD-ROM drive. У некоторых моделей Xbox DVD-ROM drive можно регулировать laser intensity, чтобы улучшить их способность читать CD-R и CD-RW media. Это потенциально рискованная операция, поскольку можно навсегда повредить DVD-ROM drive неправильной регулировкой power output laser, но многие хакеры сообщали, что правильно

выполненная procedure приводит к better media compatibility. Я предлагаю web-search по последним новостям и techniques, поскольку style и model DVD-ROM drive, используемого в Xbox, часто меняются. Кроме того, Xbox DVD-ROM drive можно outright заменить стандартным PC DVD-ROM. Проблема этого method двоякая. Во-первых, обычный PC DVD-ROM drive не может читать оригинальные Xbox game disks из-за physical security measures, встроенных в Xbox game disk. Во-вторых, PC DVD-ROM drive нужно адаптировать к custom DVD power and traystate connector на материнской плате Xbox.

Самый легкий, но самый уродливый способ - установить стандартный PC DVD-ROM drive, но оставить Xbox DVD-ROM drive подключенным через его proprietary cable. В этом method серый IDE cable подключается к стандартному PC DVD-ROM drive (установленному в slave mode через jumper configurations на drive), а power берется из power connector hard drive с использованием стандартного power splitter cable. Xbox DVD-ROM drive остается на месте, но с пустым IDE connector и с установленным proprietary yellow power-and-tray-state cable. Назначение Xbox DVD-ROM drive - служить dummy drive, который используется для manual relay state DVD drive tray к Xbox. Другими словами, пользователю нужно вручную replicate state tray стандартного PC DVD-ROM с использованием tray Xbox DVD-ROM во время media change event.

Точная процедура эксплуатации Xbox в такой configuration меняется в зависимости от конкретной модели PC DVD-ROM drive и нюансов hardware configuration Xbox, поэтому снова я предлагаю web-search по latest information. Есть также некоторые websites, описывающие, как адаптировать selected PC DVD-ROM drive models для работы с proprietary tray state and power connector Xbox. Такой project - хороший intermediate-level для хакеров, которые в основном comfortable with soldering and screwdrivers. Модификации, выполняемые на standard DVD-ROM drive, позволяют state tray стандартного drive accurately transmitted to Xbox. Однако они не позволяют вам играть в original games, если Xbox не был модифицирован additional hardware, которая circumvents security checks на DVD ROM drive. Даже без способности играть в games это все равно полезная technique для ferreting out Xbox-Linux installation problems и для улучшения способности Xbox rip your CD collection или watch DVDs. (Обратите внимание, что возврат IDE connector обратно к Xbox DVD-ROM drive восстановит original gaming functionality Xbox.)

Xbox hard drives также время от времени требуют замены. Серьезные software developers для Xbox считают advantageous установить hard drive большей емкости в Xbox, и users со сломанными hard drives также хотят заменить свои hard drives. К сожалению, OEM Xbox hard drive содержит copyrighted Microsoft programs. Xbox hard drives также защищены firmware lockout, что делает установку нового hard drive с original gaming functionality довольно challenging, особенно с точки зрения legal issues. Firmware lockout также unique to each hard drive, не позволяя заменить ваш hard drive used Xbox hard drive. Однако если вы хотите только запускать Xbox-Linux или другие homebrew programs и не заботитесь об игре в games, установка нового hard drive в Xbox так же легальна и так же проста, как установка hard drive в любой PC.

Сноски

[1] [назад] Схемы на chip обычно окружены квадратами металла («bond pads»), которые проводами соединяются с pins на packaging chip. Chip называется bond-pad limited, когда area, требуемая для кольца bond pads, превышает area, требуемую circuitry внутри chip. Стоимость excess pins становится еще выше в случае, когда chip is bond-pad limited.

Глава 11. Разработка программ для Xbox

Хотя фокус этой книги - обучение читателей путям аппаратного взлома и безопасности, одна конечная цель взлома Xbox - запуск homebrew software. Эта глава посвящена описанию некоторых homebrew software projects, находившихся в работе для Xbox на момент написания.

Xbox-Linux

Цель проекта Xbox-Linux - создать удобный для пользователя и легальный порт GNU/Linux и GNU/Linux applications на аппаратную платформу Xbox. Благодаря преданности и вкладам хакеров по всему миру проект Xbox-Linux добился большого успеха в продвижении к своим целям. Фотографию core Xbox-Linux project team можно увидеть на рисунке 11-1, а боковые вставки в этой главе и в главе 9 содержат интервью с членами команды проекта Xbox-Linux. (Homepage проекта Xbox-Linux - http://xbox-linux.sourceforge.net.)

Существенно, что проект Xbox-Linux и его principal hackers не являются anti-Microsoft. Они pro-«freedom to tinker», а не puerile Microsoft-haters; у них agenda, которая касается сохранения самих свобод мысли и речи, приведших технологию туда, где она находится сегодня.

Xbox-Linux - не ultimate software project для Xbox; наоборот, это только начало software hacking Xbox. Перенос знакомой development environment GNU/Linux на Xbox позволяет более широкой базе software hackers присоединиться к проекту взлома Xbox. С GNU/Linux Xbox может запускать широкий диапазон application software, от свободных open-source video games до word processing applications и clustering software для построения computer clusters в стиле Beowulf.

Установка Xbox-Linux

В настоящее время, чтобы запустить Xbox-Linux, нужно установить GNU/Linux boot ROM с использованием alternate firmware device. Это требует вскрыть Xbox. Глава 10 описывает методы сборки и установки alternate firmware device для Xbox через интерфейс LPC. Несколько vendors теперь предлагают easy-to-install LPC interface alternative firmware devices. Особенно Xodus/Matrix device - первое alternative firmware device на рынке с полностью solderless installation procedure. Все tools, нужные для установки Xodus/Matrix device, описаны в главе 1, «Voiding the Warranty», а само Xodus/Matrix device поставляется с easy-to-follow instructions о том, как programming and use the alternate firmware device.

Профиль: Michael Steil

Michael, можете рассказать нам немного больше о себе?

Я родился в 1979 году в Erding/Germany, я студент computer science в Technische Universität München. Я преподаю Assembly студентам первого семестра, и планирую получить MA degree в следующем году. Я работаю с computers с десяти лет; моим первым компьютером был Commodore 64, за которым вскоре последовал 386 PC. Моими главными интересами всегда были hardware и operating systems, и меня особенно fascinates разнообразие hardware architectures (Commodore, PC, Amiga, Macintosh, . . .), а также popular embedded systems, такие как gaming consoles. (Вы знали, что «SEGA CD» имеет три CPUs, один Z80 и два M68000?) Поэтому я купил много video game systems

для experimentation, таких как Nintendo SNES, SEGA Genesis и Nintendo Game Boy. Я также посмотрел на Linux для SEGA Dreamcast, но никогда не видел Linux для Sony Playstation 2, поскольку весь комплект был для меня действительно слишком дорог, и для experimentation, и для real use.

Как вы пришли к Xbox hacking и, в частности, к проекту Xbox-Linux?

30 апреля 2002 года я купил Xbox, убежденный, что это будет отличная игрушка для hacking и хорошо подходит для Linux. После часа или двух изучения system software (я не купил ни одной game) я открутил Xbox. Поиск информации о hacking the box сначала разочаровал меня: я не нашел much more than how to connect the hard disk to a PC, и site about Xbox Linux с практически no information on it. Поэтому я решил начать свой собственный Xbox hacking site и помещать туда информацию, которую выяснял, подключая hard disk к PC.

Xboxhacker.net и original Xbox Linux mailing list были большой помощью; оба привлекали excellent hackers и публиковали valuable information. Недовольный original infrastructure проекта Xbox Linux, я решил перейти на Sourceforge 23 мая. Теперь каждый contributor мог добавить что угодно на website без необходимости проходить через maintainer. Но в то время все еще было довольно theoretical: без появления modchips мы не могли делать much more than write code that «should theoretically work». Filtror Andy Green ускорил все: этот mod сделал возможным закончить bootloader и, с помощью Milosch Meriac, адаптировать Linux kernel за очень короткое время.

«Anonymous donor», подошедший ко мне в июне, не только привел к дополнительной publicity проекта и поэтому к еще большему числу contributors, но и к моей личной friendship: Walter Meyer, creator BioXX (OpenXbox) modchip, оказалось, живет всего в 20 kilometers от моего места. Среди прочего, он много помогал мне с modding my boxes, поскольку я не really a soldering iron person.

Когда Linux уже работал на Xbox, в декабре 2002 года Xbox Linux core team (Andy Green, Milosch Meriac, Franz Lehner и я; Edgar Hucek, к сожалению, не смог приехать) впервые встретилась лично на Chaos Computer Club Congress в Berlin.

Моя original motivation для всего была просто в том, что это fun, и я мог многому learn by doing it. Я начал это не потому, что хотел harm Microsoft - все же я согласен, что Microsoft harms their customers, не позволяя им использовать software, которое они хотят использовать, на hardware, которое они купили, и поэтому проект Xbox Linux особенно важен.

> NB. Microsoft может и будет пересматривать layout своей материнской платы и систему безопасности, поэтому перед покупкой уточняйте у vendor устройства совместимость с вашей конкретной system hardware. Вам также понадобится converter cable Xbox gameport to USB, если вы хотите использовать standard keyboard and mouse с Xbox; его можно купить через aftermarket retailers, таких как Lik-Sang (http://www.lik-sang.com), или собрать самостоятельно, следуя step-by-step guide в главе 4.

Перед установкой вашего alternative firmware device вам нужно будет запрограммировать его ROM image, который загружает GNU/Linux kernel. «Cromwell» - open-source, clean-room (то есть не содержит code Microsoft) boot ROM для Xbox, способный загружать GNU/Linux. Существенно, что информация, содержащаяся в source code и binary image Cromwell, не может быть использована для bypass каких-либо native copyright control mechanisms, встроенных в Xbox. Другими словами, трудно утверждать, что Cromwell является каким-либо copyright control circumvention tool. (Cromwell можно скачать с Xbox-Linux website на сервере Sourceforge.net по адресу http://xbox-linux.sourceforge.net.)

После прожига Cromwell ROM на ваш alternate firmware device и установки device в Xbox вам нужно будет прожечь на CD/RW media GNU/Linux install image, который можно скачать с Xbox-Linux website (снова http://xbox-linux.sourceforge.net). Этот install image поставляется как довольно hefty

(100+ MB) ISO image, compressed using bzip2, и содержит все software, interfaces и tools, необходимые, чтобы поднять user-friendly GNU/Linux distribution на Xbox. При прожиге этого ISO image нужно использовать option burn image в CD burner software. Не копируйте ISO image на CD как один большой file. (ISO images - literal bit patterns для CD, так что ISO image уже содержит complete filesystem description. Прожиг ISO image как regular file, вместо image, encapsulates ISO image in a new filesystem, так что ISO просто appears as a «bag of bits» вместо filesystem with files.)

Возможно, вам также понадобится второй disk, прожженный с boot program для Xbox-Linux. Эта boot program поставляется как smaller ISO image, который должен быть доступен там же, где вы скачали main GNU/Linux install image. Этот boot image позволяет boot Linux installation, просто бросив его в Xbox, как запуск game. (Можно также скопировать contents этого disk на hard drive с использованием third-party dashboard и boot Xbox-Linux directly from hard drive, если вы предпочитаете не иметь дело с separate boot disk.)

Прожиг хорошего CD/RW image - возможно, одна из самых tricky parts установки Xbox-Linux. Laser, используемый внутри DVD-ROM drive Xbox, не очень хорошо подходит для чтения writeable CD media, поэтому Xbox очень finicky about kind of media и kind of burner, а также burner settings, used to create the CD image. Более того, exact details того, как laser is degraded, меняются от Xbox к Xbox и зависят от model drive, который оказался установлен. Users found, что немногие Xbox могут reliably read CD-R media, поэтому нужно использовать CD/RW media. Кроме того, помогает прожигать CD/RW media на slowest burner setting, используя либо fresh, blank CD/RW, либо CD/RW, который был fully erased (в отличие от quick erase, который только resets filesystem и не actually destroy previously written data).

Перед тем как commit to a particular type of CD/RW media, попробуйте использовать WMA ripping tools обычного Xbox Dashboard, чтобы скопировать contents CD/RW, который вы прожгли с music, на hard drive. Если это работает reliably and without error, вы, вероятно, можете использовать этот kind of CD/RW media для установки Linux. (Многие installation problems Xbox-Linux были traced to problems reading data off CD/RW drive.) На момент написания нет distributions, available in hard-pressed CD-ROM media. В сообществе Xbox-Linux есть some talk of ordering set of custom CD-ROM images, поскольку это решило бы many of the CD/RW headaches, которые users испытывали. (Также обратите внимание, что возможно установить в ваш Xbox after-market DVD-ROM drive, который имеет better compatibility with writeable CD formats, как обсуждалось в предыдущей главе.)

> NB. Помните, что Xbox-Linux - active project, который постоянно evolves. Самые up-to-date instructions по установке GNU/Linux на Xbox можно найти на Sourceforge Xbox-Linux website, и эти instructions были переведены как минимум на полдюжины languages на момент написания. Если вы заинтересованы внести свои talents в проект Xbox-Linux, на Sourceforge Xbox-Linux website есть list of projects to-do, а также instructions о том, как join the developer's mailing list.

Michael Steil продолжает:

> [We're] not «Anti-MS» or «MS-haters.» We dislike their market in strategy, so we have a rational reason to work against them.

Есть что-нибудь еще, что вы хотели бы сказать о $200k prize for Xbox-Linux?

Я думаю, award не привлек людей, которые хотели увидеть some money: сейчас, через месяц после deadline, money все еще не распределены, и ни один человек не отправил мне single question о том, когда он получит money. Award привлек press; мы получили больше publicity, и таким образом получили больше hackers. Но никто не делал это because of the money. Поэтому мы не хотим, чтобы

нас считали being paid for the job by Michael Robertson. Хорошее доказательство в том, что мы все еще all active after the deadline.

Можете рассказать нам больше о вашем «MIST X-Code hack»?

Через некоторое время после original hack bunnie Andy полностью извлек MCPX ROM, и Steve, Paul и я начали анализировать code, а я reverse-engineered X-Code interpreter, contained within it. Когда я искал bugs, которые можно было использовать, чтобы escape the X-Code interpretation loop, я обнаружил, что часть code уже была written with our attacks in mind. Это мой original disassembly:

```
cmp    ebx, 80000880 ; ISA Bridge, MCPX disable?
jnz     short not_mcpx_disable
              ; BUG: too specific: bits 24 to 30
              ; undefined and ignored by PCI hardware!
and    ecx, not 2    ; clear bit 1 (MCPX ROM will be
                     ; turned off by setting bit 1)
not_mcpx_disable:
mov     eax, ebx
mov     dx, 0CF8h
out    dx, eax       ; PCI configuration address
add     dl, 4
mov     eax, ecx
out    dx, eax       ; PCI configuration data
jmp     short next_instruction
```

Я раньше работал с «PCI configuration», поэтому знал, что test for the attack был слишком specific: похожие codes делали бы то же самое, но проходили test. Так что у MS developers была good idea, но implementation была wrong, таким образом рассказав нам об их idea этим способом!

Я отправил свою idea Andy, Steve и Paul, и они за короткое время verified, что `0x88000880` worked just as well as `0x80000880`, чтобы turn off MCPX ROM and exiting the interpreter by mapping the interpreter code out of memory!

«Project B»

Идет work in progress, называемая разработчиками Xbox-Linux «Project B», чтобы найти способ установить и загрузить Xbox-Linux без каких-либо аппаратных модификаций. Moniker Project B происходит из criteria, defined for awarding $200,000 prize, offered by Michael Robertson, CEO Lindows. Prize «Project A» был $100,000 и был awarded первой group, добившейся Linux running on an Xbox with hardware modifications. Оставшиеся $100,000 будут awarded individual or group, которая completes Project B. Asymmetric division prize money намекает на challenge completing Project B. (Больше details on Project B можно найти на Sourceforge Xbox-Linux website по адресу http://xbox-linux.sourceforge.net/articles.php?aid=2002354043211.)

Есть ряд Project B strategies, pursued by various groups. Самый conceptually simple approach - factor 2048-bit RSA key, используемый для подписи Xbox game disks. Этот approach pursued by OperationProjectX (http://sourceforge.net/projects/opx) using a distributed computing approach. Проще говоря, если 2048-bit RSA key факторизован и раскрывает private key Microsoft, любой может forge Microsoft's digital signature and create bootable game disks for the Xbox, при условии что Microsoft никогда не удалит из Xbox kernel способность load programs from regular CD or CD/RW media. Существенно, Microsoft ships its games on 2-layer DVD-9 format disks with special security structures. Xbox firmware могла бы быть configured by Microsoft to only boot from disks that have this particular structure, regardless of the digital signature check. Поскольку сейчас impossible burn 2-layer DVDs using a common DVD burner drive, требование secured DVD-9 media как единственного source for executables стало бы impairment для распространения Xbox-Linux через free downloads off the Internet. Другая проблема этого approach в том, что chance successfully factoring Xbox's private key

through brute force search очень, очень мал. (Глава 7, «A Brief Primer on Security», содержит боковую вставку «Very Difficult Problems», которая пытается communicate computational difficulty этой задачи.) Если private key успешно recovered within a reasonable amount of time этим approach, это significantly reduce people's confidence in RSA algorithm. (С другой стороны, you can never win the lottery if you don't buy a ticket, and running a free distributed factoring client using the spare cycles on your CPU is much cheaper than a Powerball ticket.)

Другой approach, related to cracking RSA-2048 bit key, - модифицировать existing, signed Xbox executable полезным образом без изменения его cryptographic hash value. Такая constructive hash collision сделала бы modified executable identical to the original as far as digital signature check is concerned. Hash, used in Xbox's digital signature algorithm, - SHA-1. SHA-1 - 160-bit hash без publicly known algorithmic weaknesses; поскольку source of the hash fixed, пришлось бы попробовать примерно 2^160 random variations, чтобы discover a collision. В качестве side note: нельзя использовать birthday attack, чтобы reduce difficulty attack to 2^80 random variations, потому что мы не trying to find two messages that hash to the same arbitrary value. Цель - generate a specific target hash, or perhaps one of a very limited set of target hashes harvested from the set of all published Xbox game titles. Следовательно, этот approach также попадает в category «Very Difficult Problems».

Рисунок 11-1. Core team Xbox-Linux на 19th annual Chaos Computer Conference, проведенной в Berlin, Germany. Сзади Michael Steil; спереди, слева направо: Andy Green, Milosch Meriac и Franz Lehner. (Photograph courtesy of Gerhard Farfeleder.)

Альтернативный approach к Project B - найти security holes в Xbox software и использовать holes, чтобы seize control of CPU's instruction pointer. Чтобы увидеть, чем это полезно, рассмотрим example: предположим, network-based buffer overrun exploit обнаружен в game и может lead to arbitrary code execution. Program, running on PC connected to Xbox via network, могла бы затем use this exploit to send packets to Xbox, которые установили бы simple bootloader for Xbox-Linux. Этот bootloader мог бы быть чем-то столь простым, как program, которая runs code at a designated location

on Xbox's hard drive or on DVD drive. Любой port, где Xbox can accept data, является vector for this kind of attack, включая USB и network port, а также hard drive и DVD-ROM drive.) Corrupted save games or file structures can be imaged onto hard drive or DVD-ROM drive, которые cause Xbox to run user-developed code. К чести Microsoft, все network interactions и save game protocols используют fairly strong and well-tested security techniques. Кроме того, я слышал на presentation about Xbox by Microsoft at MIT, что весь game code inspected by a buffer overrun checker и что Microsoft имеет contractual remedies against game developers, found guilty of putting deliberate back doors into their game code. Это указывает, что code base Xbox более secure, чем typical Microsoft product, что делает ее тем более interesting as a problem for hackers to work on. (Если вы заинтересованы участвовать в hacking on Xbox как часть «Project B», я призываю сначала проверить Project B Prize Rules web page по адресу http://xbox-linux.sourceforge.net/articles.php?aid=20030023081956.)

Недавно buffer overrun exploit был обнаружен в том, как saved games handled by Electronic Arts' «007: Agent Under Fire» game. Exploit first divulged by hacker, known simply as «habibi_xbox», 29 марта 2003 года через posting на XboxHacker.net BBS. Существенно, exploit был identified in undisclosed number of games, но «007: Agent Under Fire» был единственной game, explicitly named in the posting. Exploit leverages unchecked string to run a short segment (a few hundred bytes) of code, который inserts a series of kernel patches. Various measures были included in design of the hack, чтобы make it very difficult to modify the hack to do anything other than run intended Xbox-Linux target. Например, hack patches original Xbox RSA public key, used for verifying digital signatures, with a new public key, while leaving digital signature check algorithm unpatched. Только Xbox-Linux bootloader, provided as part of the hack, appropriately signed with corresponding new private key. Другие hackers должны были бы factor the new public key, чтобы use this hack to run other executables. Также сама game «007: Agent Under Fire» performs an independent digital signature check on all saved games, так что modifying exploit code in hacked savegame file is not trivial. Включение таких security measures in the hack - laudable decision со стороны implementer hack, поскольку это помогает ensure that hack is not directly useful for applications such as piracy. Реализация security measures, которые protect Microsoft's interests, может помочь save Xbox-Linux project from wrath of Microsoft and U.S. Department of Justice.

Профиль: Milosch Meriac

Можете рассказать нам немного о себе?

Моя general history довольно проста. Я родился в 1976 году в Czechoslovakia. Мои parents (моя mother - teacher, father - civil engineer) escaped during the Cold War to western Germany из-за repressions by communist regime. Мне было около трех лет, когда мы arrived in Germany. В German kindergarten я immediately learned German language. С этого момента все было really simple - в десять лет я получил свой first computer после нескольких месяцев whining. Things started to roll.

После school leaving exams и weird intermezzo at German Federal Armed Forces Military Duty я начал studying cybernetics and computer science, но через три года решил quit university and concentrate as long-term objective on my own company. During my studies я established some valuable business connections, поэтому было легко work as freelancer for various companies in Germany. Я делал reverse engineering projects, developed realtime embedded linux systems with small footprint, занимался lowlevel programming вроде realtime extensions for Windows systems, и developed software based harddisk safeguard для famous German company. Сейчас я живу со своей girlfriend в Berlin, и мы having a great time there.

Почему вы hack?

Получив больше опыта в programming, я начал discover, что beautiful and bright entity computer world на самом деле fragile patchwork.

Вначале hacking был для меня like a game. Можно было walk around inside your computer system, discovering worlds of new code and possibilities every single day. Occasionally one could challenge application authors to a duel by trying to analyze and circumvent their copy protections. Иногда это было like playing chess; other times it was like a deathmatch.

С одной стороны, я was excited to see my knowledge growing, а с другой это naturally a great ego boost для 14-year-old child - circumvent security systems of overpaid godlike hardcore programmers. Во время senior high school я revised this view: while programming tools and applications for some local companies during school vacations, я встретил some genuine programmers - and was disappointed: they were neither gods, nor godlike.

Через некоторое время я realized, что writing a cool demo, hacking application X, or finding a nifty hack for Y не changes the world more than a sack of rice toppling down somewhere in China. Поэтому я начал choosing my realms more wisely - technologies of everyday life like telephones, computers, networks and satellites. Я found out, что one has the power to change things by explaining technology to average users or by helping companies to secure their products.

Сегодня я aware of my power as whitehat hacker. Every person in today's life affected by information technologies: surveillance techniques, data mining, information warfare, Digital Millennium Copyright Act, TCPA, digital rights management, new interpretations of copyright and patent law растут like mushrooms after monsoon rain. Как и в прошлом, я ache to peek behind these beautiful and bright entities, and hopefully find bugs and traps before they find us.

Можете рассказать нам о вашем опыте с проектом Xbox-Linux?

Я joined Xbox Linux project и помог get the kernel running, что было tricky, потому что Xbox architecture имеет some traps and differences compared to personal computer. Я created early Linux distributions for Microsoft's Xbox. Это было важно, потому что у нас было only 1 MB flash available to store complete distribution and kernel, and hard disk wasn't unlocked yet. Я также provided a console driver for Andy Green's filtror device, так что мы могли see kernel boot messages and get a linux console using his device as some sort of remote interface. This distribution already included network drivers, soundcard drivers, mp3 support, a telnet server, webserver, NFS support, and a broad range of standard linux tools. Это enabled us to get rid of our custom-made hardware и позволило hundreds of people join project, either as code contributors or as test persons. У нас еще не было screen output, поэтому я added framebuffer interface to Xbox Linux kernel и made many other contributions.

Число contributing developers начало enormously grow. Мы получаем awesome help со всего мира, чтобы make Xbox Linux possible. Some stay hidden because they are afraid of legal uncertainties like DMCA in United States, while others can contribute freely.

Есть ли другие comments, которыми вы хотели бы поделиться?

Некоторые people may ask, почему full-grown people like me fiddle about with this Xbox toy. У каждого человека, конечно, свои reasons; моя reason - improve my skills and learn more about recent technologies. Microsoft Xbox, например, predecessor of a TCPA/Palladium protected computer, со всеми technical and social implications. Это fine playground for my research on more secure computer systems without pressing users.

Одна из главных причин - наше community. It is really fun and a great pleasure to work together with these bright geeks - online and especially offline in a pub with pints of fine beer. Я amazed every day by growing strength of our community. Thanks to all for making this possible!

Оглядываясь вперед, success Project B может означать либо new age for Xbox hacking, либо demise of Xbox hacking. Даже несмотря на то что Project B hackers demonstrated social conscience and good will, пытаясь protect Microsoft's interests, невозможно prevent less scrupulous hackers from reverse engineering the hack and eventually figuring out how to reproduce the technique in some less Microsoft-friendly form. Конечный результат мог бы быть либо harsh crackdown by Microsoft upon all

hacking activity, либо Microsoft exiting video game business altogether, since their revenue stream would be cut off like Sega's in the Dreamcast piracy debacle. Или Microsoft could just elect to plow more money into the business and release redesigned console, incorporating patches and countermeasures for known security holes. Outcome будет heavily depend upon how events unfold in the next few months. Однако, with deep price cuts on the horizon for Xbox and rumors of a thoroughly redesigned «shrink» version of console floating around, seems that Microsoft's near-term strategy is to focus its energies on storming the market instead of stemming fair-use or piracy. After all, every Playstation2 or Gamecube sold probably has a worse effect on Microsoft's business than every Xbox converted to run GNU/Linux, or even an Xbox converted to run pirated games.

OpenXDK

Многие interesting and useful projects для Xbox, такие как XboxMediaPlayer и MAME-X (Multiple Arcade Machine Emulator for the Xbox), были developed for native Xbox gaming platform. К сожалению, эти programs были developed using unauthorized versions of Microsoft Xbox SDK (Software Development Kit). Microsoft Xbox SDK supposed to be available only to approved, licensed developers. Однако SDK leaked even before console was launched, и с тех пор многие использовали leaked Xbox SDK для создания собственных Xbox programs. Хотя proprietary Xbox SDK convenient and easy to use, его также technically illegal to use. Lack of legal SDK for native Xbox platform затрудняет привлечение large base of open-source developers.

Проект OpenXDK был создан, чтобы address need for legal alternative to Xbox SDK. Stated goal OpenXDK - create legal development kit for creating Xbox Executables (XBEs). OpenXDK позволит users create native XBE files, которые, when signed with appropriate digital signature, could run on vanilla Xbox. Since this appropriate digital signature is as of yet unknown, this work is done in anticipation of legal technology that enables interoperability with programs developed using OpenXDK.

Несмотря на свою utility, проект OpenXDK все еще в nascence и ищет developers. Больше об OpenXDK project можно найти по адресу http://openxdk.sourceforge.net. Project managers OpenXDK - Dan Johnson (also known as SiliconIce, creator of XboxHacker BBS) и Aaron Robinson (also known as caustik; caustik also leading CXBX executable relinker and CXBE Xbox emulator projects).

Глава 12. Caveat Hacker

Reverse engineering и закон об интеллектуальной собственности имеют несколько хитрых юридических взаимодействий. С одной стороны, инновация заслуживает справедливой награды. Право изобретателей или авторов исключительно производить или продавать плоды своего труда должно быть защищено. С другой стороны, свободный и конкурентный рынок также необходим, чтобы сохранять инновацию и обеспечивать честные рынки. Изучение принципов проектирования, воплощенных в существующих продуктах, и способность производить улучшенные производные продукты являются важной частью конкурентного marketplace.

Эта глава дает обзор закона об интеллектуальной собственности и некоторых более важных частей, которые вам, как хакеру, нужно знать. Неведение не является действительной защитой, и закон предписывает суровые наказания для тех, кто игнорирует законы, регулирующие reverse engineering и права интеллектуальной собственности. Некоторые акты нарушения интеллектуальной собственности наказуемы как felonies, вместе с крупными штрафами.

Большая часть этой главы была написана Lee Tien, Senior Staff Attorney в Electronic Frontier Foundation. Lee (и Joseph Liu) были моими counsel в период, когда я пытался опубликовать свои findings о системе безопасности Xbox. Глава 8 содержит боковую вставку «The Legal Challenges of Hacking», описывающую мою борьбу с MIT за публикацию моей статьи.

Содержимое этой главы представлено с намерением предоставить informational resource для хакеров. Если вы думаете, что можете находиться в юридически compromising situation, нет замены обращению к attorney и получению proper legal advice по вашей конкретной ситуации.

Профиль: Lee Tien

Lee Tien - Senior Staff Attorney в Electronic Frontier Foundation, специализирующийся на free speech law, включая пересечения с intellectual property law и privacy law.

До прихода в EFF Lee был sole practitioner, специализировавшимся на litigation по Freedom of Information Act (FOIA). Mr. Tien публиковал articles о children's sexuality and information technology, anonymity, surveillance и First Amendment status публикации computer software. Lee получил undergraduate degree in psychology в Stanford University, где был очень active in journalism в Stanford Daily. После года работы news reporter в Tacoma News Tribune Lee пошел в law school в Boalt Hall, University of California at Berkeley. Lee также выполнял graduate work in the Program in Jurisprudence and Social Policy at UC-Berkeley. [1]

Electronic Frontier Foundation

Electronic Frontier Foundation (EFF) предоставила мне legal counsel в период, когда я пытался опубликовать свою статью о системе безопасности Xbox. Следующие paragraphs вводят, что делает EFF и кто они.

Представьте мир, где technology может дать всем нам power делиться knowledge, ideas, thoughts, humor, music, words and art with friends, strangers and future generations.

Этот мир здесь и сейчас, сделанный возможным электронной сетью - Internet - с power соединить всех нас. А будущие developments in technology позволят нам access information and communicate with others еще более powerful ways.

Но governments and corporate interests worldwide пытаются prevent us from communicating freely through new technologies, точно так же, как когда люди in positions of power контролировали production and distribution of - or even burned - books, которые они не хотели, чтобы people read in the Middle Ages. Но только by fighting for our rights to speak freely whatever the medium - whether books, telephones, or computers - can we protect and enhance the human condition.

Electronic Frontier Foundation (EFF) была создана, чтобы defend our rights to think, speak, and share our ideas, thoughts, and needs using new technologies, such as the Internet and the World Wide Web. EFF первой identifies threats to our basic rights online and advocates on behalf of free expression in the digital age.

Based in San Francisco, EFF - donor-supported membership organization, работающая to protect our fundamental rights regardless of technology; educate the press, policymakers and the general public about civil liberties issues related to technology; and act as a defender of those liberties.

Среди наших various activities EFF opposes misguided legislation, initiates and defends court cases preserving individuals' rights, launches global public campaigns, introduces leading edge proposals and papers, hosts frequent educational events, engages the press regularly, and publishes a comprehensive archive of digital civil liberties information на одном из most linked-to websites in the world: http://www.eff.org.

Источник: From the EFF website, http://www.eff.org/abouteff.html.

Caveat Hacker: введение в интеллектуальную собственность, Lee Tien

Reverse engineering - процесс извлечения know-how или knowledge из artifact; на marketplace его называли «time-honored technique of figuring out just what makes a competitor's product tick». [2] Но любой, кто сегодня изучает mass-marketed products, должен быть осведомлен о legal minefield вокруг reverse-engineering. Anti-circumvention provisions of the Digital Millennium Copyright Act (DMCA), [3] contractual terms, запрещающие reverse-engineering, и Economic Espionage Act [4] - несколько опасных юридических областей, о которых technologists должны знать. Эта глава кратко обозреет эти области, чтобы дать хакерам rough idea of the issues.

Здесь есть два общих вопроса. Во-первых, lawful ли reverse engineering? Во-вторых, даже если вы можете reverse engineer продукт, можете ли вы публиковать то, что узнаете из reverse engineering?

Классическое право интеллектуальной собственности: обзор

Intellectual property law традиционно означало copyrights и patents. Оба создаются и ограничиваются federal statutes, основанными на clause об интеллектуальной собственности Constitution: «Congress shall have the Power . . . To promote the Progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries.» [5] Computer programs обычно защищаются как copyrighted «literary works», но они также могут быть patented. [6]

В последнее время люди стали думать о trade secrets как еще об одном виде intellectual property. Trade secrets изначально защищались courts under case law, но теперь они также являются предметом и state, и federal statutes. В отличие от copyrights и patents, trade secrecy law исторически основан на principles of unfair competition.

В United States authors и inventors не имеют «natural rights». [7] Вместо этого их права основаны на понятии public welfare. Society will benefit, если authors и inventors получают некоторую protection, потому что у них не будет adequate incentives создавать, если другие смогут свободно использовать их work. Но эта protection ограничена, чтобы assure that the public ultimately benefits. [8] Например, copyright и patent rights действуют только for «limited times»; eventually protected works must enter the public domain. [9] Короче, intellectual property law устанавливает terms for a «bargain» between the public and authors or inventors.

Авторское право

Copyright law защищает original works of expression, которые «fixed» in a tangible medium, и дает author (or assignee) exclusive rights over reproduction, distribution, adaptation, public display and public performance of the work. Он не защищает against independent creation.

Works - не то же самое, что copies или phonorecords (copies of sound recordings). Когда вы покупаете book, вы владеете copy, но copyright owner сохраняет rights to the work itself. Обратите внимание, кстати, что doctrine «first sale» позволяет lawful owners of copies продавать или передавать эти lawfully owned copies, [10] with certain exceptions. [11]

Есть много разных types of works, с множеством different rules для каждого type. Поэтому copyright law довольно complex, а technology не упростила matters. Рассмотрим copyrighted song. Song или musical composition (MC) защищена copyright, обычно held by songwriter. Чтобы record the song, нужно permission from the MC copyright owner. [12] После записи возникает independent copyright in the sound recording (SR), который защищает actual recorded sounds, including singer's interpretation of the underlying song, as well as efforts of producer and sound engineers. Record companies обычно владеют SR copyrights. В результате, если вы хотите использовать copyrighted sound recording of the song in a TV commercial, вам нужно permission и MC copyright owner, и SR copyright owner.

Большинство rights copyright owner довольно obvious, но некоторые - нет, особенно когда involved computers. Например, computers load programs into RAM, создавая copy for copyright purposes. Copyright act содержит specific exemption, который позволяет owner of a copy of a computer program copy the program into computer memory. [13] Это иллюстрирует general strictness of copyright law: то, что нельзя использовать copyrighted work for its intended purpose without making a copy, не означает, что making the copy is not copyright infringement. Implications этой strictness для Internet серьезны, поскольку Internet dissemination generally involves the making of copies.

Right over adaptation также может сбивать с толку. Adaptations, или «derivative works», - works based on a copyrighted work: foreign-language translations, movies based on books, and so on. В одном much-criticized case суд found, что вырезание pictures из lawfully owned copies и mounting pictures onto ceramic tiles создавало infringing derivative works. [14] Большинство courts disagree with this result. [15]

Copyright protection начинается automatically when a work is created и generally lasts for the life of the author plus 70 years. [16] Works become free for all to use, то есть enter the public domain, once the copyright term expires.

Есть много exceptions to copyright. Правило, что copyright protects expression, означает, что он не запрещает никому использовать ideas или facts, revealed in the work. «Ideas» includes plots of stories. More generally, copyright doesn't protect utilitarian aspects of a work, так что вы можете написать computer program, который does the same thing as another program, so long as you don't copy its expression.

Facts considered «outside» copyright, потому что они discovered, not authored. Это включало бы, например, discovery of new prime numbers. Но можно иметь copyright in the selection, sequence or arrangement of facts или anything else that is not itself copyrightable. Classic example - anthology of public-domain poetry. Можно иметь copyright to the compilation, even though individual pieces are unprotected, if the selection, sequence or arrangement is sufficiently original. Alphabetical arrangement of facts in typical telephone «white pages» directory fails the constitutional originality requirement. Вы не получаете protection merely because you invested money, time or effort into collecting the phone numbers.

Copyright doesn't cover many «ordinary» uses of the work. Само по себе чтение book не subject to copyright, потому что не infringes any of the copyright owner's rights. Singing a song in the shower is a performance, but copyright owner only has a right over public performances. Здесь снова, однако, Internet изменил things. Когда вы читаете document in your web browser, copy of the document was probably made by your computer. Thus many formerly ordinary uses now entail making a copy, which raises copyright issues.

Сегодня много controversy about «fair use». Fair use - defense to copyright infringement, intended to allow people to make some unauthorized use of copyrighted works. Fair use allows book reviewers to quote from books. Это very complicated area of law; whether a use is «fair» depends on factors like purpose, nature, amount, and economic effect of the use. [17]

Патенты

Patent law защищает inventions и дает inventor (or his assignee) право exclude others from making, selling, or using the invention for 20 years from the date of filing of the patent. Unlike copyright, patent law protects against independent invention by another person.

Bargain here is that in return for the patent, the inventor must provide enough information in the patent application to enable one «skilled in the art» to create the invention without much experimentation. Once a patent is awarded, the application is made public. By making the information public, the patentee contributes to society's store of knowledge.

A patent confers no affirmative rights, however; if you patent an improvement to someone else's invention, you can't practice the improvement without infringing on the underlying patent. If you invent and patent a new drug, you may still need regulatory approval before you can sell the drug.

Чтобы быть patentable, invention must be useful, novel, and «nonobvious» to one «skilled in the art». Novelty and nonobviousness requirements mean that the invention must be a sufficient development in technology before the right to exclude is given. Developments that do not meet these high standards are denied protection.

Коммерческие тайны

Третья область law - trade secrecy - также considered part of intellectual property law, although it is not really property. Trade secret - commercially valuable business or other information, known to the user but not to competitors. Secrecy, although not absolute secrecy, is the essence of a trade secret; нужно take reasonable precautions to protect the trade secret against disclosure.

Есть obvious relationship between patents and trade secrets, because both protect useful information. If the useful information isn't patentable at all, there's no choice. But one might not want to patent a patentable invention for several reasons. You might not want to disclose information in the patent application. Also, if you don't expect the technology to be valuable for very long, it might not be worth getting a patent that lasts 20 years.

Main downside of trade secrecy is that it provides no protection against independent invention or against reverse-engineering. Trade secrecy therefore unwise if the secret can be figured out from the product. If, on the other hand, the invention is a process used in making the product, it might be hard to reverse-engineer. Even though Coca-Cola has been on the market for many years, apparently no one has figured out how to duplicate it.

Конституционная сделка авторского права

Intellectual property rights are a means to an end - to promote the progress of knowledge and technology. Как однажды сказал Supreme Court, «the monopoly privileges that Congress may authorize are neither unlimited nor primarily designed to provide a special private benefit.» [18]

Passage above указывает, что intellectual property law давно concerned about limiting potential monopoly power, conferred by copyright and patent law. For instance, first-sale doctrine prevents patent and copyright owners from controlling the market once patented products or copies of copyrighted works are sold.

Также copyright law давно interpreted by courts and crafted by Congress to preserve a balance with freedom of speech. Doctrines like idea/expression dichotomy, fair use doctrine, and copyright's limited term generally viewed as reducing potential conflict between copyright and freedom of expression. [19]

Interestingly, concern about monopolies historically linked to concern for free speech. English copyright law had long functioned as a kind of state-sponsored cartel; in return for private monopolies over writings, publishers agreed to act as policemen of the press in service of government censorship - in particular, Bible and other religious works. [20] Similarly, copyright law's idea-expression dichotomy ensures that uncopyrightable facts and ideas and unpatentable functional principles remain in the public domain for future creators to build on.

Традиционный взгляд на Reverse Engineering

Historically, reverse engineering has always been a lawful way to gain information embodied in mass-marketed products. For many technology firms, reverse-engineering competitors' products to study their innovations is a standard practice. Indeed, U.S. courts have also treated reverse engineering as an important factor in maintaining balance in intellectual property law, and the Supreme Court has called reverse engineering «an essential part of innovation».

The law recognizes three main purposes of legitimate reverse engineering. Competitive reverse-engineering intended to create a direct substitute. Compatibility or interoperability reverse-engineering aimed at figuring out how to make a product that works with the reverse-engineered product. And of course, researchers often reverse-engineer products in order to gain knowledge with no commercial purpose.

Trade Secrecy и «Improper Means»

In general, a trade secret is misappropriated only if a person or firm misuses or discloses the secret in breach of an agreement or confidential relationship, engages in other wrongful conduct (e.g., bribery, coercion, trespass) to obtain the secret, or acquires the secret from a misappropriator knowing or having reason to know that the information was a misappropriated trade secret.

Most states, like California, explicitly provide that reverse engineering is a lawful way to acquire a trade secret. Several reasons support reverse engineering as a sound principle of trade secret law. [21] Buying a product in the open market generally gives the buyer personal property rights in the product, which include the right to take the product apart, measure it, subject it to testing, and the like. The law also regards sale of a product in the open market as a publication of innovations it embodies and a dedication of them to the public domain unless the creator has obtained patent protection for them.

Vulnerability of trade secrets to reverse engineering is part of the overall constitutional scheme. In Bonito Boats v. Thunder Craft Boats, the Supreme Court struck down a Florida law that forbade manufacturers of boats from using existing boat parts as «plugs» for a direct molding process that yielded competing products because the law «prohibit[ed] the entire public from engaging in a form of reverse engineering of a product in the public domain.» [22] The court explained that reverse engineering is «an essential part of innovation», likely to yield variations on the product that «could lead to significant advances in technology». Indeed, «the competitive reality of reverse engineering may act as a spur to the inventor» to develop additional patentable ideas.

In cases like Bonito Boats, the question is whether a state law is «preempted» by federal law. When federal and state law conflict, either directly or as a matter of federal policy goals, the state law loses under the doctrine of «conflict» preemption. This stems from the Constitution's Supremacy Clause, under which federal law generally trumps state law. [23] Copyright law also contains a specific preemption clause, discussed below.

Авторское право и проблема промежуточного копирования

Until recently, copyright law didn't need to worry about reverse engineering, because there was little reason to reverse engineer books, art, or music. Now that computer programs are «literary works», things are much different. Since many computer programs are distributed only in object code, the reverse engineering process commonly requires an initial decompilation into source code - which entails making a copy.

U.S. courts have found that copyright law does not necessarily prohibit reverse-engineering, because copying incidental to reverse engineering can be a «fair use»: «The Copyright Act permits an individual in rightful possession of a copy of a work to undertake necessary efforts to understand the work's ideas, processes, and methods of operation.» [24] This can be true even when the ultimate goal of the reverse engineering is commercial. The courts generally rely on the Constitutional purpose for copyright protection: «the promotion of 'the Progress of Science....'» [25] The fair use doctrine advances this Constitutional objective by «encourag[ing] others to build freely upon the ideas and information conveyed by a work.» [26]

Key case here was Sega Enterprises Ltd. v. Accolade, Inc. [27] Accolade disassembled Sega game programs in order to get information necessary to make its games compatible with Sega Genesis game console. Accolade then sold its own games in competition with games made by Sega and its licensed developers. Accolade raised a fair use defense to Sega's claims that the disassembly copies were infringing. The court accepted Accolade's defense for the reasons described above. It also noted that if Accolade could not disassemble Sega's code, Sega would get «a de facto monopoly over [the unprotected] ideas and functional concepts [in the program]», which is only available under patent law. [28]

The court's holding, however, was limited to reverse engineering undertaken for a «legitimate reason», such as to gain access to the functional specifications necessary to make a compatible program, and then only if it «provides the only means of access to those elements of the code that are not protected by copyright.» [29]

Патентное право

There is no general fair use defense or reverse-engineering exemption in patent law. In theory, you shouldn't need to reverse-engineer a patented product, because the patent specification should inform the relevant technical community of the best way to make the invention.

Some reverse engineering activities will not infringe a patent. The buyer of a machine embodying a patented invention, for example, is generally free to disassemble it to study how it works under patent law's first-sale principle. Buying the product means that you have the right to use it, and simply studying it doesn't infringe the patent owner's exclusive rights to make or sell the invention. Nevertheless, courts sometimes enforce contractual restrictions on reverse engineering. [30]

Also, one who tries to make a patented invention to satisfy scientific curiosity may have an «experimental use» defense. Under U.S. law, this defense is narrow and probably does not include research uses that may lead to development of a patentable invention or a commercial product. [31]

The clash between these three areas can be seen if we look again at the Sega situation. Suppose Sega had a patent on an algorithm used in all of its game programs. By disassembling Sega programs, Accolade arguably «makes» or «uses» the patented algorithm, even if it did so inadvertently. In short, the intermediate copying problem reappears in the patent context.

Новые вызовы для reverse engineers

Importance of reverse engineering has only grown with the rise of commercial cryptography in mass-marketed products, because it is impossible to make systems more secure without trying to break them. Ironically, growing use of encryption has contributed to laws against reverse engineering. The entertainment industry, for example, now relies on encryption and other technologies to protect digital information like music on CDs and movies on DVDs against unauthorized copying.

Unsurprisingly, new laws have been enacted to prevent people from «circumventing» encryption and other forms of security.

Legal encroachments to reverse engineering haven't been limited to encryption. In the 1970s and 1980s some states forbade the use of a direct molding process to reverse-engineer boat hulls. [32] In the late 1970s and early 1980s, semiconductor industry sought and obtained legislation to protect chip layouts from reverse engineering to make clone chips. [33] A major international agreement on intellectual property rights says nothing about reverse engineering. [34]

Digital Millennium Copyright Act и проблема unauthorized access

DMCA is one of the most important laws that now regulate reverse engineering. One part of the DMCA - its «anti-circumvention» provisions - gives legal protection to technical measures that effectively control access to or prevent copying of a copyrighted work. Unfortunately, the DMCA is extremely complex; for instance, the DMCA makes it unlawful to bypass «effective technical protection measures» without clearly specifying what that term means.

Несанкционированный доступ

DMCA essentially creates a new right of «access» for copyright owners. Spokesmen for the copyright industry liken the act of circumventing a technical protection system to «breaking and entering» a home.

One simple example is censorware programs used by schools and libraries to prevent children from viewing inappropriate images. These programs often contain encrypted «blacklists» of censored websites, which vendors typically treat as trade secrets. Suppose a researcher finds that a particular program blocks sites that are wholly appropriate for children, and wants to read the blacklist in order to figure out how many appropriate websites are being wrongly blocked. [35] Because the vendor has encrypted the blacklist in order to prevent people from gaining access to its content, and the list is arguably a copyrighted compilation of facts, the encryption is a technical protection measure applied to a copyrighted work and unauthorized decryption would be an unlawful act of circumvention - except that the DMCA currently has a temporary exemption for decrypting censorware blacklists.

Another example: the movie industry uses an encryption scheme called Content Scrambling System (CSS) to protect movies on DVDs. In the 2600 case, [36] CSS was held to be a technical measure that «effectively» controls access to movies. Bypassing CSS without the copyright owner's authorization is an unlawful «act of circumvention» under the DMCA. Note that the courts have not found that the fair use doctrine applies to the DMCA (as opposed to copyright law). Thus, if the use of CSS prevents you from fast-forwarding through the commercials on a DVD movie - it is still unlawful to «circumvent» that restriction.

Note here that the notion of «effective» here is not connected to cryptographic efficacy. Even weak encryption is «effective» under the DMCA because the ordinary person could not defeat it.

Технологии обхода

DMCA protects technical measures in a second way: its «anti-device» provisions outlaw the manufacture and distribution of technologies that enable circumvention. [37] Continuing the «breaking and entering» metaphor, spokesmen for the copyright industry liken circumvention technologies to «burglars' tools», which are illegal in many states.

Section 1201 of the DMCA states that «[n]o person shall manufacture, import, offer to the public, provide, or otherwise traffic in any technology, product, service, device, component, or part thereof» if it has one or more of the following three characteristics: (1) if it is «primarily designed or produced for the purpose of circumventing [technical] protection», (2) if it has «only limited commercially significant purpose or use other than to circumvent [technical] protection», or (3) if it is «marketed by that person or another acting on its behalf with that person's knowledge for use in circumventing technical protection».

Note that these provisions apply not only to the new right of «access control», but to the rights of copyright owners generally. Thus, technologies that would circumvent copy-protection measures for CDs can be unlawful under these provisions.

Recall the two examples just given. In the 2600 case, at issue was the DeCSS program, which enables people to decrypt DVD movies protected by CSS. DeCSS was found to be a prohibited circumvention technology. In the censorware example, the DMCA exemption permits the act of decryption, but it says nothing about whether a censorware researcher can make available the computer program used to decrypt the encrypted blacklist, or even the details of the method of decryption.

Навигация по исключениям DMCA

Just as you can't reverse-engineer object code without decompiling or disassembling it, you can't reverse-engineer a technical protection measure without circumventing it. Moreover, you often need a technological device or tool to actually perform reverse engineering, so the ban on circumvention technologies also restricts reverse engineering.

In combination, these DMCA provisions create major barriers to cryptographers and security researchers who want to analyze the security measures used in real, mass-marketed products. A commercial reverse engineer who discovers a problem with another firm's technical measure and offers suggestions about how to improve it is at risk of being indicted on criminal DMCA charges.

Even an academic reverse engineer is at risk of being sued for publishing a paper about the weaknesses in a firm's security measures, because such a paper could be labeled a «tool of circumvention». [38] One example is Princeton professor Edward Felten, who assembled and entered a team of scientists in the music industry's «SDMI Challenge», a contest to crack digital watermarking and other technologies being considered by the Secure Digital Music Initiative for protecting digital music. Felten and his team entered the contest with the intent of using the SDMI Challenge as a real-world security case study, and they eventually authored a peer-reviewed academic paper that was to be presented at a conference. Before the paper was actually presented, the Recording Industry Association of America (RIAA) sent Felten and the conference organizers a letter warning him that publishing the paper would violate intellectual property laws, including the DMCA.

The DMCA also contains several exemptions relevant to reverse engineering: circumvention of a technical protection system when necessary to achieve interoperability among computer programs; circumventions conducted in the course of legitimate encryption research; and circumvention for purposes of computer security testing. Unfortunately, each of these exemptions is both complex and narrow. Even when the act of reverse-engineering is allowed, the DMCA strictly regulates what can be done with the resulting information.

1201(f): обратная разработка для совместимости

This exemption allows the circumvention of technical protection measures for interoperability reverse engineering. It also allows, to a very limited extent, the dissemination of information gained from reverse-engineering. Note that 1201(f) would not have exempted Felten's attack on the SDMI watermarks, because it had no relation to interoperability.

The 2600 case, mentioned earlier, concerns the publication of a computer program known as «DeCSS» on the website of 2600 Magazine. DeCSS can be used to bypass CSS, the technical protection measure used to control access to DVD movies. EFF, which represented 2600 Magazine, argued that DeCSS

qualifies for the interoperability privilege of 1201(f). DeCSS was designed, we argued, to enable people to build software that would enable them to play legitimately purchased DVD movies on their platform of choice, namely, Linux computer systems. The courts rejected this argument, saying that 1201(f) only permitted circumvention for purposes of achieving program-to-program interoperability, whereas DeCSS enabled program-to-data interoperability that 1201(f) did not cover. This ruling is dubious, because there are computer programs as well as data on movie DVDs.

While 1201(f) seems to follow Sega in permitting interoperability reverse-engineering, it is more restrictive in several ways: interoperability is the only legitimate purpose for which reverse engineering may be done; only program-to-program interoperability qualifies, even though circumvention may be needed to achieve hardware-to-program interoperability or program-to-data interoperability; and the information resulting from reverse engineering cannot be freely published.

1201(g): исследование шифрования

The DMCA also contains an express exemption for encryption research. Unfortunately, it is also very narrow. For one thing, this exception only applies if the cryptographer has asked (even if he or she has not received) permission from the copyright owner to engage in an act of circumvention before the circumvention is accomplished. Second, the statute emphasizes the need for a cryptographer to be an expert in order to qualify for this exemption, even though some of the most brilliant minds in the field of cryptology lack formal training. Third, the statute permits a cryptanalyst to make tools to bypass access controls, but is silent on whether tools to bypass use or copy controls are permissible (that is, it contains an exception to one but not both of the anti-device rules). Fourth, it regulates the cryptologist's ability to disseminate the results of decryption.

Consider again Prof. Felten's SDMI research: it would not be exempted by 1201(g) because digital watermarks are not encryption.

1201(j): исследование безопасности

The DMCA's security research exemption has a similar structure: it applies only if the tester asks in advance and likewise allows making tools only to bypass access controls, not copy or use controls. Like 1201(g), it too regulates the tester's dissemination of the results of the testing.

Even in this narrow form, it is not clear whether Felten's research would be covered. Sec. 1201(j) only permits making a tool to bypass an access control. Is a digital watermark an access control or a copy control? The answer to this question depends to a large extent on how the watermark is used. EFF argued that, as contemplated by the RIAA, the SDMI watermark technologies were both access and copy control technologies.

Лицензионные соглашения и договорные запреты обратной разработки

Intellectual property isn't the only obstacle to reverse engineering. It's common for software licenses to prohibit reverse engineering. A typical license clause might say: «You may not, and you may not permit others to, (a) disassemble, decompile or otherwise derive source code from the Software, (b) reverse engineer the Software, (c) modify or prepare derivative works of the Software, (d) copy the Software, except as expressly permitted in this Agreement, (e) rent or lease the Software, or (f) use the Software in any manner that infringes the intellectual property or other rights of Licensor or another party.»

Companies argue that such provisions legally bind purchasers not to reverse engineer their software. If they do so anyway, they have breached a contract and can be sued for damages. The problem, of course, is that the anti-reverse-engineering provision gives the copyright owner rights beyond those it would have under, say, the Sega decision.

Whether this kind of contractual prohibition is enforceable is a hotly disputed issue. Courts have sometimes rejected reverse engineering defenses in trade secrecy cases because this activity exceeded the scope of licensed uses of the software. [39] Courts have sometimes refused to enforce software shrinkwrap license restrictions against reverse engineering because of a conflict between the clause and federal intellectual property policy. In Vault Corp. v. Quaid Software Ltd., [40] the maker of copy-protection software tried to enforce an anti-reverse-engineering clause under Louisiana law against a firm that had reverse-engineered its copy-protection scheme. The court held that federal law preempted the contractual clause as a matter of federal policy, the same argument used in Bonito Boats to override the Florida boat hull law.

In addition, Section 301 of the Copyright Act preempts state-created or state-enforced rights «that are equivalent to any of the exclusive rights within the general scope of copyright . . . .» As might be expected, there's a debate about what «equivalent» means. Courts have said that contract provisions enforceable under state law are «equivalent» to federal copyright when the conditions for infringement are the same. But if infringement of the state-created right requires an «extra element», it is not «equivalent».

Such a contractual clause was recently found enforceable. In Bowers v. Baystate Technologies, Inc., [41] an inventor marketed a patented computer-aided design (CAD) software «toolkit» with an anti-reverse-engineering license clause. Baystate, a competitor, reverse engineered Bowers' software and then marketed a competing CAD toolkit. After some complicated litigation, the court eventually held, among other things, that Baystate breached its contract with Bowers.

The court held that the license wasn't preempted because a contract has an «extra element» - the parties must agree. [42] It follows that federal copyright law can never preempt a contractual prohibition. The problem with the Bowers decision is that it focuses only on the specific preemption clause of the Copyright Act and completely ignores constitutional «conflict» preemption. [43]

The Uniform Computer Information Transactions Act (UCITA) is a state legislative attempt to address these issues, but it is also mired in controversy.

Коммерческие тайны и Economic Espionage Act

The Economic Espionage Act (EEA) [44] created the first federal cause of action for trade secrecy misappropriation. But it has no reverse engineering defense. This is troubling because rights granted under the EEA arguably implicate certain reverse engineering activities previously thought to be lawful. In particular, it's unclear whether decompilation and disassembly of computer programs may violate EEA rules that forbid duplicating trade secrets.

Ответственный хакер: незнание не защищает

In general, there are two ways you can violate intellectual property laws. Direct infringement means that you actually infringed. Indirect infringement means that you facilitated actual infringement by someone else. For example, in the Betamax case, the issue was whether Sony, by selling VCRs, could be found liable for its customers' copyright infringement.

Гражданские и уголовные правонарушения и наказания

The legal theories we've talked about carry a broad range of potential penalties. The main concern is civil liability, either economic damages or an injunction against the activity or both. Damages are usually tied to the amount of harm caused by the infringement.

In patent law, for example, the usual basis for damages is that of a «reasonable royalty». The court will calculate how much you should have paid the patent owner in royalties if you had contracted for a license. Damages can also be based on the infringer's profits or the patent owner's lost profits. «Willful» infringement is treated more harshly. The patent statute permits a court, in its discretion, to increase damages up to three times the base damages (and also to pay the patent owner's attorney's fees) if the infringer knew about the patent and did not consult with competent patent counsel.

The current trend in intellectual property law is toward greater attention to criminal penalties. Under the first federal copyright act in 1790, copyright infringement was a purely civil matter. It was not until 1897 that Congress added criminal penalties to the copyright act, and criminal copyright infringement was classified as a misdemeanor. [45] Moreover, criminal copyright infringement was rarely used.

Today, the risk of criminal prosecution appears considerably higher, and the criminal penalties are much greater. Amendments to the copyright act in 1982 and 1992, for instance, classified certain kinds of infringement as felonies. Even then, however, criminal infringement had to be undertaken willfully and for commercial advantage or private financial gain.

The 1997 No Electronic Theft Act (NET Act) criminalized the reproduction or distribution of one or more copies of copyrighted works that have an aggregate retail value of over $1,000 during any 180 day period, regardless of how those copies are created or distributed. It retained the willfulness requirement, but eliminated the requirement that the defendant's infringement be motivated by profit or commercial gain.

The DMCA also contains criminal provisions, which were invoked in the prosecution of Dmitry Sklyarov and the company he worked for, ElcomSoft. Elcomsoft produced and distributed software that can be used to convert digital books from Adobe's eBook format into Adobe's PDF format. In the course of the format conversion, the use restrictions imposed by the eBook format are stripped away. It was undisputed that the Elcomsoft software can be used to facilitate noninfringing uses of eBooks (e.g., fair use excerpting, or to facilitate automated translation into Braille for blind readers). Sklyarov himself was never accused of infringing a copyright, or assisting in the infringing activities of any third party. Nevertheless, for his part in developing the software, the FBI arrested him and held him in custody for 3 weeks. [46] He and Elcomsoft were indicted by a grand jury; based on the indictment, Sklyarov faced a maximum of 25 years in prison and a fine that could exceed $2 million. [47] ElcomSoft and Sklyarov eventually were found not guilty of violating the DMCA.

Reverse Engineering as «The Freedom to Tinker» and Other Legal Issues

Edward Felten, a computer science professor at Princeton University, views reverse engineering as a part of the «the freedom to tinker», which should include the freedom to «take them apart, to discuss them, to explore how they work, to modify them, to make them better». Felten argues that «as more and more of our world is experienced through electronic devices, and communications and culture are more and more mediated by these devices, it becomes increasingly important that we be able to tinker with them, to be able to understand this part of our world».

The freedom to tinker should also include the right to talk about tinkering. But as we've seen, many of the new intellectual property rules limit the right of reverse-engineers to share what they learn from tinkering. These limits not only raise serious First Amendment free-speech issues, they go to the heart of the constitutional basis for copyright and patent law: progress in the arts and sciences. One of the major issues raised by the DMCA is its chilling effect on scientists. [48]

Сноски

[1] [назад] From the EFF website, http://www.eff.org/homes/lee_tien.html
[2] [назад] Joel Miller, Reverse Engineering: Fair Game or Foul?, IEEE Spectrum, Apr. 1993, at 64, 64.
[3] [назад] 17 U.S.C. § 1201-1204.
[4] [назад] 18 U.S.C. § 1831-39.
[5] [назад] U.S. Const. Art. I, §8, cl. 8. When the Constitution was written, the word «science» was often used as a synonym for «knowledge».
[6] [назад] See Diamond v. Diehr, 450 U.S. 175 (1981); In re Alappat, 33 F.3d 1526 (Fed. Cir. 1994).
[7] [назад] In Europe, copyright has traditionally been viewed as protecting an inherent inalienable personal right of the creator of a work.
[8] [назад] «The economic philosophy behind the clause empowering Congress to grant patents and copyrights is the conviction that encouragement of individual effort by personal gain is the best way to advance public welfare through the talents of authors and inventors in 'Science and useful Arts.'» Mazer v. Stein, 347 U.S. 201, 219 (1954).
[9] [назад] Feist Publications, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 348-49 (1991) («This result is neither unfair nor unfortunate. It is the means by which copyright advances the progress of science and art.»)
[10] [назад] See 17 U.S.C. Sec. 109. «The whole point of the first sale doctrine is that once the copyright owner places a copyrighted item in the stream of commerce by selling it, he has exhausted his exclusive statutory right to control its distribution.» Quality King v. L'Anza Research Int'l, 523 U.S. 135, ___ (1998).
[11] [назад] For instance, phonorecords and stand-alone computer programs are treated differently than books under Sec. 109.
[12] [назад] Under current copyright law, the MC reproduction copyright is controlled by compulsory license provisions, which means that you automatically get permission by paying a statutory rate.
[13] [назад] 17 U.S.C. § 117 (permitting making of copy or adaptation «as an essential step in the utilization of the computer program in conjunction with a machine.»).
[14] [назад] Mirage Editions, Inc. v. Albuquerque A.R.T. Co., 856 F.2d 1341, 1344 (9th Cir. 1988), cert. denied, 489 U.S. 1018 (1989).
[15] [назад] See, e.g., Lee v. Deck The Walls, Inc., 925 F. Supp. 576 (N.D. Ill. 1996), aff'd sub nom. Lee v. A.R.T. Co., 125 F.3d 580 (7th Cir. 1997) (rejecting reasoning of Mirage Editions); Precious Moments, Inc. v. La Infantil, Inc., 971 F. Supp. 66, 68-69 (D.P.R. 1997); Paramount Pictures Corp. v. Video Broadcasting Sys., Inc., 724 F. Supp. 808 (D. Kan. 1989).
[16] [назад] Under the first copyright act, protection lasted for only 14 years.
[17] [назад] 17 U.S.C. § 107.
[18] [назад] Sony v. Universal City Studios 464 U.S. 417, 429 & 432 (1984).
[19] [назад] See generally Neil Weinstock Netanel, Locating Copyright Within the First Amendment Skein, 54 Stan. L. Rev. 1 (2001).
[20] [назад] See generally L. Ray Patterson, Free Speech, Copyright, and Fair Use, 40 Vand. L. Rev. 1 (1987).
[21] [назад] See generally Pamela Samuelson & Suzanne Scotchmer, The Law and Economics of Reverse Engineering, 111 Yale L. J. 1575 (2002).
[22] [назад] The Court went on to say that «[w]here an item in general circulation is unprotected by a patent, '[r]eproduction of a functional attribute is legitimate competitive activity.'»
[23] [назад] U.S. Const. art. VI, cl. 2.
[24] [назад] Atari Games Corp. v. Nintendo, 975 F.2d 832, 842 (Fed. Cir. 1992); see Sony Computer Ent. Corp. v. Connectix Corp., 203 F.3d 596 (9th Cir. 2000).
[25] [назад] Id., quoting U.S. Const. Art. I, §8, cl. 8.
[26] [назад] Feist Publications, Inc., v. Rural Telephone Serv. Co., Inc., 499 U.S. 340, 350 (1991).
[27] [назад] 977 F.2d 1510 (9th Cir. 1992).
[28] [назад] Id. at 1526-1527.
[29] [назад] Id. at 1518.
[30] [назад] See Pioneer Hi-Bred Int'l, Inc. v. DeKalb Genetics Corp., 51 U.S.P.Q.2d (BNA) 1797 (S.D. Iowa 1999) (enforcing a «bag tag» prohibiting purchasers of PVPA-protected corn seed from using the seed for breeding or research purposes).

[31] [назад] See Roche Prod. v. Bolar Pharmaceutical Co., 733 F.2d 858, 858-63 (Fed. Cir. 1984); Rebecca S. Eisenberg, Patents and the Progress of Science: Exclusive Rights and Experimental Use, 56 U. Chi. L. Rev. 1017, 1023 (1989).
[32] [назад] These state laws were struck down by the Supreme Court in Bonito Boats.
[33] [назад] Semiconductor Chip Protection Act, Pub. L. No. 98-620, 98 Stat. 3347 (1984) (codified at 17 U.S.C. § § 901-914 (1994)). We will not discuss this statute except to note that it contains a specific reverse-engineering privilege that permits the copying of protected chip designs in order to study the layouts of circuits, and also the incorporation of know-how discerned from reverse engineering in a new chip. Interestingly, reverse engineers must engage in enough «forward engineering» to develop an original chip design that itself qualifies for SCPA protection.
[34] [назад] Agreement on Trade-Related Aspects of Intellectual Property Rights (TRIPS), Apr. 15, 1994, Marrakesh Agreement Establishing the World Trade Organization, Annex 1C, Legal Instruments - Results of the Uruguay Round vol. 31, 33 I.L.M. 81 (1994). The trade secrecy provision of the TRIPS Agreement is Article 39, 33 I.L.M. at 98.
[35] [назад] See, e.g., http://www.sethf.com.
[36] [назад] Universal City Studios v. Reimerdes, 111 F.Supp. 294 (S.D.N.Y. 2000), aff'd 273 F.3d 429 (2d Cir. 2001).
[37] [назад] The DMCA covers two different kinds of technologies based on what they protect: technologies that «effectively control access to [copyrighted] works», Sec. 1201(a)(2), and technologies that «effectively protect[] a right of a copyright owner . . . in a work or a portion thereof», Sec. 1201(b)(1).
[38] [назад] While this seems odd, consider that many academic papers in security include computer program code.
[39] [назад] E.g., Technicon Data Sys. Corp. v. Curtis 1000, Inc., 224 U.S.P.Q. (BNA) 286 (Del. Ch. 1984); see also DSC Communications Corp. v. Pulse Communications, Inc., 170 F.3d 1354 (Fed. Cir. 1999).
[40] [назад] 847 F.2d 255 (5th Cir. 1988).
[41] [назад] 302 F.3d 1334 (Fed. Cir. 2002).
[42] [назад] The court relied on an earlier case, ProCD, Inc. v. Zeidenberg, 86 F.3d 1447, 1454. (7th Cir. 1996).
[43] [назад] EFF has submitted an amicus brief supporting Baystate's petition for rehearing en banc in the case. [add cite]
[44] [назад] Economic Espionage Act of 1996, Pub. L. No. 104-294, 110 Stat. 3488 (codified at 18 U.S.C. § § 1831-1839 (Supp. V 1999)).
[45] [назад] See generally Lydia Loren, Digitization, Commodification, Criminalization: The Evolution of Criminal Copyright Infringement and the Importance of the Willfulness Requirement, 77 Wash. U. L.Q. 835, 840 (1999).
[46] [назад] See Professor Larry Lessig, «Jail Time in the Digital Age», N.Y. Times (July 30, 2001); Declan McCullagh, «Hacker Arrest Stirs Protest», Wired News (July 19, 2001); Jennifer 8 Lee, «U.S. Arrests Russian Cryptographer as Copyright Violator», N.Y. Times, July 18, 2001.
[47] [назад] See Brad King & Michelle Delio, «Sklyarov, Boss Plead Not Guilty», Wired News (Aug. 30, 2001).
[48] [назад] See generally Electronic Frontier Foundation, Unintended Consequences: Four Years under the DMCA (2003) [cite to EFF website]

Глава 13. Вперед!

State of the art во взломе Xbox постоянно продвигается. Тысячи хакеров постоянно исследуют, innovating, discovering и sharing new methods and techniques, чтобы сделать Xbox более useful and valuable piece of hardware для ее end users. Keeping abreast of latest developments in hacking can be overwhelming. Надеюсь, чтение этой книги дало вам способности понимать latest posts and news на различных websites и web fora, dedicated to Xbox hacking. Эта глава обсуждает, куда можно пойти, чтобы узнать больше о latest hacks, где ask for help, и как вы можете contribute your unique abilities and perspective to the community. Эта глава также обсуждает некоторые larger challenges, которые будут стоять перед hackers in the future, namely the trusted PC initiatives.

Хакерское сообщество

Xbox hackers - anarchistic community, которая работает mostly underground, поддерживая связь и sharing information through various Internet «fora» (fora is the plural of forum, as data is the plural of datum). Большая часть Xbox hacking community keeps a low profile, и hackers often use pseudonyms to protect their identities. Причины использования pseudonyms vary, но generally anonymity carries the benefit of greater operating freedom. Hackers are more inclined to share their results and findings, if they know they can back away unscathed in case things get ugly. Использование pseudonyms также levels the playing field. Hackers judge each other primarily on the basis of the quality and frequency of their contributions, and little else. То, что вы можете быть young, не умаляет вашего first impression или street credibility, как это могло бы быть в других situations. Likewise, many hackers have no qualms about being blunt when you've made an error, и у них even less patience for stupidity presented as erudition or rude assertions. С другой стороны, many hackers are more than happy to extend a hand to those who have made an honest effort to read the FAQs, search the web and generally try their best to check and make sure that the question hasn't already been answered.

Хакерские форумы

Xbox hacking community имеет many civic fora для sharing their results and airing their concerns. Most popular fora are web-based BBSes such as www.XboxHacker.net and www.xbox-scene.com, and IRC channels such as #xboxhacker. Web-based BBSes typically feature news logs, FAQs, and useful links to information. More importantly, BBSes include fora where people can share information and post questions. Through these fora, you can tap the collective knowledge of all the hackers that frequent these BBSes. Logged history of these fora also contains a wealth of Xbox hacking information (and misinformation). Я encourage readers who have unanswered questions from this book to check out these fora for answers.

One of the first Xbox hacking fora to be created was the XboxHacker BBS (www.xboxhacker.net). Many of the best and brightest Xbox hackers have contributed to its fora. Например, one of the forum threads documents, in real-time, the adventures of Andy Green (known as numbnut on the XboxHacker BBS) as he hacked the version 1.1 security scheme of the Xbox. Я многому learned from reading the forum postings of the XboxHacker BBS. Я также met some of the most interesting people through the BBS fora. Founder of the XboxHacker BBS, Dan Johnson (also known as SiliconIce), tells his story in the sidebar «Profile: Dan Johnson».

Another resource for finding more information about the Xbox in general is the web search engine Google (www.google.com). As more hackers become involved with the Xbox, Google is becoming an increasingly

important tool for casting a wide net and discovering the latest tools and techniques. For example, at the time of writing Google started indexing a number of resources for replacement Xbox components. This can be useful for those who do not want to go through the trouble of adapting an ATX power supply to work with the Xbox.

Try to use keywords that are as specific as possible when searching with Google. For example, typing Xbox hacking into Google will return a large number of links related to the general topic of Xbox hacking, but few specifics. For example, today's top hit on Google for «Xbox hacking» is a LWN.net article titled «LWN: Lindows CEO funds Xbox hacking contest (News.com).» This seems fairly removed from information about how to install a new hard drive or the details about the Xbox security system. When narrowing down your search, try to figure out what the de facto jargon and spelling is for your concept. Suppose you are looking for information on the new Xbox security system. If you search on new Xbox, you hardly get any technical information. However, if you search on xbox v1.1 the search returns many more useful technical results. One of the best ways to harvest the current jargon and acronyms is by browsing the hacking BBSes.

Вклад в сообщество

If you are looking for a way to contribute to the Xbox hacking community, keep in mind that most hackers have a unique skill or strength that typically corresponds to his or her area of greatest interest, and that most hackers hack for fun. Например, I really enjoy hardware, especially when it requires building something. Also, while I can write code, I don't particularly enjoy it. Thus, my contribution to the hacking community is primarily through hardware projects. Likewise, I have a hard time motivating myself to engage in software projects. So, most of the time I just sit back and enjoy watching what other people are doing on the Xbox. It is both educational and entertaining.

When trying to think about what you can do for the Xbox hacking community, don't worry yourself about trying to jump in quickly and rush into a project that you aren't in love with, or that you aren't comfortable executing. You will know when your time has arrived: the project will just shout your name, and you will naturally be compelled to hack.

Профиль: Dan Johnson (a.k.a. SiliconIce)

Can you tell us a little bit about yourself, and how you got into hacking?

For some time, I had been interested in electronics hacking, though for the most part my interest had been limited to merely reading about such feats. Hacked devices such as Sega's Dreamcast and the Netpliance I-Opener had caught my attention from time to time. However, my first real experience with electronics hacking came with the «ePods», a discontinued internet appliance/web-tablet device.

I read about these interesting devices online and found my way to Ken Segler's I-Appliance BBS (http://www.linux-hacker.net), home of the famous I-Opener hacks. After reading about the neat things people were doing with these tablets, I was intrigued and spent a good deal of my saved up money at the time ($200) on one of the units. After receiving my new toy, I spent a lot of time performing many of the documented hacks, mostly software based. This was to be my first glimpse into the world of electronics hacking.

After the ePods, I moved on to another internet appliance, the Gateway Connected Touchpad. A slick-looking 10" touchscreen with a 400MHz Crusoe and 96MB RAM housed behind it; it looked like a great deal and perfect for a fun project. The device ran a custom build of Linux for AOL off of a 32MB

CompactFlash card. I swapped this out with a Microdrive and after much swapping of this disk between laptops and USB readers and the Gateway, the device booted a pared down version of Windows using «98Lite». The device seemed perfect for use as a finger-operated, network-attached mp3 jukebox (among other things), so I began working on a custom player that would allow easy operation via the touchpad. The whole Gateway experience made for a neat summer project between my Junior and Senior years of high school.

How did the XboxHacker BBS come about?

It was during this time period that I got the idea for XboxHacker.Net. To me, the Xbox seemed to have the potential to be the ideal computing device for under the TV . . . as soon as it could be programmed. The hardware was most impressive for the time, and more than adequate for what I hoped would be accomplished on the device. Unlike other consoles, the Xbox also was to have a hard drive and ethernet port built in. Once hacked, the Xbox could be used to emulate old consoles such as NES, SNES, N64, and even PSX, to conveniently play back any type of media file, such as mp3s or DivXs, on your home entertainment system, to play streamed media from a home network, or even to double as a basic PC.

Some argued that the cost of the device at $299 made it out of range to bother hacking, as a PC with similar specs could be constructed without the need for hacking for not much more in cost. However, the Xbox had the advantage that it could also play Xbox games and was already designed to sit with your TV. The hacks would just be added value. After my short foray into the world of electronics hacking, I thought that hacking the Xbox could be an interesting project to help organize as I was intrigued by the possibility of having a convenient box for my TV to perform the tasks mentioned above. It was not until many months later that I would actually acquire the XboxHacker.Net domains.

How was your experience growing and running the XboxHacker BBS?

From the beginning, XboxHacker.Net focused on a few primary goals: providing and spreading technical information about the Xbox, and providing a place for fellow hackers to discuss technical information related to hacking the Xbox. Though due to my limited technical knowledge and experience I could do little in the way of actual hacking to contribute to the effort, I knew plenty enough to help facilitate the effort by collecting and distributing relevant information and moderating a discussion board.

Not long after the site launched, we were fortunate enough to receive some links from such high-profile websites as Mike Magee's The Inquirer and Van Smith's Van's Hardware. The XboxHacker.Net BBS quickly became one of the primary places on the Internet to discuss any material related to hacking the Xbox and the XboxHacker.Net news page had the most up-to-date news on the status of the Xbox. A few weeks after the site came online, it received mention in an article on CNET, and from then on the activity level steadily increased. It wasn't long before XboxHacker.Net had outgrown the small shared server we were on, so the site moved to a much larger account which was also outgrown in a matter of weeks. Traffic to the forums continued to increase rapidly, so I made the decision to move the site to its own dedicated server where we could have room to grow and not worry about every few MBs of bandwidth usage or how many concurrent forum users were online.

It didn't take long before the activities of XboxHacker.Net captured the attention of Microsoft. Early in the site's history, I was contacted a couple of times with requests to remove materials. The first instance was of a screenshot of a developer tool I had received without much explanation that showed «security sectors» on the disks. The second instance was a request to remove a link someone had posted in the forums to a BIOS image of the Xbox. Aside from these few minor incidents, contact between Microsoft and XboxHacker.Net was virtually non-existent. It was made a policy early on to steer clear of issues of questionable legality such as discussion about the backing up of games, linking to copyrighted materials, or posting Microsoft code from the BIOS.

In the beginning, though interest was high, progress seemed to be relatively slow . . . there was not a lot that could be done with the Xbox before the security was broken. There were many individuals who

contributed notably early on. Some of the earliest contributors to the XboxHacking world were Andy and Luke, whose Web-pages contained a plethora of information on file formats and other valuable tidbits of Xbox information. Steve «SurferDude» Gehlbach contributed to the design of a VGA converter circuit for the Xbox and Ken Gasper improved on this to create a true VGA adapter for the Xbox. Bunnie's early analysis and hardware info page generated interest in XboxHacking as well after its appearance on Slashdot. Indeed, there are numerous others who made their mark as well. Privileged to count such skilled hackers among the contributing members on the boards, XboxHacker.Net grew into a hub for technical Xbox information, discussion, and news.

The crown jewel of XboxHacker.Net has no doubt always been the XboxHacker BBS. The site was centered around the forums and activities of our members. Often, the latest XboxHacking news would simply be links to forum topics and clips from posts made by our members. The purpose of the forums was to enable the diverse, multi-national group of people interested in hacking the Xbox to work together towards this common goal. The forums provided a great place to share information and discuss ideas with fellow hackers and also served to bring together many talented hackers who may otherwise have had no means of collaborating. It was a great feeling to watch the progress that the forums enabled. One particular discussion that comes to mind took place over the course of several days. A number of members were discussing the new security system in the second generation Xbox, looking for flaws and ways around it. It was exciting to watch the events unfold as hackers got closer to the solution and eventually broke the security on the revised Xbox. Our hundreds of contributing forum members were in large part responsible for the success of XboxHacker.Net and the progress of the XboxHacking world in general.

Besides the activity on the XboxHacker.Net BBS, there were no doubt many other groups working to hack the Xbox in secret. One such group I knew of on IRC contained contributing forum members as well as others from the electronics hacking underground. Names here will go unmentioned. Many bits of information from independent groups or individuals were relayed to me for reporting over time. However, despite the growing interest in XboxHacking and the number of people involved in the effort, it would be some time before any major breakthroughs became public knowledge.

The XboxHacking scene picked up pace fairly rapidly after the public availability of modchips near summer 2002. By this time, the Xbox security system had been cracked by multiple groups and it was only a matter of time before modchips were readily available to the public. At this point, the work shifted away from hardware hacking for the most part and more towards software development. A sister site to XboxHacker.Net, XboxDeveloper (http://www.xboxdeveloper.net) was started to help catalog the software that became available for the Xbox, though it never took off to the level that XboxHacker.Net did. Using leaked copies of the Microsoft Xbox SDK, programmers began writing and porting various applications for the Xbox, including media players and emulators such as MAME. The Xbox-Linux project (http://xbox-linux.sourceforge.net/) under the leadership of Michael Steil, successfully allowed the Linux operating system to run on the Xbox. I pushed to start the OpenXDK project (http://sourceforge.net/projects/openxdk/), an open source developer kit that would allow development of software for the Xbox unhindered by legal issues, though that project has met with only mixed success. It is now under the leadership of «Caustik» (Aaron Robinson), a CS student at Case Western University. Due to the more accessible nature of software development compared with hardware hacking, the number of contributing members to the general XboxHacking effort increased rapidly. The amount of «homebrew» software available for the Xbox today is amazing and includes everything from media players, console emulators, and utilities to originally written games.

Доверенные вычисления

Trust is the cornerstone of security, and in order to have faith in your security system, you need to have faith in the hardware and software you are running. The trusted computer is a machine that has been architected to be resistant to attacks that could compromise the trustability of the machine. There are many approaches to building a trustable computer, from the Automated Teller Machine model of physical security and tamper-resistance, to less hardware intensive solutions such as those used in the Xbox. The Xbox fits into the broader picture of trusted computing since it is one of the first high-profile, widely-deployed trusted PC implementations. In a way, the Xbox gives us a hint of what we might expect down the road for trusted computing.

Trusted computing is a potentially disruptive emerging technology. The rise of trusted clients in an ad-hoc network like the Internet harbors the promise of enabling safe and private on-line financial transactions, of reducing or eliminating the occurrence of computer viruses, and of reducing or eliminating spam email. Trusted PCs can also be used to securely store sensitive data, such as your medical and financial records and your naughty or embarrassing secrets. Another application of trusted PCs is to reliably enforce digital content access rights and management policies upon users. The digital rights management (DRM) aspect of trusted computing could fundamentally change the way we use computers today; many of us enjoy the benefits of pseudo-free content and flexible copyright implementations.

The fundamental problem is not the existence of content management policies - indeed, content management policy can be beneficial for consumers. The real problem is when the policies which govern your rights can be set unilaterally by content providers. In the current trusted PC proposals, the user is not trusted to have control over certain key secrets inside their machines. Instead, the attestation information (information required to establish trustability) of a user's machine is partially maintained by a third party. Can we rely on a non-elected, unregulated third party with business interests to determine who can do business, send email, and otherwise be recognized as a trustable entity?

Trusted computing is like a gun. It's great to have one as long as you're the one controlling the trigger. Unfortunately, many trusted PC opponents fear that in practice, systems will be deployed with preset rights, policies and third-party trust affirmation resources pointed in the wrong direction for consumers. A company, during good times, may set the privacy and security policy in favor of users to attract a larger customer base, but once Chapter 11 comes knocking, or the company is sold or otherwise changes hands, these policies can and will shift. What is to prevent businessmen from promoting trusted computing initially with user-friendly policies, and then suddenly shifting into a wallet-squeezing DRM mode? Ethics?

Шаг назад

There is a problem with the phrase «trusted computing»: it has become synonymous with cryptographically secured trusted computers. Let's take a step back and just talk about alternative approaches to building trusted computers.

Trustability has always been important in computers. However, back in the early days of computing, machines were so expensive that the hardware necessary to enforce strong trust policies was not within the reach of consumers. For example, many early machines shipped with a socket for a hardware Memory Management Unit (MMU) chip. The MMU was one of the first steps toward trustable hardware memory models; part of an MMU's job is to enforce page-level memory access protections. MMUs were sold as an option because they were quite pricey at the time. Unfortunately, the move toward trustable hardware stopped at the MMU, partially because computer networks didn't exist in any major form until relatively recently. In a non-networked world, data needed to be protected only from programmer errors and from access by a few select users with physical access to the machine. Today, computers need something

stronger than just an MMU, something that can provide trust in the face of viruses and remote attackers attempting to exploit subtle software weaknesses to run malicious code.

The natural extension to the MMU's hardware-enforced paged virtual memory model might be address capabilities with a tagged memory model. A memory tag is a set of bits that record the type of data or code stored in a memory location. In a tagged memory model, every memory location has a set of tag bits, kind of like how every memory location in a conventional error-correcting memory implementation is associated with some ECC bits. Tag bits help the hardware enforce data type management policies; for example, a memory location tagged as piece of data can never be accidentally or intentionally executed as code. A capability is a pointer granted by a trusted kernel that cannot be forged. The unforgeability property is preferably enforced by hardware through tag bits. Many architectures also include the ability to enforce access boundaries as part of a hardware capability. [1]

Capabilities and memory tags are not new ideas; in 1961, the Burroughs B5000 used capabilities (then called descriptors) and tagged memory to guard, in hardware, against buffer overflow attacks, and to isolate code from data. [2] The MIT PDP-1, Intel i432, IBM System/38, and the Mach and Amoeba operating systems also implemented capabilities in some form, and this is by no means a complete list of systems that have used capabilities or tagged memory. The security properties of capabilities have also been demonstrated in many academic studies, such as the EROS (Extremely Reliable Operating System). [3] Unfortunately, capabilities and tagged memory never made their way into the heart of mainstream PC architecture. Security and reliability has always taken a back seat to cost, backward-compatibility, and performance.

This brief history lesson demonstrates that trusted computing does not require the cryptographic approach that is being proposed today by Palladium and the Trusted Computing Platform Alliance (TCPA). In fact, cryptography on its own does not provide any security. Secure key management is really what provides all the security in Palladium/TCPA. Cryptographic algorithms simply transfer the security of the key into the user's domain.

This being said, one can draw an analogy between a capability and a cryptographic key. Both require a trusted OS to manage their creation, dissemination and destruction. Both are equally weak if the system cannot protect against forged keys or capabilities. The big difference is that if a secret key leaks, all security is lost, eternally. On the other hand, capabilities are created and destroyed dynamically, so the leakage of a capability might lead to a security breach, but the scope and duration of the breach is limited. To this extent, capabilities provide a more robust solution for computer security.

Note that relying solely on cryptographic techniques for hardware security still leaves machines open to classic buffer-overrun style attacks and security holes due to programming errors. «Measurements» of the software's state help mitigate this weakness by detecting code alterations before executing security-critical operations, but measurements are not a perfect solution. On the other hand, buffer-overrun attacks are impossible in systems using hardware enforced capabilities with bounds checking.

Memory tags can also be used to implement security features that are not feasible using a purely cryptographic approach to trusted computing. One example is the trustable concurrent processing of compartmentalized secrets. [4] In this example, multiple threads with varying levels of security clearance are operating on a single processor. The hardware enforces a policy where all threads impress their security level upon the data that they access. In other words, every computation simultaneously operates in two domains: the conventional arithmetic domain, and the security domain.

Suppose an unclassified thread adds two unclassified numbers and creates a piece of data named foo. foo's security tag is also computed in parallel with the add arithmetic operation. In this case, the security tag's result is «unclassified». Now, suppose a top-secret thread touches foo: foo's security tag now changes to «top secret». Unclassified threads can no longer read foo, even if the unclassified thread has a valid pointer to foo; foo must be explicitly reclassified before it can be read again by unclassified threads.

Such a strictly compartmentalized security system can be used, for example, to ensure that no internal kernel structures are ever accessible to user processes, even in the presence of bugs and memory leaks (including scenarios where kernel memory is deallocated and reassigned to a user process without a memory clearing step). This scheme can also be used to establish security audit trails which are useful for helping programmers to trace the root cause of a security breach, and as damage control measures in case of a security breach.

To be fair, one advantage of the cryptographic approach to trusted computing is that if a rogue does get hold of data through some hardware means - by eavesdropping on the hardware or stealing a hard drive - the data cannot be deciphered. However, users can elect to use cryptography for data protection in any computer implementation, including those using hardware capabilities and tagged memory. The problem of secure key management is still a difficult problem, but perhaps it can be solved in part by integrating cryptographic smart card readers into PCs.

There is no essential reason why a trusted PC implementation must use the cryptographic techniques proposed in Palladium and TCPA. The techniques reviewed in this section, namely capabilities and tagged memory, can be used to implement a secure, trustable PC in a manner that does not involve the risk of users losing the ability to set their own access policies. Users would be in full control of their machine and their secrets at all times.

Palladium против TCPA

There is much confusion these days about the current trusted PC proposals, namely Microsoft's Palladium and the Trusted Computing Platform Alliance (TCPA). There are enough similarities between the two proposals that many people think they are one and the same, but the goals of each initiative are different.

The TCPA is a multi-corporation alliance to create computers with some nominal amount of trust. Significantly, much of its specifications can also be applied to non-PC platforms. In TCPA the trust is planted in a secure hardware module called the TPM (Trusted Platform Module). The TPM contains features to ensure that the secrets contained within the module are never leaked through software attacks. It also contains features, such as secured system «measurements», that attempt to transfer the trust contained within the TPM to the host machine.

Palladium, on the other hand, is a whole-system PC-centric security concept created by Microsoft alone. One of its components is a TPM-like security module, but Palladium also calls for sweeping changes to the hardware chipset and the way I/O ports are implemented. The chipset is required to enforce memory security policies from all potential DMA sources, such as the graphics card. Palladium also calls for encryption of the I/O to the keyboard and video subsystems.

Palladium's requirement for cooperation between chipset vendors, OEMs and Microsoft is a potentially large flaw. There isn't enough margin in the commodity PC hardware industry today to support the overhead of an extensive cryptographic security overhaul. Too, many chipset vendors do not have any experience with implementing secure systems. On top of the language barrier faced by many overseas chipset vendors, chipsets are usually developed on a short fuse and with a keen eye on the pocketbook. Can chipsets developed under these conditions be expected to protect sensitive secrets?

The Xbox is an example of what can go wrong when security policies are defined by one body and implemented by another very different organization. Microsoft wrote a specification for a trustable piece of hardware, namely, the Xbox. Strong cryptographic algorithms are used liberally in the Xbox, and the master key for the system is locked deep inside a complex piece of silicon. However, experience has demonstrated that the Xbox's security system can be bypassed using a combination of an unsecured debug port, a flaw in the hardware initialization scheme, and a bug in the handling of a boundary case of

the instruction pointer in the CPU. These three minor oversights, committed by three independent parties (the assembly contractor, the Xbox firmware designer, and Intel), conspire to provide a convenient method for defeating Xbox security.

Each of these oversights on their own does not represent a significant security problem. This leads to the disconcerting question of how many security breaches in a particular Palladium implementation will be caused by the stacking of multiple benign flaws. Every complex consumer electronics system has minor bugs or design oversights, especially when systems are composed of components built by multiple independent entities whose primary interest is in turning a profit.

In the consumer electronics industry, one can either ship a perfect product, or one can make money. Products that don't make money are quickly cancelled. Thus, it is very rare to find a consumer product that is technically perfect in all respects. As a result, the only practical way to guarantee the security of consumer electronics system as complex as Palladium is to throw it into the wild, and let the hackers have their way with it for many years until all of the big security holes have been discovered and plugged.

On the other hand, the TCPA's TPM is a device created to solve a certain set of problems that is smaller in scope than Palladium's. Thus, the TPM is not as exciting from a market perspective, but it may be more practical and serviceable for its intended purpose. Both the TPM and Palladium are weak to hardware attacks, but the TPM doesn't attempt to extend security requirements as far into third-party system design territory. The TPM is primarily a secured key management module that can detect most modifications and intrusions to the host system. The software layers built on top of this substrate do the rest of the heavy lifting, caveat emptor.

Хакинг доверенного PC

The current proposals for the trusted PC are weak against some fairly simple hardware attacks, even in the absence of any integration oversight or bug-related back doors.

The first attack is one that I call the «Surreptitious BIOS», or SPIOS (pronounced «Spy OS») attack. SPIOS can be used to defeat DRM policies that rely on the cryptographically sealed storage feature of the trusted PC to prevent unauthorized user access to data. The basic idea is to boot the PC with an unmodified BIOS into trusted mode and extract all the desired data into system RAM, then to perform a warm reset of the system while swapping the BIOS image.

The modified BIOS image can be used to read out the desired data from system RAM. The desired data may be a session key stored in memory, or the actual decrypted data itself, depending upon how the program structures and caches its data in memory. Since the current trusted PC specifications call for an LPC bus based BIOS, inexpensive alternate firmware devices (similar to those used on the Xbox) can be used to execute this attack. There are techniques that application programmers can use to complicate this attack, such as only decrypting a single block of data each time into system memory, but many of these techniques severely degrade system performance. The degradation of system performance may be especially pronounced if file caching and prefetching is disabled.

Another attack is one that I call the «Surreptitious RAM», or SPAM attack. The goal of this attack is to spoof the trusted routines responsible for measuring the fitness of the system state. A device, such as an FPGA or ASIC, is installed on the plug-in memory cards in between the DRAM chips and the memory connector. This device monitors the pattern of addresses going by, or it may have an extra connector that sniffs the state of the wires connected to the I/O pins of the cryptomodule responsible for authenticating the system.

Either way, when a system measurement is in progress, the SPAM device presents a memory image that is consistent with an unmodified, trusted system state. However, during all other operating modes, the SPAM device presents a memory image that is modified to do whatever the user pleases. This modification can be very subtle: just a couple of bits flipped at the right locations is all it takes to modify key branch instructions in the security kernel.

This device is more powerful than the SPIOS since it works on a system that is powered-up and supposedly trustworthy. It can be applied to effectively defeat a wider range of DRM schemes as well as some authenticated transactions between the local machine and the server. SPAM alone cannot be used, however, to falsely identify a system as another registered, trusted system, since SPAM lacks the secret shared between the tamper-resistant secure cryptomodule in the local machine and the authentication server. False system identification would require either extracting the key from a tamper-resistant secure cryptomodule (possible, but not trivial and most likely destructive to the module), or somehow tricking a secure cryptomodule from another registered, trusted machine into providing the falsified identity.

The SPAM device can be manufactured for relatively little (high performance FPGAs can cost as little as $50 today in single quantities), and can be very easy to install. The SPAM can be either integrated directly into a memory module (in which case it functions as both a trust violation device and as a memory expansion device), or it can be provided as a device that is installed in a stacked configuration in between the motherboard's memory slot and the existing memory device. In some memory card configurations, particularly ones that employ heat shields, it may be possible to hide the SPAM device and pass the module off as a regular memory expansion device. While elaborate, this may be a worthwhile attack against a large corporation or bank that stores high-value secrets on a trusted PC-based server.

Взгляд вперед

When considering the prospect of trusted computing, we need to first consider whether the currently proposed schemes will offer all the benefits that they promise, and then weigh those against the potential harm to consumers' rights and the potential benefits to criminals (enhanced privacy can be used for both good and ill). If trusted computing could provide perfect security for online businesses, then that might be worth the potential risks. However, the scenarios outlined in this chapter indicate that the trusted PC's security may be less than perfect.

Consider the Xbox. The Xbox is a trusted PC implementation that can be hacked with just a $50 solderless module. This places a fairly strong bound on the value of secrets that can be trusted to an Xbox. Hardware modchips are so inexpensive that they pay for themselves with the cost of a copied game title, or two games if you elect to pay someone to install the chip for you.

Of course, there are always the moral and social implications of stealing content too, as well as new legislation, such as the DMCA, which aims in part to make such acts a crime. Unfortunately, the current trusted PC proposals on the table are also weak in the face of similarly inexpensive hardware attacks. Thus, it is unlikely that they will provide the level of security required for high-value or very embarrassing secrets.

The fact of the matter is that hacking technology will be developed whether or not it is illegal, and whether or not the intention is good or evil. Thus, it is in the best interests of consumers and companies to educate the population about hacking, and for everyone to understand the limitations of their «trusted PC». The worst-case scenario would be if billions of dollars were invested in trusted computing, only produce a net result of no greater safety or privacy for consumers, while severely curtailing consumers' rights with poor content policy implementations.

The good news for trusted PC proponents is that shrinking feature sizes in integrated circuits is driving greater integration throughout the PC. Within a decade, today's PC will fit on a single piece of silicon. Once the RAM and the BIOS are fully integrated into a single piece of silicon, hacking a system becomes much more difficult - but not impossible. Focused Ion Beam (FIB) machines, a tool used by chip designers and failure analysis labs, can cut and jumper nano-scale features. Another upside for trusted machine designers is that public key processors could become so small and cheap to integrate into a chip that individual chips, especially memory chips, could start using strong crypto to authenticate and encrypt their I/O.

Another technology that could aid the implementation of trusted PCs is integrated tamper-resistant or tamper-detecting features. For example, a time-domain reflectometer (TDR) could be built into a chip's I/O cells. A TDR can detect the presence of an eavesdropper on a wire by recognizing certain changes to the wire's electrical properties. In addition to the ability to detect eavesdroppers, integrated TDR devices are desirable for high performance I/O since they can be used to calibrate the drive impedance and equalization/pre-emphasis filters required for multi-gigabit speed communications.

Заключительные мысли

This book has taken you through a whistle-stop tour of Xbox hardware hacking, from the basics of soldering and disassembling to the latest projects and techniques. It has also introduced you to the social aspects of Xbox hacking: the people who hack, and the interplay between society, law, and hacking.

While the details of how to install a blue LED in an Xbox may be irrelevant in a few years, the skills you learn executing the installation will last a lifetime. Moreover, the social and legal issues confronting hackers and consumers will extend beyond the Xbox and into every part of our emerging information-centric way of life.

The material in this book is just a starting point; there is a world of hardware out there waiting to be explored. I hope this book has provided the novice readers with a strong starting point for becoming an explorer, fixer, and innovator in a world increasingly filled with, controlled by, and dependent upon electronics.

Happy hacking!

• bunnie@xenatera.com

Сноски

[1] [назад] An efficient, high performance hardware implementation of precise object boundaries using tagged capabilities can be found in a tech note titled «A capability representation with embedded address and nearly-exact object bounds» by Jeremy Brown, J.P. Grossman, Andrew Huang and Tom Knight. http://www.ai.mit.edu/projects/aries/Documents/Memos/ARIES-05.pdf
[2] [назад] «The Architecture of the Burroughs B5000 - 20 Years Later and Still Ahead of the Times?» by Alastair J.W. Mayer. http://www.ajwm.net/amayer/papers/B5000.html
[3] [назад] Originally conceived at the University of Pennsylvania by Jonathan Shapiro (http://www.eros-os.org/)
[4] [назад] More about security systems like this can be found in a tech note titled «A Minimal Trusted Computing Base for Dynamically Ensuring Secure Information Flow» by Jeremy Brown and Tom Knight. http://www.ai.mit.edu/projects/aries/Documents/Memos/ARIES-15.pdf

Приложение A. Где взять хакерское снаряжение

Существенный психологический барьер для начала аппаратного хакинга - воспринимаемая недоступность хакерских инструментов. Хотя Radio Shack держит небольшой запас components and tools, components дороги, а большинство tools трудно использовать с сегодняшними миниатюризированными components. Это приложение перечисляет нескольких vendors components and tools, у которых solid selection по reasonable prices. Это приложение также включает order numbers Jameco для basic tools, required to do the projects in this book. Вы можете ввести эти order numbers directly into Jameco's secure ordering website, чтобы помочь quickly get started.

Поставщики для любителей

Существует wide variety of component and equipment distributors, от тех, кто caters primarily to large businesses, до тех, кто обслуживает individuals, hobbyists, and repairmen. Вы можете испытать некоторую frustration при dealing with distributor, который caters to large businesses. Эти distributors service high-volume accounts, поэтому accounts обслуживаются designated sales representative. В результате может быть difficult получить comprehensive catalog of parts или pricing and inventory information for small orders. Vendors, listed in this section, friendly to individual orders, and their collective inventory will stock most of the parts you will ever need.

Таблица A-1. Таблица vendors of components and equipment.

Vendor Name	Vendor Contact Info	Specialties and Notes
Digi-Key	www.digikey.com; 1-800-344-4539 (phone); 1-218-681-3380 (fax)	Wide selection of original manufactured components. Hobbyist-friendly, prompt service. $5 handling charge on all orders under $25. Excellent website features real-time inventory and pricing check information, data sheets and parametric search tools.
Jameco	www.jameco.com; 1-800-831-4242 (phone); 1-800-237-6948 (fax)	Economical hobbyist-friendly vendor. Components are a mixture of original and surplus stock. Good selection of tools at a reasonable price. $5 handling fee on all orders under $20. A wide selection of do-it-yourself kits available.
MCM Electronics	www.mcmelectronics.com; 1-800-543-4330 (phone); 1-800-765-6960 (fax)	Specialty consumer electronics service/repair tools and parts as well as common tools and components. Sells hard-to-find security bits and replacement parts. $2.95 flat handling charge, no minimum order requirement for internet purchases.
Mouser	www.mouser.com; 1-800-346-6873 (phone); 1-817-804-3899 (fax); orders@mouser.com	Tools and component vendor that stocks a good selection of original components. Hobbyist-friendly, no minimum order, no handling charge.
Newark	www.newark.com; 1-800-463-9275 (phone)	Larger, traditional component distributor. Broad component selection but not all components are stocked; you may have to wait some weeks for certain orders to get filled. A good place to go for components you cannot find at other hobbyist-friendly distributors. $5 handling fee on all orders under $25.

Vendor Name	Vendor Contact Info	Specialties and Notes
Fry's Electronics	www.frys.com	Retail electronics store that stocks a reasonable selection of components and tools. A great place to go and browse if you do not like catalog shopping. Only found in California, Texas, Oregon and Arizona.
McMaster Carr	www.mcmastercarr.com	Vendor of mechanical hardware, metal stock and machine tools. A place to get raw sheet metal and fasteners to finish off a project. Website is comprehensive and helpful.
FindChips	www.findchips.com	Automated parts search engine. Search dozens of distributors for parts. Most hobbyist-friendly vendors are listed in this search engine.

Готовые формы заказа оборудования

Этот раздел содержит selected list of tools, sold by Jameco, которые можно заказать, чтобы gear up for hacking. «Basic» order form содержит all the bits and drivers necessary for opening up an Xbox, as well as some basic soldering tools. «Advanced» order form содержит set of tools and utilities, которые useful to have on hand, but are expensive enough that those on a budget may consider ordering them only as they require them.

In addition to the tools in these order forms, you will need a few small parts for the projects described in Part I. Please refer to the introduction of each respective chapter for a list of parts that you will require.

Базовая форма заказа

Таблица ниже перечисляет basic tools, которые понадобятся для projects, described in this book. You can order these parts by going to Jameco's website, www.jameco.com, and clicking on the yellow «Quick Order» button on the left hand side. Enter in the Jameco part numbers listed here and get started hacking! (Prices may vary.)

26-piece SAE Hobby Tool Set includes all of the bits, drivers, and screwdrivers necessary to disassemble an Xbox, including the T-10, T-15, and T-20 torx bits. Я включил 1/32" conical bevel tip в order form above, because the soldering iron comes with a horrendously large 1/8" chisel tip. A 1/8" tip is useful for soldering together small countries and creating solder shorts all over your fine-pitched components. Conical bevel and chisel tips feature a flattened region that enables rapid heat transfer, and thus are easier to use than conical sharp tips. Я также включил some soldering iron tip cleaner and some rosin paste flux. Они немного pricey, но они will make your life a lot easier, trust me.

Description	Jameco Part Number	Price
16 to 30 Watt Variable Soldering Iron	116572	$18.95
1/32" Conical Bevel tip for soldering iron	35326	$4.49
Small pack of 60/40 Kester flux core solder	73576	$2.59
2 oz Kester rosin paste flux	73584	$1.59
Desoldering Wick	175986	$2.49

Description	Jameco Part Number	Price
Soldering iron tip cleaner and conditioner	132986	$4.95
26-Piece SAE Hobby Tool Set	170069	$14.95
Wire stripper, 22-30 gauge	127870	$4.95
Total price		$54.96

Расширенная форма заказа

Таблица ниже перечисляет несколько tools that are nice to have around, but pricey enough that you probably should only get them if you anticipate doing a lot of hacking.

- Soldering iron stand and sponge are a must if you plan on doing even a moderate amount of soldering. Stand safely stows the hot soldering iron tip, reducing the chance of burns and fires.
- Try soldering with the 1/32" semi-chisel tip. You may find that you like it better than the conical bevel tip.
- Anti-stat wrist-strap is always a good idea to have on hand, especially if you tend to wear clothes and shoes that generate static electricity.
- SMT removal kit comes in handy when trying to remove smaller SMT components.
- Multimeter is a great all-purpose test, measurement, and diagnostic tool to have on hand at all times, and has applications all around the house or dorm. Many models of multimeters are available; the one listed here offers the most features for the lowest price. I have tried it out and it works well.
- Wire stripper and crimping tool listed here are very helpful for the power supply replacement project. The crimping tool is sold by Digi-Key, not Jameco. The crimping tool is a bit on the pricey side, but unlike sub ten dollar tools it features shaped crimp dies for higher-quality solderless crimps.

Description	Part Number	Price
Soldering iron stand	36329	$4.25
Sponge for soldering iron stand	134631	$2.99
Semi-chisel 1/32" tip	35078	$4.49
Adjustable anti-stat strap	159257	$7.95
SMT component removal kit	141305	$21.95
Multimeter (volts, ohms, amps, farads, temperature, frequency, diode/transistor)	177480	$49.95
Wire stripper, 16-26 Gauge	127861	$4.96
Crimping tool (order from Digi-Key)	WM9999-ND	$36.19

На этом список рекомендованного хакерского снаряжения заканчивается.

Приложение B. Техники пайки

Это приложение объяснит основы пайки, а также некоторые более продвинутые техники пайки surface-mount. Вы получите от этого приложения максимум, если будете экспериментировать с техниками пайки по мере чтения. Jameco Electronics (www.jameco.com) продает wide variety of project kits, требующих through-hole assembling, которые можно использовать для basic soldering practice. Companies such as TopLine (www.topline.tv) offer economy practice kits using component blanks, если вы хотите практиковать свои SMT assembly techniques, а MCM Electronics (www.mcmelectronics.com) предлагает practical Surface Mount Solder Practice Kit. (Я настоятельно рекомендую попрактиковать свои SMT soldering skills перед тем, как пытаться прикрепить SMT component, который вам не безразличен.)

Введение в пайку

Basic technique for soldering довольно проста: вклиньте tip вашего soldering iron в пространство между component lead и circuit board pad, чтобы нагреть обе части. Когда они достаточно горячие, постепенно подавайте немного solder wire в joint, пока не будет создан nice, smooth solder fillet. В реальности soldering требует немного practice and experience, прежде чем человек сможет паять типичную board с более чем thousand joints без плохих solder joints.

Чтобы был создан хороший solder joint, горячий жидкий solder должен «wet» subject pieces, чтобы connection была made. Поэтому настоящее искусство hand-soldering - понимание того, как гарантировать solder wetting.

Можно сказать, когда liquid solder wets piece of metal, просто глядя на него. Wetted joint выглядит так, как будто molten solder lost all surface tension; liquid solder shiny и smoothly flows over the work area. В opposite situation solder has a dull sheen, и он tends to ball up around the soldering iron tip вместо flowing outward.

Используйте флюс

Solder fails to wet the subject metal because oxygen in the air or dirt and grease have reacted with the metals. В этом случае можно apply a flux to your workpiece, чтобы break down these foreign compounds. Word flux comes from Latin fluxus, which means flowing. Most solder comes with a core of flux built in to enhance solderability. Если внимательно посмотреть на cut piece of solder, можно увидеть flux core, surrounded by solder alloy. Always use solder with a flux core, or you will be in a world of pain trying to get the solder to wet. Almost all solders made specifically for electronics have a flux core, but there are solders that you can accidentally purchase in a hardware store that have no flux, and that are made for purposes such as joining pipes. Когда вы нагреваете flux-core solder, поднимается small puff of vaporized flux smoke. Small fan placed near the work area will blow the fumes away and prevent inhalation.

Common novice mistake - иметь слишком много faith in flux-core solder. Frequently, flux contained in flux core solder is not enough to get the solder to wet. В этом случае нужно apply extra flux. Raw flux usually comes as liquid or paste, so application is easy. In liquid form, single drop of flux can be applied with half of a toothpick. Break the toothpick in half, leaving the break slightly jagged. Dip the broken end of the toothpick into the flux, and a small drop will cling to the end. Applying liquid flux to a large area can be done with a fine-tipped artist's brush, but be sure to clean the brush when you are done or it will end up

gummy and unusable in a couple of days. Flux dropper is also handy, but expensive. Flux dropper is a bottle with a thin capillary needle on top; when you invert the bottle, flux slowly drips out of the capillary. In paste form, flux can be applied by dipping any piece of scrap, such as a toothpick or a piece of solid wire, in the flux paste. Finally, flux pens are handy for beginners because they combine flux storage and dispensing in a convenient and inexpensive package. Flux pens don't have the accuracy or quality of other flux application techniques, but they are convenient and good for occasional use.

Many fluxes require clean-up after use. Fluxes can harden with time, making repairs difficult in the future, they can slowly attack the board, and they can absorb water and become conductive. Traditional soldering flux is resin flux. Resin fluxes require use of strong solvents that are flammable and toxic. As a result, I tend to recommend water soluble fluxes or no-clean fluxes. Water soluble fluxes can be removed by just washing down the board with water. It is better to use distilled deionized water, but I have found that most warm tap water is pretty effective as well. When I have to clean a large batch of boards, I throw them into a dishwasher (with the food trap cleaned and no detergent!). After the boards are washed, set them on a conductive, clean cookie sheet or aluminum foil, and put them into an oven on low heat (around 200°F) to bake for about an hour or two, or until all the water is driven away. Be sure not to turn the oven up too high, or water trapped inside the pores of components will turn into steam and cause them to crack or explode. Most parts are designed to be «process sealed», so washing them with water is fine. Be careful with connectors and switches, however; you may need to tape over them with a piece of Kapton tape to prevent contamination with water.

> Warning
>
> Do not use acid-type fluxes on circuit boards. They will attack the board and components and cause failures with time. Acid fluxes frequently plague novices who use solders and fluxes that are intended for pipe soldering.

Советы для начала

Components with many leads need to be aligned and tacked in place before soldering. If the component is of a surface-mount type, it can be held in place by soldering down two pins on opposite corners of the device. After tacking the component in place, re-inspect the alignment in case the component drifted during the tacking process. If the component is of a through-hole type, you will need masking tape to hold it in place while you turn the board over. Tack the corner leads of the component in place, and check that it is level to the board before soldering all of the leads. (Masking tape has a bit of play in it and frequently you will have to heat one of the tacked corners while pressing down on the component to level the component with the board.)

Kapton tape is a fairly handy thing to keep around the workbench. Made by DuPont, Kapton can withstand temperatures up to 500°F, well above the melting point of solder. It is handy for masking off nearby regions where you do not want solder. However, Kapton tape is expensive so use it only in situations where it will come in contact with hot solder.

Solder fails to wet metal that is too cold. This is a common problem when soldering large joints, or when soldering joints attached to large sheets of copper. In these cases, the connected metal conducts enough heat away so that the junction never reaches the melting point of solder. The solution to this is either to use a more powerful soldering iron (but be careful - the heat of very large soldering irons can also cause the board traces to fall off), or to leave your soldering iron in contact with the junction for a longer period of time before applying the solder. One trick to heating up the work area faster is to feed just a touch of solder into the tip of the iron where it is heating things up. Even though the solder will not wet on the board, the liquid solder on the iron's tip increases the effective contact area between the iron and the board, and heat

will be transferred more rapidly into the board. After applying solder to the iron's tip, you will sometimes have to rotate the iron, maintaining contact at the joint, in order to get the molten solder in contact with the board.

You can tell when a component lead or a board pad is sufficiently hot by carefully observing the way light reflects off of it. Component leads and pads on a circuit board typically have some kind of plating on them, usually made out of solder. This solder plating has a slightly dull sheen under normal conditions. When made sufficiently hot, however, the sheen goes from dull to almost perfectly reflective. To get a feel for what this looks like, try heating up a large square pad with your soldering iron, using the technique described above. You can usually watch the melting front propagate across the pad as the soldering iron heats the board.

> Warning
>
> If you are using a non-temperature controlled soldering iron, use the lowest wattage soldering iron you can to get the job done. This will help prevent board damage, as excess heat can cause the copper traces to delaminate from the board.

Пайка поверхностного монтажа

Mastering the skill of surface mount soldering requires a bit of patience, practice, and good tools. The basic tools that are required beyond the basic soldering kit are tweezers and a magnifying lens.

A good pair of fine-tipped tweezers is an essential soldering accessory for surface mount components. Tweezers are required for the safe handling of surface mount components during soldering, as small components will rapidly heat up and become hot enough to burn your finger. Tweezers are also necessary to hold small components in place, preventing them from being pulled around by the surface tension of the liquid solder as it melts and cools. The point on the tweezers should be small enough to fit in between the pins of the finest surface-mount part that you intend to work on. This way, you can use the tweezers to manipulate individual pins during soldering and inspection.

There are many grades of tweezers. The grading is based on the sharpness, quality and durability of the tips, the alignment of the tips, and the spring action of the tweezers. High-grade tweezers are a little bit pricey, but well worth the investment if you intend to do a lot of surface mount soldering. Distributors that focus on production supplies, such as Future-Active Electronics (www.future-active.com), sell a reasonable selection of good tweezers.

The biggest challenge in surface mount soldering is being able to see what you are working on. Your hands have the ability to easily and repeatedly manipulate objects smaller than the naked eye can see. The ideal magnifying solution for soldering is an optical stereoscope, like the kinds used for the inspection of biological specimen. Unfortunately, these microscopes are very expensive, with the better models selling for around the price of a used car.

A more economical solution is to use a tabletop magnifying lens. Many drafting/art supply stores sell these kinds of lenses, and most office supply stores sell at least one model of lamp with an integrated magnifying lens. These magnifying lenses will help assembly, but they lack the power to do a thorough inspection of your soldering job. A relatively inexpensive high-power portable magnifier, such as a jeweler's loupe or a proofing magnifier, should be used to inspect finished solder joints.

Техника для простых компонентов

Simple surface mount components, such as resistors, capacitors, inductors, and small semiconductor devices such as transistors and diodes, are easy to mount on a circuit board. Let's look at the technique for mounting these components.

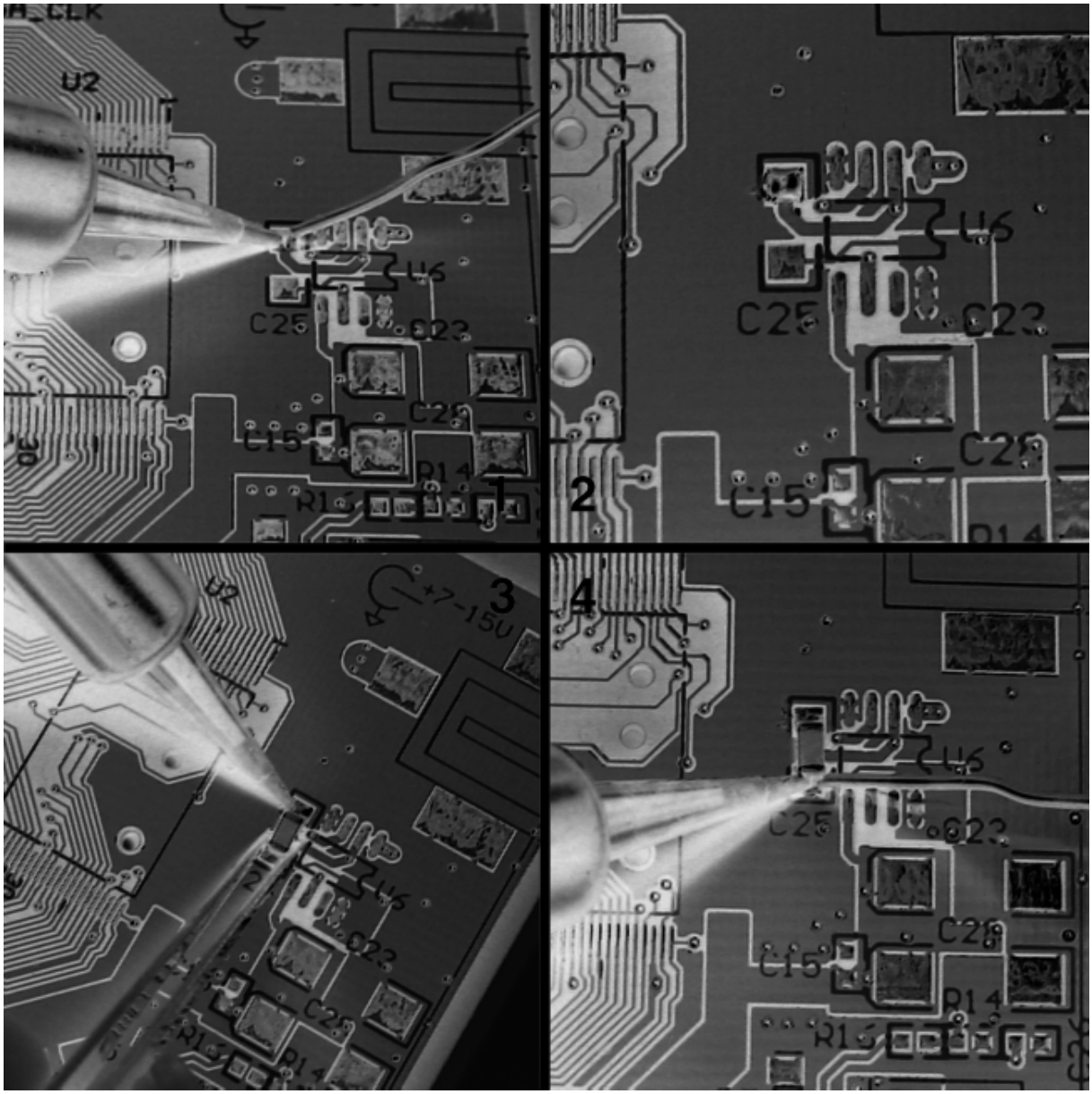


Рисунок B-1. (1) Apply a touch of solder to one of the target component's pads. In this case, the target component is C25. (2) Picture of the solder blob after application. (3) Use tweezers to align the target component and then apply heat with a soldering iron until the initial solder blob has flowed around the component's leads. (4) Solder the remaining component pads.

First, place a blob of solder on one of the component's pads, then place and align the component over its pads using a pair of tweezers. Once you feel comfortable with the component's placement, apply heat to the blob of solder until it melts and the component sinks into place on its pads. Keep applying the heat while you adjust the alignment of the component. Remove the heat, and wait for the solder to cool down and solidify. Double-check the alignment of the component, and then solder the rest of the component's contacts to the board. If the first solder joint looks dull or weak, reheat it with a touch more solder once all the other pins have been soldered down (see рисунок B-1). This technique works for components with two

or three pins, and for multi-pin components with a wide pitch, such as the 50-mil pitch small outline IC packaged devices.

The most difficult situation you may encounter when soldering down these components occurs when one or more of the component leads are attached to a large area of copper, such as a power plane. In this case, a large amount of heat must be applied to the PCB pad before the solder will wet and adhere to the circuit board. Beware when this happens, as the solder will oxidize and ball up, and make poor electrical contact with the board. Add a touch of solder flux if this situation occurs to help enhance the wetting action of the molten solder.

> Tip
>
> A «dry» soldering iron tip will have trouble heating large patches of copper, or pads that are connected to power planes. One way to increase the rate of heat transfer from the soldering iron is to start the iron out with a blob of molten solder on the tip, and touch the iron to the board through the molten solder blob. This blob will spread out and heat the board locally, and will establish a good thermal connection that more effectively transfers heat. This will ultimately lead to a better solder connection once more solder is added.

Техника для сложных компонентов

Most integrated semiconductor components today are available in some kind of a fine-pitch surface mount package. This kind of packaging can be intimidating at first, but a little bit of practice with a few simple techniques and tips is all it takes to attach these parts with a high degree of confidence. The process of attaching a fine-pitch surface mount component is very similar to that of attaching a simple surface mount component, and it should take no more than a few minutes for a typical mid-sized surface mount package.

The first step is to tack the component on the circuit board by soldering down only two corner pins on the chip. If the alignment is not correct on your first try, it is easy to correct by just heating one of the two corners while pushing the chip in the desired direction for alignment (see рисунок B-2). Be very mindful of the accuracy of this alignment: a misaligned chip will lead to no end of frustration while trying to solder the chip down.

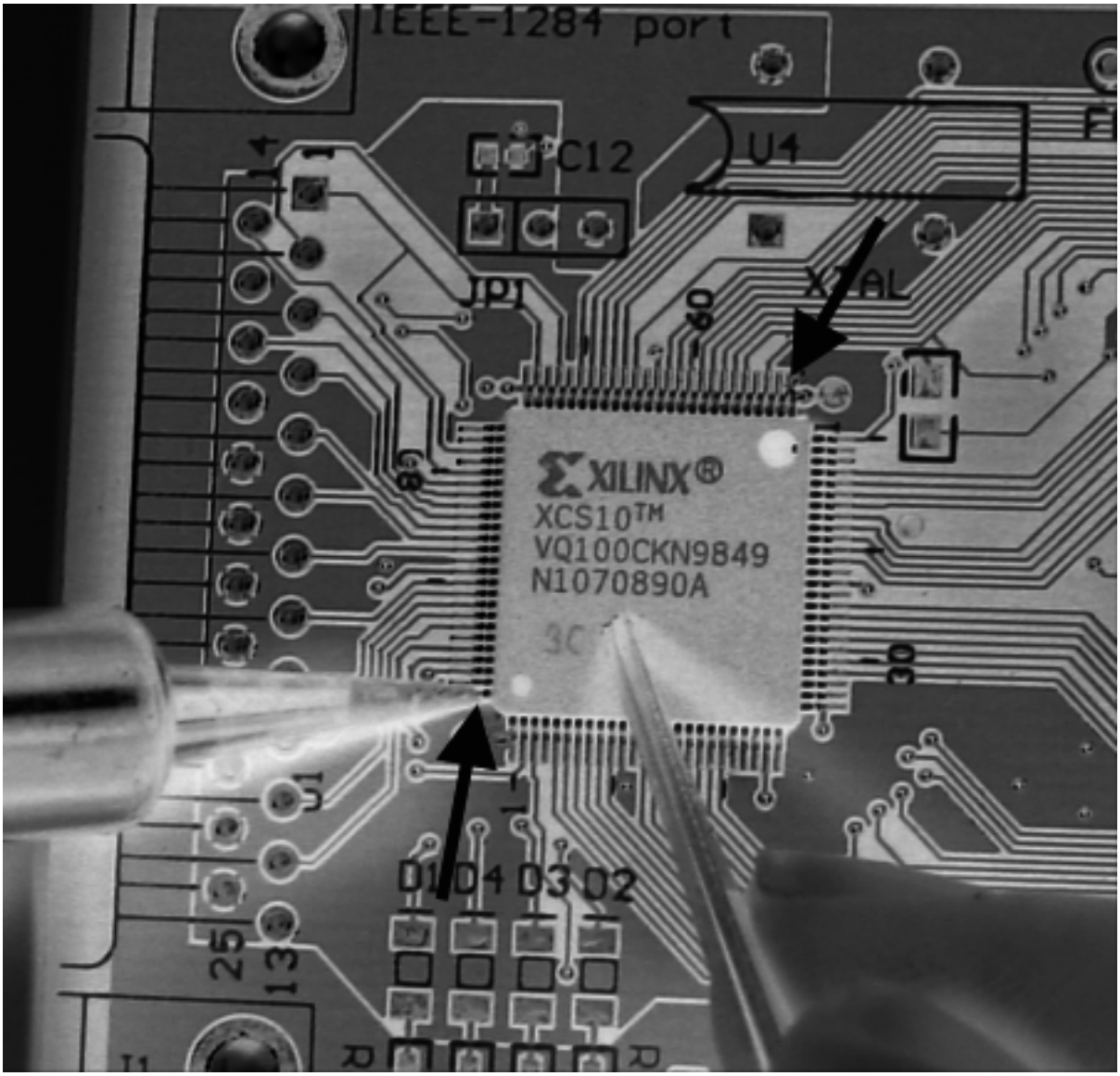


Рисунок B-2. First, solder down one pin on each of two opposite corners of a chip. The arrows on this diagram indicate the corners used in this example.

The next step is to apply a thin film of solder flux around the chip. This will enhance the wetting action of the solder on the chip pins and cause the solder to wick onto the component pads, instead of in between the leads, which would cause a short (see рисунок B-3).

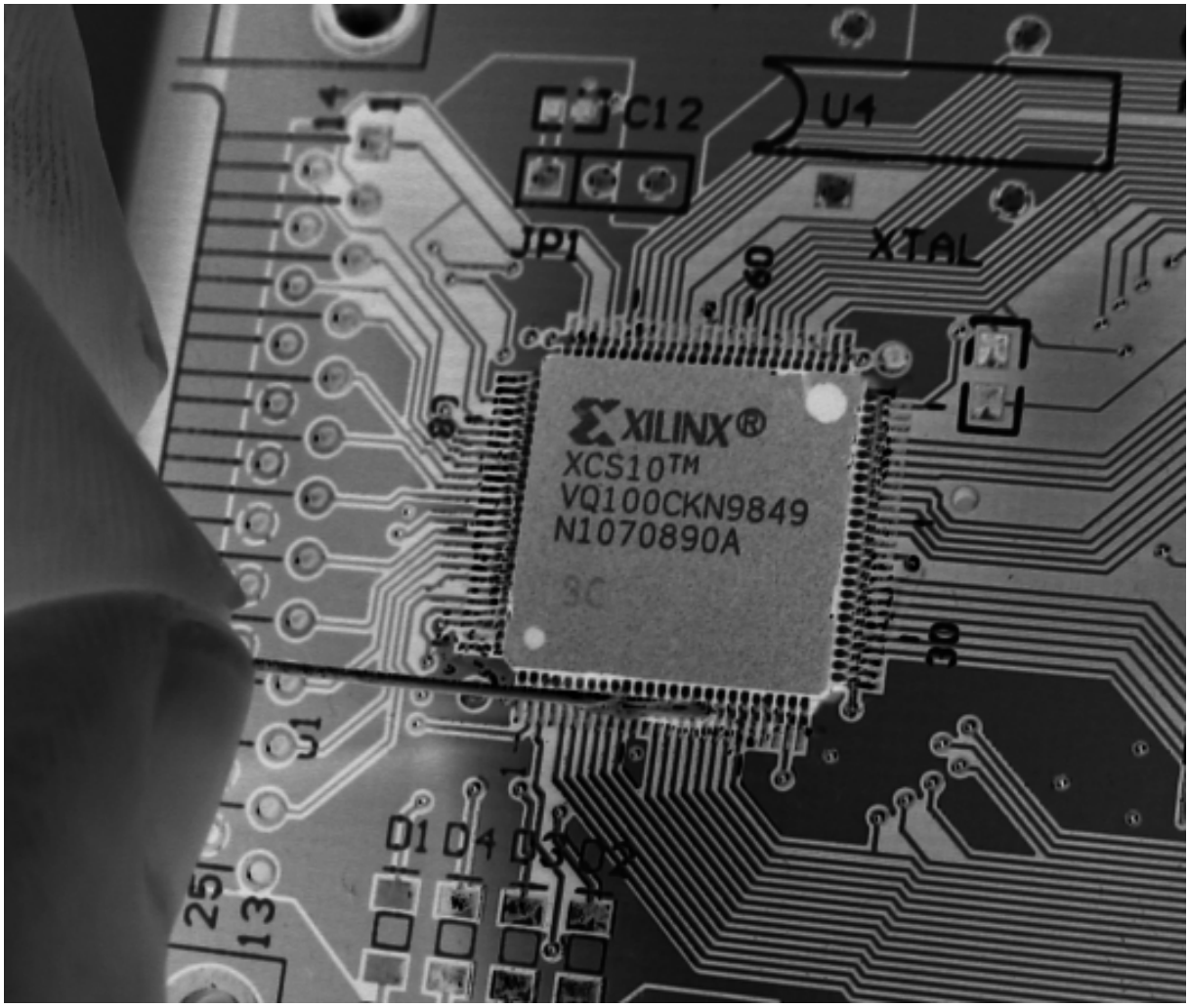


Рисунок B-3. Apply a thin film of solder flux to the pins around the chip. In this illustration, a paste flux is being applied using a scrap piece of wire that is frequently dipped in the flux container.

> Tip
>
> When laying out a board, use extra-long pads for fine-pitched surface mount components. The extra length will make hand-soldering easier, though it will also make routing a little bit more difficult and cause the board size to grow slightly. The extra pad length acts to wick away excess solder from the chip leads, thus making solder bridges less likely.

Once all of the leads have been uniformly coated with the soldering flux, load the tip of a soldering iron with a tiny ball of solder, and press this ball up against the unsoldered leads. The ball of solder will wick into the space underneath and around the component leads. Repeat this process until all of the leads have been coated with solder. Do not worry at this point if excess solder bridges multiple pins. Once you are finished soldering all the pins, use a copper desoldering braid (solder-wick) to remove any solder bridges, as shown in рисунок B-4.

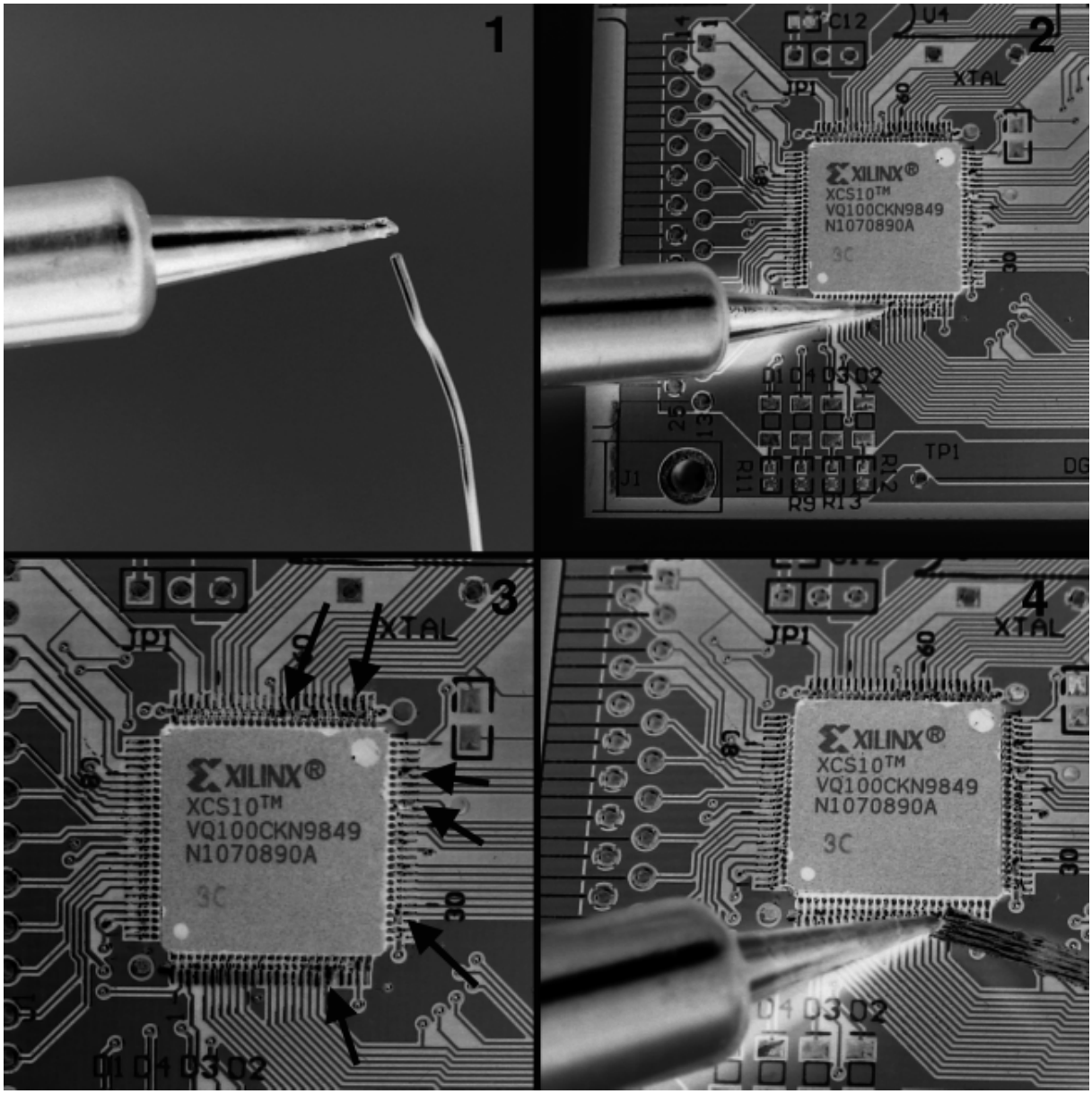


Рисунок B-4. (1) melt a small amount of solder onto the tip of a hot iron. (2) Transfer this solder onto the pins of the component. Repeat steps (1) and (2) until all pins are soldered. (3) Solder bridges, indicated with arrows, will have formed across many pins. (4) Remove the solder bridges using a piece of copper desoldering braid.

You are almost done. Finally, make sure that all the pins are firmly soldered down. It is difficult to do this inspection visually without a microscope. Instead, pull on the pins with a needle or the tips of a tweezer, dragging the tweezer along the pins as shown in рисунок B-5 using a firm, steady motion. Pins that are not soldered in place will move slightly. If they do, repair them by pushing them back into place, and applying a touch of solder to the pin.

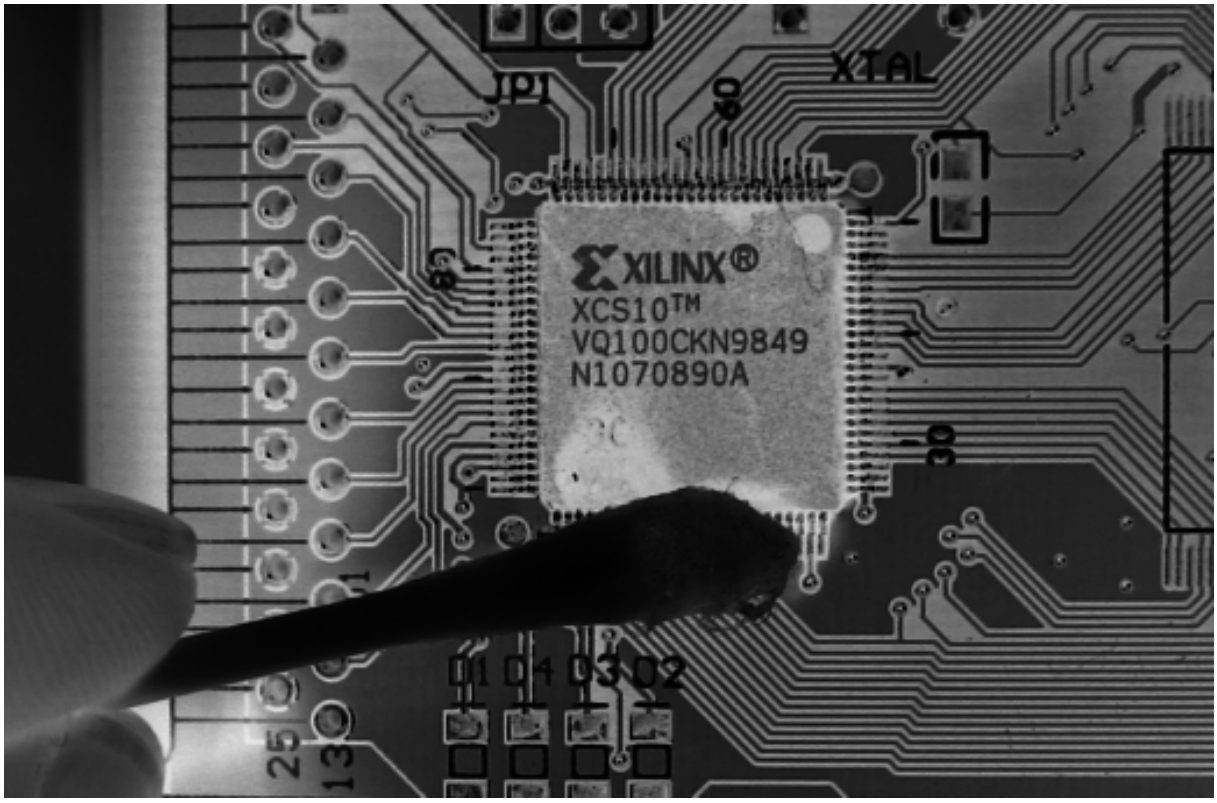


Рисунок B-5. Stroke the pins along the side of the chip using a pair of tweezers or a needle using a firm, steady force. Pins that are not soldered well will bend slightly. The arrow indicates the direction of motion.

> Caution
>
> Always check for power shorts after soldering a circuit board. Some solder bridges can be microscopic dendrites, and some solder bridges will form behind the pins underneath the chip. Check for power shorts using a multimeter set to measure resistance. Do not use the audible continuity mode, as large, high speed computer boards usually have a low (few ohms) resistance between power rails, which is low enough to register a short on many continuity meters. Also, when measuring the resistance between power lines, wait a second or two for the measurement to settle. The initial resistance between power rails will be very low while the large capacitors that sit on the power lines charge up.

Once your component has been soldered down, clean up the excess flux using a mild solvent, such as Isopropyl alcohol, and a cotton swab as shown in рисунок B-6. Clean-up is important because it makes visual inspection easier, and it also makes it easier to probe the chip pins during debugging. It is also important because many fluxes tend to crust over and trap contaminants with age, hampering future repairs.

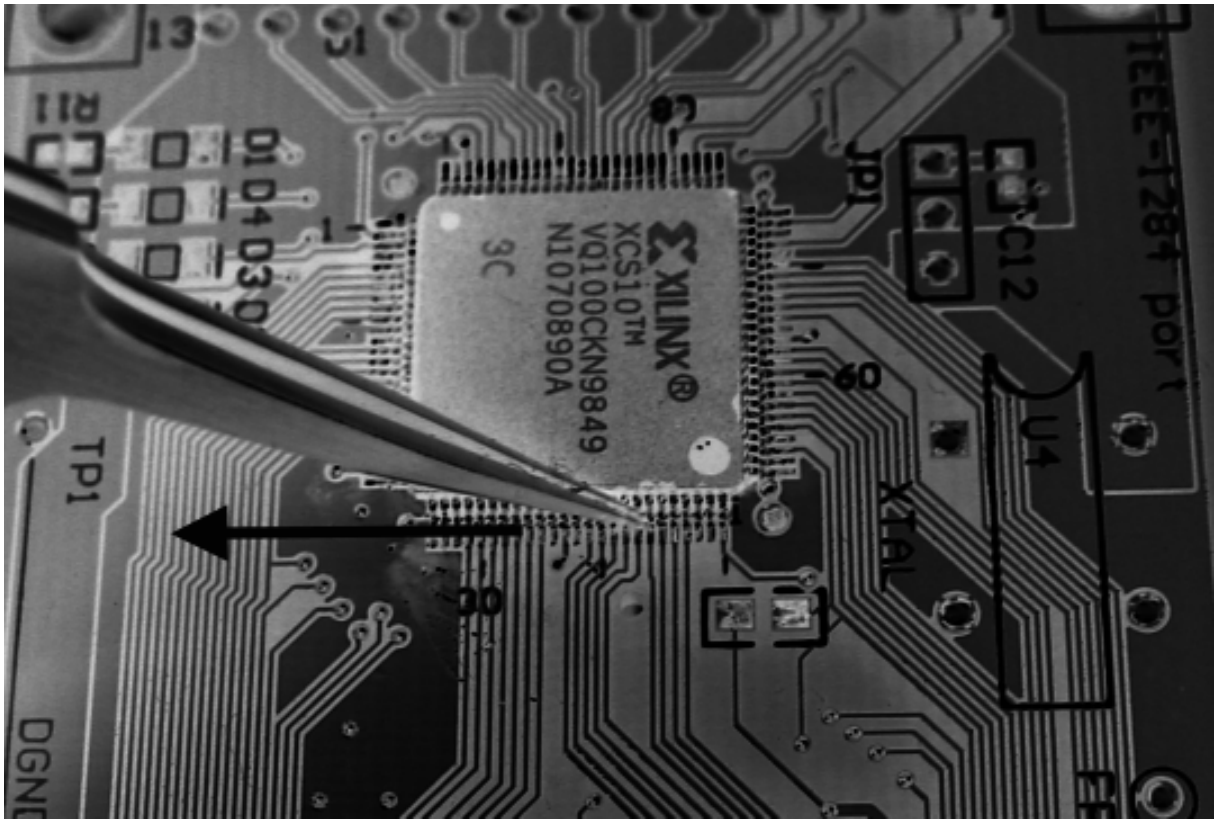


Рисунок B-6. Clean up the flux residue using a cotton swab dipped in a mild solvent.

Техника снятия компонентов

There are many techniques for removing surface mount components, and many of them require special tools, such as tong-style soldering irons or hot air guns. Tong-style soldering irons are good for removing smaller surface mount components, especially on boards that are densely packed. They are fairly quick and efficient, and a proper tong-style soldering iron will leave the pins of the component in relatively good condition. However, they are also fairly expensive, and a bit of practice is needed in order to figure out the right timing and pressure to use when removing a chip.

An easier and cheaper solution is to use a heat gun, like the ones sold in hardware stores for removing linoleum floor tiles. The heat gun will heat the entire region of a board, and components will float off their pads with minimal damage to their leads. The downside of this technique is that it is not very precise, so it is not an ideal solution for removing chips on boards that are going to be used again. As a result, heat guns are most effective for salvaging good chips from broken boards. The other problem with using heat guns is that the heat can warp boards or cause destructive failures in chips that have absorbed moisture into their packages. This failure mode, called «popcorning», happens when moisture trapped inside a chip boils but cannot escape, causing a pressure buildup that culminates in a destructive release event. Chips and boards operating in a humid environment or that have been washed in water should be baked in an oven at around 200 to 250°F for a few hours before desoldering with a heat gun.

Another option that is particularly appealing for hobbyists is to re-alloy the solder joint so that it melts at a very low temperature, below 300°F. A company called Chip Quik (www.chipquikinc.com) holds the patent on this technique, and removal kits can be purchased from many vendors, including Jameco (order number 141305). This technique is appealing because it requires no special equipment, and it is easy and fairly safe, as shown in рисунок B-7.

To do so, first coat the chip pins with soldering flux, then melt the chip removal alloy onto the pins of the component to be removed. This will create a large bead of low melting temperature alloy all around the chip. Next, heat the entire bead by dragging the tip of the soldering iron through the bead. The stored heat in the chip and the alloy will keep it molten long enough for the whole chip to be easily slid off its pads. Finally, clean up the low melting temperature alloy by heating it with a soldering iron and then wiping it away with a cotton swab. The alloy will wipe off fairly easily from both the board and the chip.

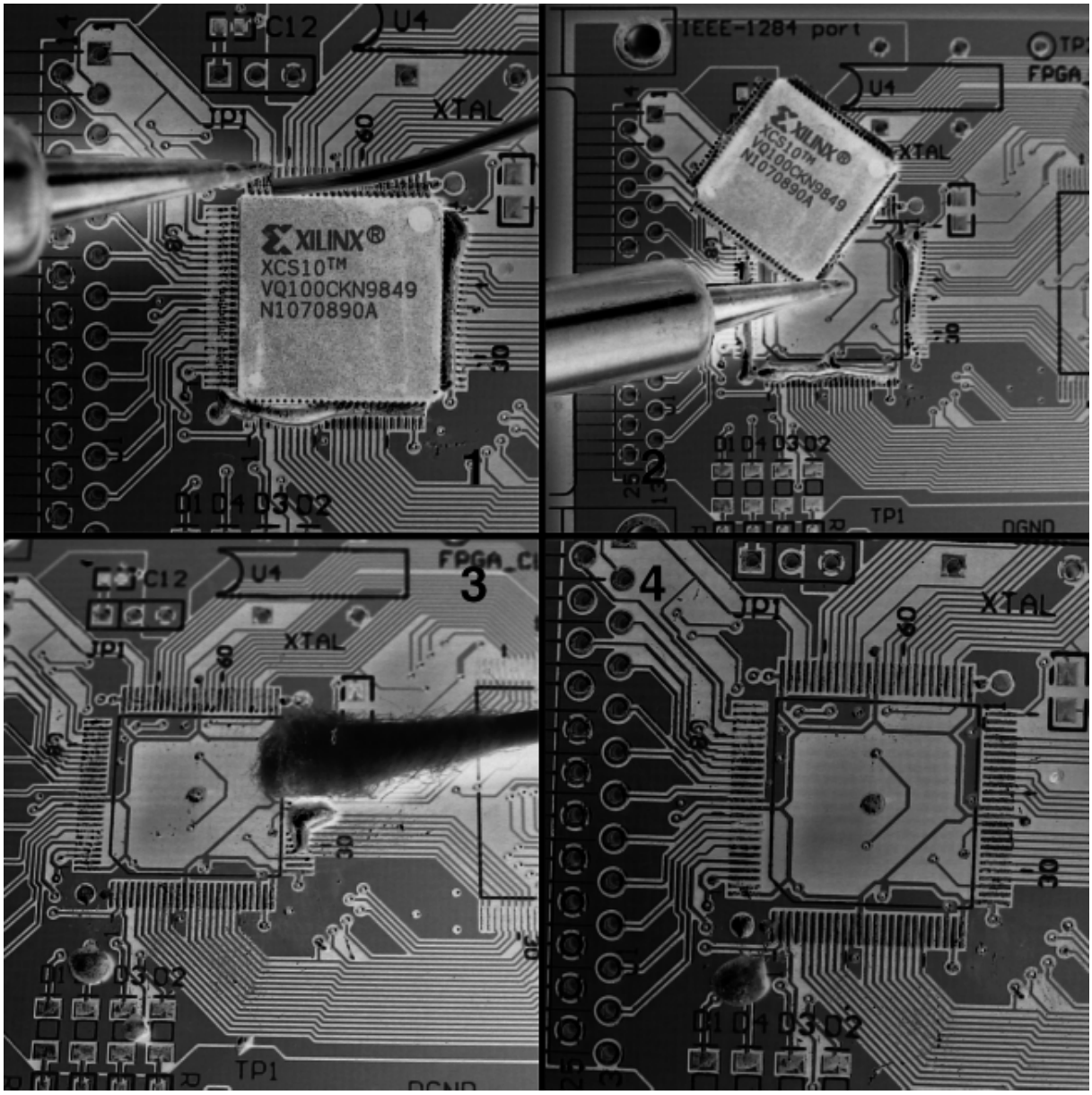


Рисунок B-7. Removing a chip by realloying the solder joint. (Prepare the chip pins with soldering flux before starting this process.) (1) Melt the realloying compound onto all of the chip pins. The realloying compound comes in wire form. (2) Heat the blobs of metal around the chip by dragging a soldering iron through the metal. All the realloyed solder will become molten, at which point the chip can be pushed off its pads. (3) Clean up the realloying compound from the board and the chip by heating it and wiping it away with a cotton swab. (4) The result, a clean chip removal.

This re-alloying technique preserves the integrity of the removed chip's pins as well as the pads on the circuit board. The downside of this technique is that cleaning up the alloy can be a bit messy, and tiny particles of alloy can get caught between the pins of neighboring chips. (Caution while cleaning up will prevent this problem.) The other downside is that removing chips consumes the removal alloy, so

removing large numbers of chips with this alloy can become expensive in the long run. Fortunately, chip removal is a fairly rare occurrence for most hobbyists.

The «ball grid array» (BGA) package is a type of surface mount package that is becoming increasing popular due to its high electrical performance and density. These packages are attached to a board through a large array of solder balls underneath the package. Clearly, this kind of package cannot be soldered using a conventional soldering iron. In addition, BGA packages are very difficult to inspect because most of the solder connections are buried deep below the package. The only viable method for attaching these components is to use an oven that heats the whole board and the component to the point where all the solder balls melt. Typically, the alignment of these BGA packages is accomplished with the aid of a machine of some kind, and the inspection of these BGA packages is accomplished with either X-rays or ultrasonic imaging.

The equipment required to install these kinds of packages is very expensive, so the best way for a hobbyist to install a BGA packaged device is to pay a professional assembly house to do it. Companies such as Naprotek (www.naprotek.com) that specialize in quick-turn, low volume prototyping will install a BGA packaged device for a fairly reasonable fee. When getting a BGA installed, always ask for a copy of the inspection photo for the part; the peace of mind brought by the photo is worth the extra cost. BGA parts can also be removed and re-balled, although this process can be much more pricey than the attachment process.

Приложение C. Введение в разводку PCB

Это приложение познакомит вас с процессом проектирования и разводки PCB, а также даст обзор доступных инструментов, которые могут понадобиться для достижения ваших проектных целей, с упором на продукты и приемы для тех, у кого бюджет на шнурке. В заключение я представлю несколько простых проектов, с которых можно начать свои аппаратные приключения.

Философия и поток проектирования

Печатные платы - это холст аппаратного хакера, а CAD-инструменты - кисти. Как и в любой инженерной или художественной дисциплине, приобретение уверенности в проектировании и разводке печатных плат требует практики. К счастью, появление недорогих сервисов прототипирования PCB и бесплатных (или почти бесплатных) CAD-инструментов сделало проектирование и разводку PCB недорогим и доступным хобби.

Проектирование и разводка PCB - две тесно связанные задачи. Компромиссы возникают и меняются на протяжении стадий проектирования и разводки. Иногда компонент может не поместиться, или деталь может оказаться недоступной, и вам придется изменить схему, чтобы учесть этот недостаток. В другой раз вы можете внести высокоуровневое изменение в проект или поймать ошибку, и PCB придется обновить, чтобы отразить эти изменения. По моему опыту, ключ к тому, чтобы быстро выдать успешный проект PCB, - быть гибким на всех фронтах проектирования.

Уточнение вашей идеи

Процесс проектирования PCB всегда начинается с вашей идеи. Первое, что нужно сделать, - получить очень ясное представление о том, что именно вы пытаетесь построить. Чем больше конкретики у вас есть заранее, тем проще окажется процесс проектирования. У вас должно быть представление о том, насколько большой должна быть итоговая плата, сколько она должна стоить и, конечно, что она должна делать. Я всегда считаю полезным рисовать эскизы, а в случае больших проектов - писать проектные документы, которые помогают мне организовать и зафиксировать мои мысли.

Для первых нескольких проектов одним из самых трудных шагов будет точно сформулировать идею, потому что вы еще не будете знать, какие виды компонентов доступны для реализации вашей идеи и вокруг каких реальных ограничений вам придется проектировать. Лучший способ начать - найти уже существующую идею, очень похожую на вашу, и смоделировать свою идею по ее образцу. (Многие производители микросхем предлагают бесплатные application notes и образцы проектов, которые образуют отличную отправную точку.) Еще один способ определить свои идеи - учиться на существующих продуктах: если вы хотите построить будильник, разберите существующие часы и посмотрите, как они были сделаны.

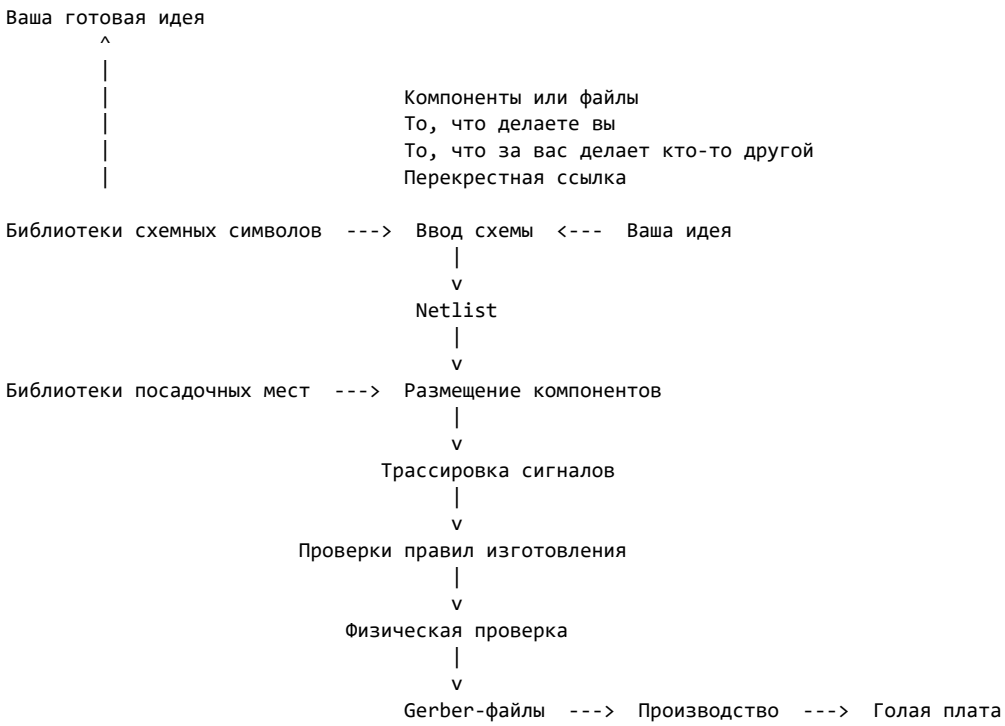
Ввод схемы

Когда у вас есть идея, нужно создать принципиальную схему. Схема - это символическое представление вашей идеи, выраженное как набор символов деталей и виртуальных проводов.

Большинство программ для ввода схем поставляются с библиотекой деталей, чтобы ускорить процесс ввода схемы. Однако если вы не найдете нужную деталь в библиотеке, вам придется создать собственный схемный символ. Все схемные символы связаны с посадочным местом PCB-компонента, то есть с рисунком меди на печатной плате, который сопрягается с компонентом.

Один распространенный источник ошибок возникает из-за того, что не проверяется связь между символом и посадочным местом: посадочное место 16-контактного DIP не подходит для 16-контактного линейного разъема поверхностного монтажа, и большинство проектных инструментов не могут отличить одно от другого. Поэтому проверяйте, что все назначения посадочных мест корректны, и дважды и трижды проверяйте схемный символ. В частности, всегда проверяйте и перепроверяйте выводы питания, поскольку они могут вызывать самые трудные и разрушительные виды ошибок. Подумайте о том, чтобы попросить друга также дважды проверить символы, чтобы избежать повторения ошибок или не пропустить ошибки по незнанию. (Такой объем избыточности может звучать глупо для простых деталей, но он становится абсолютно необходимым для деталей с сотнями и тысячами выводов, когда ваш мозг размякает на середине процесса проверки.)

Внимание к деталям с самого начала - самый важный навык при вводе схемы, и оно может избавить вас от головной боли из-за раздражающих ошибок позже. Каждый вывод каждого компонента существует по какой-то причине, и если какой-либо вывод оставлен неподключенным, вы должны понимать, почему это допустимо или недопустимо. Для этого читайте datasheet'ы продуктов, включая каждую страницу и каждую сноску. Не игнорируйте мелкий шрифт, который требует подтягивающего резистора для задания условий запуска или конденсатора для фильтрации шума либо стабилизации системы, иначе вы, скорее всего, получите еще больше раздражающих ошибок.



```
Физические ограничения  --->  Инструмент разводки PCB

Компоненты  --->  Спецификация материалов  --->  Дистрибьютор  --->  Сборка

Правиловые и эвристические проверки
```

Рисунок C-1. Процесс проектирования платы, от идеи до готового продукта.

Проверки правил проектирования полезны для нахождения некоторых ошибок в схемах, но проверки, которые они выполняют, обычно довольно элементарны и поэтому ловят только самые грубые ошибки. Типичный набор проверок может включать обнаружение дублирующихся обозначений деталей, висящих сетей и плавающих входов.

Помимо обычных проверок правил проектирования, вы также можете придумать собственные простые проверки. Например, многие инструменты проектирования не выполняют "проверки четности netlist", которые можно довольно легко сгенерировать с помощью perl-скрипта или табличной программы. Проверки четности netlist - это эвристические проверки, которые подсчитывают количество компонентов, подключенных к каждой сети, а затем сортируют этот подсчет по числу соединений. "Одноконтактные сети" окажутся наверху отсортированного списка. Одноконтактная сеть почти всегда указывает на типографическую ошибку (неверно написанное имя netlist) во время ввода схемы. Полезно бегло просмотреть весь отсортированный список, чтобы убедиться, что все группы сигналов имеют одинаковое число соединений. Большинство сигнальных шин имеют одинаковое число соединений для каждого сигнала в шине. За один день можно создать инструмент, который позволит выполнять проверки подсчета соединений netlist для вашей программы ввода схем, и вы, несомненно, сэкономите себе бесчисленные часы усилий и денег, найдя ошибки, которые он поймает.

Перед экспортом схемы в инструмент разводки платы следует заказать все детали из спецификации материалов схемы. Часто критические детали будут в дефиците, и ввод схемы придется переработать, чтобы учесть нехватку. Нередко единственная необходимая переработка - изменение назначения посадочного места компонента. Изменение схемного проекта с учетом дефицита у производителя позволяет избежать трудной задачи внесения изменения в уже завершенную разводку платы.

Разводка платы

Входом для программы разводки платы служит netlist, аннотированный информацией о посадочных местах компонентов. Netlist - это промежуточное представление каждого компонента и вывода, а также их связности. Извлечение netlist - хорошо автоматизированный процесс, но сопоставление схемных символов с посадочными местами печатной платы не всегда хорошо управляется.

Трудность сопоставления символа и посадочного места возникает из-за наличия нескольких вариантов корпусовки для одной детали. Например, символ транзистора может одинаково подразумевать крошечное устройство в корпусе SOT-23 или огромное устройство в корпусе TO-3, и именно вам нужно убедиться, что при извлечении netlist выбран правильный корпус. Всегда полезно проверять подразумеваемое посадочное место тогда, когда компонент помещается в схему, а не проверять все сразу во время трансляции netlist или, что еще хуже, во время размещения или финальной проверки проекта.

Когда у вас есть законченный netlist, вы готовы выполнять разводку платы.

Другой внешний вход для программы разводки платы - правила проектирования. Правила проектирования задаются компанией-изготовителем плат и включают спецификации минимальной ширины дорожки и минимального зазора между дорожками, минимального размера отверстия,

минимального кольца сквозного отверстия, а также количества слоев питания и трассировки. Точные правила проектирования зависят от выбранного процесса, который, в свою очередь, определяется тем, что вы можете себе позволить. Лучшие процессы предлагают дорожки шириной до 2 mil (mil - это 1/1000 дюйма, или 25,4 микрона) и лазерно-сверленые глухие/скрытые переходные отверстия примерно такого же диаметра, но цена изготовления выходит далеко за пределы типичного любительского бюджета меньше ста долларов. Более типичный любительский процесс имеет правила 6 mil дорожка/зазор, минимальный готовый размер отверстия 15 mil и два либо четыре слоя меди. (Список компаний-изготовителей плат вы найдете ближе к концу этого приложения.)

Разводка платы состоит из двух фаз: размещения и трассировки. Разумное размещение деталей сильно упростит задачу трассировки. В общем случае цель - разместить все детали так, чтобы соединения были как можно короче, с как можно меньшим числом переходных отверстий, чтобы минимизировать шум, задержку и потери сигнала. Размещение некоторых деталей, таких как разъемы, переключатели и силовые компоненты, жестко ограничено и оставляет вам мало выбора. Для остальных деталей понимание проекта поможет определить, какие детали должны получить лучшее размещение.

Когда размещение закончено, распечатайте проект в масштабе 1:1 и проверьте, что компоненты подходят к своим посадочным местам, укладывая реальные компоненты на распечатанную разводку. Если вы собираетесь использовать панельку с компонентом, обязательно используйте панельку при проверке чертежа 1:1, поскольку панельки требуют больше места, чем сам компонент. Эта проверка гарантирует, что все компоненты у вас в правильном типе корпуса, что все контуры компонентов корректны и что между каждым компонентом достаточно зазора для удобной сборки. Еще одна важная вещь, которую нужно проверить на чертеже 1:1, - ориентация и распиновка всех разъемов, потому что очень легко перевернуть разъем или использовать посадочное место разъема неправильного пола на печатной плате. Будьте осторожны и при обращении с микросхемами, особенно с теми, у которых мелкий шаг выводов поверхностного монтажа. Следите, чтобы не погнуть выводы, и соблюдайте правильный протокол защиты от статического электричества.

Общие рекомендации по размещению и трассировке

Вот короткий список некоторых рекомендаций по размещению и трассировке. Помните, это всего лишь общие предложения, и, несомненно, будут ситуации, где они неприменимы.

Оставляйте место для fanout-переходов у устройств поверхностного монтажа

Устройства поверхностного монтажа дают большое преимущество в плотности по сравнению со старой компонентной базой со сквозными выводами, которая раньше была стандартом de facto. Однако компоненты поверхностного монтажа все равно требуют сквозных переходных отверстий для трассируемости, особенно в сложных и/или автоматически трассируемых проектах. Эти трассировочные переходы называются fan-out vias для SMD-площадок. Рисунки C-2 и C-3 демонстрируют использование fan-out vias на детали поверхностного монтажа.

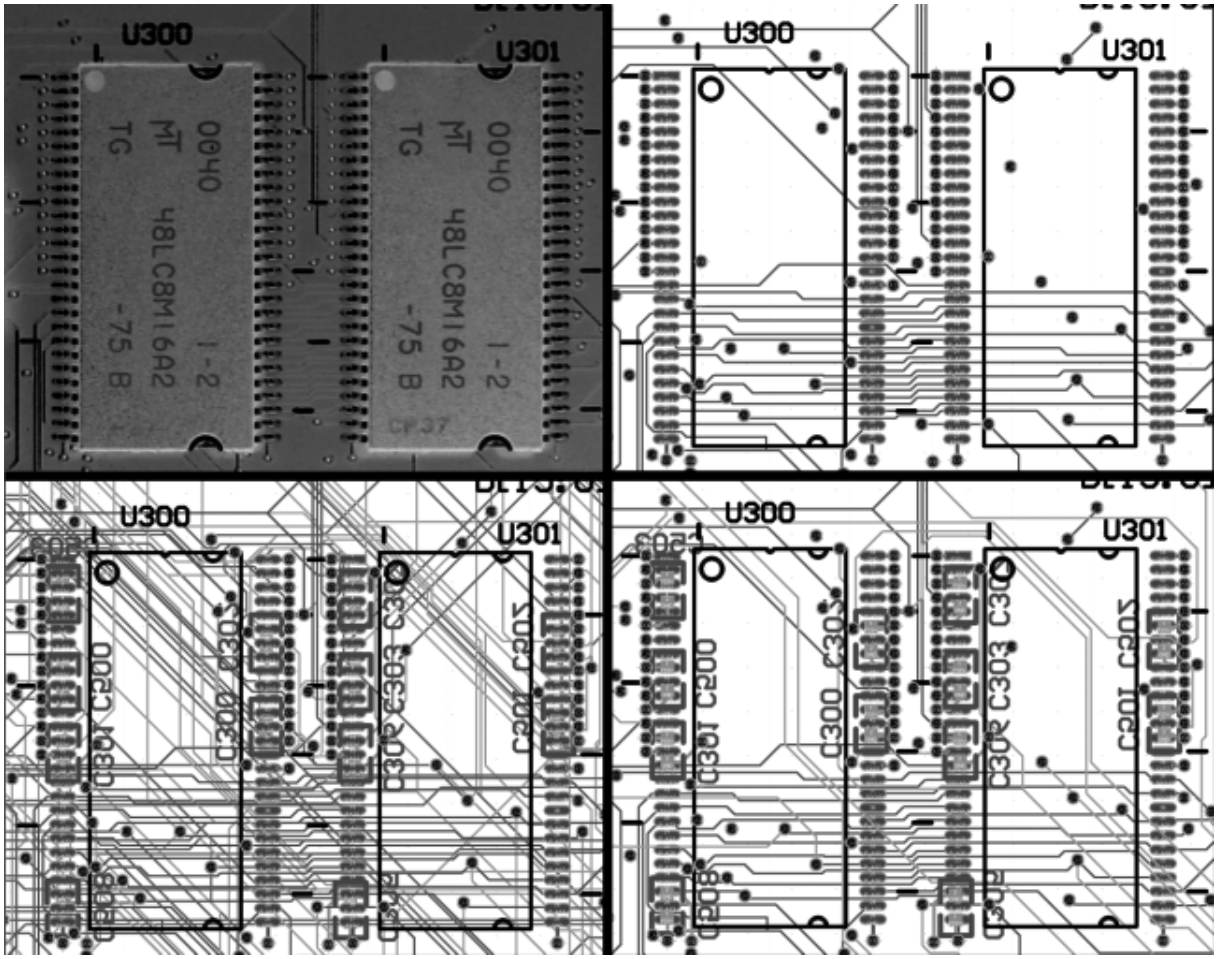


Рисунок C-2. Четыре вида разводки печатной платы. Сверху слева, по часовой стрелке: изготовленная печатная плата с компонентами; вид верхнего слоя печатной платы в программе разводки PCB; вид всех слоев печатной платы в программе разводки PCB; вид только верхнего и нижнего слоев, демонстрирующий двухстороннюю SMT-разводку.

Развязывающие конденсаторы хорошо помещаются под SMD-площадками

Самый распространенный пассивный компонент в типичном цифровом проекте - развязывающий конденсатор. Эти крошечные конденсаторы есть повсюду, и они могут съедать ценное пространство для трассировки и fanout-переходов, если их неправильно разместить. Если вы готовы создать двухстороннюю плату поверхностного монтажа, развязывающие конденсаторы можно разместить на стороне платы, противоположной площадкам целевого компонента. Размещая эти компоненты под областью площадок компонента, вы не расходуете никакую область fanout. На самом деле удачно размещенный развязывающий конденсатор может использовать тот же переход питания, который используется выводами питания компонента. Вид в левом нижнем углу рисунка C-2 дает ясную иллюстрацию этой техники. (Есть некоторые особые случаи, когда вы можете не захотеть так делать, как отмечено в следующем разделе.)

```
площадки компонента
области fanout
корпус компонента
```

Рисунок C-3. Области fan-out вокруг посадочного места SMD-компонента.

Знайте свои особые дорожки

Хорошая новость о разводке цифровых печатных плат состоит в том, что большинство дорожек требуют мало размышлений, в отличие от типичной аналоговой платы. Плохая новость состоит в том, что если остальные дорожки вы сделаете неправильно, плата будет демонстрировать странное и раздражающее поведение, которое будет трудно отлаживать. Поэтому трассировка этих особых дорожек - немного черная магия. Этот раздел дает лишь несколько рекомендаций по обращению с такими особыми дорожками, но я советую заинтересованным читателям найти текст, посвященный разводке плат, чтобы по-настоящему изучить и оценить эти техники. Две книги, которые я рекомендую, - *Digital Systems Engineering* Уильяма Дж. Далли и Джона У. Поултона (Cambridge University Press), а также *High Speed Digital Design: A Handbook of Black Magic* Говарда У. Джонсона и Мартина Грэма (Prentice-Hall PTR).

Обычно виды дорожек, которые требуют особого внимания при трассировке печатной платы, таковы:

• Дорожки питания
• Сигналы опорного времени (тактовые)
• Высокоскоростные дорожки
• Аналоговые/смешанные сигнальные дорожки

Как общее правило, дорожки питания должны быть толще средней сигнальной дорожки, особенно если вы используете один из более высококлассных процессов изготовления, которые предлагают узкие (~ 5 mil) ширины дорожек. Дорожки питания нужно утолщать, чтобы противодействовать как резистивному нагреву, так и паразитной индуктивности. Узкие дорожки питания, особенно рядом с ключевыми точками распределения питания, будут действовать как резисторы и нагреваться, просаживая напряжение питания до уровня, который вызывает неопределенные неисправности в ваших схемах.

Правильный размер дорожки питания зависит от толщины меди. Типичные платы используют "1-oz copper" толщиной 1,35 mil (один квадратный фут медной фольги толщиной 1,35 mil весит одну унцию). Наружная дорожка шириной 12 mil в меди 1 oz требуется, чтобы пропускать 1 ампер тока с повышением температуры на 10 градусов Цельсия. Для скрытых слоев требуются более толстые дорожки при сходной токонесущей способности.

При трассировке дорожек питания между слоями помните, что переходные отверстия также имеют сопротивление. Одного переходного отверстия недостаточно для соединения критических дорожек питания между слоями. Критические дорожки питания должны иметь несколько переходных отверстий, соединяющих их между слоями, чтобы удерживать паразитные сопротивления и индуктивности низкими. Распределенные плоскости питания на нескольких слоях также должны иметь щедро распределенные переходные отверстия, чтобы гарантировать сохранение общего потенциала.

> NB. В высокопроизводительных или малошумящих приложениях размещение переходного отверстия между развязывающим конденсатором и выводом питания может иметь слишком высокую цену для целостности электрического сигнала ради удобства трассировки. Переходные отверстия нарушают распространение высокоскоростных (сотни мегагерц) электрических волн. Поэтому оптимальное местоположение развязывающего конденсатора в таких приложениях - между выводом компонента и переходом питания.

Сигналы опорного времени включают тактовые сигналы и strobes. Многим устройствам памяти требуются асинхронные управляющие strobes с чувствительными требованиями к таймингу. Эти сигналы должны быть правильно согласованы и разведены способом, совместимым со стратегией согласования, обычно маршрутом "daisy chain". Маршруты daisy chain не имеют ответвлений, поэтому у волнового фронта сигнала есть только один путь движения.

Электрические сигналы перемещаются по печатной плате примерно с 1/4 скорости света, или около трех дюймов за наносекунду. Поэтому высокоскоростные дорожки должны иметь согласованные длины, иначе сигналы могут приходить со значительным сдвигом фазы относительно опорного времени. Длины дорожек согласуются путем удлинения более коротких дорожек до длины самой длинной дорожки. Удлинение длины дорожки выполняется с помощью serpentine-дорожек, которые извиваются и увеличивают эффективную длину дорожки, не меняя размещения конечных точек дорожки.

Аналоговая и смешанная сигнальная трассировка далеко выходит за рамки этого приложения. В среднем любительском цифровом проекте большая часть аналоговой схемотехники будет изолирована источниками питания. Любые специальные требования к разводке для конкретного компонента источника питания обычно хорошо документированы в datasheet'е компонента.

Держите в голове, что электрические сигналы ленивы и неразборчивы: сигнальный ток всегда пойдет по пути наименьшего сопротивления, а сигналы будут наводиться в соседние дорожки. Кроме того, ток должен сохраняться, поэтому у каждого пути сигнального тока должен быть путь обратного тока, явный или нет. Держите эти простые правила в голове, когда разводите любые аналоговые участки на своей печатной плате.

Печатные платы служат хорошими радиаторами

При трассировке мощных компонентов, таких как силовые регуляторы и высокопроизводительные микропроцессоры, помните, что медь в печатной плате - отличный проводник тепла. При определенных условиях вы можете сэкономить себе радиатор, просто разведя большую область меди, соединенную с теплоотводящей площадкой или выводами земли целевой детали. Если вы используете многослойный проект платы с плоскостями питания, используйте несколько переходных отверстий, чтобы помочь проводить тепло во внутренние слои.

Теплоотводящие способности печатной платы также могут быть неудобством при ручной сборке. Хорошая теплопроводность меди затрудняет нагрев вывода компонента, который также соединен с большой областью меди. При соединении маломощных компонентов с плоскостями питания подумайте об использовании переходных отверстий с тепловыми разгрузками. Тепловая разгрузка - это набор небольших зазоров в соединении переходного отверстия с плоскостью питания, который уменьшает теплопроводность без существенного влияния на электрические характеристики соединения. (Заметьте, что большая группа плотно упакованных переходов питания с тепловой разгрузкой вокруг области меди может привести к несоединенным или плохо соединенным островкам меди.)

Установите предпочтительные направления трассировки для каждого слоя

Установление доминирующего направления трассировки для каждого слоя может упростить трассировку плотных плат. Например, сделайте верхний слой горизонтальным слоем трассировки, а нижний - вертикальным слоем трассировки. Если нужно провести сигнал между двумя компонентами, расположенными по диагонали через плату, сначала проведите горизонтальную дорожку по верхнему слою, а затем вертикальную дорожку по нижнему, чтобы соединить два компонента. Альтернативная стратегия, например просто провести дорожку диагонально по верхнему слою платы, снижает общую трассируемость между двумя половинами платы наполовину: единственный способ попасть с одной половины на другую теперь - идти по нижнему слою. Исключения из этого правила допустимы, особенно если приходится делать компромисс между целостностью сигнала и трассируемостью.

Складывайте плату с ортогональными слоями

После установления предпочтительных направлений трассировки для каждого слоя сложите слои так, чтобы никакие два слоя не имели параллельных предпочтительных направлений трассировки. Эта ортогональность помогает свести к минимуму помехи сигналов между слоями. Если у вас есть слои питания, попробуйте поместить их между слоями, чтобы они помогали экранировать помехи между сигнальными слоями.

На двухслойных платах используйте пальцы для шин питания

На двухслойной плате часто возникает соблазн просто провести питание и землю кольцом по внешнему краю платы. Это не идеальная ситуация, потому что кольцо морит голодом сердце платы, а также повышает вероятность больших паразитных токовых петель, которые ухудшат работу схемы. Вместо этого используйте чередующиеся и/или наложенные "пальцы" питания. Эти пальцы зададут доминирующее направление трассировки каждого слоя и должны быть разведены до трассировки каких-либо сигналов.

Советы по использованию auto-router

Auto-router'ы - смешанное благословение: они могут сэкономить часы трассировки, но также могут вызвать часы раздражающих проблем. Первое правило использования auto-router - никогда не позволять ему работать с вашей единственной копией файла проекта печатной платы. Вместо этого создайте копию проекта и дайте auto-router выполнить свою магию на копии.

Второе правило - изучите ошибки auto-router на простых тестовых проектах, прежде чем применять его к финальному проекту. Auto-router'ы часто имеют критические ошибки или ограничения, которые нужно понять до использования инструмента. При изучении ошибок auto-router обращайте особое внимание на то, как он обрабатывает заблокированные дорожки, залитые полигоны и неудобные размеры дорожек. Некоторые auto-router'ы фактически удаляют заблокированные дорожки (дорожки, проложенные вручную и помеченные как неподвижные), тогда как другие игнорируют их или не работают в их присутствии. Это может быть особенно раздражающим, если вы потратили часы на разводку критических сетей питания и тайминга перед запуском auto-router.

Наконец, не рассчитывайте, что auto-router полностью разведет сложную плату. Auto-router'ы отлично подходят для быстрой трассировки первых 90 процентов платы, но они сильно замедляются по мере того, как плата становится более загруженной. Заметьте, что тонкие

изменения в размещении компонентов могут сделать auto-router успешным или сорвать его. Многие auto-router'ы не распознают шины или прямые сквозные соединения без специальной аннотации или идеального размещения компонентов.

CAD-инструменты

Инструменты проектирования плат за последние несколько лет значительно подешевели. Инструмент, который я чаще всего использую для проектирования плат, - Protel 99SE. (Я еще не купил более новую версию, Protel DXP.) Protel - высокоинтегрированный инструмент, включающий ввод схем, моделирование, управление библиотеками и разводку плат с проверкой правил проектирования и автоматической трассировкой - все интегрировано в одном инструменте. (Похоже, с каждым выпуском ПО в среду проектирования интегрируется какая-нибудь новая функция, к лучшему или к худшему.)

Вы можете скачать 30-дневную полностью функциональную демонстрационную версию ПО Protel с их веб-сайта www.protel.com. Хотя полноценная лицензия на продукт стоит тысячи долларов, это все равно выгодно выглядит по сравнению со многими другими программными пакетами, которые предлагают такую же глубину функциональности и число функций. Другие высококлассные поставщики PCB CAD включают Mentor (PADS), Cadence (OrCAD) и Altium (P-CAD). Интересно, что Altium также владеет пакетом ПО Protel.

Если вы только начинаете и хотите немного поразводить платы ради практики, некоторые компании-изготовители плат предлагают бесплатные полнофункциональные привязанные инструменты проектирования. ExpressPCB (www.expresspcb.com) предлагает бесплатный инструмент ввода схем и разводки PCB для клиентов, использующих их сервис изготовления. Их инструмент функционален, но немного ограничен с точки зрения проверок правил проектирования и практической сложности, для которой его реально можно использовать. Однако ExpressPCB - отличный стартовый инструмент для новичков, и он способен реализовать почти любой аппаратный проект выходного дня.

Перед отправкой любого законченного проекта в производство просмотрите экспортированные файлы сторонним просмотрщиком файлов, чтобы защитить свои вложения в изготовление от ошибок в проектных инструментах. Самый распространенный формат файлов, используемый для изготовления плат, - формат "Gerber". Хороший бесплатный просмотрщик Gerber, которому я доверяю, сделан Graphicode (http://www.graphicode.com/).

Компании-изготовители плат

Компании-изготовители плат различаются по возможностям так же широко, как и CAD-инструменты. Некоторые компании выполняют только большие производственные заказы, тогда как другие зарабатывают на хлеб обслуживанием рынка быстрых прототипов и любителей. Вот несколько моих любимых компаний-изготовителей плат вместе с кратким описанием их базовых предложений.

Sierra Proto Express

Расположенная в Сан-Хосе, Калифорния, Sierra Proto Express предлагает одни из самых конкурентных тарифов на быстрые прототипы. На момент написания Sierra Proto Express предлагает линейку процессов "No Touch Product". Эти производственные процессы имеют строгие требования к правилам проектирования, но они очень доступны по цене. Например, вы можете изготовить двухслойную печатную плату за четыре дня по цене $34 за плату (минимальный заказ - две платы), или можете изготовить четырехслойную печатную плату за четыре дня по цене $51 за плату (минимальный заказ - две платы). Технология, предлагаемая по этим ценам, - это правило 6 mil дорожка/зазор с готовыми отверстиями 15 mil. Sierra Proto Express также предлагает процессы с более быстрым сроком изготовления, шириной дорожек до 5 mil и готовыми отверстиями 10 mil. Для дополнительной информации посетите www.sierraprotoexpress.com.

Data Circuit Systems

Data Circuit Systems, также расположенная в Сан-Хосе, Калифорния, - мой предпочтительный поставщик для проектов, требующих агрессивных правил проектирования или специальных вариантов обработки, которые плохо вписываются в предложения более дешевых quick-turn компаний. Их всеобъемлющий "Process Capabilities Survey" (доступный для скачивания на их веб-сайте) является подробным и ясно написанным, так что он убирает много догадок из интерпретации правил проектирования. Они также выполняют довольно строгий набор заводских проверок вашего отправленного проекта, которые часто ловят тонкие ошибки разводки, способные вызвать проблемы позже. Я обнаружил, что их сотрудники компетентны и дружелюбны, и хотя их цены немного выше, чем у большинства производителей инженерных прототипов, их хорошо документированный процесс и проверки правил проектирования помогают снизить риск агрессивных проектов, и в конечном счете дополнительная стоимость, вероятно, того стоит. Посетите www.datacircuits.com.

Advanced Circuits

Advanced Circuits из Авроры, Колорадо (www.4pcb.com), имеет на своем веб-сайте функцию мгновенного расчета цены. Уже одна эта функция делает их хорошим выбором для плат промежуточной сложности, которые не подходят ни под какие дисконтные quick-turn правила процесса. Вы можете использовать функцию мгновенной цены, чтобы оптимизировать выбор технологии реализации по цене. Кроме того, они часто предлагают скидочные quick-turn спецпредложения.

Alberta Printed Circuits

Alberta Printed Circuits (AP Circuits), расположенная в Альберте, Канада, - одна из оригинальных quick-turn прототипных PCB-фабрик, с быстрым обслуживанием. Процесс P1, который они предлагают, базовый: без soldermask и silkscreen. В результате трудно выполнять проекты поверхностного монтажа с мелким шагом, потому что припой склонен попадать повсюду на этапе сборки. Однако они изготовят и отправят вашу плату в процессе P1 всего за один день по неслыханной цене, без требований к минимальному заказу. Базовая плата за производственный запуск составляет около $45 на момент написания, плюс примерно $0,65 за квадратный дюйм сверх базовой платы. Технология - 8 mil дорожка/зазор с минимальным размером сверла 20 mil (28 mil, если вы хотите придерживаться самого дешевого варианта процесса). AP Circuits отлично подходит для плат, которые нужно сделать в крайнем случае и при строгом бюджете, особенно если вы используете компоненты со сквозными выводами или крупные SMT-компоненты, которые легко собрать без soldermask. Посетите www.apcircuits.com.

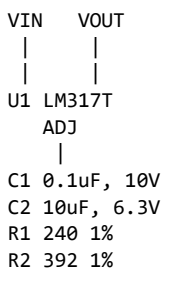
Стартовые проекты

В главе 5, "Замена сломанного блока питания", вам объясняется, как заменить блок питания Xbox стандартным блоком питания ATX. Одна проблема состоит в том, что полярность сигнала включения питания инвертирована между Xbox и стандартным ATX-блоком. Хакерское решение, предложенное в главе, - всегда оставлять блок питания включенным, а вместо этого включать и выключать Xbox, сначала включая блок питания и затем нажимая кнопку питания Xbox.

Довольно легко спроектировать и развести плату, которая позволит инвертировать полярность сигнала питания, чтобы вы могли управлять состоянием питания Xbox только с передней панели Xbox. Вы также можете правильно регулировать резервное питание, вместо использования двух диодов. Такая плата состояла бы из микросхемы инвертора, например 74HCT04, и регулятора, например LM317K. LM317K - регулируемый стабилизатор, который можно настроить так, чтобы снизить резервное напряжение +5V, предоставляемое ATX-блоком, до резервного напряжения +3,3V, требуемого Xbox. Пример принципиальной схемы этой платы показан на рисунке C-4.

Выбор разъемов для этой платы остается за вами. Самым простым решением было бы просто использовать отверстия и припаять провода через отверстия. На этой плате всего пять соединений. Три идут к блоку питания: провод +5VSB (фиолетовый), провод земли (черный) и выход включения питания (зеленый). Оставшиеся два, +3.3VSB (вывод 6 на разъеме питания) и вход включения питания (вывод 11 на разъеме питания), идут к Xbox.

Обязательно проверьте выходное напряжение регулятора перед установкой готовой платы. Довольно легко ошибиться номиналом резистора или перепутать вывод, и оба эти состояния могут привести к опасно высоким напряжениям, подаваемым в Xbox. Кроме того, при постоянной установке платы обязательно изолируйте нижнюю и верхнюю стороны платы от случайного контакта с корпусом Xbox или другими компонентами Xbox.



U1: вывод 3 = VIN, вывод 1 = ADJ, вывод 2/Tab = VOUT

```
U2 74HCT04
вывод 12
вывод 14
вывод 7

+5.0V Standby от блока питания
(фиолетовый провод)

+3.3V Standby к материнской плате Xbox
(вывод 6 на разъеме питания Xbox)

Сигнал Power On к блоку питания
(зеленый провод)

Сигнал Power On от материнской платы Xbox
(вывод 11 на разъеме питания Xbox)

Земля от блока питания
(черный провод)
```

Рисунок С-4. Пример принципиальной схемы платы-адаптера для замены ATX-блока питания. Резисторы R1 и R2 задают выходное напряжение стабилизатора напряжения U1 равным +3,3V.

Приложение D. Начало работы с FPGA

Интеграция - проклятие аппаратных хакеров. Нам нравится разбирать вещи, изменять их и улучшать, но тенденция состоит в том, чтобы заталкивать все в одну или две ASIC (Application Specific Integrated Circuit, специализированная интегральная схема). Такая интеграция недоступна простым смертным, поскольку стоимость набора масок, используемых для определения структур на микросхемах, быстро приближается к одному миллиону долларов. Это один миллион долларов за каждую уникальную ревизию микросхемы. Если допущена ошибка, которая требует нового набора масок, вам придется потратить еще один миллион долларов, чтобы ее исправить.

К счастью, миллион долларов наличными авансом за микросхему - это слишком много даже для многих корпораций, и это создало рынок для FPGA - универсальных программируемых ("реконфигурируемых") аппаратных устройств, которые можно использовать вместо ASIC во многих приложениях.

Что такое FPGA?

FPGA означает field programmable gate array. Другими словами, это массив вентилей, который может быть запрограммирован в поле конечными пользователями. Можно думать об FPGA как о заказном кремнии, который вы можете построить в удобстве собственного дома, хотя тенденция к частичной реконфигурируемости и контекстно-зависимой реконфигурации добавляет FPGA измерение, которого нет в ASIC. Хотя ASIC дешевле за штуку при больших объемах и могут иметь гораздо более высокие тактовые частоты, FPGA закрепились как предпочтительный инструмент для приложений малых и средних объемов, а также для прототипирования.

Базовая архитектура FPGA - это массив аппаратных примитивов, встроенный в гибкую сеть трассировки. Сила FPGA возникает из того факта, что сложные вычисления можно разбить на последовательность более простых логических функций. Эти более простые функции, в свою очередь, можно разбивать дальше, пока все вычисление не будет описано ничем, кроме последовательности базовых логических операций, которые можно отобразить на аппаратные примитивы FPGA. Таким образом, одну и ту же FPGA можно использовать для реализации микропроцессора, видеоконтроллера или игры в крестики-нолики, просто изменяя конфигурацию аппаратных примитивов и сети трассировки.

Виды аппаратных примитивов, реализованных архитектурой FPGA, сильно влияют на эффективность реализации FPGA для данного целевого приложения. Современные FPGA предоставляют проектировщикам в основном примитивы шириной в один бит: lookup table с 4 или 5 входами и 1-битным выходом, а также один бит синхронизированного по времени хранения, известный как flip-flop. Lookup table используется как логический примитив, потому что ее можно запрограммировать для выполнения любой логической операции с таким числом членов, сколько входов имеет lookup table. Затем эти примитивы соединяются в огромную программируемую сеть проводников; типичная high-end FPGA может иметь многие десятки тысяч таких примитивных элементов.

Оказывается, что хотя однобитные структуры очень универсальны, они могут быть очень неэффективны по ресурсам в приложениях, где естественная ширина данных велика. В частности, область, отведенная под собственно логические примитивы, во многих случаях составляет около 1 процента, а остальное - конфигурационная память и межсоединения. Вся эта проводка нужна, чтобы обработать множество перестановок трассировки, которые могут понадобиться для приложений шириной в один бит.

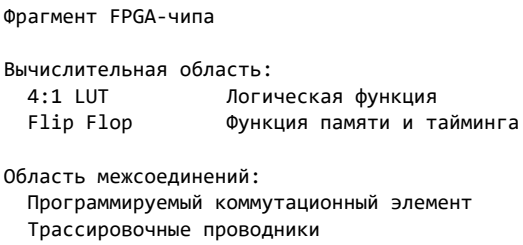


Рисунок D-1. Блок-схема типичной структуры FPGA, иллюстрирующая диспропорцию между количеством проводников на FPGA и количеством вычислительной логики. Типичная современная FPGA будет содержать несколько десятков тысяч таких базовых ячеек.

Чтобы повысить эффективность площади, многие FPGA также включают несколько крупнозернистых примитивов, таких как блоки RAM или блок умножителя. FPGA Xilinx Virtex II-Pro даже включают несколько ядер PowerPC на кристалле. Хотя это звучит впечатляюще, фактическая площадь, занимаемая таким ядром, удивительно мала: процессор PowerPC, вероятно, занимает чуть больше 1 мм2 площади кремния, тогда как площадь FPGA составляет сотни квадратных миллиметров.

Самые свежие FPGA на рынке имеют очень гибкие I/O в дополнение к очень гибкому вычислительному аппаратному обеспечению. Типичная FPGA может взаимодействовать со всеми наиболее популярными высокоскоростными стандартами сигнализации, включая PCI, AGP, LVDS, HSTL, SSTL и GTL. Кроме того, большинство FPGA также могут обрабатывать сигналы с DDR-тактированием. Если эти аббревиатуры ничего вам не сказали, базовая идея такова: FPGA можно использовать, чтобы разговаривать почти с любой аппаратной частью, которую можно найти на типичной материнской плате PC, такой как Xbox. Это чрезвычайно хорошая новость для аппаратных хакеров, потому что означает, что FPGA можно использовать для эмуляции или мониторинга почти любой микросхемы, найденной в PC. (Конечно, PC, возможно, придется замедлить по тактовой частоте в случаях, когда FPGA не может поспеть за скоростью PC.)

Проектирование для FPGA

Для типичного потока проектирования FPGA у вас есть несколько вариантов ввода проекта. Если вы предпочитаете мыслить графически, большинство потоков проектирования поддерживают инструмент ввода схем. Хотя ввод схем часто более интуитивен для аппаратных проектов, их может быть труднее сопровождать и изменять. Например, изменение всех экземпляров имени сети может быть утомительным, если вам приходится щелкать по каждому проводу и вводить новое имя. Кроме того, размер любого отдельного уровня иерархии проекта ограничен размером листа схемы, так что сложный проект потребует значительного планирования и предусмотрительности уже только для ввода схемы.

В результате hardware description languages (HDL) являются предпочтительным инструментом для реализации сложных проектов. На первый взгляд HDL выглядят очень похожими на обычные языки программирования. Например, синтаксис Verilog очень похож на синтаксис C или Java. Однако семантика языка может быть несколько трудной для понимания.

Аппаратное обеспечение обладает присущим ему параллелизмом, который процедурные языки вроде C выразить не могут. Если задуматься, каждый вентиль и каждый flip-flop на FPGA может вычислять параллельно, тогда как в программе на C номинально предполагается один поток выполнения. В результате HDL представляют аппаратное обеспечение как набор процессов, работающих параллельно; именно программист должен сгруппировать все функции в правильные процессы, чтобы компилятор понял, как превратить процесс в вентили.

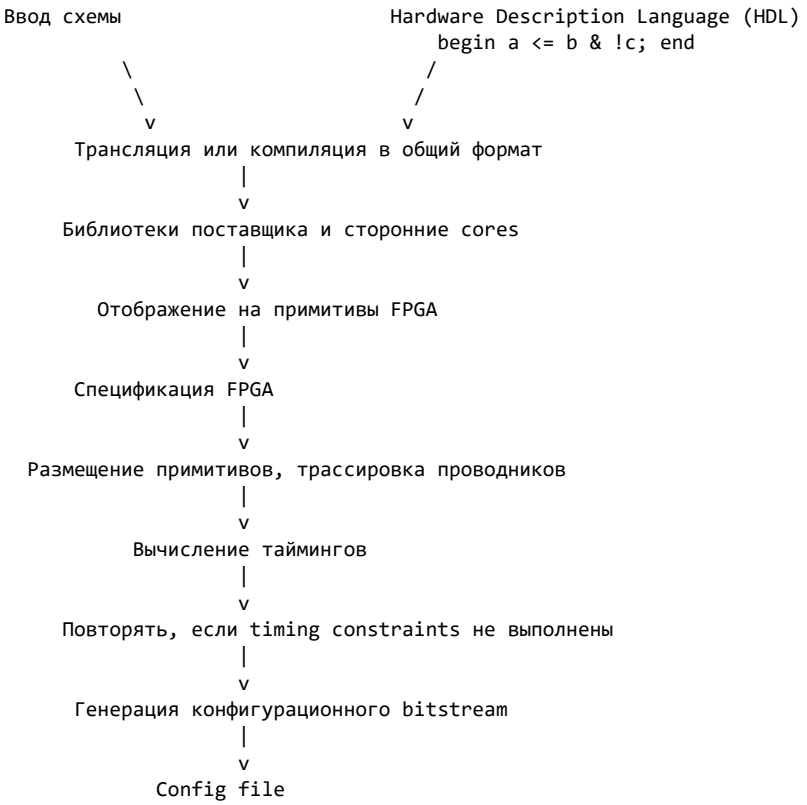


Рисунок D-2. Типичный поток проектирования FPGA.

Например, одиночный тактируемый элемент хранения (flip-flop) в Verilog - это "process", который обычно имеет структуру, похожую на эту:

```
input inData;        // declare your inputs and outputs
input clock;
reg   bitOfStorage;  // declare the storage bit as a reg
type
always @(posedge clock) begin
  bitOfStorage <= inData;
end
```

Этот код берет значение на входном порту `inData` и на каждом нарастающем фронте тактового сигнала сохраняет `inData` во flip-flop, выход которого называется `bitOfStorage`. Несколько процессов, ограниченных синтаксисом `always @( ... ) begin ... end`, могут существовать в одном проекте, и все процессы выполняются параллельно.

Комбинационная логика также может быть выражена как процесс. Например, следующий код Verilog реализует двухвходовый мультиплексор без тактового сигнала:

```
input  a;
input  b;
input  select;
output out;
reg    c;
always @(a or b or select) begin
  if( select == 1'b1 ) begin
    c <= a;
  end else begin
    c <= b;
  end
end
assign out = c;  // assign statements can contain logic
                 // functions as well
```

В этом примере содержимое блока в скобках после ключевого слова `always` содержит sensitivity list, включающий все входы, которые могут повлиять на выход. Если оставить параметр вне sensitivity

list, это означает, что выход не изменится, даже если этот параметр изменится. Например, если бы вы опустили `a` и `b` из sensitivity list, то единственный момент, когда выход изменялся бы, был бы при изменении `select`: вы построили бы latch, который хранит либо `a`, либо `b` в зависимости от состояния `select`. Однако желаемая операция мультиплексора - передавать изменения либо на `a`, либо на `b` на выход постоянно, даже когда `select` не переключается, поэтому и `a`, и `b` должны быть частью sensitivity list.

Есть множество тонкостей в изучении HDL, которые выходят за рамки этой книги, но два приведенных выше фрагмента кода должны дать вам вкус того, чего ожидать. Опытному software-программисту может быть труднее приспособиться к HDL, чем новичку, потому что многие программные приемы, которые считаются само собой разумеющимися, очень плохо переводятся в прямую аппаратную реализацию. Массивы, структуры, умножение и примитивы деления - все это принимается как должное в программном мире, но каждая из этих конструкций переводится в потенциально большие и неэффективные блоки аппаратуры. Кроме того, в аппаратной реализации все возможные случаи в операторе `case` существуют независимо от того, намеревались вы этого или нет; если пренебречь полной спецификацией оператора `case` с помощью `default`, это часто означает, что будет синтезирована дополнительная аппаратура для обработки неявных случаев. В Google проиндексированы многочисленные учебники и справочники синтаксиса Verilog; `verilog syntax` и `verilog tutorial` - оба хорошие наборы ключевых слов для начала поиска справочников синтаксиса или учебников. На сайте Xilinx также есть хороший справочник Verilog для проектировщиков FPGA, а Sutherland HDL, Inc. имеет бесплатное краткое справочное руководство по Verilog по адресу http://www.sutherland-hdl.com/on-line_ref_guide/vlog_ref_body.html.

> Разгон проектов FPGA
>
> Стоит отметить, что timing-модели, используемые для FPGA, довольно консервативны. Это означает, что весьма вероятно, что FPGA будет правильно работать на частотах, намного более высоких, чем признает timing analyzer. Фактически тщательная ручная разводка логики FPGA может вытянуть производительность FPGA намного дальше ее заявленных спецификаций.
>
> Например, FPGA (Xilinx Virtex-E), использованная для реализации отвода шины Xbox Hypertransport, специфицирована только для обработки скоростей данных около 200 Мбит/с/вывод, но приложение требовало 400 Мбит/с/вывод. Причина, по которой мне удалось это сделать, состоит в том, что фактическая логика и элементы хранения могут работать очень быстро, но большая часть производительности сгорает в проводниках и повторителях, которые несут сигналы между логическими элементами. В частности, некоторые проводники будут иметь настолько большую задержку при 400 Мбит/с, что фактически сохраняют данные на один тактовый цикл.
>
> Я определил, какие проводники были медленнее остальных, захватив последовательность данных и сравнив ее с шаблоном, который ранее обнаружил с помощью осциллографа. После того как медленные пути были идентифицированы, я инвертировал тактовый сигнал и/или вставил flip-flop'ы на каналах, у которых было слишком мало задержки. Конечным результатом был набор сигналов с исправленным временным перекосом. Затем эти сигналы можно было тривиально демультиплексировать на более низкую тактовую частоту, где могли использоваться обычные техники проектирования HDL, скомпилированного стандартным способом.
>
> Хотя эта техника очень мощная, она не является применимой в общем случае, потому что величина задержки, вызванная проводником, меняется от микросхемы к микросхеме и может зависеть от таких параметров, как температура окружающей среды и качество напряжения питания. Однако для одной конкретной микросхемы в контролируемых обстоятельствах я смог получить 2x от номинальной производительности. Еще одно

> важное отличие между этим приложением и более общим приложением состоит в том, что bit error rates порядка 1 ошибки на несколько тысяч были допустимы, поскольку я мог просто взять три трассы и XOR'ить их, чтобы восстановить любую информацию, потерянную из-за случайных источников шума. Однако bit error rates 1 к 10 000 неприемлемы для нормальных приложений; более типичны невосстановимые error rates лучше чем 1 к 10 000 000 000 000. Все это возвращается к моей поговорке: "Легко сделать что-то один раз, но сделать что-то миллион раз идеально - трудно".

Еще одно преимущество подхода ввода проекта через HDL - доступность бесплатных и платных "softcores". Веб-сайты вроде www.opencores.org предлагают HDL-ядра с лицензией для широкой публики для таких функций, как USB-интерфейсы, DES- и AES-криптодвижки, а также различные микропроцессоры. Кроме того, почти каждая стандартная функция предлагается сторонними поставщиками, которые продадут вам cores за плату.

После ввода проекта я настоятельно рекомендую симулировать ваш проект перед компиляцией его в аппаратную реализацию. Пытаться отслеживать ошибки, подкручивая код, отправляя его в железо и прощупывая изменения, очень неэффективно. Симуляция позволяет проверить любой узел схемы нажатием кнопки. Кроме того, усилие, необходимое для симуляции изменения кода, очень мало, особенно по сравнению с усилием по проталкиванию изменения весь путь до железа.

Когда проект введен и просимулирован, его нужно скомпилировать или транслировать в общий формат netlist. Этот формат netlist подается в программу, которая отображает примитивы netlist на аппаратные примитивы целевой FPGA, после чего отображенные примитивы размещаются и трассируются. Получившийся проект анализируется на соответствие набору ограничений, заданных проектировщиком. Если проект не удовлетворяет спецификациям проектировщика, он итеративно уточняется через последовательные проходы place and route. Когда проект проходит свои проектные ограничения, он поступает в генератор конфигурационного bitstream, где внутреннее представление FPGA переводится в двоичный файл, который FPGA может использовать для конфигурации самой себя. (Все эти шаги происходят довольно бесшовно по нажатию кнопки в более поздних версиях инструментов проектирования FPGA.)

Идеи проектов

Теперь, когда вы немного знаете, что такое FPGA и как их можно программировать, какие вещи можно с ними делать?

Как оказывается, в наши дни FPGA имеют достаточно логической емкости и производительности, чтобы выполнять очень впечатляющий диапазон задач. Очевидное промышленное применение FPGA - эмуляция проектов, предназначенных для жестко разведенного кремния. Стоимость создания заказной микросхемы резко взлетела, и скоро наступит ситуация, когда одна критическая ошибка может стоить сотни тысяч долларов, если не миллионы, чтобы ее исправить.

С другой стороны, исправление ошибки, сделанной в HDL-описании FPGA, в основном стоит только времени и проектных усилий; вы не выбрасываете никакие детали, и вам не нужно покупать новые детали. Поэтому многие компании приняли стратегию полного моделирования макета проекта в FPGA перед tape-out финального кремния. Побочный выигрыш от такого подхода состоит в том, что команды software и hardware, которые являются пользователями заказного кремния, могут начать проверять свои проекты с использованием FPGA-макета, пока заказной кремний изготавливается; этот процесс иногда может занимать пару месяцев.

Для хакеров FPGA - своего рода панацея для всех видов сложных проектов. Они являются отличным выбором для реализации криптографических функций, если вы заинтересованы в

brute-force поисках ключей или быстром шифровании больших объемов данных. Они также очень полезны для реализации функций обработки сигналов, особенно с учетом существования бесплатных multiplier и digital filter cores. FPGA могут достигать более высокой производительности при меньшей мощности, чем DSP, и поэтому они имеют уникальную нишу в приложениях вроде робототехники с батарейным питанием. FPGA также полезны для embedded controller applications: небольшое микропроцессорное ядро, эквивалентное PIC или лучше него, сегодня легко помещается в FPGA. Добавьте все свои заказные аппаратные периферийные устройства, такие как serial port и PWM timing generators, - и дело пошло.

FPGA также полезны в ситуациях, где фокус не на большом перемалывании чисел. FPGA - отличный кусок glue logic в тесном месте, а удачно размещенная FPGA может избавить вас от необходимости когда-либо добавлять проволочную перемычку для исправления платы из-за ошибки логического проекта. FPGA также являются дешевой альтернативой логическому анализатору для тех из нас, кто не может позволить себе mainframe Tek TLA за $10 000. Высокоскоростные I/O-возможности последних FPGA в сочетании с большими автоматически сгенерированными встроенными памятями, сконфигурированными как FIFO, превращают проектирование системы захвата и анализа сигналов в простую задачу.

Наконец, FPGA имеют применения в смешанных сигнальных ситуациях, которые не сразу очевидны. Самое распространенное смешанное сигнальное применение, вероятно, - использование FPGA для управления аналоговыми сигналами VGA-монитора. Пара резистивных делителей или удачно выбранный тип выходного драйвера - все, что вам нужно, а весь тайминг и логика, необходимые для генерации цветных изображений, могут обрабатываться логикой внутри FPGA. FPGA также можно тривиально использовать как PWM D/A-преобразователи или даже как часть sigma-delta D/A- или A/D-преобразователей.

Где покупать

Вы, вероятно, думаете, что любой инструмент, настолько универсальный и мощный, должен стоить целое состояние. Хотя это было верно примерно десять лет назад, сегодня можно купить FPGA на 100 000 вентилей намного дешевле $50, а инструменты проектирования часто бесплатны для образовательных пользователей и/или любителей.

Конечно, FPGA сама по себе не так полезна; ее нужно установить на плату с правильными соединениями, чтобы ее можно было использовать. Для этого компания XESS (www.xess.com) делает линейку довольно доступных стартовых комплектов FPGA. Их продуктовая линейка меняется по мере появления новых FPGA, но текущая плата FPGA начального уровня - плата XSA-50, которая поставляется с FPGA на 50 000 вентилей примерно за $150. Плата также включает несколько мегабайт RAM, параллельный порт, VGA-порт, порт PS/2-клавиатуры и несколько других необходимых вещей.

Другой вариант - построить собственную плату с нуля, если вы чувствуете смелость. Другие приложения в этой книге описывают, как войти в разводку и изготовление плат и как прикреплять FPGA-устройства с мелким шагом к вашим платам. На самом деле пытаться строить собственные платы довольно приятно, и я рекомендую попробовать; стоимость изготовления платы сегодня значительно ниже $100, так что вы не потеряете слишком много, даже если ваша плата в итоге не заработает.

Если вы делаете собственную плату, вам нужно будет купить FPGA у дистрибьютора Xilinx. Веб-страница Xilinx (www.xilinx.com) имеет самые свежие ссылки на дистрибьюторов. На момент написания один из более удобных дистрибьюторов - NuHorizons (www.nuhorizons.com), поскольку они предлагают информацию о наличии продуктов и ценах на своей веб-странице без требования регистрации или специальной учетной записи клиента.

ПО разработки FPGA обычно можно получить по низкой цене или бесплатно. Например, Xilinx предлагает бесплатную среду разработки для своих линеек деталей Virtex-II (до 300K gates), Spartan II-E и CoolRunner. Среда разработки называется Xilinx ISE WebPACK и доступна для скачивания после регистрации на www.xilinx.com. Эта бесплатная среда имеет впечатляющий список функций, включая схемный и HDL-ввод, HDL-синтез, floorplanner, timing-driven place and route, timing analysis и инструменты power analysis.

Xilinx также предлагает версию своего ПО под названием "Xilinx Student Edition" через Prentice-Hall. Это ПО поставляется в комплекте с рядом учебников и документации, которые могут помочь вам войти в проектирование FPGA. На сайте Xilinx под вкладкой "Education" вы найдете большое разнообразие полезных учебников и лекций.

Приложение E. Отладка: подсказки и советы

Не паникуйте!

Развитие навыков отладки так же важно, если не важнее, чем развитие навыков проектирования. Самый важный отдельный совет - никогда не паниковать: случайное подкручивание и изменение вещей внесет больше неопределенности и ошибок, чем исправит.

Отладка проста, когда дана полная видимость системы и полное знание ожидаемого состояния правильно работающей системы. Простое сравнение наблюдаемого состояния с ожидаемым прояснит, что пошло не так. К сожалению, мир редко работает таким образом. Микросхемы - черные ящики, и единственная видимость внутреннего состояния микросхемы идет через ее выводы. Многие сигналы также слишком трудно напрямую измерять или записывать. Кроме того, спецификации, предоставляемые производителями, часто расплывчаты или трудны для интерпретации. Поэтому настоящее искусство отладки состоит в прослеживании набора симптомов до корневой причины несмотря на недостаток видимости и полного знания системы.

Поймите систему

Пытаться отлаживать систему, сначала не поняв, что именно вы пытаетесь отлаживать, - все равно что пытаться читать японский комикс без какого-либо знания японского. На поверхностном уровне можно понять, кто плохой парень, а кто хороший, но вы действительно теряетесь в том, какое именно отношение ко всему этому имеет парящий кот. Чтобы полностью понять сюжет, вам нужен японский словарь и много времени и терпения. Аналогично, базовые принципы электроники и интуиция доведут вас до точки, где вы примерно знаете, чего ожидать, но просветление приходит только после того, как вы прочитали datasheet'ы компонентов. Чем больше вы понимаете о системе, тем легче будет выяснить, почему что-то пошло не так. Делайте заметки по мере того, как читаете больше о системе, и думайте про себя о способах, которыми проблемы могли бы проявиться, если что-то действительно пошло бы не так. Также помогает видеть другие системы, похожие на ту, которую вы пытаетесь починить, и помогает понимание теории работы.

Наблюдайте симптомы

Ошибки проявляют себя через симптомы, и именно вам нужно вывести корневую причину, наблюдая несколько симптомов и deducing виновника. Пустой экран на TV, который должен показывать видеовыход вашей консоли, - пример симптома. Есть много причин, по которым экран TV может быть пустым, например сломанный видеокабель, сломанный TV, сломанный видеоразъем, сломанный видеоисточник, пустой носитель в видеоисточнике или даже отсутствие питания системы. Как общее правило, следует наблюдать по меньшей мере два, предпочтительно три, симптома, согласующихся с причиной, прежде чем заключать, что вы нашли корневую причину. Держите в голове, что самые показательные симптомы часто не очевидны внешне и потребуют измерения или эксперимента, чтобы их найти. В примере с нашим пустым TV-экраном наши измерения так же просты, как посмотреть, включается ли индикатор питания на TV, или выходит ли звук из TV без видео.

Базовая стратегия отладки - начать с очевидного симптома и изолировать различные части системы, чтобы определить, какая часть является непосредственной причиной симптома. Непосредственная причина определяется как что-то, что напрямую влияет на наблюдаемый симптом. Непосредственные причины отказа видео на TV - отсутствие сигнала к TV, сломанный TV или отсутствие питания; ненепосредственными причинами были бы аппаратный отказ в вашем видеоисточнике или фаза Луны. Другими словами, при симптоме A подумайте обо всех возможных непосредственных причинах X, Y и Z, а затем проверьте каждую, чтобы определить, какая является фактической причиной. Когда вы изолировали проблему, подумайте, что могло заставить ее отказать, и повторяйте процесс, пока не обнаружите корневую причину.

Изоляцию причины ошибок можно облегчить использованием заведомо исправных эталонов. В нашем примере можно исключить TV как источник отказа, подав на него сигнал от заведомо исправного DVD-плеера. Чтобы эксперимент с заведомо исправным эталоном был действительным, нужно сохранять постоянным все, кроме части, которую вы заменяете эталоном. Подключение исправного DVD-плеера к другому входу TV, не тому, который использует консоль, скажет нам только, что работает дисплейная часть TV. Путь от консольного входа к TV не проверен. Правильное выполнение эксперимента подключило бы DVD-плеер к видеовходу, используемому консолью.

Такого рода паранойя или врожденное недоверие к системе становится очень важным при выслеживании тонких аппаратных ошибок. Не принимайте как должное никакой фактор, который мог бы повлиять на наблюдаемую систему, и никогда, никогда не игнорируйте необъяснимое или противоречивое поведение, даже если оно периодическое. Например, иногда система будет работать правильно или ломаться, если вы коснетесь определенного места на печатной плате или помашете рукой возле определенной области; иногда система будет демонстрировать другое поведение на короткий момент после включения питания. Есть соблазн списать такие наблюдения как аномалии или пустяки, но факт в том, что они произошли, и у них должно быть объяснение. Один конкретный пример - касание печатной платы и наблюдение изменения состояния системы. Где вы коснулись? Как вы коснулись? Ваши руки влажные или сухие? Когда вы касаетесь печатной платы, ваше тело действует как малая емкость и большое сопротивление. Это может слегка замедлить сигналы или разрядить высокоимпедансные узлы, такие как неподключенный цифровой вход. Если вы сильно нажали на плату, вы могли изгибать плату таким образом, который меняет электрические свойства треснувшей дорожки или плохого паяного соединения.

Есть некоторые симптомы, которые часто неправильно интерпретируют как причины. Перегоревшая дорожка или поврежденный компонент обычно является симптомом, а не причиной проблемы. Другими словами, неисправность где-то еще в схеме обычно ответственна за отказ компонента. Самопроизвольный отказ компонента - относительно редкое явление. Предположим, вы отлаживаете сломанную стереосистему. Вы чувствуете запах горения, идущий из стереосистемы, и видите большой резистор, почерневший от перегрева. Скорее всего, если вы просто замените этот резистор, замена просто снова сгорит. Реальная причина может быть закороченным транзистором или поврежденной цепью питания, но они проявляют себя не так очевидно, как сгоревший резистор.

Еще одна мощная техника наблюдения - сравнение с заведомо исправной системой. Если вы пытаетесь отладить сломанное устройство, найдите работающее и сравните напряжения и другие рабочие характеристики между ними. Если вы пытаетесь отладить собственную самодельную систему, постройте симуляцию схемы, если возможно, или найдите схему с похожим проектом. Эти заведомо исправные системы можно использовать, чтобы быстро изолировать аномальное поведение. Более того, вы можете контролируемым образом вызывать отказы в заведомо исправном образце, чтобы проверить, действительно ли вы нашли корневую причину проблемы. Эта техника особенно применима к симулированным системам.

Распространенные ошибки

Самый распространенный источник аппаратных ошибок в самодельных проектах - плохие паяные соединения и неправильно установленные поляризованные компоненты, такие как конденсаторы, диоды, IC и разъемы. Кроме того, разъемы - особенно печально известные источники отказов, потому что они подвергаются наибольшему физическому насилию, и обычно трудно определить, находится ли разъем в хорошем состоянии, только визуальным осмотром. Ниже приведен список распространенных ошибок, приблизительно упорядоченный по убыванию популярности.

1. Плохое паяное соединение. Это включает холодные паяные соединения, мостики и забытые соединения. Внимательный визуальный осмотр может поймать многие случаи плохих соединений. Припой между всеми соединениями должен выглядеть гладким и блестящим, и припой должен демонстрировать мокро выглядящий мениск поверх площадок печатной платы и выводов компонентов. Фотографии хороших и плохих паяных соединений можно найти в приложении B: "Техники пайки". Плохие паяные соединения также можно быстро определить на многих корпусах поверхностного монтажа, мягко протягивая жесткую проволоку, например кончик пинцета или скрепку, по выводам вдоль корпуса. Плохо соединенные выводы слегка согнутся. Изгибание платы также может помочь выявить плохие паяные соединения. В других случаях вам может понадобиться омметр, чтобы проверить качество паяного соединения. (Если у вас был грязный опыт пайки компонентов, очистите плату мягким растворителем, таким как изопропиловый спирт, с помощью ватной палочки перед осмотром.) Наконец, помните, что видеть - значит верить: используйте увеличительное стекло, чтобы помочь своим осмотрам. Предпочтителен микроскоп средней мощности, но любая закрепленная лупа (например, те, что стоят на чертежных лампах) или кольцевая лупа вроде используемой ювелирами чрезвычайно поможет.

2. Неправильные номиналы компонентов. Неправильный номинал компонента может случиться, когда похожий внешне, но имеющий другой номинал компонент случайно установлен на печатную плату. Это особенно проблематично с пассивными компонентами поверхностного монтажа, которые часто не маркированы или маркированы неясно. Держите в голове, что единственный способ правильно проверить номинал компонента - снять его с платы и затем проверить. Установки на платы неправильных компонентов можно избежать, если быть очень внимательным и методичным в хранении компонентов в ясно промаркированных пакетах или коробках во время сборки.

3. Плохие разъемы. Это включает разъемы, которые установлены задом наперед, или, что еще хуже, спроектированы с неправильными назначениями выводов. Обращайте внимание, где находится вывод 1, и на систему нумерации, используемую разъемом. Некоторые разъемы используют зигзагообразную систему нумерации выводов, а другие используют круговую систему нумерации выводов. Разъемы wire-to-board также трудно собирать вручную. Осмотрите все точки, где провода взаимодействуют с контактами разъема, на плохие обжимы, лишнюю изоляцию или плохие паяные соединения. В худшем случае используйте вольтметр, чтобы проверить непрерывность разъема.

4. Недосмотры конфигурации в результате непрочитанного datasheet. Сложные микросхемы часто поддерживают несколько режимов работы, которые выбираются притягиванием набора выводов к высоким или низким логическим уровням. Микросхемы также часто требуют внешних резисторов для нагрузки или смещения вывода для правильной работы. Иногда микросхемам также требуются сети из конденсаторов, резисторов и индукторов, чтобы стабилизировать внутренние функции. Держите в голове, что неиспользуемые входы часто требуют терминации

на фиксированное напряжение для правильной работы, так что не игнорируйте части datasheet только потому, что вы не используете определенные функции.

5. Проблема проекта или проблема реализации. Иногда ошибка вызвана прямой ошибкой проектирования или проблемой трансляции между правильной схемой и разводкой платы. Проблемы трансляции часто вызваны опечатками при задании имен схемных сетей или неявными именами питания на схемных символах. Неявные имена питания часто используются на цифровых компонентах ради удобства, но могут вызвать значительные проблемы в проектах, использующих несколько напряжений питания. Эти виды проблем можно поймать до перехода к разводке с помощью эвристической программы проверки netlist, как описано в приложении C. Нарушения правил высокоскоростного проектирования представляют другой вид проблемы реализации. Схемы, работающие на высоких частотах (25+ MHz) или имеющие быстрые фронты (< 5 ns), требуют особого внимания к электрическому импедансу и терминации линии передачи.
6. Блок питания вне спецификации. Проверяйте напряжения питания как можно ближе к точке использования, поскольку провода могут уменьшить фактически доставленное напряжение. В некоторых случаях со схемой все в порядке, а блок питания просто неспособен предоставить достаточно сока, чтобы запустить ваш проект. Также проверяйте изменения напряжения питания во времени. Избыточный шум на питании может вызвать проблемы, а системы, использующие большие объемы высокоскоростной CMOS-логики, могут иметь очень требовательные скачки потребления тока, которые способны вести к коротким провалам и выбросам напряжения питания.
7. Сломанные или поврежденные дорожки печатной платы. Это может быть проблемой, если вы вручную собрали плату и у вас были трудности с установкой компонента. Избыточное тепло во время сборки может заставить дорожки отслоиться от печатной платы. Также знайте своего поставщика плат. Некоторые поставщики плат (особенно quick-turn дисконтные поставщики прототипов) не выполняют полный электрический netlist-тест вашей печатной платы. Ищите перетравленные дорожки, которые истончились за пределы допуска, и также проверяйте, что каждое via-отверстие имеет серебристое кольцо вокруг отверстия. Иногда сверла смещены или наклонены во время сверления платы, и неправильно просверленное отверстие в итоге разрывает электрические соединения.
8. Latch-up или проблема последовательности питания. Latch-up - потенциально катастрофическое явление, при котором паразитное короткое замыкание создается между питанием и землей внутри подложки микросхемы. Latch-up запускается введением тока в подложку. Это может случиться в системах со смешанными напряжениями, где прикладываются входные напряжения, которые выше напряжения питания микросхемы. Во многих случаях latch-up сопровождается перегревом микросхемы, который может привести к постоянному повреждению микросхемы. Рекомендуемая практика при первом включении системы - использовать амперметр, чтобы отслеживать, сколько тока потребляет система, и касаться всех компонентов, чтобы увидеть, не становятся ли какие-либо чрезмерно горячими. Если компонент вошел в latch-up, вы обычно будете наблюдать избыточное потребление тока порядка сотен миллиампер.
9. Тепловая проблема. Это проблема прежде всего с линейными стабилизаторами напряжения и мощной цифровой схемотехникой. Проверьте, что все мощные компоненты правильно посажены на радиаторы и что радиаторы правильно изолированы, когда они контактируют с электрически активной частью корпуса микросхемы.
10. Непреднамеренное короткое замыкание на открытую медь. Это проблема с разъемами и микросхемами, у которых на нижней стороне есть открытые области металла, способные закоротить открытые области платы, такие как vias. Это также проблема вокруг областей, где

для удержания платы используются винты. Головка металлического винта может непреднамеренно соприкоснуться с via, размещенным слишком близко к отверстию под винт.

11. Загрязнение платы. Эта проблема вызвана остатками паяльного флюса или другими технологическими остатками на плате, создающими пути утечки малого тока. Некоторые остатки флюса имеют не пренебрежимо большое (меньше одного мегаома) сопротивление, и это может вызвать проблемы с высокоимпедансными схемами, такими как R-C-сети с большой постоянной времени.
12. Неисправное тестовое оборудование. Это особенно проблема, если вы используете подержанное или старое тестовое оборудование. Тестовые щупы со временем получают перегибы и ошибки калибровки, так что иногда паршивый сигнал, который вы видите на осциллографе, на самом деле является результатом плохого тестового щупа или плохого выбора земли щупа. Калибруйте свое тестовое оборудование по заведомо исправному сигналу, чтобы исключить проблемы тестового оборудования.
13. Наименее вероятная проблема - плохая микросхема или неисправный компонент. Производители компонентов идут на большие меры, чтобы гарантировать, что детали, отправленные вам, функциональны. Типичные уровни отказов измеряются единицами parts per million для простых и умеренно сложных деталей. Часто нам нравится воображать, что причина нашей проблемы - плохая микросхема от производителя, но это почти никогда не так. Обычно, если найдена плохая деталь, она была повреждена либо технологической проблемой (грубое обращение или проблемы сборки), либо проблемой проекта где-то еще в схеме, которая вызывает наблюдаемый отказ.

Восстановление после поднятой дорожки или площадки

Поднятие или разрыв медных дорожек на печатной плате - распространенная проблема, с которой сталкиваются люди, пытающиеся установить after-market модификации с использованием навесных проводов. Это расслаивание дорожек медной фольги обычно вызвано избыточным теплом от паяльника. Еще одна распространенная причина - натягивание присоединенного провода модификации, как можно сделать при снятии изоляции с конца провода после того, как он был припаян к печатной плате. К счастью, обычно довольно легко восстановиться после этой проблемы.

> Совет. Лучшее решение - профилактика. Не используйте слишком мощный паяльник для работы с печатными платами. Предпочтителен паяльник с контролем температуры, но недорогой маломощный (15 ватт) паяльник также будет работать. Кроме того, если припой, кажется, не пристает к плате, прекратите прикладывать тепло. Вместо этого нанесите немного флюса на плату и провод и очистите жало паяльника кондиционером для жала или губкой, смоченной дистиллированной водой (водопроводная вода содержит химикаты, которые могут ухудшать жала паяльников). Это улучшит паяемость, так что вам не понадобится прикладывать столько тепла или силы, чтобы выполнить соединение.

Первое, что нужно сделать, когда вы видите, что дорожка или площадка поднимается от печатной платы, - ОСТАНОВИТЬСЯ! Не усугубляйте проблему дальше; худшее, что можно сделать, - заставить всю дорожку отслоиться назад, продолжая тянуть провод. Снимите провод, если он все еще соединен, едва коснувшись паяльником соединения и позволив проводу отпасть. Рисунок E-1 иллюстрирует такую сцену бедствия.

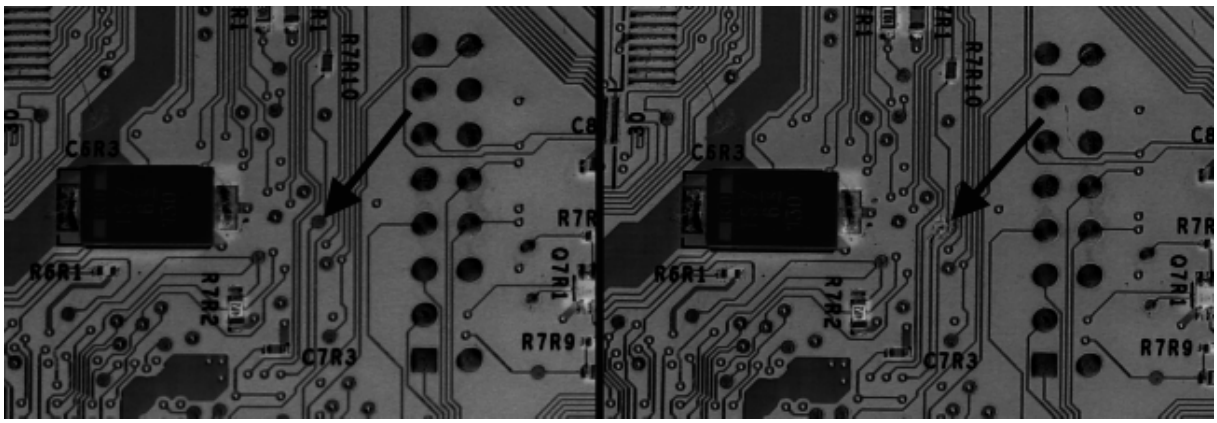


Рисунок E-1. Слева стрелка указывает на исходную площадку, к которой выполняется пайка. Справа площадка была сорвана из-за избыточного тепла и силы.

Стратегия восстановления после сломанной дорожки состоит в том, чтобы удалить soldermask, исправить дорожку перемычкой и найти альтернативную точку для пайки, проследив дорожку до соседнего компонента или via.

Удаление soldermask открывает лежащие под ней медные дорожки. Короткую перемычку можно припаять к этим оголенным дорожкам, чтобы исправить разрыв, вызванный порванной дорожкой. Оголенная область также служит удобной отправной точкой для использования continuity meter, чтобы найти альтернативную точку для крепления перемычки. Удалите soldermask либо мелкозернистой (200 или мельче) наждачной бумагой, либо соскабливанием поверхности острым hobbyist's knife. При удалении soldermask будьте осторожны, чтобы не зацепить куски сломанной дорожки и дальше не оторвать дорожку от платы. Когда soldermask удалена, очистите область мягким растворителем, таким как rubbing alcohol, с помощью ватной палочки. Затем нанесите очень тонкий слой паяльного флюса на область и проведите чистым жалом паяльника вдоль открытых дорожек. Небольшие количества припоя, приставшие к жалу, перетекут на печатную плату и покроют дорожки, предотвращая окисление голой меди. Если жало паяльника слишком чистое, нанесите на него каплю припоя, слегка вытрите жало о влажную губку и попробуйте снова. Не пытайтесь залудить открытые дорожки шариком расплавленного припоя на жале. Будет нанесен лишний припой, который может привести к коротким замыканиям. (Заметьте, что паяльный флюс необходим для получения равномерного тонкого покрытия припоя на дорожках. Не пропускайте нанесение паяльного флюса.) Рисунок E-2 иллюстрирует, как дорожки будут выглядеть до и после процесса лужения.

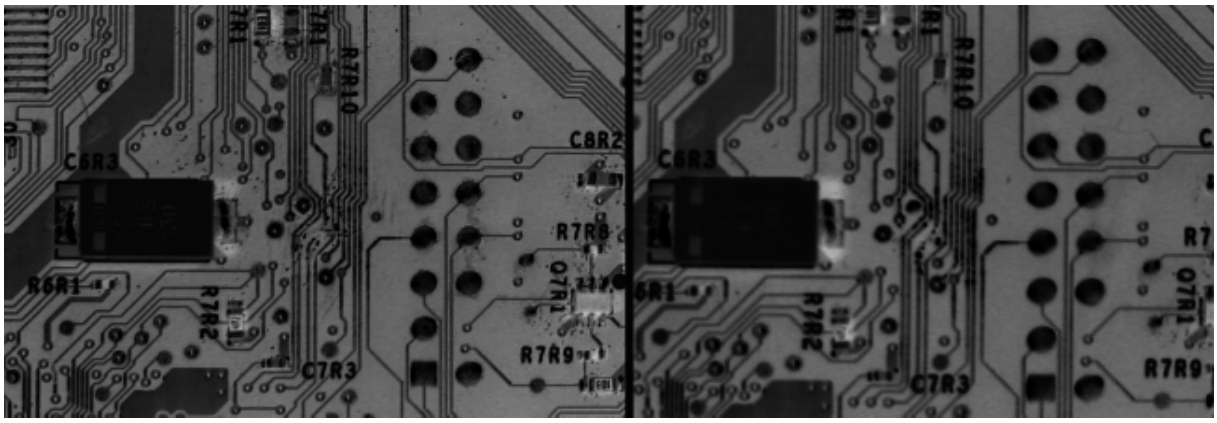


Рисунок E-2. Слева область после удаления soldermask мелкозернистой наждачной бумагой. Справа область после того, как она была залужена (подготовлена заново для пайки).

На этом этапе вы можете захотеть использовать continuity meter, чтобы определить альтернативную точку для присоединения провода модификации. Большинство вольтметров имеют функцию звукового continuity meter. Когда она выбрана, вольтметр издает тон всякий раз, когда сопротивление между щупами очень низкое. И vias, и выводы компонентов служат хорошими альтернативными точками присоединения.

Если вы решите использовать via, нужно соскоблить soldermask и подготовить via перед присоединением провода модификации. Рисунок E-3 иллюстрирует использование continuity meter для нахождения альтернативной точки пайки. Держите в голове, что иногда вам придется проследить через несколько vias, чтобы найти лучшую альтернативную точку присоединения.

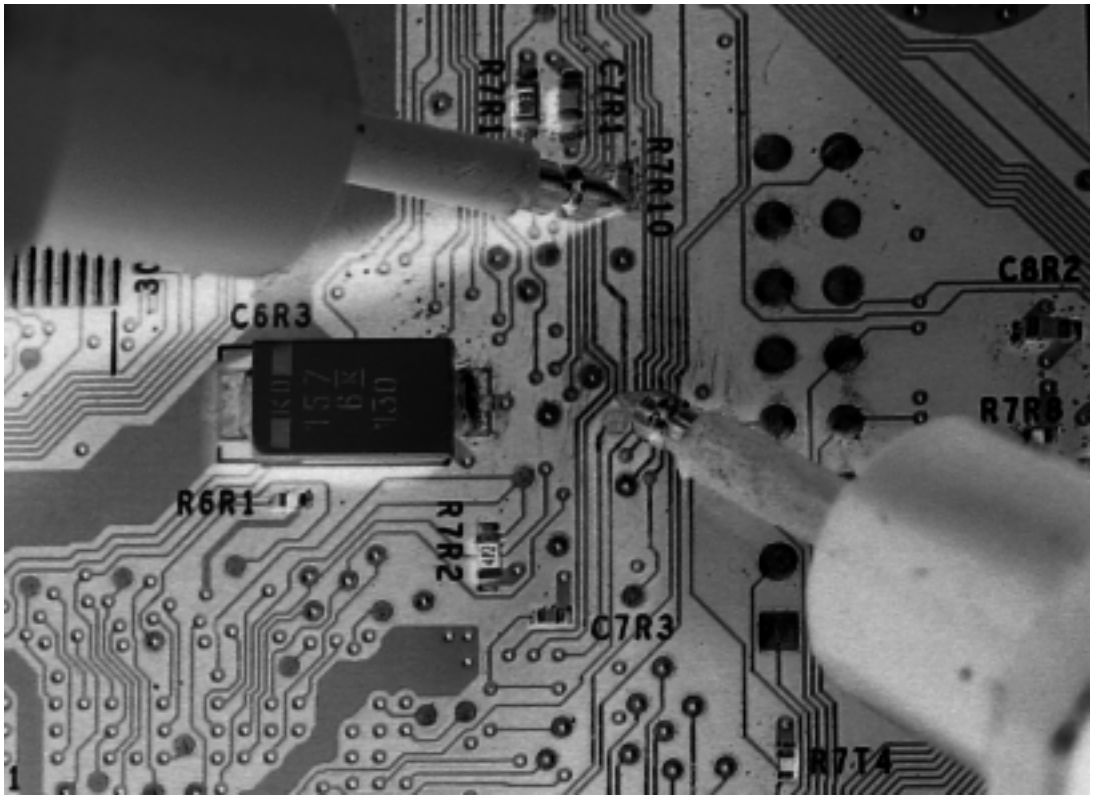


Рисунок E-3. Использование continuity meter для нахождения альтернативной точки присоединения. В этом случае R7R10 оказывается хорошей альтернативой.

Следующий шаг - присоединить короткую перемычку через сломанную дорожку. Нанесите еще немного паяльного флюса поверх области сломанной дорожки. Отрежьте кусок тонкого провода (примерно 30-gauge) примерно равный длине рассматриваемого разрыва. Поместите провод над разрывом, используя липкость паяльного флюса, чтобы помочь процессу размещения. Удерживайте провод на месте пинцетом и прикладывайте тепло паяльником, пока обе стороны не соединятся с краями сломанной дорожки. Проверьте, что провод на месте, осторожно толкнув его пинцетом; провод не должен двигаться. Также проверьте наличие коротких замыканий на соседние дорожки с помощью continuity meter. Если обнаружено короткое замыкание, просто нагрейте перемычку, пока она не отпадет от платы, и попробуйте снова. Рисунок E-4 иллюстрирует, как выглядит восстановленная дорожка.

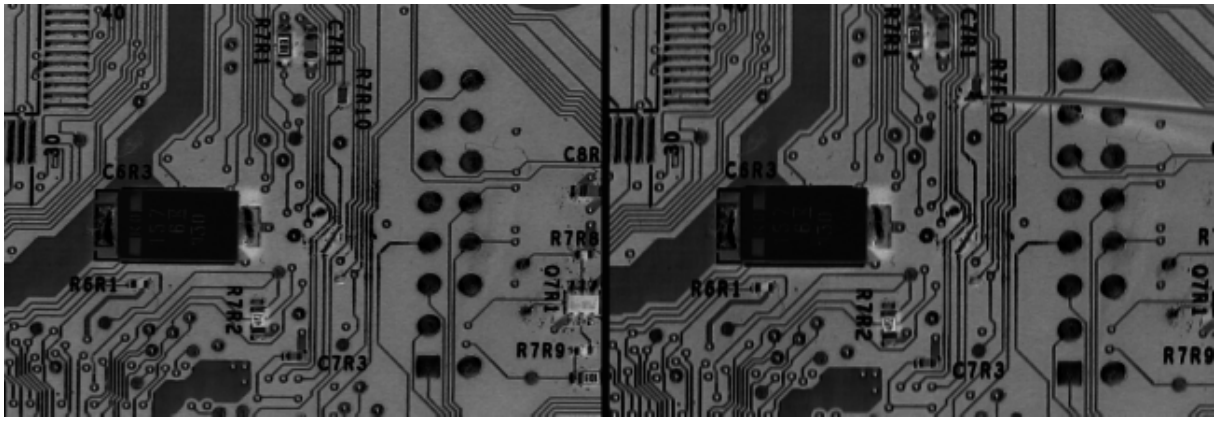


Рисунок E-4. Слева перемычка установлена поверх поврежденной дорожки. Справа провод модификации успешно присоединен к альтернативной точке пайки.

Наконец, присоедините провод модификации к альтернативной точке пайки, которая была ранее найдена с помощью continuity meter.

Приложение F. Аппаратный справочник Xbox

Это приложение суммирует распиновки основных разъемов, используемых в аппаратуре Xbox.

Распиновка блока питания

Блок питания, используемый в Xbox, - импульсный источник, рассчитанный максимум на 96 ватт, с пиковой импульсной способностью 160 ватт менее чем на 10 секунд. Microsoft покупает этот блок питания у нескольких поставщиков, включая Delta Electronics, Inc. (www.deltaww.com). Блок Delta используется в Xbox для Соединенных Штатов, и datasheet на эту деталь можно найти через их веб-сайт или через веб-поиск.

Таблица F-1. Распиновка основного разъема питания. Цвета проводов могут немного различаться в зависимости от конкретной модели блока питания, использованной в вашем Xbox. Эта таблица относится к блоку питания Delta DPSN-96AP A version.

Pin	Описание	Цвет провода
1	+12V	Желтый
2	+5V	Красный
3	+5V	Красный
4	+5V	Красный
5	+3.3V	Оранжевый
6	+3.3V Standby	Коричневый
7	GND	Черный
8	GND	Черный
9	GND	Черный
10	GND	Черный
11	Power On	Белый
12	Power OK	Синий

Распиновка видеоразъема

Распиновка видеоразъема - небольшая загадка, потому что некоторые его сигналы не показывают очевидных или узнаваемых форм сигналов при probing, и потому что один разъем поддерживает несколько режимов отображения. Есть несколько веб-сайтов, которые публикуют распиновки видеоразъема, но перекрестная проверка опубликованной информации измерениями выявляет некоторые расхождения. Здесь я сделал все возможное, чтобы собрать воедино и согласовать две отдельные публикации из XboxHacker BBS и веб-страницы ucon64 на Sourceforge.net. Оригинальные публикации можно найти по адресам http://www.xboxhacker.net/index.php?do=article&id=10&page=1 и http://ucon64.sourceforge.net/ucon64misc/conn.html. Определение всех восьми видеорежимов, выбираемых сигналами MODE1-3, перечислено на веб-странице XboxHacker BBS. Мои измерения показывают, что все соответствия композитного видео и аудиосигналов корректны, но я не смог проверить соответствия SDTV, HDTV и RGB, данные в публикации Xboxhacker BBS. Заранее извиняюсь, если какие-либо из этих сигналов неверны.

Заметьте, что выводы 12 и 24 имеют более длинные штыри на разъеме Xbox, что указывает на то, что они используются для подачи питания на периферийные устройства, подключенные к видеоразъему, во время событий горячей вставки. Более длинные штыри позволяют схемотехнике периферийного устройства включиться до получения сигналов в случае, если периферия подключается при включенном Xbox. Это помогает предотвратить потенциально разрушительную ситуацию внутри микросхем периферии, называемую latch-up.

Таблица F-2. Распиновка разъема питания жесткого диска.

Описание	Цвет провода
+12V	Желтый
GND	Черный
GND	Черный
+5V	Красный

Левая сторона	Правая сторона
1	12
13	24

Рисунок F-1. Аудио-видео разъем Xbox, вид при взгляде на заднюю панель Xbox снаружи.

Нумерация выводов, используемая таблицей F-3 для видеоразъема Xbox, показана на рисунке F-1. Распиновки, полученные из Xboxhacker и из ucon64, расходятся в своей схеме нумерации, поэтому я выбрал схему нумерации где-то между ними. Интересно, что вывод 24 на рисунке F-1 отображается на квадратную площадку на видеоразъеме Xbox, указывая, что схема нумерации, выбранная мной здесь для разъема, не согласуется со схемой нумерации производителя (квадратные площадки обычно обозначают вывод 1, тогда как круглые площадки обозначают все остальные выводы). Это не должно влиять на корректность таблицы, поскольку схемы нумерации выводов произвольны и должны быть только согласованы с таблицей определения выводов.

Таблица F-3. Распиновка видеоразъема.

Pin	Имя сигнала	I/O	Комментарий
1	Right Audio	Out	Аудиовыход, правый канал
2	GND	Power	

Pin	Имя сигнала	I/O	Комментарий
3	SPDIF	Out	Аудиовыход Sony/Philips Digital Interface (S/PDIF)
4	VSYNC	Out	Vertical Sync (режим VGA output)
5	GND	Power	
6	GND	Power	
7	GND	Power	
8	GND	Power	
9	Pb / B	Out	Pb для HDTV mode, Blue для RGB mode
10	GND	Power	
11	Y / G	Out	Y в режимах SDTV и HDTV, Green в RGB mode
12	GND	Power	Имеет более длинные штыри для hot-insertion
13	CVIDEO	Out	Composite video out
14	GND	Power	
15	C / Pr / R	Out	C в SDTV, Pr в HDTV, Red в RGB mode
16	GND	Power	
17	STATUS	Out	Status pin SCART (Syndicat des Constructeurs d'Appareils Radio Récepteurs et Téléviseurs)
18	MODE3	In	Выбор видеорежима output, pin 3
19	MODE2	In	Выбор видеорежима output, pin 2
20	MODE1	In	Выбор видеорежима output, pin 1
21	HSYNC	Out	Horizontal Sync (режим VGA output)
22	GND	Power	Экран аудиокабеля левого канала
23	Left Audio	Out	Аудиовыход, левый канал
24	+5V	Power	Питание +5V, имеет более длинные штыри для hot-insertion

Распиновка USB-разъема

Xbox использует производную USB для портов игровых контроллеров. На передней панели Xbox есть четыре порта игровых контроллеров, и все они имеют одинаковую распиновку, как показано на рисунке F-2.

Контакт
+5V USB
USB Data -
USB Data +
Video Sync
GND

Рисунок F-2. Распиновка игрового контроллера, вид снаружи корпуса Xbox на разъем.

Сигнал "video sync" - это 3,3V CMOS- или TTL-совместимый сигнал. Это базовая последовательность импульсов положительной полярности 15,734 kHz, синхронизированная со временем горизонтальной строки композитного видеовыхода, с одним более длинным импульсом в начале каждого видеополя. Этот сигнал позволяет периферийным устройствам, направленным на TV-экран, таким как light pen или light gun для стрелковых игр, выводить информацию о положении.

Игровые контроллеры подключаются к Xbox через промежуточный break-away connector. Назначение этого break-away connector - предотвратить повреждение консоли (особенно повреждение жесткого диска) из-за перетаскивания или рывка в случае, если шнур запутается вокруг ноги пользователя. Распиновка этого break-away connector показана на рисунке F-3.

Контакт
+5V
Data -
Data +
Video Sync
GND

Рисунок F-3. Распиновка break-away разъема игрового контроллера, вид прямо в разъем, расположенный ближе к Xbox.

Игровой контроллер Xbox имеет два слота расширения для карт памяти, микрофонов и другой периферии. Эти слоты также предоставляют USB-совместимый интерфейс. Игровой контроллер содержит USB hub (микросхему hub Atmel AT43USB401), который повторяет входящий USB-сигнал к слотам расширения. Распиновка разъема расширения показана на рисунке F-4.

1	2	3	4	5
+5V	GND	Video Sync	USB Data -	USB Data +

Рисунок F-4. Распиновка слота расширения игрового контроллера, вид внутрь слота игрового контроллера при кнопках, обращенных вверх.

Распиновка Ethernet-разъема

Ethernet-порт на Xbox - стандартный 10/100 base-TX витая пара RJ-45. Распиновки и цвета таблицы F-4 основаны на стандарте EIA/TIA 568B. Рисунок F-5 иллюстрирует нумерацию выводов разъема.

Таблица F-4. Распиновка Ethernet 10/100 RJ-45.

Pin	Описание	Цвет провода
1	Transmit +	Оранжевая полоса
2	Transmit -	Оранжевый
3	Receive +	Зеленая полоса
4	Not connected	Синий
5	Not connected	Синяя полоса
6	Receive -	Зеленый
7	Not connected	Коричневая полоса
8	Not connected	Коричневый

Первый контакт	Последний контакт
1	8

Рисунок F-5. Распиновка Ethernet-разъема Xbox, вид снаружи в сторону задней панели Xbox.

Распиновка ATA-разъема

Xbox использует стандартную шину Advanced Technology Attachment (ATA) для связи со своим жестким диском и DVD-приводом. Шину ATA обычно (но технически неправильно) называют шиной IDE (Integrated Drive Electronics). Большинство современных приводов квалифицируются как IDE-style drives; например, SCSI-приводы также имеют интегрированную электронику привода. Однако годы (не)правильного использования сделали термин IDE синонимом шины ATA.

Таблица F-5 дает распиновку ATA-разъема, как он виден на материнской плате Xbox при взгляде сверху на разъем, когда задняя сторона Xbox направлена к наблюдателю (разъем должен быть с правой стороны). Заметьте, как нумерация выводов идет зигзагом: все нечетные выводы на одной стороне, а четные - на другой.

Таблица F-5. Распиновка ATA-разъема.

Pin	Сигнал	Комментарий	Pin	Сигнал	Комментарий
1	Reset		2	Ground	
3	Data 7		4	Data 8	
5	Data 6		6	Data 9	
7	Data 5		8	Data 10	
9	Data 4		10	Data 11	
11	Data 3		12	Data 12	
13	Data 2		14	Data 13	
15	Data 1		16	Data 14	
17	Data 0		18	Data 15	
19	Ground		20	Key	Пустой вывод для поляризации
21	DMARQ	DMA Request	22	Ground	

Pin	Сигнал	Комментарий	Pin	Сигнал	Комментарий
23	DIOW-	I/O Write	24	Ground	
25	DIOR-	I/O Read	26	Ground	
27	IORDY	I/O Ready	28	CSEL	Cable Select
29	DMACK-	DMA Acknowledge	30	Ground	
31	INTRQ	Interrupt Request	32	IOCS16-	16 bit I/O
33	DA1	Device Address Bit 1	34	PDIAG-	Passed Diagnostics
35	DA0	Device Address Bit 0	36	DA2	Device Address Bit 2
37	CS0-	Chip Select 0	38	CS1-	Chip Select 1
39	DASP-	Dev. Active/Slave Present	40	Ground	

Разъем питания DVD-ROM

Xbox использует проприетарный разъем питания DVD-ROM. Этот разъем не только подает питание, но также несет несколько управляющих и статусных сигналов. Эти сигналы передают информацию о состоянии привода и лотка привода. Распиновки, приведенные здесь, взяты с Xboxhacker BBS, а оригинальная публикация Ken Gasper, на которой основана эта распиновка, находится на форуме Xboxhacker.

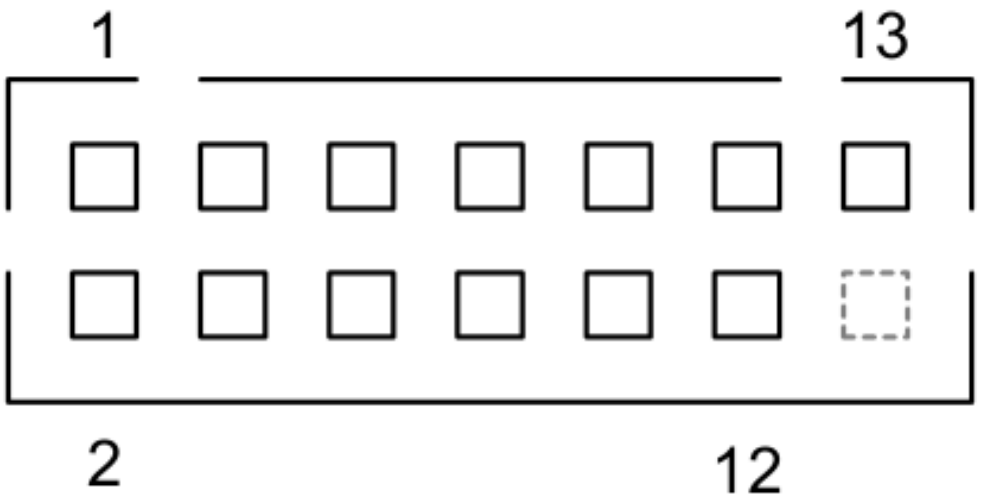


Рисунок F-6. Нумерация выводов разъема питания DVD-ROM, вид сверху на материнскую плату Xbox.

Нумерацию выводов разъема, как она видна при взгляде сверху на материнскую плату Xbox, можно найти на рисунке F-6, а распиновку - в таблице F-6.

Таблица F-6. Распиновка разъема питания DVD (вид на материнской плате).

Pin	Сигнал	Комментарий	Pin	Сигнал	Комментарий
1	12VDC	Питание +12 Volts	2	5VDC	Питание +5 Volts
3	GND	Current return, reference	4	EJECT-	Active low tray eject
5	TS0	Traystate status 0	6	TS1	Traystate status 1
7	TS2	Traystate status 2	8	ACTIVITY-	Disk seek/data transfer
9	12VDC	Питание +12 Volts	10	5VDC	Питание +5 Volts
11	GND	Current return, reference	12	GND	Current return, reference
13	Key	Not connected		Blank	Для поляризации

LPC-разъем

Xbox имеет debug and test port, основанный на шине LPC (Low Pin Count). Эта шина была первоначально определена Intel для использования с Southbridge chipsets, чтобы уменьшить число выводов, тем самым сэкономив стоимость, при сохранении поддержки устаревших PC I/O functions. Эти устаревшие I/O-функции раньше сидели на почти вымершей шине ISA, и они включают клавиатуру, мышь, serial port, parallel port и boot ROM. Спецификацию Intel для шины LPC можно найти по адресу http://www.intel.com/design/chipsets/industry/25128901.pdf.

LPC debug connector особенно значим, потому что его можно использовать для подачи альтернативного ROM image в Xbox в случае, если встроенный ROM отсутствует или поврежден таким образом, что ROM кажется отсутствующим или пустым. Эта функция может использоваться и использовалась для создания простой в установке альтернативной boot ROM для Xbox.

Распиновка Xbox LPC debug connector, кажется, основана на *Installable LPC Debug Module Design Guide* от Intel, http://www.intel.com/technology/easeofuse/LPC_mod_spec72.pdf, с некоторыми небольшими изменениями, как отмечено в таблице F-7. В частности, функция вывода 16 неясна, поскольку его парный вывод 15 был переназначен как вывод питания на материнской плате Xbox. Назначение вывода 15 как вывода питания выводится из толстой дорожки и близкого развязывающего конденсатора, выделенного этому выводу. Если бы вывод 15 предназначался для использования как постоянно высокий сигнал SPDA1, была бы использована более узкая дорожка без conditioning питания.

Таблица F-7. Распиновка LPC-разъема (вид на материнской плате).

Pin	Сигнал	Комментарий	Pin	Сигнал	Комментарий
1	LCLK	33 MHz clock	2	VSS	Current return
3	LFRAME#	Start, end of LPC transactions	4	KEYWAY	Blank for polarizing
5	LRST#	LPC Reset	6	VCC5	+5V power
7	LAD3#	Muxed Address/Data	8	LAD2#	Muxed Address/Data
9	VCC3	+3.3V power	10	LAD1#	Muxed Address/Data

Pin	Сигнал	Комментарий	Pin	Сигнал	Комментарий
11	LAD0#	Muxed Address/Data	12	VSS	Current return
13	SCL	I2C serial clock	14	SDA	I2C serial data
15	VCC3	+3.3V power (was SPDA1 in Intel spec.)	16	SPDA0	Address select for serial EEPROM device (?)

Разъем вентилятора

Разъем вентилятора в Xbox - трехконтактный header, где выводы 1 и 3 подключены к регулирующему по температуре pulse-width-modulation (PWM) контроллеру скорости вентилятора, а вывод 2 подключен к источнику питания +12 volt.

Разъем передней панели

Функции передней панели Xbox, а именно мигающий LED, выключатель питания и eject switch, подключены к материнской плате Xbox через разъем передней панели. Распиновка этого разъема дана в таблице F-8. Распиновка отражает нумерацию выводов разъема на материнской плате Xbox, как видно при взгляде на разъем сверху.

Таблица F-8. Разъем передней панели (вид на материнской плате).

Pin	Комментарий	Pin	Комментарий
1	Ground	2	Power switch
3	Ground	4	Eject switch
5	Green LED	6	Red LED
7	Red LED	8	Green LED
9	Not connected but wired	10	No pin (polarization)

На этом аппаратный справочник Xbox заканчивается.