

The Hardware Hacker. Приключения в создании и взломе аппаратуры

Русский перевод книги The Hardware Hacker о производстве, открытой аппаратуре, реверсе и практическом аппаратном хакинге.

Больше крутых материалов в моём телеграм-канале [@grogrigs](#)

Предварительные материалы

Отзывы о The Hardware Hacker

«Аппаратура, говорит bunnie, - это мир без секретов: если углубиться достаточно далеко, даже самый важный ключ выражен в кремнии или плавких перемычках. Мир bunnie - это мир без тайн, только с неисследованными пространствами. Это взгляд внутрь ума, которому нет равных».

- Эдвард Сноуден

«Tour de force, соединяющий множество гениальных карьер одного из великих хакеров-коммуникаторов мира: практическую, теоретическую, философскую и часто взрывающую мозг».

- Кори Доктороу, автор Little Brother и активист в области технологий

«bunnie живет в мире аппаратуры, там, где припой встречается с печатной платой. У него больше практического опыта, и он лучше объясняет, как работает аппаратная экосистема, чем любой другой человек, которого я встречал, а я знаю в этой области многих. Он превратил этот опыт и экспертизу в потрясающую книгу - библию с хакерской точки зрения для каждого, кто пытается работать в возникающем и развивающемся мире аппаратуры, понимать его и действовать в нем».

- Джои Ито, директор MIT Media Lab

«bunnie - предельный экскурсовод по аппаратному хакингу в его нынешнем виде, с вниманием к возвышенному искусству того, как вещи на самом деле производятся. The Hardware Hacker проведет вас по фабрикам мира, рассматривая и технические, и этические последствия тех "вещей", которые мы производим и покупаем».

- Лимор "ladyada" Фрид, основатель и инженер Adafruit Industries

«Любопытно, как появляются устройства, которыми мы пользуемся каждый день? Хотите изготовить собственный проект? В этой хорошо написанной книге bunnie описывает все тонкости производственного процесса в Китае. Очень увлекательное и информативное чтение».

- Митч Альтман, изобретатель TV-B-Gone

«The Hardware Hacker по своей сути - это букварь для понимания культуры создания вещей в Китае, того, как выпускать их тысячами, и того, почему открытая аппаратура работает».

- Hackaday

Титульная страница

The Hardware Hacker

Andrew "bunnie" Huang

Приключения в создании и взломе аппаратуры

Авторские права и сведения об издании

The Hardware Hacker. Авторское право (с) 2017 Andrew "bunnie" Huang.

Все права защищены. Никакая часть этой работы не может быть воспроизведена или передана в какой-либо форме или какими-либо средствами, электронными или механическими, включая фотокопирование, запись либо любую систему хранения или извлечения информации, без предварительного письменного разрешения правообладателя и издателя.

20 19 18 17 1 2 3 4 5 6 7 8 9

ISBN-10: 1-59327-758-X

ISBN-13: 978-1-59327-758-1

Издатель: William Pollock

Выпускающий редактор: Alison Law

Дизайн обложки и суперобложки: Hotiron Creative

Дизайн внутреннего оформления: Beth Middleworth

Редактор-разработчик: Jennifer Griffith-Delgado

Литературный редактор: Rachel Monaghan

Верстальщик: Alison Law

Корректор: Emelie Burnette

Составитель указателя: BIM Creatives, LLC.

Изображения на следующих страницах воспроизведены с разрешения: страницы 58-59 (с) David Cranor; страница 124 (с) m ss ng p eces; страницы 216, 227-228 (с) Scott Torborg; страница 248 (с) Joachim Strombergson; страницы 253 (внизу) и 254-255 (с) Jie Qi; страница 256 (вверху) (с) Chibitronics; страница 310 (с) Nadya Peek; страница 326 (вверху) из Eva Yus et al., "Impact of Genome Reduction on Bacterial Metabolism and Its Regulation," Science 326, no. 5957 (2009), перепечатано с разрешения AAAS; страница 349 (с) Sakurambo, использовано по лицензии CC BY-SA 3.0.

Интервью на следующих страницах изначально были опубликованы онлайн и воспроизведены с разрешения: страницы 190-204, изначально опубликовано как "MAKE's Exclusive Interview with Andrew (bunnie) Huang - The End of Chumby, New" Phillip Torrone в Make: (April 30, 2012), <http://makezine.com/2012/04/30/makes-exclusive-interview-with-andrew-bunnie-huang-the-end-of-chumby-new-adventures/>; страницы 357-372, изначально опубликовано на китайском как "Andrew "bunnie" Huang: 在 China Software Developer Network (July 3, 2013), <http://www.csdn.net/article/2013-07-03/2816095>; страницы 372-382, изначально опубликовано как "The Blueprint Talks to Andrew Huang" в The Blueprint (May 15, 2014), <https://theblueprint.com/stories/andrew-huang/>.

За информацией о распространении, переводах или оптовых продажах обращайтесь непосредственно в No Starch Press, Inc.:

No Starch Press, Inc.

245 8th Street, San Francisco, CA 94103

телефон: 1.415.863.9900; info@nostarch.com; www.nostarch.com

Каталогизационные данные Библиотеки Конгресса

Имена: Huang, Andrew, автор.

Название: The Hardware Hacker: приключения в создании и взломе аппаратуры / Andrew "bunnie" Huang.

Описание: 1-е изд. | San Francisco: No Starch Press, Inc., [2017]

Идентификаторы: LCCN 2016038846 (печать) | LCCN 2016049285 (ebook) | ISBN 9781593277581 (pbk.) | ISBN 159327758X (pbk.) | ISBN 9781593278137 (epub) | ISBN 1593278136 (epub) | ISBN 9781593278144 (mobi) | ISBN 1593278144 (mobi)

Темы: LCSH: электронные аппараты и приборы -- проектирование и конструирование. | Электронные аппараты и приборы -- технологические инновации. | Компьютерное оборудование ввода-вывода -- проектирование и конструирование. | Реверс-инжиниринг. | Электронная промышленность. | Huang, Andrew.

Классификация: LCC TK7836 .H83 2017 (печать) | LCC TK7836 (ebook) | DDC 621.381092--dc23

Запись LC доступна по адресу <https://lcn.loc.gov/2016038846>

No Starch Press и логотип No Starch Press являются зарегистрированными товарными знаками No Starch Press, Inc.

Другие названия продуктов и компаний, упомянутые здесь, могут быть товарными знаками соответствующих владельцев. Вместо того чтобы использовать символ товарного знака при каждом появлении охраняемого названия, мы используем эти названия только в редакционных целях и в интересах владельца товарного знака, без намерения нарушить права на товарный знак.

Информация в этой книге распространяется на условиях "As Is", без гарантии. Хотя при подготовке этой работы были приняты все меры предосторожности, ни автор, ни No Starch Press, Inc. не несут ответственности перед каким-либо лицом или организацией за любые потери или ущерб, причиненные или предположительно причиненные прямо или косвенно информацией, содержащейся в ней.

Посвящение

Всем замечательным, терпеливым и принимающим людям, которые поддерживали этого чудаковатого хакера.

Благодарности

Спасибо всем трудолюбивым сотрудникам No Starch Press за то, что помогли этой книге состояться. В особенности спасибо Биллу Поллоку за то, что он придумал и поддержал этот труд, и спасибо Jennifer Griffith-Delgado за то, что она собрала, отредактировала и выстроила мои тексты в форму этой книги.

Предисловие

Когда Билл Поллок, основатель No Starch Press, впервые обратился ко мне с идеей издать сборник моих текстов, я отнесся к этому скептически. Я не думал, что материала хватит хотя бы на сотню страниц. Похоже, я ошибался.

Моя мать часто говорила: «Неважно, что у тебя в голове, если ты не можешь рассказать людям, что именно там находится», и, когда я учился в седьмом классе, она записала меня на внеклассные занятия по написанию эссе. Тогда я ненавидел эти занятия, но, оглядываясь назад, благодарен ей. Начиная с эссе для поступления в колледж и до сегодняшнего дня способность приводить свои мысли в связную прозу была для меня бесценной.

Большая часть материалов этой книги изначально была опубликована в моем блоге, но, как вы вскоре увидите, эти записи не были рекламными текстами, написанными ради дохода от объявлений. Одна из причин, по которым я пишу, состоит в том, чтобы укрепить собственное понимание сложных тем. Легко верить, что ты понимаешь предмет, пока не попробуешь строго объяснить его кому-то другому. Письмо - это способ перегнуть интуицию в структурированное знание; я пишу только тогда, когда нахожу что-то интересное для письма, а затем публикую это под лицензией CC BY-SA, чтобы побудить других делиться материалом.

В эту книгу вошла подборка моих текстов о производстве, интеллектуальной собственности (с акцентом на сопоставлении западного и китайского взглядов), открытой аппаратуре, обратной разработке, а также биологии и биоинформатике. Хорошие редакторы No Starch Press также отобрали пару интервью, которые я давал в прошлом и которые оказались особенно информативными или глубокими. Общая нить, проходящая через все эти разные темы, - аппаратура: как ее создают, какие правовые рамки ее окружают и как ее разбирают. И да, биологические системы - это тоже аппаратура.

Меня всегда тянуло к аппаратуре, потому что, хотя я не особенно одарен в абстрактном мышлении (отсюда и потребность писать, чтобы упорядочивать мысли), я довольно хорошо работаю руками. У меня куда больше шансов понять вещи, которые я могу увидеть собственными глазами.

Все мое понимание мира всегда строилось на серии простых физических переживаний, начиная с детства, когда я складывал кубики и опрокидывал их. В этой книге я делюсь некоторыми из своих более поздних переживаний. Надеюсь, читая о них, вы глубже поймете мир аппаратуры без необходимости десятилетиями складывать кубики и опрокидывать их.

Удачного хакинга,

• b.

Часть 1. Приключения в производстве

Глава 1. Сделано в Китае

До своей первой поездки в Китай я был убежден, что Акихабара в Токио - главное место, куда нужно ехать за самой новой электроникой, безделушками и компонентами. Все изменилось в январе 2007 года, когда я впервые увидел SEG Electronics Market в Шэньчжэне. SEG - это восемь этажей любых компонентов, каких только может пожелать аппаратный энтузиаст, и лишь позже я узнал, что это всего лишь верхушка айсберга электронного района Hua Qiang.

В то время я был ведущим инженером по аппаратуре в Chumby и приехал в Китай вместе с тогдашним CEO Steve Tomlin, чтобы понять, как производить chumbys (устройства доставки контента с открытым исходным кодом и Wi-Fi) дешево и вовремя. С такими ценами, как в SEG, мы определенно оказались в правильной стране, чтобы сделать успешной по крайней мере первую часть этой миссии.





SEG Electronics Market в Шэньчжэне - новая электронная мекка. Акихабара, завидуй молча!

Огромный блошиный рынок электронных компонентов

Когда я впервые вошел в здание SEG, меня накрыл вихрь электронных компонентов: ленты и катушки резисторов и конденсаторов, микросхемы всех типов, индуктивности, реле, тестовые точки с подпружиненными контактами, вольтметры и лотки с микросхемами памяти. Как полный новичок в массовом производстве, я был поражен всем, что увидел в SEG.

Все эти детали были втиснуты в крошечные киоски шесть на три фута, в каждом из которых продавец что-то делал на ноутбуке. Одни продавцы играли в го, другие считали детали. Некоторые киоски были настоящими семейными лавками: матери присматривали за младенцами, а дети играли в проходах.



Пара семейных лавок компонентов.

Другие киоски представляли собой профессионально устроенные точки с персоналом в униформе, и они работали как бар - со стульями и всем прочим, только для электронных компонентов.



Шикарный профессиональный продавец деталей.

В SEG никто не говорит: «О, вы можете купить 10 таких светодиодов или пару таких реле», как можно услышать в Акихабаре. Нет-нет. Эти киоски специализируются, и если вы видите компонент, который вам нравится, обычно можно купить несколько туб, лотков или катушек с ним; можно взять достаточно, чтобы уже на следующий день запустить изделие в производство.

Оглядывая рынок, я увидел женщину, сортировавшую стопки mini-SD карт емкостью 1 Гбайт, словно фишки для покера. Мужчина укладывал планки памяти Kingston на 1 Гбайт в розничные упаковки, а рядом с ним девушка считала резисторы.



Нижний левый угол этой витрины был плотно набит всевозможными SD-картами.

В другом киоске лежали стопки источников питания, варисторов, батарей и программаторов ROM, а еще в одном были микросхемы самых разных видов: Atmel, Intel, Broadcom, Samsung, Yamaha, Sony, AMD, Fujitsu и другие. Некоторые микросхемы явно были выпаяны из бывшего в употреблении оборудования и перемаркированы, а некоторые лежали в совершенно новой OEM-упаковке с лазерной маркировкой.



Одно только количество микросхем, продававшихся в одном киоске SEG, было невероятным.

Я видел микросхемы, которые никогда не смог бы купить в Соединенных Штатах, катушки редких керамических конденсаторов, о которых мог только мечтать по ночам. Чувства обострились; голова пошла кругом. Я не мог сдержать улыбку предвкушения, когда повернул за следующий угол и увидел лавки, от пола до потолка заваленные, вероятно, сотней миллионов резисторов и конденсаторов.



Катушки и еще катушки компонентов в каждой витрине.

Sony CCD и CMOS элементы камер! Я не смог бы купить их в Соединенных Штатах, даже если бы вырывал зубы торговым представителям. (У некоторых продавцов за прилавком даже есть технические описания; всегда спрашивайте.) Затем я заметил стопку микросхем регуляторов Micrel, а после нее - выставленный на продажу DSP-чип Blackfin. Неподалеку женщина считала DRAM-микросхемы на 256 Мбит: лотки по 108 компонентов, уложенные по 20 в высоту, примерно в 10 рядов.



Эквивалент всего склада DRAM-микросхем Digi-Key лежал прямо передо мной!

А напротив нее стояло еще с полдюжины маленьких лавок, набитых микросхемами, как и ее лавка. В одной точке мужчина гордо стоял над лотком NAND-флеш-микросхем емкостью 4 Гбит. Все это можно было получить за небольшой торг, немного наличных и поспешное прощание.



Крупный план лотка с флеш-микросхемами 4 Гбит.

И это только первые два этажа SEG. Есть еще шесть этажей компьютерных компонентов, систем, ноутбуков, материнских плат, цифровых камер, камер безопасности, USB-накопителей, мышей, видеокамер, высококлассных графических карт, плоских дисплеев, уничтожителей бумаги, ламп, проекторов - чего угодно. По выходным там слоняются промо-девушки в нелепых блестящих боди с брендом Acer и пытаются заманить вас купить их товар. У этого рынка энергия круглогодичного CES, скрещенного с Computex, только вместо простого показа новейших технологий цель в том, чтобы затащить вас в эти киоски и продать вам эту аппаратуру. Выставки всегда немного напоминают стриптиз: вы зависаете над недоступными товарами под стеклом, а ваше дыхание оставляет на нем призрачные кольца.

Но SEG - не стриптиз. Это оргия потребительских и промышленных закупок электроники, где можно заполучить почти любую единицу оборудования, если у вас в кошельке достаточно kuai[1]. Запахи, суэта, торговый напор - все вместе делает SEG предельным блошиным рынком электронных компонентов. Как если бы Digi-Key сошел с ума, пустил обезьян на свой склад в Миннесоте, а возникший хаос выплеснулся на блошинный рынок в Китае.

Конечно, многие детали, которыми я восхищался в 2007 году, теперь уже антиквариат. Например, флеш-микросхемы на 4 Гбит - мусор, а флеш-диски на 1 Гбайт - вчерашний день. Но тогда эти вещи были большим делом, и SEG по-прежнему остается лучшим местом, где можно оптом достать новейшие технологии.

Следующая технологическая революция

В трех кварталах вниз по улице от SEG находился Shenzhen Bookstore[2]. Первая и самая заметная стойка была отделом иностранных книг, набитым классикой вроде The Design of CMOS Radio-Frequency Integrated Circuits профессора Stanford University Thomas Lee и несколькими книгами профессора UCLA Behzad Razavi. Я взял книгу Lee, и она стоила 68 kuai, то есть 8,50 доллара США. Ничего себе! Книга Jin Au Kong об уравнениях Maxwell? 5 долларов. Jin Au Kong преподавал мне уравнения Maxwell в MIT.

Я пустился во все тяжкие и набил сумку шестью или семью книгами, вероятно, примерно на 700 долларов, если бы покупал их в Соединенных Штатах. На кассе я купил их меньше чем за 35 долларов вместе с дополнительными CD, сэкономив около 665 долларов. Это эквивалент покупки билета экономкласса до Гонконга!

В Китае знания дешевы. Компоненты дешевы. Знания в книгах Shenzhen Bookstore были настоящими, детали для применения этих знаний находились ниже по улице в SEG, а в пределах часа езды на север, вероятно, было 200 фабрик, способных взять любую идею электроники и буквально выпускать ее корабельными партиями. И это не какие-то отсталые фабрики. Своими глазами я видел, как там собирали и тестировали фирменные одномодовые магистральные оптоволоконные трансиверы на 1550 нанометров. Шэньчжэнь - плодородная почва, и его нужно увидеть, чтобы понять.

Шэньчжэнь ощущается как радиолюбительская барахолка в Silicon Valley восьмидесятых, когда все крупные компании только основывались и начинали работу, только это усилено 25 годами прогресса закона Мура и скорости потока информации через интернет. В этом городе с населением 12 миллионов человек большинство связано с технологиями или производством, многие учат английский, и все готовы много работать.

Где-то там должны быть свои Jobs и Wozniak, тихо создающие следующую революцию. Но я тоже часть Шэньчжэня, и меня до сих пор пробивает дрожь ужаса и восторга при мысли о том, что я являюсь частью этой революции. Это моя история, начинающаяся с той открывшей глаза поездки в Шэньчжэнь ради Chumby.

Экскурсия по фабрикам с Chumby

В сентябре 2006 года Chumby была командой всего примерно из полудюжины человек, и мы только что раздали около 200 ранних прототипов устройств chumby на FOO Camp, конференции, организованной Tim O'Reilly. Участники FOO Camp хорошо приняли устройства, так что я получил добро на создание азиатской цепочки поставок.

В ноябре мы со Steve отправились в Китай, чтобы посетить потенциальные фабрики, но перед отъездом попросили надежного поставщика в Соединенных Штатах назвать свою лучшую цену за эту работу как базовый ориентир для переговоров с китайскими производителями. Затем мы обзвонили множество друзей с опытом работы в Китае и договорились примерно о шести экскурсиях по фабрикам. Мы попали в довольно разные места: от специализированных фабрик размером всего 500 человек до мегафабрик с более чем 40 000 человек.

Поездку в Китай для экскурсии по фабрике ничем заменить нельзя. Фотографии могут рассказать только историю, выстроенную фотографом в кадре, и невозможно почувствовать масштаб и качество предприятия, не увидев его своими глазами. В целом фабрики охотно показывают производство, и я не стал бы работать с фабрикой, которая не позволила бы мне приехать. Однако большинство фабрик ценят предупреждение за неделю, хотя по мере развития отношений с ними все должно становиться более открытым и прозрачным.

Кстати об открытости: природа Chumby как проекта с открытым исходным кодом очень помогла процессу выбора фабрики. Во-первых, мы не боялись, что кто-то украдет наш дизайн (мы уже отдавали его бесплатно), поэтому устранили трение вокруг NDA (соглашений о неразглашении) при обмене критически важной информацией вроде ведомости материалов. Думаю, благодаря этому фабрики в Китае лучше к нам относились; они, казалось, охотнее открывались нам, потому что мы были готовы открываться им. Во-вторых, ни у одной фабрики не было сомнений, что ситуация конкурентная. Любой мог и стал бы давать цену и участвовать в торгах за нашу работу.

(На самом деле мы получили несколько незапрошенных коммерческих предложений, и они были вполне конкурентоспособными), так что это сэкономило один раунд возмущенного пыхтения.

После рассмотрения нескольких производственных вариантов мы со Steve в итоге решили работать с компанией PCH China Solutions. Сама PCH владеет лишь несколькими предприятиями, но у нее есть всеобъемлющая сеть доверенных и проверенных поставщиков, главным образом в Китае, но также в Европе и Соединенных Штатах. Неудивительно, что фабрики, которым PCH передавала работу на субподряд, были одними из лучших предприятий, которые мы посетили в Китае. Фактически штаб-квартира PCH находится в Ирландии - поэтому большинство ее штатных инженеров ирландцы, - так что языкового барьера для нас тоже не было. (Инженеры PCH также трудолюбивы, находчивы и хорошо обучены, а в качестве бонуса они, похоже, всегда знают, где лучше всего найти пинту, где бы ни находились. Я и не представлял, что в Китае так много кранов Guinness!)

Во время экскурсии даже по одной фабрике нужно воспринять очень многое, не говоря уже о полудюжине, и легко перегрузиться и потеряться в зыбких подробностях производства электроники. Но во время моих фабричных туров для Chumby и в работе с PCH над воплощением chumby в жизнь были некоторые ключевые детали, которые я нашел особенно захватывающими.

Масштаб в Шэньчжэне

Одна из ошеломляющих сторон работы в Китае - сам масштаб этого места. Я не был на автомобильном заводе в Michigan и на заводе Boeing в Seattle, но у меня есть ощущение, что по масштабу Шэньчжэнь способен дать фору и тому и другому. В 2007 году в Шэньчжэне было 9 миллионов человек.

Чтобы дать представление о масштабе шэньчжэньской фабрики: местная фабрика New Balance нанимала 40 000 человек и могла производить более миллиона пар обуви в месяц. По моей оценке, путь от сырой ткани до готовой обуви занимал около 50 минут, и каждый идеально прошитый пучок пластика и кожи сшивался вручную на промышленной швейной машине. Рабочие места устроены так, чтобы каждый этап процесса занимал у работника около 30 секунд.

Конечно, фабрика New Balance выглядит маленькой рядом с Foxconn, фабрикой, где делают iPod и iPhone.



Вы понимаете, что стали большими, когда у вас есть собственный съезд с автомагистрали.

Foxconn - гигантское предприятие, по-видимому, с более чем 250 000 сотрудников, и у него есть собственный особый статус свободной торговли. Все предприятие обнесено стеной, и я слышал, что, чтобы попасть внутрь, нужно показать паспорт и пройти таможенно. До ядерных робопсов из корпоративно-государственных анклавов в «Лавине» Нила Стивенсона остается совсем немного.

Кормить фабрику

Есть старая китайская поговорка: *min yi shi wei tian*. Буквальный перевод был бы таким: «люди считают пищу божественной» или «для людей еда близка к небесам». Можно смотреть на нее и как на совет правителю: «мандат правительства [синоним небес] крепок лишь настолько, насколько крепка пища на тарелках людей». Или можно истолковать ее как оправдание прокрастинации: «сначала поедим [ведь это важно, как небеса]».

Как ни поверни, мне кажется, что эта поговорка в Китае по-прежнему работает. Один важный показатель того, насколько хорошо фабрика обращается с сотрудниками, - качество еды, потому что фабричных рабочих обычно размещают, кормят и обслуживают прямо на территории.

На некоторых фабриках еда действительно довольно хорошая. Например, когда я ел вместе с рабочими на фабрике, производившей печатные платы *chumby*, мне подали смесь из рыбы на пару, запеченной свинины, яичных рулетов, чистых жареных овощей и какого-то сочетания маринованных овощей с мясом. Рис, суп и яблоки также давали в количестве «бери сколько хочешь».



Обед с фабрики, которая производила печатные платы *chumby*.

На каждом предприятии, которое я посетил, для гостей также были отдельные приборы и тарелки. На одной фабрике мою еду подали на тарелке из Styrofoam с одноразовыми палочками, тогда как фабричный рабочий, с которым я ел, получил еду на стальной тарелке со стальными палочками. Я не проходил медицинский осмотр этой фабрики, поэтому мне выдали одноразовые инструменты для еды, чтобы я не занес на фабрику потенциальные чужеземные болезни.

Возвращаясь к масштабу: некоторые фабричные пищевые операции впечатляюще велики. Я слышал, что рабочие Foxconn потребляют 3000 свиней в день. От свиней до iPhone - все это происходит прямо здесь, в Шэньчжэне!



Грузовик со свиньями съезжает с автомагистрали в сторону Foxconn.

Преданность качеству

После того как я начал работать с PCN над фактическим производством shumby, примерно в июне 2007 года я столкнулся с ситуацией, которая показала мне, насколько шэньчжэньские фабричные рабочие преданы правильному выполнению своей работы.

Я обновил материнскую плату shumby, добавив электретный микрофон со встроенным предварительным усилителем на полевом транзисторе (FET). Микрофон нужно было вставить в правильной ориентации относительно схемы, чтобы FET получал надлежащий ток смещения.

Первые образцы, которые я получил с фабрики PCN, имели микрофон, установленный наоборот, и я позвонил на фабрику, чтобы попросить изменить его полярность. На следующей неделе я собирался приехать на фабрику и хотел увидеть исправленные образцы. Когда я приехал и проверил микрофон, к своему огорчению обнаружил, что микрофоны все еще не работают.

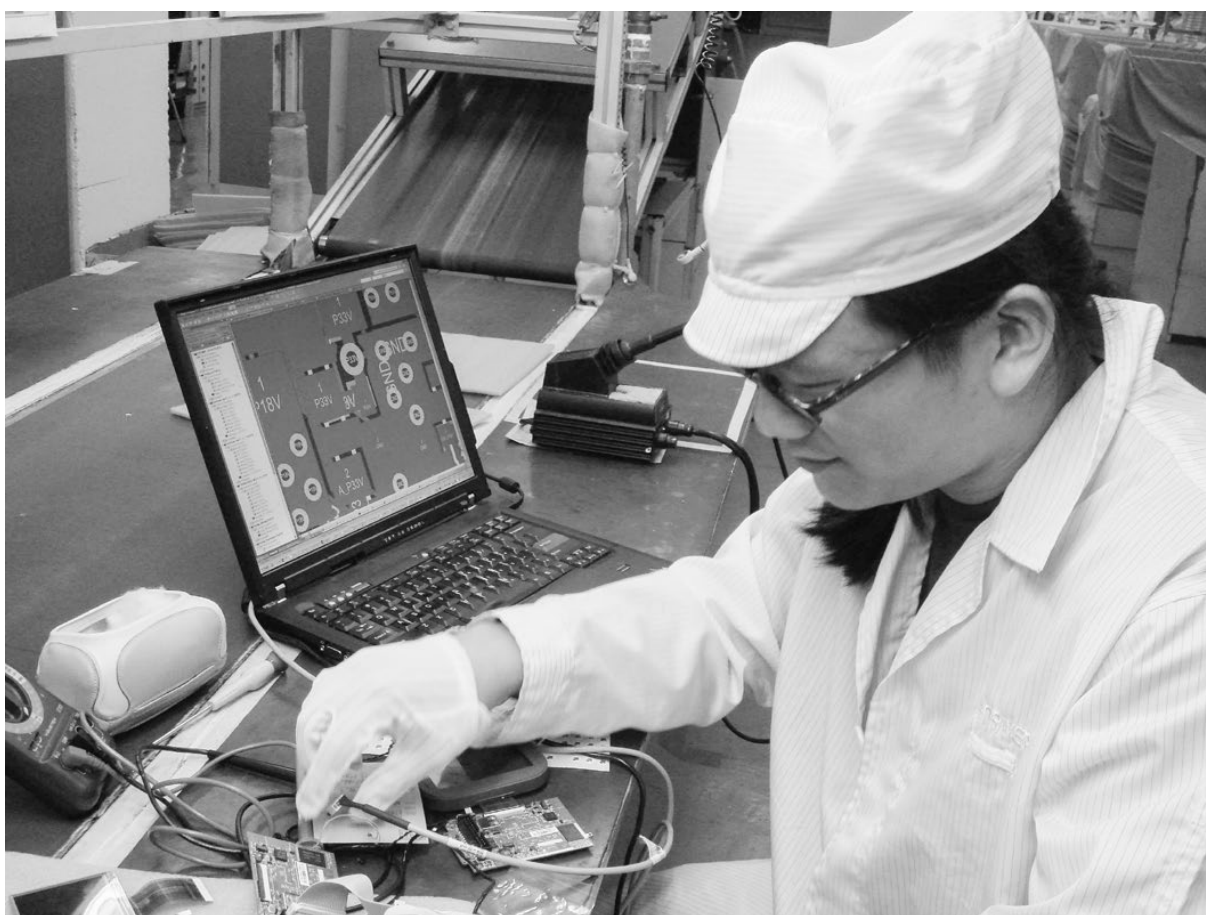
Как такое возможно? Есть только два способа подключить микрофон.

Оказалось, что на линии микрофон собирали два оператора. Один припаивал красный и черный провода к микрофону. Следующий припаивал эти красный и черный провода к печатной плате. Операторам сказали поменять порядок, и оба добросовестно выполнили указание - в результате я получил микрофон, который все еще был припаян наоборот, но цвета проводов поменялись местами. (Вообще-то это довольно типичная история для проблем в Китае.)

На следующий день фабрика должна была изготовить первый пилотный тираж из 450 печатных плат. Все должно было пройти идеально, чтобы производственный график Chumby остался по расписанию. Мы заново изготовили паяльные трафареты (мы также отлаживали проблему выхода годных изделий с аудиокодеком в корпусе QFN), и примерно к полудню они были готовы; около 18:00 у меня в руках были первые платы для тестирования. Я запустил финальный фабричный тест, и устройство снова отказало - на микрофоне. Для всех на фабрике это был совсем не счастливый момент, поскольку фабрика несла ответственность за любые производственные дефекты.

Я надел халат и отправился на линию, чтобы начать отлаживать проблему.

Остаток ночи я провел на фабрике, и вместе со мной там оставался каждый менеджер и каждый техник, участвовавший в производстве chumby. Давление было огромным: прямо рядом с нами линия выпускала 450 потенциально дефектных печатных плат, а я не хотел дергать стоп-кран, потому что все еще не знал первопричины и нам нужно было оставаться в графике.



Я отлаживал схемы в 3 утра в день финального фабричного теста chumby.

Буквально всю ночь рядом со мной дежурила группа фабричных работников, готовых принести мне все, что понадобится: паяльники, измерительное оборудование, новые платы, рентгеновские аппараты, микроскопы. Примечательно, что ни один человек не колебался; ни один человек не жаловался; ни один человек не терял фокуса на проблеме. Люди без малейших колебаний

отменяли планы ужина с друзьями. Все, кто не был нужен в конкретный момент, занимались наблюдением за другими аспектами проекта. Я не видел такой слепой преданности делу с тех пор, как работал с командой автономной подводной робототехники в MIT.

И это продолжалось до 3 утра.

В конце концов, к моему смущению, проблема оказалась не ошибкой PCN. Проблема была в новой версии прошивки, которую я получил ранее в тот же день от команды в Соединенных Штатах. В ней был баг, отключавший микрофон из-за хака, случайно внесенного в дерево сборки.

Еще более впечатляюще то, что, когда PCN об этом узнала, никто не рассердился и никто не жаловался. (Ну, продавщица устроила мне взбучку, но я это заслужил: она была достаточно добра, чтобы всю ночь сопровождать меня на производственной линии и быть моим переводчиком, поскольку мой мандарин был недостаточно хорош.) Все просто испытали облегчение, что это была не их вина.

Мы разошлись, и на следующий день в 11 утра, хорошо выспавшись, я вернулся на фабрику. Я увидел Christy, проектного менеджера фабрики по производству плат chumby. Я спросил, во сколько она пришла на работу, и она сказала, что всегда должна являться к 8 утра. Мне стало по-настоящему неловко: Christy засиделась допоздна из-за нашего бага, а утром пришла рано, пока я отсыпался. Я спросил ее, почему она оставалась так поздно, хотя знала, что должна быть на работе к 8 утра. Она могла уйти домой, и мы продолжили бы на следующий день.

Она просто улыбнулась и сказала: «Моя работа - проследить, чтобы это было сделано, и я хочу хорошо выполнить работу».

Создавать технологию, не используя ее

Вот еще одна интересная история. Однажды, когда мы выходили из фабричного цеха, Xiao Li (менеджер по обеспечению качества на фабрике, где мы делали chumby) спросила меня: «Что делает chumby?» Я не очень хорошо говорил по-китайски, и она тоже не очень хорошо говорила по-английски, поэтому я решил начать с нескольких базовых вопросов.

Я спросил ее, знает ли она, что такое World Wide Web. Она сказала, что нет.

Я спросил ее, знает ли она, что такое интернет. Она сказала, что нет.

Я был ошеломлен и не знал, что сказать. Как описать синий цвет слепому?

Xiao Li была экспертом в сборке и тестировании компьютеров. На некоторых проектах она, вероятно, собирала PC и загружала Windows XP сотни тысяч раз снова и снова. (Одному богу известно, сколько раз я слышал этот проклятый звук запуска во время истории с микрофоном, потому что прямо рядом со мной стояла группа станций финального теста для материнских плат ASUS.) Но она не знала, что такое интернет.

Я предполагал, что если человек прикасается к компьютеру, то он также благословлен дарами интернета. В один миг я почувствовал себя избалованным снобом и свиньей за то, что забыл: Xiao Li, вероятно, не могла позволить себе компьютер, не говоря уже о широкополосном доступе в интернет. Получи она возможность, ей, конечно, хватило бы ума всему этому научиться, но она была слишком занята зарабатыванием денег, которые, вероятно, отправляла домой семье.

В итоге лучшее, что я смог сделать, - сказать Xiao Li, что chumby - устройство для игр.

Квалифицированные рабочие

Рабочие Шэньчжэня могут знать не так уж много обо всем, что они производят, но, помимо преданности делу, они обладают высокой квалификацией. Однажды я наблюдал за парнем, работавшим на той же фабрике, где шили сумки для chumby, и, клянусь, он мог сшивать косметички со скоростью 5 секунд на сумку. И он даже не был на 100 процентов сосредоточен на своей задаче: во время шитья он слушал iPod.

И, судя по всему, он не был их самым быстрым сотрудником! У них был человек примерно вдвое быстрее, и он работал в компании около семи лет. Я пошел посмотреть на более быстрого рабочего, но тот уже ушел на обед, потому что все закончил; рядом с его рабочим местом стояли два огромных контейнера готовых косметичек.

В покое ключе меня поразило, когда я узнал, как в Китае делают прорезиненные бирки (те самые, которые вы видите на одежде повсюду). Я всегда думал, что их прессуют машиной, но ошибался. Все эти слова, цвета и буквы рисуются вручную. Человек просто кладет трафарет логотипа поверх пустой бирки, с удивительной точностью закрашивает его и переходит к следующей бирке в очереди. Когда цветов несколько, для каждого цвета есть отдельный человек, чтобы процесс шел быстро.

Я спросил РСН, есть ли у них механизированные фабрики для подобных вещей. Они сказали, что такие предприятия существуют, но минимальный объем заказа огромен (сотни тысяч, иногда миллионы) из-за чрезвычайно низкой себестоимости продукта и относительно высокой стоимости оснастки для автоматизированного процесса. Это согласуется с тем, что я слышал об игрушках McDonald's Happy Meal. Их обычно скрепляют винтами, потому что дешевле заплатить человеку за завинчивание игрушки на протяжении всего производственного тиража, чем изготовить стальную пресс-форму для литья под давлением с допусками, необходимыми для защелкивания игрушек[3].

В аппаратуре chumby был похожий компромисс. На внутренней электронике chumby было четыре разъема. У поставщиков из США, у которых я мог закупаться, лучшая цена одного разъема составляла около 1 доллара США, а трех других - около 0,40 доллара за каждый. Очень талантливый специалист РСН по поиску поставщиков (ее репутации боялись и уважали все поставщики) сумела найти для меня разъемы стоимостью соответственно 0,10 и 0,06 доллара, сэкономив почти полные 2 доллара себестоимости. Есть одна загвоздка: у разъемов не было жертвенной пластиковой площадки автоматической установки компонентов, которая позволила бы устанавливать их машиной.

Решение? Конечно, человек.



Этот человек вручную устанавливал более дешевые разъемы на каждый chumby примерно за пять центов за штуку. Благодаря ему chumby стали на 2 доллара дешевле, что освободило нам, потребителям, больше денег для Starbucks.

Потребность в мастерах

Я хотел бы представить вам человека, которого знаю просто как мастера Chao. Я познакомился с ним в процессе производства chumby, и я почти уверен, что за свою жизнь вы пользовались или видели что-то, созданное им.

Когда я пришел в комнату образцов на фабрике, где работал мастер Chao, я был потрясен тем, как много предметов на их полке я сам покупал, использовал или видел в магазине в Соединенных Штатах. Потребительские бренды высшего уровня производят свои вещи на этой фабрике, и, насколько мне известно, в то время на фабрике был всего один главный лекальщик: мастер Chao. Он приложил руку к созданию косметичек для Braun, чехлов для аксессуаров Microsoft и медицинских биндажей для крупных брендов, продаваемых в аптеках, среди множества других продуктов.



Мастер Chao - человек на переднем плане; на заднем плане Joe Perrott, отличный проектный инженер Chumby из PCH China Solutions.

Мастер Chao - ремесленник в традиционном смысле. Когда-то лучшую мебель проектировали и строили только с помощью интуиции и умения мастера. Теперь мы все идем в IKEA и покупаем CAD-спроектированные, управляемые цепочкой поставок наборы мебели для сборки по картинкам - и, несмотря на все это, выглядит она не так уж плохо. В результате слово "ремесло" было оттеснено к описанию какого-нибудь набора для скрапбукинга или вышивания, который покупаешь в Michaels и собираешь в медленные выходные. Мы забыли, что в эпоху до машин ремесло было единственным способом построить что-либо качественное.

Однако оказывается, что традиционное мастерство по-прежнему важно, потому что CAD-инструменты не принесли нам способности моделировать наши ошибки до того, как мы их совершим.

Создание плоской выкройки для текстильных изделий - хороший пример процесса, которому требуется мастер. Плоская выкройка - это набор двумерных форм, используемых для раскроя тканей. Эти формы вырезают, сгибают и сшивают в сложную трехмерную форму. Отобразить проекцию произвольной трехмерной формы на двумерную поверхность с минимальной площадью отходов между деталями уже достаточно трудно. А тот факт, что материал растягивается и деформируется, иногда в разных направлениях, и что шитье требует достаточных допусков для хорошего выхода годных изделий, превращает создание выкройки в проблему, которую трудно автоматизировать.

Чехлы chumby добавляли еще один уровень сложности, потому что предполагали пришивание куска кожи к мягкой пластиковой рамке. В такой ситуации, когда вы пришиваете кожу, рамка слегка

деформируется и растягивает кожу, создавая швейное смещение, зависящее от направления и скорости шитья. Эта сила фиксируется в швах и вносит вклад в окончательную форму чехла. Я предлагаю кому-нибудь создать инструмент компьютерного моделирования, который сможет точно учесть эти силы и предсказать, как будет выглядеть такой продукт после сшивания.

И все же каким-то образом мастерство Chao в искусстве построения выкроек позволило ему очень быстро и за очень малое число итераций создать и подстроить выкройку, компенсировавшую все эти силы. Его результаты, полученные только с помощью картона, ножниц и карандашей, были поразительно умными и проницательными. Будьте благодарны за его навыки старого мира; вероятно, они сыграли роль в производстве чего-то, чем вы пользовались или от чего получили пользу.



В офисе мастера Chao не было ни одного компьютера, но продукты, которые я видел здесь, окружали широкий набор высокотехнологичных устройств.

Автоматизация сборки электроники

До работы в Chumby я думал, что почти все производится машиной. Конечно, экскурсии по текстильным фабрикам очень быстро исправили мое впечатление; однако высокотехнологичные вещи вроде сборки электроники все же, как правило, сильно автоматизированы, даже в Китае. Единственными исключениями, которые я увидел во время фабричных туров, были, по иронии, самые дешевые продукты, например игрушки. В этих цехах по-прежнему доминировали линии рабочих, вручную вставлявших компоненты и паивавших платы погружением.

Одна интересная дихотомия, связанная с автоматизацией, - бимодальное распределение продуктов, использующих технологию chip-on-board (CoB). Сборка CoB напрямую соединяет кремниевый кристалл с PCB. Готовые сборки CoB имеют характерный вид «капли эпоксидной смолы», в отличие от готового вида пластикового корпуса. Высококласные плотные электронные сборки часто используют технологии CoB. В свое время я сделал пару CoB-разработок для некоторых оптических трансиверов на 10 Гбит, и они были недешевы. Однако одновременно почти все игрушки используют технологию CoB, чтобы устранить стоимость корпуса микросхемы! Это свидетельство упорства игрушечных фабрик в снижении затрат: они готовы купить автоматический

проволочный бондер и поставить его рядом с линиями, формирующими головы кукол и сшивающими мягкие игрушки, потому что собственный бондер экономит пять центов.

Типичный проволочный бондер присоединяет провод толщиной с человеческий волос к площадке на кремниевом кристалле, ненамного превышающей диаметр этого провода, и делает это несколько раз в секунду. Проволочные бондеры - очень быстрые и точные машины. Соединение происходит так быстро, что плата кажется плавно поворачивающейся, но на самом деле при вращении она останавливается 16 раз, и на каждой остановке между микросхемой и платой присоединяется провод.

Однако непосредственно перед бондингом микросхему очень аккуратно приклеивают к плате вручную, а сразу после бондинга оператор-человек инкапсулирует микросхему, очень аккуратно нанося эпоксидную смолу вручную. Это означает, что проволочный бондер - единственная автоматизированная машина на сборочных линиях простых игрушек. Увидев этот процесс, я по-новому оценил, что вложено в тех говорящих кукол Barney, которые продаются за 10 долларов в Target.

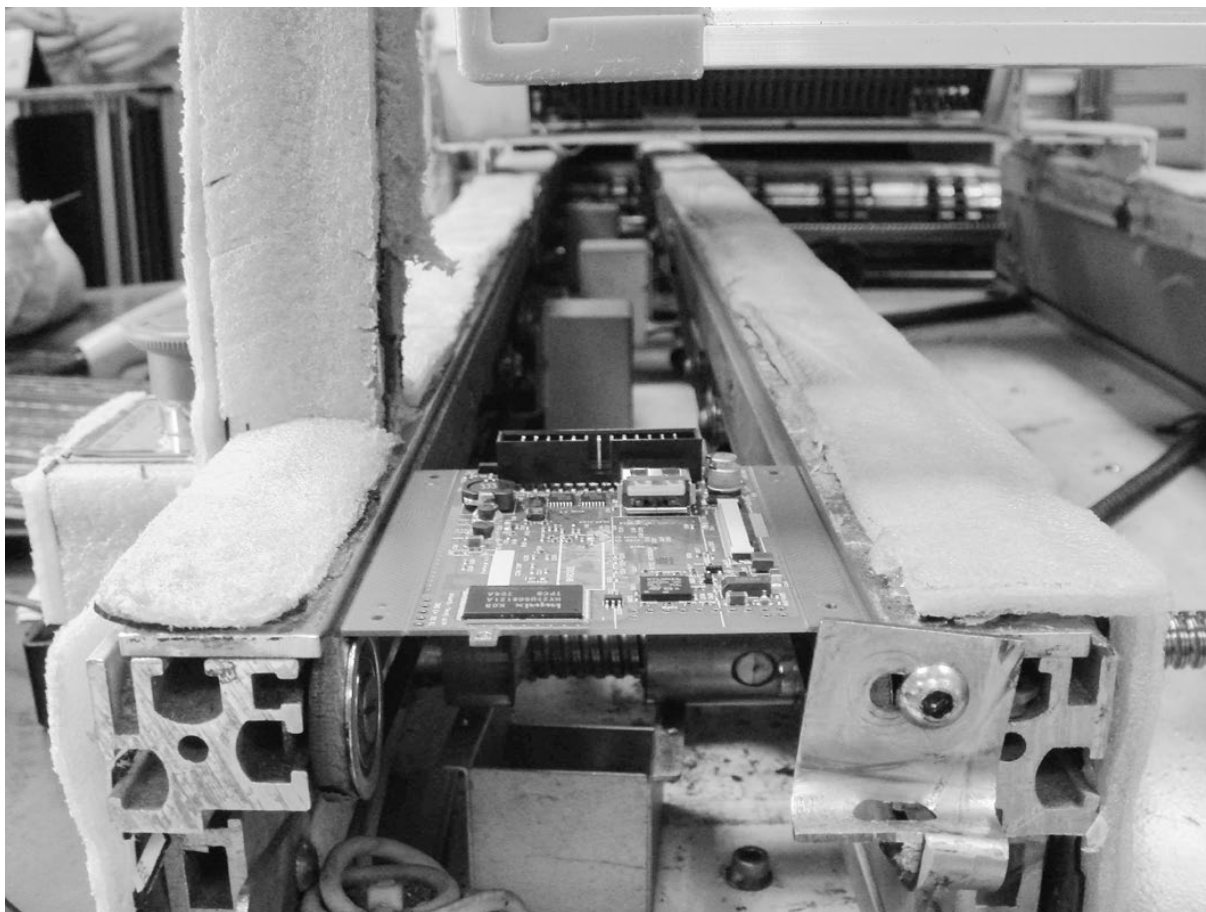
Процесс производства chumby тоже использовал немного автоматизации благодаря автоматам установки микросхем. Автоматы установки микросхем (как и автоматы установки компонентов) устанавливают компоненты поверхностного монтажа на РСВ, чтобы затем эти компоненты можно было припаять.



Китайская фабрика сборки РСВ для chumby имела десятки линий, заполненных проверенными временем автоматами установки микросхем Fuji.

Наблюдать автомат установки микросхем в работе абсолютно завораживает. Автоматы установки микросхем на фабрике сборки РСВ для chumby могли устанавливать от 10 000 до 20 000 компонентов в час на машину. Это означает, что каждая машина могла ставить от 3 до 6 компонентов в секунду. Роботизированные узлы движутся быстрее, чем успевают увидеть глаз, и

все превращается во внушающий трепет размытый поток. Установщик компонентов, который я видел на фабрике chumby, работал примерно как пулемет Gatling: сама «пушка» компонентов была неподвижна, а плата танцевала под ней. Установщик фактически «смотрел» на каждый компонент и поворачивал его в правильную ориентацию, прежде чем опустить на плату.



Это конец линии сборки основной платы chumby!

Фабрика, которую мы использовали для сборки PCB chumby, также производила фирменные материнские платы PC и, похоже, без проблем выпускала значительно больше 10 000 таких сложных сборок каждый день. Но даже если процессы вроде установки компонентов можно автоматизировать, есть вещи, которые машина просто не может сделать.

Точность, литье под давлением и терпение

В ходе инженерной разработки chumby мне также пришлось узнать о литье под давлением, потому что печатная плата должна была находиться внутри какого-то корпуса. Для электронщика с небольшим механическим опытом это был совсем не маленький холм для восхождения. Концепция кажется простой: вы делаете полость из стали, под высоким давлением загоняете в нее расплавленный пластик, даете ему остыть - и вуаля, готовая деталь выходит наружу, прямо как из формочек Play-Doh в начальной школе.

О, если бы процесс был настолько прост.

Да, пластик течет, но он не особенно жидкий. Он движется медленно и остывает по мере движения. На цвет пластика влияют изменения температуры, а при использовании неправильно спроектированной формы в готовом продукте можно даже увидеть линии течения и линии спая. Есть также целый набор вопросов, связанных с тем, как готовая деталь извлекается из формы, как

форма изготавливается и отделяется, где находятся литники и каналы для попадания пластика внутрь формы, и так далее.

К счастью, у РСН были специалисты в Китае, которые знали обо всем этом, и я в основном учился, наблюдая.

Если бы мне нужно было описать литье под давлением одним прилагательным, это было бы слово «точное». Когда все сделано правильно, формы точны до допусков тоньше волоса, хотя изготовлены из твердой стали. Достичь такого уровня точности в столь долговечном материале - немалое достижение, и впечатляет наблюдать, как машина вырезает форму из сырой стали.

Станок, который резал формы для корпуса chumby, имел подвижный стол, быстро перемещавший блок стали, вероятно, весивший несколько сотен фунтов; он фрезеровал металл в большой спешке!



Станок для резки форм, использованный при производстве chumby's. Сравните его с людьми, стоящими рядом, чтобы оценить масштаб.

Но механическая обработка - лишь самый грубый этап изготовления формы. После вырезания черновой формы ее помещают в электроэрозионный станок (EDM), где всплеск электронов выбивает микроскопические кусочки со стальной поверхности. Это чрезвычайно утомительный процесс: я видел, как работают многие EDM, и это похоже на наблюдение за высыханием краски. Однако EDM чертовски точны и дают впечатляющие, повторяемые результаты.

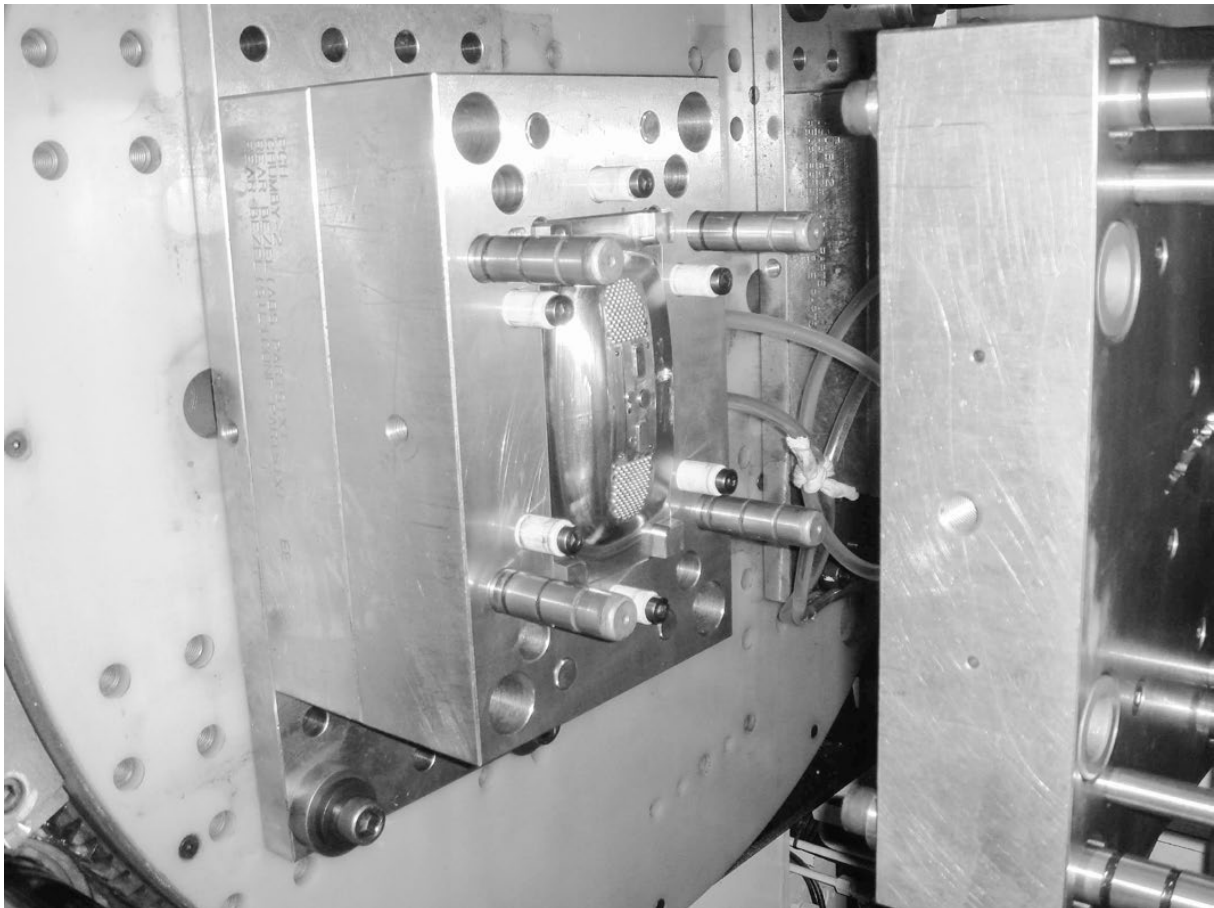
С точки зрения управления проектом самым большим откровением для меня стали феноменально длинные сроки подготовки производственного качества для пластика, отлитого под давлением. В общей сложности форма chumby превратилась из блока сырой стали в инструмент первого впрыска за четыре-шесть недель, и мне пришлось поехать в Китай и увидеть, как

инструментальный цех делает свою работу, прежде чем я убедился, что в график не заложено какое-то огромное количество лишнего времени.

Еще более страшным с точки зрения управления рисками было отсутствие хороших инструментов моделирования, позволяющих предсказать, как пластик будет течь через форму. Если мы видели заметные дефекты вроде линий течения и линий спая, нам приходилось ждать четыре-шесть недель, чтобы узнать, стала ли новая форма лучше. Ох!

К счастью, инструментальщики, которых Chumby использовала в Китае, предвидели эти проблемы и делали инструменты так, чтобы ошибаться в сторону избытка стали, потому что удалить материал для исправления проблемы гораздо проще, чем добавить материал. Это как старая поговорка плотника: дважды отмерь, один раз отрежь, а если уж режешь неправильно, режь с запасом.

Форма, использованная для создания задней рамки chumby, была особенно сложной, потому что включала процесс, называемый формованием поверх детали. Если у вас есть chumby classic, посмотрите на заднюю сторону. Там есть резиноподобный TPE вокруг твердой рамки ABS. Многие люди предполагали, что это приклеенная резиновая лента. На самом деле TPE формируется на месте на задней детали. Для этого требуется двухкомпонентная форма.



Финальная форма для задней рамки chumby внутри машины литья под давлением.

Формы фактически было две, и одна сторона формы вращалась, чтобы чередующиеся системы материалов могли формироваться в правильные моменты процесса.

В скромные пластиковые детали, которые вы видите каждый день, вложено много тяжелой работы, и все это часть создания качественных продуктов. Но одновременно существует и вполне реальная необходимость соответствовать ожиданию низких цен.

Проблема качества

Очевидно, что вместе с ожиданием низкой стоимости товаров, сделанных в Китае, приходит серьезная задача управления качеством. Посмотрите на освещение в медиа таких тем, как свинцовая краска в игрушках, промышленные химикаты в пище и другие изделия, сделанные в Китае, и вы увидите сомнительные решения, принятые ради удержания цен.

Размышляя о таких случаях, я считаю важным применять бритву Hanlon. Перефразируя: «Никогда не приписывайте злему умыслу то, что можно удовлетворительно объяснить невежеством».

У британцев тоже есть удачная, емкая версия этого афоризма: «сначала бардак, потом заговор».

Некоторые производители действительно стремятся заработать любой ценой, но я думаю, что большинство ошибок совершается по невежеству. Большинство рядовых работников фабрик не знают, для чего в конечном счете используется их продукт, и под сильным давлением снижения затрат принимают такие сомнительные решения.

Фабрикам также приходится иметь дело с продуктами, ужасно плохо специфицированными, а также с клиентами, которые заваливают их всевозможными легкомысленными требованиями, - и большинство клиентов не доводит дело до конца ни в том ни в другом случае. В итоге фабрики играют в игру «отгрузим и узнаем», и если клиент не замечает отсутствующую спецификацию, значит, спецификация, должно быть, была неважной. Игра не лучшая, и это означает, что клиентам нужно всегда сохранять бдительность в аудитах и поддержании стандарта качества.

Разрыв между Америкой и Китаем

Одна фундаментальная проблема, лежащая за этой игрой, состоит в том, что многие жители Китая не понимают и не ценят базовые вещи, которые мы в Америке принимаем как должное, и наоборот. Многие китайские фабричные рабочие хорошо образованы, но они не выросли в такой «культуре гаджетов», какая есть у нас в Соединенных Штатах, поэтому нельзя ничего предполагать об их способности субъективно интерпретировать спецификации продукта.

Например, вы можете сказать американскому инженеру: «Я хотел бы кнопку на этой панели», - и, вероятно, получите что-то довольно близкое к ожидаемому по виду и ощущению, поскольку у вас и инженера есть общие переживания и ожидания относительно кнопки на панели. Если сделать то же самое в Китае, вы, вероятно, получите что-то слегка неловко выглядящее и топорно ощущающееся, но чертовски дешевое, а также очень простое в изготовлении и тестировании. Хотя последние свойства желательны по практическим причинам, американские ценители гаджетов просто не купят вещь, которая эстетически неуклюжа или ощущается топорной.

И все же в конечном счете именно эти потребители хотят - нет, требуют - дешевых товаров, и именно эта потребность подталкивает к решению производить в Китае. Проблема в том, что, кроме надписи на продукте «Made in China» или «Made in the USA», потребителей на самом деле не волнует производственный процесс. Какую наценку вы готовы заплатить за гаджет с надписью «Made in the USA»? Премия к стоимости труда в США в 10 раз выше, чем в Китае. Подумайте об этом: может ли средний американский фабричный рабочий быть в 10 раз производительнее среднего китайского фабричного рабочего? С таким множителем трудно конкурировать.

Я не говорю, что в отечественных поставщиках нет ценности: для меня было бы куда меньше усилий и меньше риска производить вещи в Соединенных Штатах. Фактически большинство ранних прототипов делают именно там из-за огромной ценности, которую могут добавить отечественные поставщики. Однако для массового продукта экономика просто не сходится. Никто не купил бы его, потому что цена не оправдывала бы набор функций. Меня даже можно было бы обвинить в лени,

если бы я просто остался с отечественным поставщиком и переложил более высокую стоимость на клиентов.

Участие в производственном процессе

В конце концов, производство в Китае - лучший способ удерживать затраты низкими, а для поддержания качества нет замены поездке в Китай и прямому участию. Почти каждая фабрика «приберется» в день вашего визита, но острый глаз и правильные вопросы позволяют увидеть сквозь любые наспех наведенные фасады.

Когда я оценивал фабрики для Chumby, я всегда посещал комнату контроля качества (QC). Я ожидал увидеть ряды хорошо поддерживаемых и явно используемых папок с проектной документацией и стандартами QC, а также эталонные образцы, то есть предпроизводственные образцы продукта. Я требовал показать содержимое случайной папки и связанный с ней эталонный образец и проверял, знают ли сотрудники, что происходит в этой папке. (Некоторые фабрики действительно набивают папки продукта случайными данными.) Я также считал хорошим знаком твердые инвестиции в оборудование: у лучших производителей, которых я посетил, было по паре комнат со сложным оборудованием для тепловых, механических и электрических предельных испытаний, и, конечно, в комнате находились операторы, реально использовавшие это оборудование. (Я вполне мог представить, как китайский производитель покупает комнату оборудования просто для вида.)

Но я подозреваю, что производители игрушек и еды не привозят таких техников, как я, на фабрики в Китае, чтобы регулярно наблюдать за процессом. Сравните это с Apple, которая регулярно отправляет группу инженеров на интенсивные двухнедельные (или более долгие) вахты на фабриках (обычно Foxconn, которую некоторые в Apple ласково прозвали Mordor). В результате я встречал многих инженеров Apple в эмигрантских барах Шэньчжэня.

Для нас в Chumby было важно, что PCN China Solutions предлагала управление производством и контроль качества западного уровня прямо на месте, в Китае. Если у нас возникала проблема с поставщиком, PCN сразу отправляла кого-нибудь на фабрику посмотреть, что происходит, - без испорченного телефона и без затяжной переписки с FedEx. А владельцы фабрик в Китае обычно очень отзывчивы, когда вы появляетесь у них на пороге.

Таким образом, подход Chumby к загадке качества был целостным. Мы начали с того, что инженер (я) почти с первого дня был на фабрике и изучал ситуацию. Важно узнать, что фабрика может делать, а чего не может. Я смотрел, что производится на линии и какие методы используются. Затем, когда пришло время проектировать продукт, я старался использовать процессы и методы, наиболее привычные для фабрики. Когда мне приходилось делать что-то новое (а любому хорошему инновационному продукту это понадобится), я выбирал свои сражения и сосредоточивался на них, потому что все новое становилось многонедельной задачей, которую нужно довести до правильного результата. Эта стратегия применима даже к самым мелким деталям: если фабрика упаковывает товары в термоусадочную пленку, а вы хотите завернуть свой продукт в бумагу, планируйте серьезно сосредоточиться на разработке процесса бумажной упаковки, потому что вполне возможно, что никто из рабочих линии на выбранной вами фабрике раньше даже не видел продукт, завернутый в бумагу.

Конечно, когда для chumby разрабатывался новый процесс, я предпочитал находиться на фабрике, и до сих пор предпочитаю. Ничто не сравнится с тем, чтобы стоять на линии и показывать рабочим, которые будут собирать ваше устройство, как именно его нужно делать. Например, я лично обучал рабочих сборочной линии chumby тому, как прикреплять кусок медной ленты к LCD-сборке, чтобы сформировать правильный EMI-экран.

Трудно описать тонкости того, как складывать ленту по сложной детали из листового металла, чтобы она обеспечивала хороший электрический контакт с заземляющими поверхностями, не создавая риска короткого замыкания на другие компоненты. Такие тонкости, как тот факт, что клей с одной стороны плохо изолирует, также требуют базового понимания физики, которого у рабочих линии просто нет. Что еще хуже, объяснение этих понятий требует технических слов, которых ваш переводчик может даже не знать.

В моем случае даже хороший трехмерный чертеж или фотография готовой сборки не смогли бы передать всю концепцию, потому что жесткость ленты требовала особого движения при складывании, чтобы она не порвалась. Объяснение процесса на расстоянии, согласование образцов по фотографиям и в конечном счете утверждение экземпляра, доставленного через FedEx, могли бы занять пару недель, но стоять перед группой рабочих и лично демонстрировать процесс заняло всего несколько минут. И, несмотря на языковой барьер, по выражениям лиц и языку тела я мог понять, осознали ли они важность конкретного шага. По этим признакам я сразу заново показывал этапы, которые были неоднозначными или трудными для освоения.

Обычно, когда вы можете продемонстрировать процесс на таком уровне детализации и близости, рабочие начинают делать его правильно за часы, а не за недели. Это одна из причин, по которой я провел так много времени в Китае во время разработки производственного процесса chumby.



Все участвовали в процессе качества chumby. На этой фотографии CEO Steve Tomlin (крайний слева) и художественный директор Susan Kare (в центре) на швейной фабрике прорабатывают детали шелкографии логотипа.

Самодельное удаленное тестирование

Однако Chumby не всегда могла отправить кого-то в Китай. Я, например, предпочитал не жить в Китае, поэтому в Chumby мы во многом полагались на РСН, чтобы следить за качеством и убеждаться, что все идет хорошо, и они выполняли это великолепно.

Часто работа на расстоянии означала, что новые процессы вводились неделями, если меня не было на месте, чтобы подстроить и утвердить их сразу, потому что каждая отдельная правка включала почти круговой рейс через FedEx. Пройдя через этот процесс несколько раз, я научился закладывать две недели на каждую правку, в отличие от нескольких часов, когда я находился в фабричном цехе.

Эти двухнедельные отрезки быстро складывались.

Учитывая трудность надзора за операциями в Китае из Соединенных Штатов, удаленный электронный мониторинг результатов тестирования продуктов был необходим. Для chumby я разработал набор тестеров, которые программировали, персонализировали, загружали, проверяли и измеряли каждое устройство, сходящее со сборочной линии. Все данные процесса тестирования записывались в журнал, а в конце дня журнал передавался на сервер в Соединенных Штатах.

Эти данные позволяли мне отлаживать множество проблем на линии. Я мог определить, что оператор на конкретном тестере испытывает трудности со сканером штрихкодов. Я также сразу знал, если в тот день была проблема выхода годных изделий или если пропускная способность была ниже ожидаемой. Наличие такой самодельной возможности аудита было очень мощным, потому что фабрика знала, что я за ней наблюдаю. На самом деле наличие такой возможности может улучшить отношения с фабрикой: фабрика несет стоимость проблем выхода годных изделий (по крайней мере изначально), поэтому она ценит, когда инженер-разработчик может дать быстрый совет и помощь, прежде чем проблемы выйдут из-под контроля.



Пара тестовых станций chumby на фабрике в Китае. Есть целая история о трудностях, через которые мы прошли, чтобы ввезти эти ноутбуки в Китай.

Дальнейшее фабричное тестирование

После завершения настройки процесса тестирования он может автономно работать на фабрике. Например, на фабрике PCB chumby первый проход финального контроля выполнялся вручную: один человек просматривал каждую печатную плату, а затем с помощью картонного шаблона другой оператор убеждался, что никакие компоненты не отсутствуют. После этого устройства переходили к автоматизированному тестированию.

Периодически и PCN, и фабрика также проводили тестирование Restriction of Hazardous Substances (RoHS) на устройствах chumby, чтобы убедиться, что нет загрязнения заданным набором потенциально вредных химических веществ, включая свинец. RoHS - стандарт безопасности опасных химических веществ, обязательный в Европе, но, по иронии, не в Соединенных Штатах. Фабрики регулярно выполняют этот тест на всех продуктах, даже на тех, что отправляются только в Соединенные Штаты, потому что скрытое загрязнение линии может помешать отгрузке в Европу других продуктов, произведенных на той же линии.

Даже после всех этих испытаний, уже в Соединенных Штатах, Chumby продолжала отбирать устройства для целей QC. Для этого мы регулярно заказывали, характеризовали и разбирали устройства, чтобы убедиться, что соблюдаются все операционные процедуры.

Ошибки все равно случаются

Несмотря на такие меры защиты, в любом продукте будут допущены некоторые ошибки. Каждый продукт проходит фазу, когда ошибки, не пойманные внутренним QA, выбиваются наружу. Вам приходится полагаться на первоклассную службу клиентского сервиса и поддержки, и вы должны планировать, что на этом этапе придется быть очень гибким и изобретательным, чтобы решать проблемы и не допускать их повторения.

Когда я работал в Chumby, если слышал об устройстве «в поле» с аппаратными проблемами, я фактически звонил клиенту, который об этом сообщил. Я хотел знать, что пошло не так, чтобы исправить проблему и убедиться, что она никогда больше не повторится ни у кого!

Однако моя самая большая надежда с chumby состояла в том, чтобы избежать того, что произошло с Microsoft и «красным кольцом смерти» у Xbox 360, когда консоли переживали серьезный аппаратный отказ, переставали работать и просто показывали красный свет вокруг кнопки питания, вызывая огромное разочарование у игроков. Эта проблема проявилась только после того, как Xbox 360 уже несколько лет была на рынке и после отгрузки миллионов устройств. Ситуации вроде «красного кольца смерти» - худший кошмар инженера продукта.

Итак, как видите, довести chumby (или любой продукт) до состояния, когда его можно отгружать потребителям, - это только начало. Настоящая задача начинается потом.

Если вы когда-нибудь окажетесь в этой точке производственного процесса, желаю вам удачи!

Заключительные мысли

Рассказанные здесь истории передают некоторые из моих приключений - и неудач - в изучении того, как строить продукты серийно. Следующие две главы более рефлексивны и менее повествовательны. Следующая глава проведет нас по виртуальной экскурсии по трем фабрикам, чтобы увидеть, чему мы можем у них научиться, а глава 3 попытается обобщить все уроки о производстве, которые я к этому моменту усвоил.

Примечания

[1] Разговорное слово для юаня, базовой счетной единицы жэньминьби (RMB), валюты Китая.

[2] Этот книжный магазин закрылся после визита, который я здесь описываю.

[3] Из-за сильной инфляции зарплат после этого конкретного визита это, вероятно, уже не соответствует действительности.

Глава 2. Внутри трех очень разных фабрик

Трудно понять, как работает компьютер, не открыв его и не заглянув внутрь. Точно так же трудно понять, как делают продукты, не войдя на фабрику и не пройдя по линии. Хотя мы часто думаем о производстве как о необходимом, но скучном шаге после инновации, на самом деле эти две вещи тесно связаны. Изобретатель думает о продукте один раз; фабрика думает о нем день за днем, иногда годами подряд.

Значение фабрик как узлов инноваций в сегодняшней взаимосвязанной глобальной экономике только растет. Реальность такова, что не существует «фабрики Apple» или «фабрики Nike». Скорее есть ряд предприятий, специализирующихся на конкретных процессах (например, изготовлении PCB или производстве застежек-молний), которые курируются знакомыми брендами. Поэтому не редкость увидеть продукты двух конкурентов, идущие бок о бок по похожим линиям на одном предприятии. Такая концентрация узкой экспертизы означает, что лучшее место, где можно узнать, как улучшить какой-то аспект вашего продукта, часто является тем же местом, где похожий аспект делают для продуктов всех остальных.

Некоторые из моих самых важных прозрений о том, как улучшить продукт, пришли из наблюдения за техниками, работающими на линии, и из того, как я видел хитрые приемы оптимизации, которые они выработали после долгого повторения одного и того же действия.

Эта глава проведет вас по трем фабрикам, которые делают повседневные вещи: PCB (в частности, те, что используются в Arduino), USB-флешки и застежки-молнии. Приоткрыв занавес, вы получите некоторое представление о проектных компромиссах, стоящих за продуктами, и о том, как их можно сделать лучше. На фабрике PCB я открыл секрет того, как они печатают карту Италии высокого разрешения на обратной стороне каждого Arduino; на фабрике USB-флешек я стал свидетелем странного брака высоко- и низкотехнологичных производственных приемов; а на фабрике застежек-молний выяснил, как даже самый скромный продукт может нести ценные уроки для проектировщиков.

Где рождаются Arduino

Был июль 2012 года, и прошло около шести месяцев с тех пор, как мой предыдущий стартап Chumby прекратил работу. Я решил взять год паузы, чтобы разобраться с жизнью и вычеркнуть несколько пунктов из списка желаний, одним из которых была поездка в Италию. Моей девушке пришла в голову отличная идея связаться с командой Arduino и узнать, смогу ли я в рамках нашего маршрута посетить их фабрику в Scarmagno (это было за годы до раскола Arduino/Genuino). Участники Officine Arduino (особенно управляющий директор Davide Gomba) любезно нашли время в своих плотных графиках, чтобы показать мне свою фабрику. Они терпеливо ждали, пока я выпускал наружу своего внутреннего любителя фотографии и общую любовь ко всему аппаратному, и я определенно уехал оттуда с множеством отличных снимков.

Scarmagno, маленький городок в северной Италии, находится примерно в полутора часах езды на машине к западу от Milan, рядом с фабриками Olivetti на окраине Torino. Город занимается всем изготовлением печатных плат, монтажом плат и распространением официально брендированных Arduino. Мне было действительно интересно увидеть фабрики, и главным событием моей экскурсии стало посещение System Elettronica, фабрики PCB, где делали PCB для Arduino.

Одна очаровательная особенность System Elettronica в том, что владелец выкрасил фабрику в зеленый, белый и красный цвета итальянского флага. В фабричном цехе я видел часть этого духа в красных и зеленых стойках, тянувшихся по всей длине предприятия.



Общий вид фабричного цеха System Elettronica в августе 2012 года.

Но вскоре я перестал уделять много внимания декору, потому что именно в этом фабричном цехе я смог проследить свежую партию Arduino Leonardos через весь производственный процесс. Вот как изготавливались эти платы.

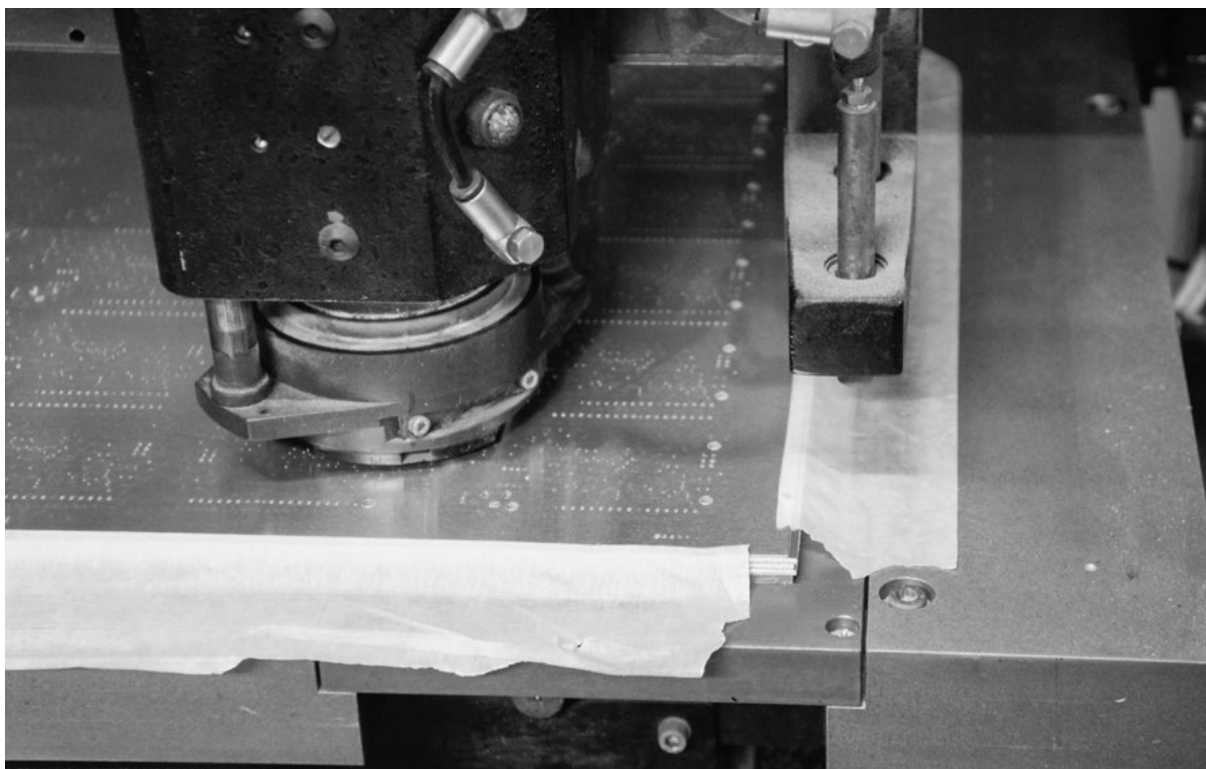
Начать с листа меди

Платы Arduino Leonardo начинаются как огромные листы первичного, покрытого медью FR-4, материала из стекловолокна и эпоксидной смолы, который большинство PCB использует как подложку, изолирующий и структурный слой между медными слоями. Листы были толщиной 1,6 мм (самая распространенная толщина PCB, соответствующая 1/16 дюйма), вероятно, около метра шириной и примерно полтора метра длиной.



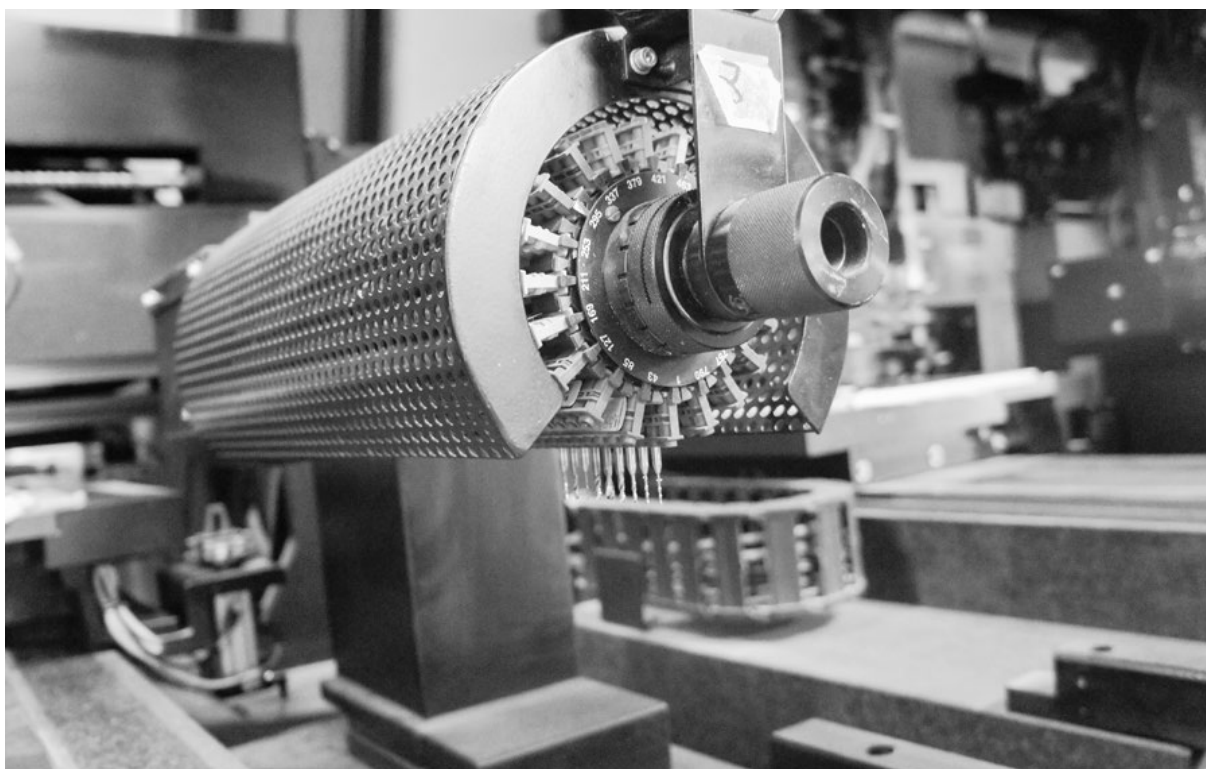
Стопка медных листов, ожидающих превращения в платы Arduino.

Первый шаг обработки РСВ - просверлить все отверстия: контактные площадки, переходные отверстия (vias, малые отверстия, соединяющие разные слои РСВ), монтажные отверстия, металлизированные пазы и так далее. При изготовлении РСВ отверстия сверлят до формирования рисунка, то есть до этапа, на котором маскирующее химическое вещество фотографически задается на листе везде, где на готовых платах должна остаться медь, включая места дорожек, площадок пайки и так далее. Некоторые просверленные отверстия используются для выравнивания масок, которые позже в процессе формируют рисунок дорожек. Сверление также грязный и неточный процесс, который мог бы повредить рисунки схем, если бы они уже были на месте.



Сверлильная головка CNC, использованная для сверления плат Arduino.

Заготовки медных панелей складывали стопкой по три, и CNC-сверло проходило один раз через все три, позволяя сверлить по три подложки за раз.

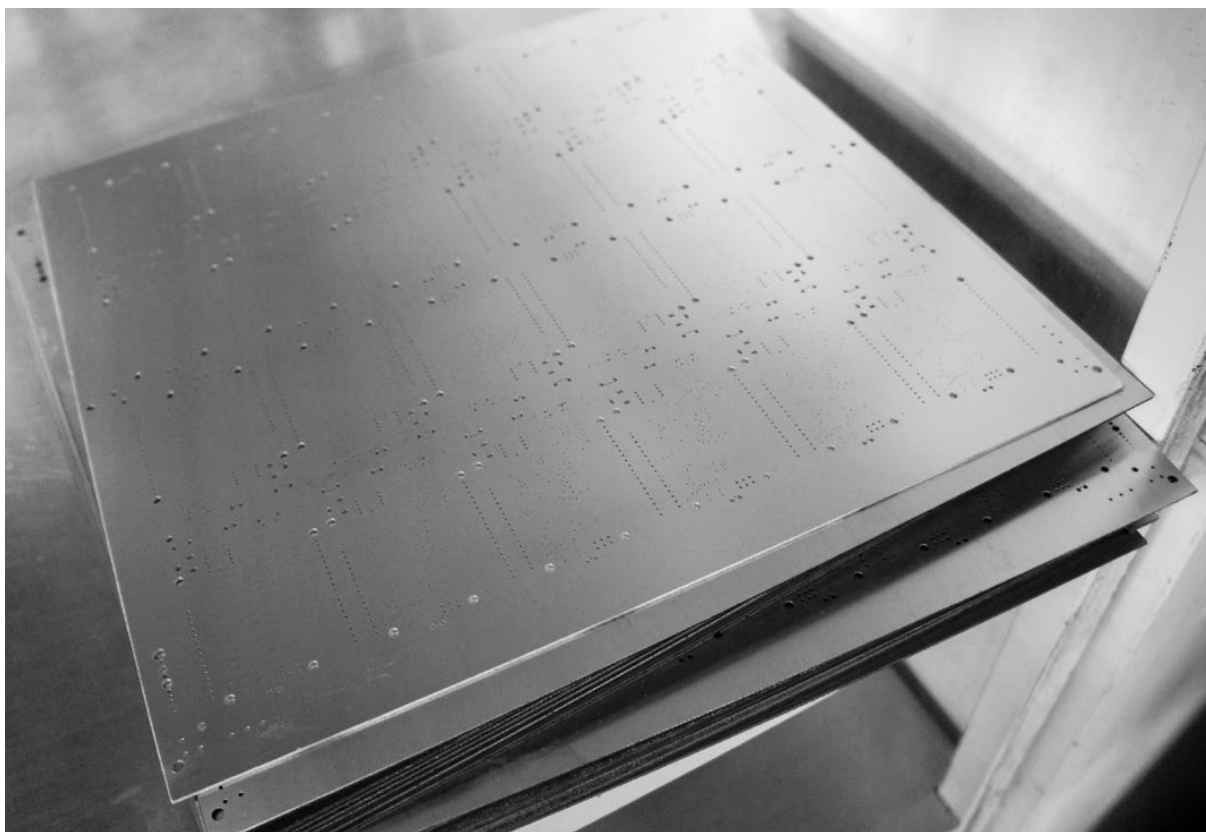


Стойка сверл, использованная CNC-сверлильным станком. Если вам когда-нибудь приходилось создавать файлы сверления NC-drill, это та самая стойка сверл.

Каждое отверстие в плате Arduino было просверлено механически, включая переходные отверстия. То же верно для любой PCB со сквозными отверстиями, поэтому количество переходных отверстий так важно при расчете стоимости PCB.

Заметьте, что конкретный сверлильный станок, который я видел в System Elettronica, был относительно небольшим. Я видел в Китае массивные сверлильные установки, где четыре или шесть сверлильных головок объединены (механически присоединены) в машину размером с грузовик, обрабатывающую десятки панелей одновременно, в отличие от трех панелей, с которыми мог справиться этот станок. Логика такого подхода в том, что точный роботизированный позиционирующий узел - дорогая часть сверлильной машины. Само сверло дешево: всего лишь вращающийся мотор, приводящий бит. Поэтому один способ увеличить пропускную способность - объединить несколько сверл на одном большом узле и двигать их согласованно. Каждое отдельное сверло все равно проходит через свою стопку панелей, но за цену одного позиционера X-Y вы получаете в четыре-шесть раз большую пропускную способность, чем у сверла, которое я видел в поездке по Италии. Эти большие машины сверлят так быстро и сильно, что земля дрожит при каждом просверленном переходном отверстии даже на расстоянии нескольких метров.

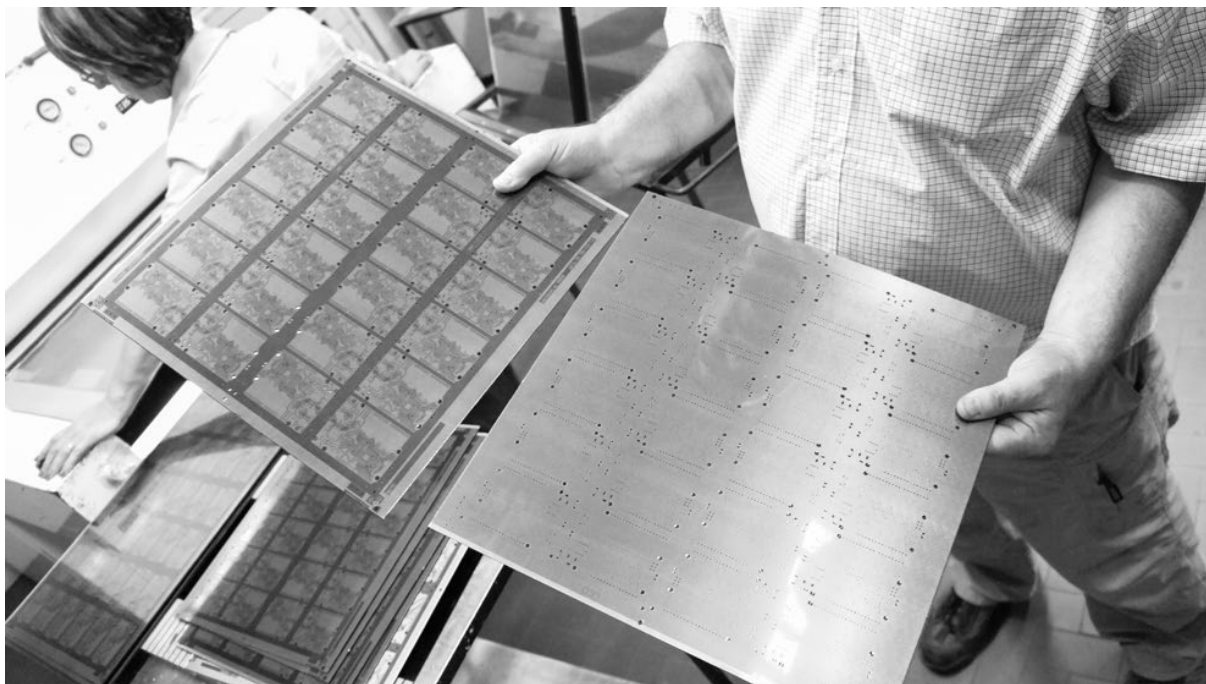
После того как панели просверлены, очищены и зачищены от заусенцев, они готовы к следующему шагу производственного процесса.



Стопка готовых просверленных панелей плат Arduino Leonardo.

Нанесение рисунка РСВ на медь

Следующий шаг - нанести на панель фоторезист, светочувствительное химическое вещество, и экспонировать рисунок. В System Elettronica этот процесс использовал световой стол и высококонтрастную пленку. Я также видел, как для нанесения рисунка на РСВ применяют прямое лазерное экспонирование в виде растрово сканирующего лазера. Прямые лазерные сканеры чаще встречаются в цехах быстрого изготовления прототипов, а пленочное изображение чаще встречается на массовом производстве.



До и после: правая панель показывает фоторезист до экспонирования, левая - после.

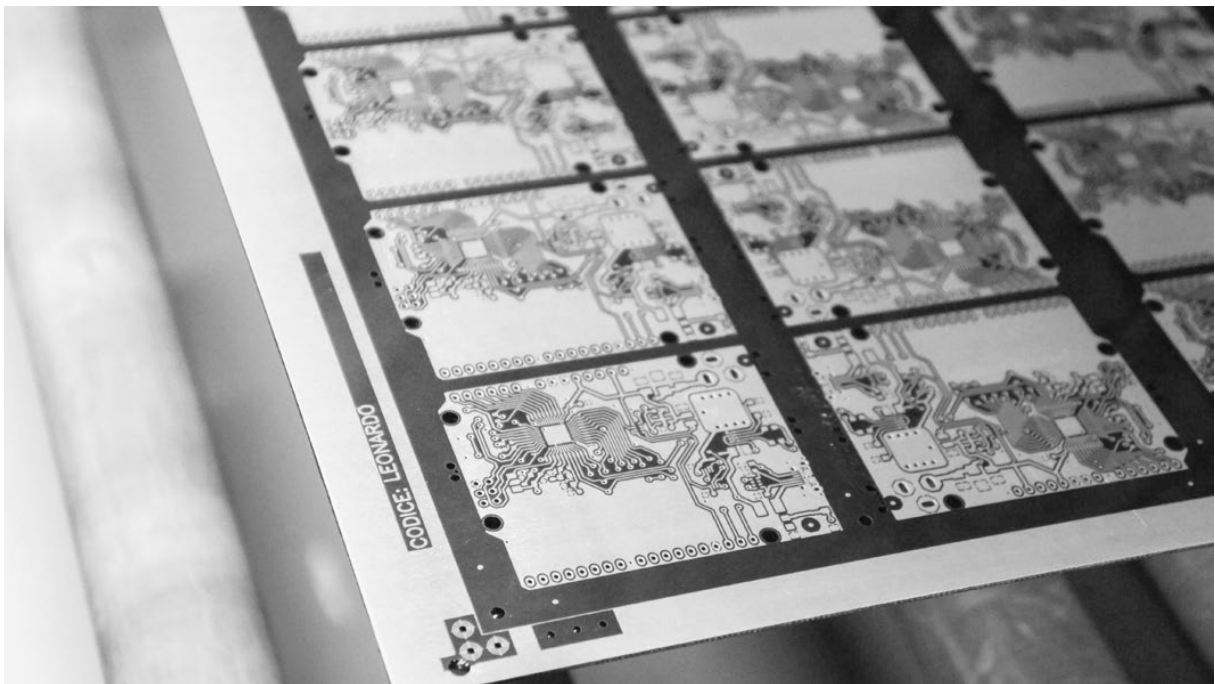


PCB устанавливают в световой стол, который экспонирует пленку ее необработанной обратной стороны.

После нанесения рисунка каждую панель плат отправляют в машину для проявления. В этом случае одна и та же машина используется для проявления и фоторезиста, и паяльной маски.



Машина, проявляющая фоторезист.



Эта фотография панели с проявленным фоторезистом - одна из моих любимых фотографий с фабрики System Elettronica. Кроме того, что-то в словах «Codice: Leonardo» звучит просто круто.

Травление РСВ

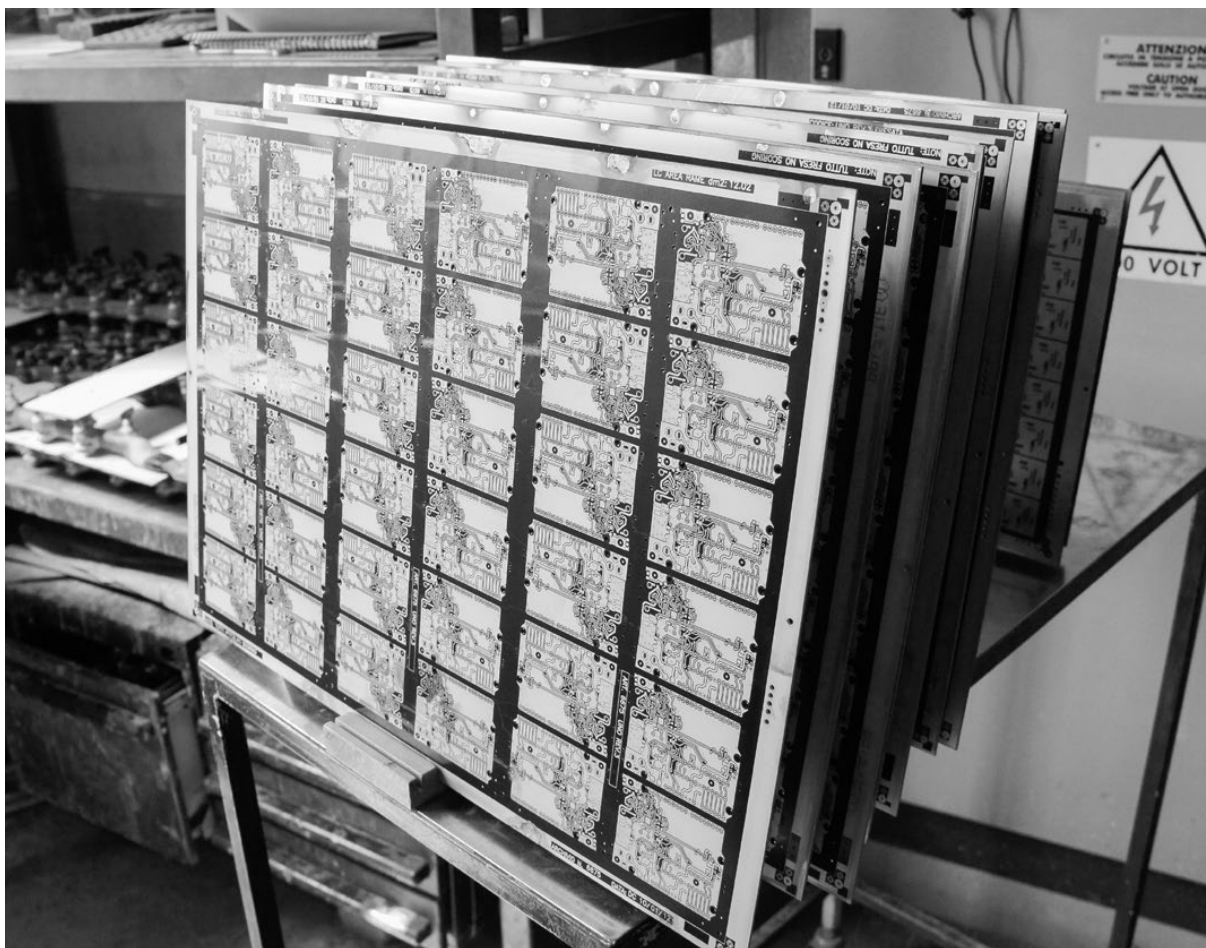
После фотообработки и проявления панели проходят через серию химических ванн, которые травят и металлизуют медь.

Панели мягко покачивают туда-сюда в химической ванне, чтобы ускорить процесс травления. Движение также отводит использованный травитель от панелей, обеспечивая более равномерную скорость травления независимо от количества меди, которое нужно удалить. Перемещение панелей через эти химические ванны в Scarmagno было полностью автоматизировано. Автоматизация необходима, потому что панели должны обрабатываться серией едких химических ванн при минимальном воздействии кислорода. Кислород может испортить панель за считанные секунды, поэтому перенос между ваннами должен быть быстрым, а время пребывания панели в ванне - постоянным. Ванны также содержат химические вещества, вредные для людей, так что намного безопаснее, когда эту работу выполняет робот.



Машина, перемещающая панели в травителе.

После обработки панелей в этой серии растворов на всех поверхностях панели, не обработанных фоторезистом, включая ранее неметаллизированные сквозные отверстия и площадки, появляется тусклое белое покрытие (я предполагаю, что это никель или олово).

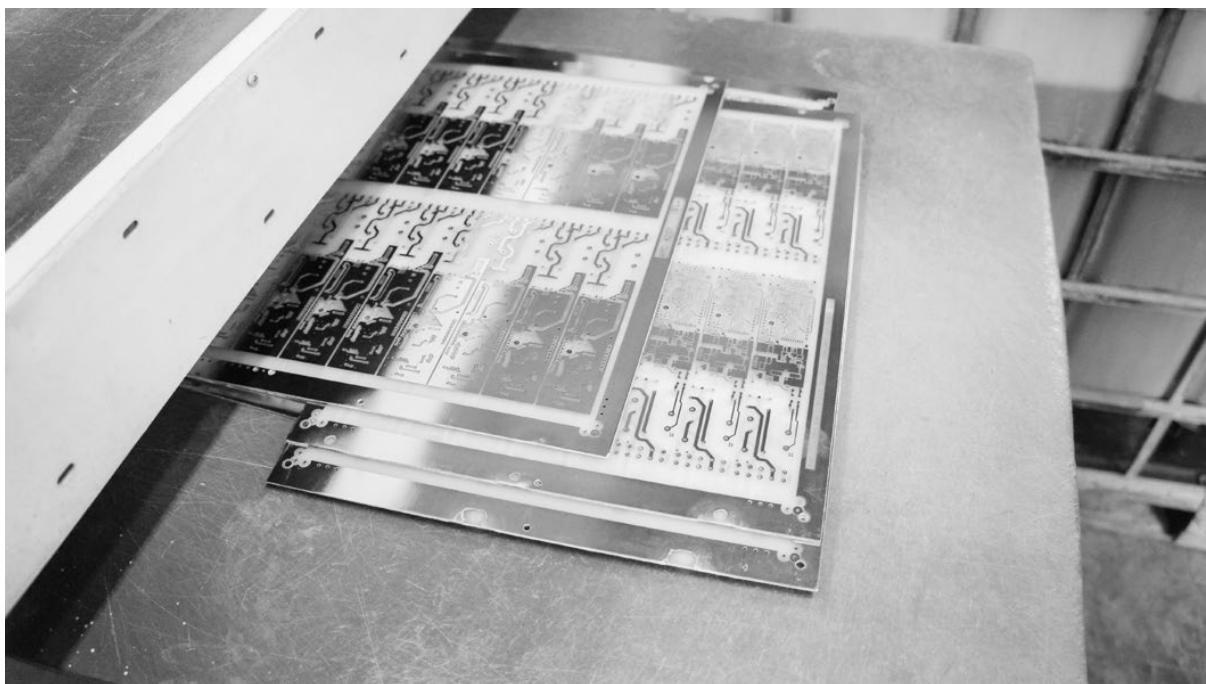


Панели плат Arduino Leonardo после прохождения серии химических ванн.

В этот момент резист и неметаллизированную медь снимают, оставляя только сырой FR-4 и металлизированную медь. Finalный этап обработки дает яркую медную отделку.



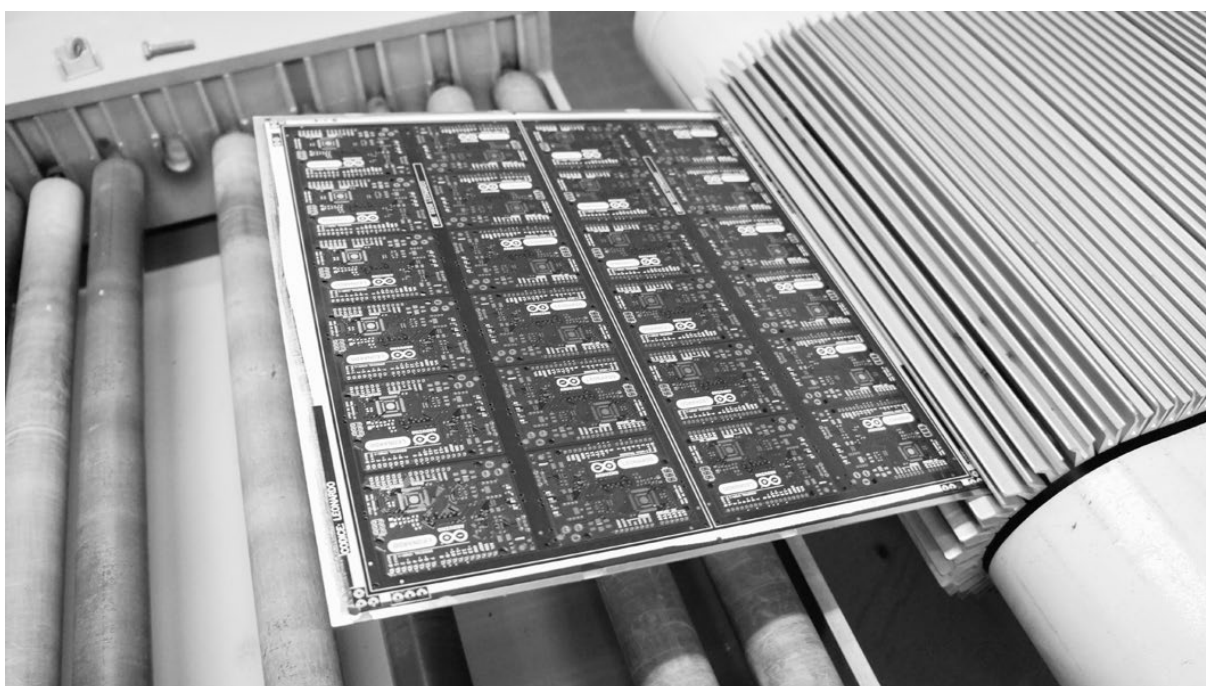
Панель, с которой вытравлена нежелательная медь.



Панели РСВ с яркой блестящей медью. На этой фотографии не показана панель Arduino, потому что они не проходили через машину, когда я ее фотографировал.

Нанесение паяльной маски и шелкографии

После полировки меди панели готовы к паяльной маске (защитному лакоподобному слою, который изолирует расположенные ниже медные дорожки и предотвращает образование паяных перемычек сверху) и шелкографии (чернилам, используемым для маркировки компонентов, нанесения логотипов и так далее). Их наносят процессом, очень похожим на процесс формирования рисунков дорожек, с использованием фотошаблона и машины проявления/снятия.



Панель плат Arduino с проявленными паяльной маской и шелкографией.

В случае Arduino шелкография фактически представляет собой второй слой паяльной маски. Для команды Arduino была закуплена очень конкретная формула белой сухопленочной паяльной маски, чтобы создать резкий, хорошо выглядящий слой, способный разрешить сложную графику, которую вы видите на платах Arduino, особенно карту Италии на обратной стороне. Среди других методов создания слоев шелкографии, которые я видел, есть высокоразрешающая струйная печать, лучше подходящая для цехов быстрого изготовления плат, и, конечно, одноименный процесс шелкографии со шваброй и краской.

Испытание и окончательная обработка плат

После всей этой химической обработки панели получают защитное покрытие припоем от машины выравнивания припоя горячим воздухом. Когда паяное покрытие нанесено, каждая плата проходит 100-процентное тестирование. У каждой дорожки измеряют непрерывность и сопротивление парой летающих щупов. Процесс, который я видел, называется тестирование летающей головкой (также тестирование летающими щупами), и в такой установке несколько пар рычагов с игольчатыми щупами быстрыми постукивающими движениями проверяют непрерывность между парами дорожек. Если учесть все дорожки на Arduino Leonardo, это очень много зондирования! К счастью, руки робота движутся размытым пятном, поскольку он может проверять сотни точек в минуту.

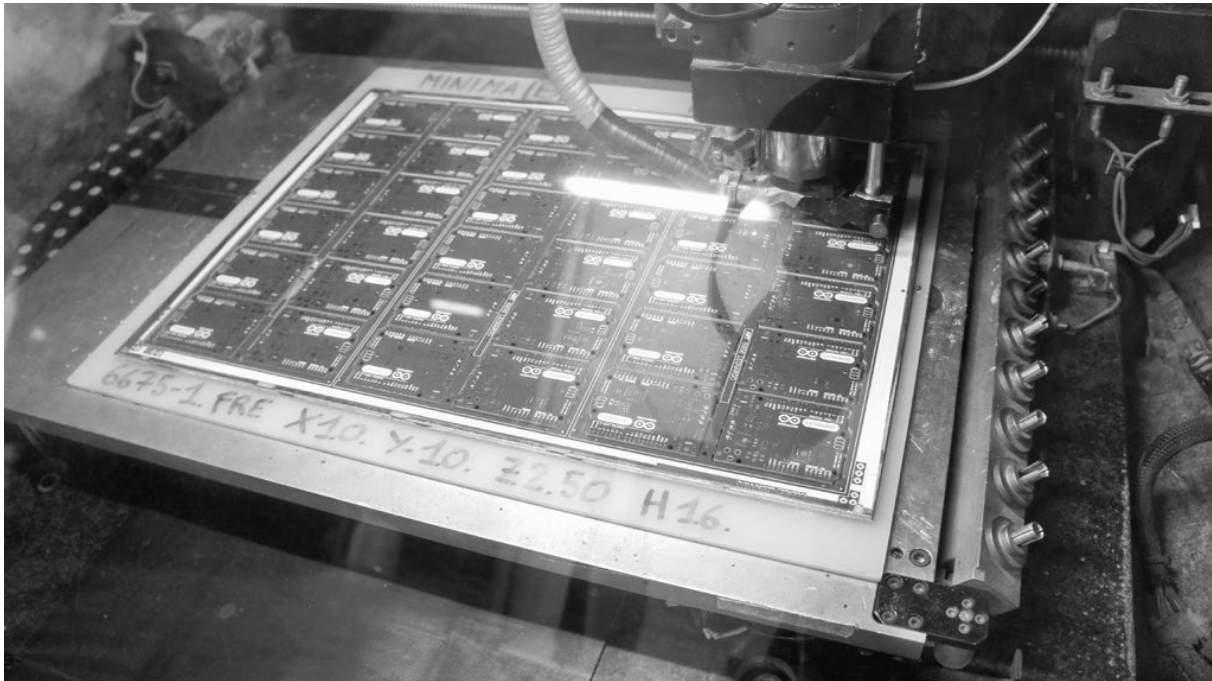
Примечание. Альтернатива тестированию летающими щупами - тестирование в зажимной оснастке, где набор пружинных контактов помещается в оснастку, способную проверить всю плату за одну механическую операцию. Однако зажимные тестовые оснастки очень трудоемки в сборке и обслуживании и требуют физической перекоммутации каждый раз, когда обновляются файлы Gerber, описывающие изображения РСВ. Поэтому при меньших объемах тестирование летающими щупами экономически эффективнее и гибче, чем тестирование в зажимной оснастке.



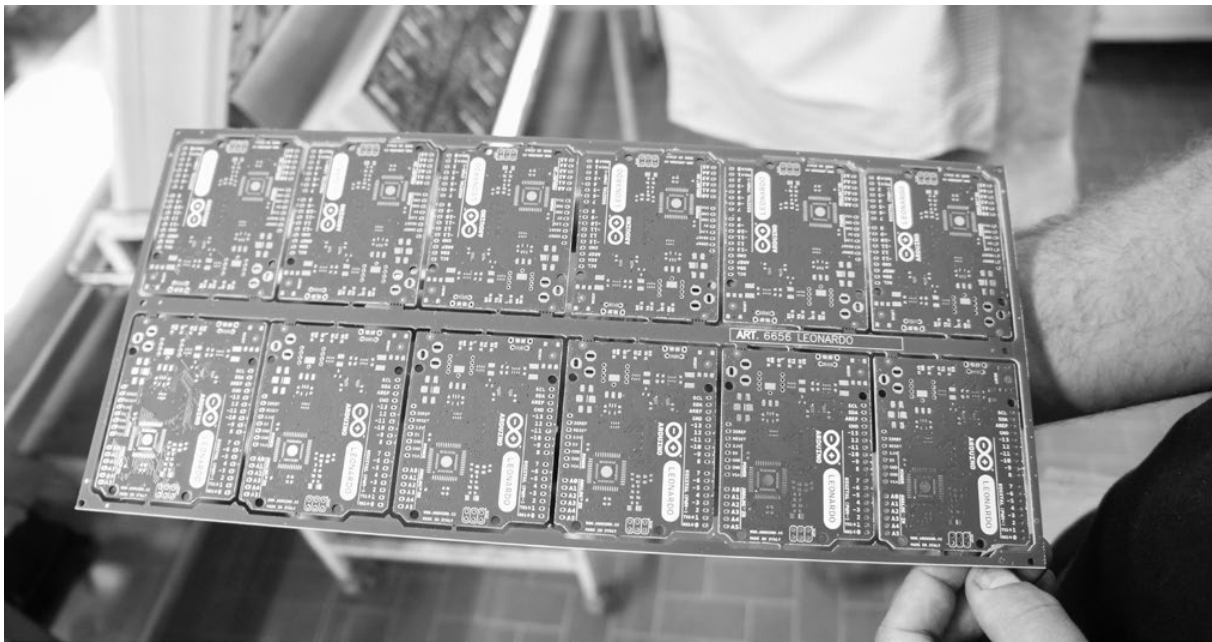
Стопка почти готовых панелей РСВ, ожидающих финального шага фрезеровки отдельных плат.

Это конкретное предприятие создавало только панели; компоненты фактически устанавливала другая фабрика. В таких ситуациях, прежде чем панели можно отправить на следующую фабрику, отдельные РСВ нужно выфрезеровать так, чтобы они помещались в машины технологии

поверхностного монтажа (SMT) для установки компонентов. Панели снова складывают стопкой и пакетно обрабатывают на машине, использующей фрезу, чтобы вырезать и освободить платы. После этого платы наконец готовы к отправке на SMT-предприятие.



Несколько панелей Arduino, сложенных для фрезеровки.



Меньшие панели 2x6 делают SMT-обработку эффективнее.



Настоящая стопка примерно из 25 000 голых PCB Arduino, готовых покинуть фабрику PCB. Оттуда их наполняли компонентами, отгружали и продавали мейкерам по всему миру!

Я рад, что сделал этот крюк и посетил фабрику PCB Arduino. Я бывал на нескольких фабриках PCB, и у каждой из них свой характер и собственный набор приемов для повышения выхода годных изделий, а также уникальные ограничения, которые проектировщикам нужно компенсировать. Было также интересно увидеть маленький трюк с использованием дополнительного слоя паяльной маски вместо шелкографии для достижения высокого косметического качества. Хотя разрешение шелкографии ограничено сеткой шелкового барьера, удерживающего краску, паяльная маска ограничена качеством оптики и химического проявления, что дает более чем десятикратное улучшение разрешения и в итоге более высокое воспринимаемое качество. Обычно более низкое качество шелкографии приемлемо, потому что конечные пользователи не видят печатные платы внутри компьютеров, но для Arduino конечный продукт - это сама печатная плата.

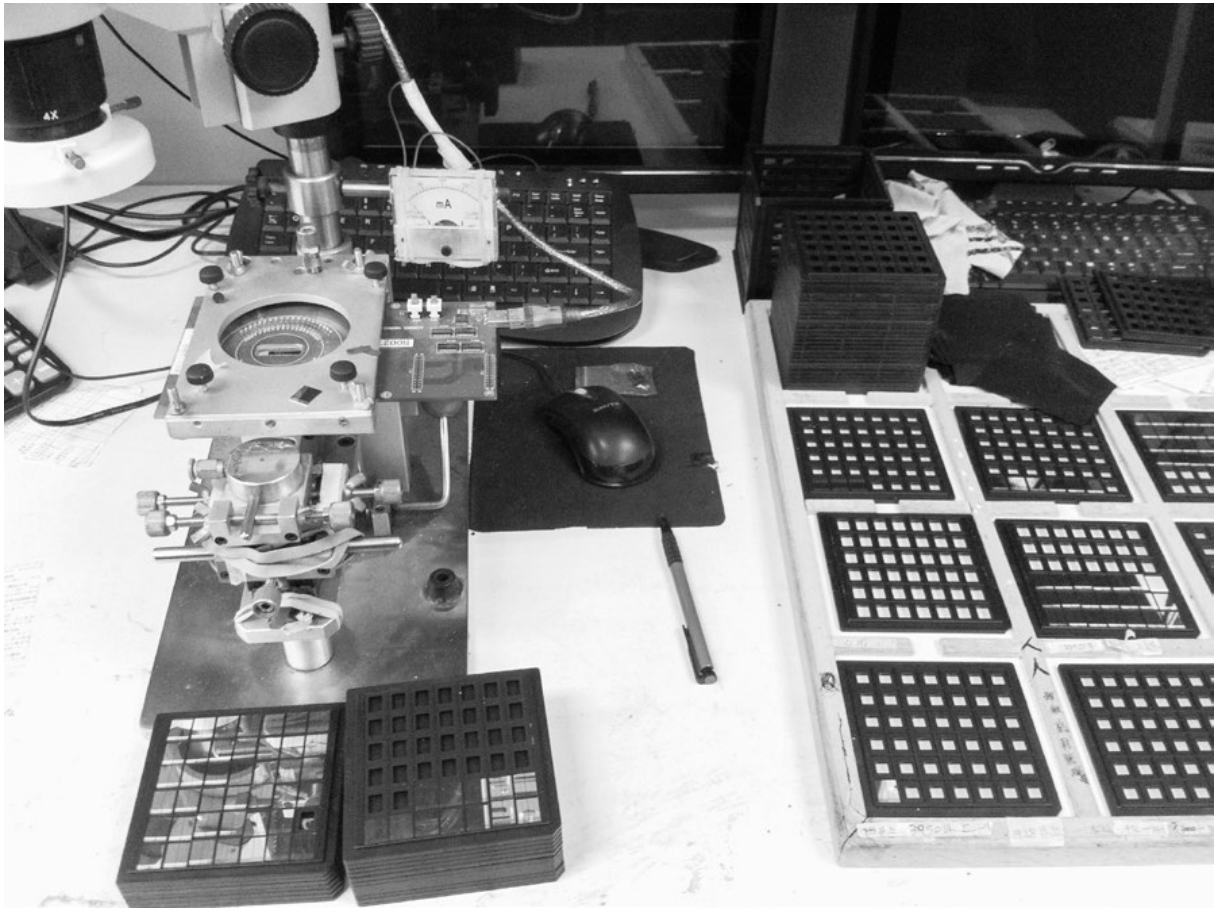
Где рождаются USB-флешки

Через несколько месяцев после моей экскурсии по фабрике Arduino мне посчастливилось быть приглашенным докладчиком на Linux Conference Australia (LCA) 2013. В своем докладе «Linux in the Flesh: Adventures Embedding Linux in Hardware» я рассказывал о том, что Linux есть во всевозможных устройствах, которые мы видим каждый день. Эта история не о Linux, но она связывает меня и, косвенно, LCA с фабрикой.

Одной из безделушек, которые я получил от организаторов LCA, была маленькая USB-флешка с Тух, пингвином-талисманом Linux, снаружи. Когда я увидел устройство, я подумал, что это забавное совпадение: примерно за неделю до конференции я был на фабрике, производившей USB-флешки точно такого же типа. Я увидел процесс сборки плат USB-флешек от начала до конца, и он, к удивлению, включал намного меньше автоматизации, чем процесс производства Arduino.

Начало USB-флешки

Жизнь USB-флешек начинается с голых микросхем флеш-памяти. До установки на PCB эти микросхемы проверяют на емкость памяти и функциональность.



Рабочее место, где проверяют микросхемы флеш-памяти. Металлический прямоугольник слева с круглым вырезом - зондовая карта.

На рабочем месте этой фабрики стопки голых кристаллов флеш-памяти ожидали тестирования и сортировки с помощью зондовой карты, у которой есть крошечные, очень точно расположенные контакты, используемые для касания площадок на поверхности кремниевой пластины, лишь немного более широких, чем человеческий волос. (Мне нравится, как рабочий на этой конкретной станции использовал резинки, чтобы прикрепить аналоговый измеритель тока к зондовой карте.)



Зондовая карта крупным планом.

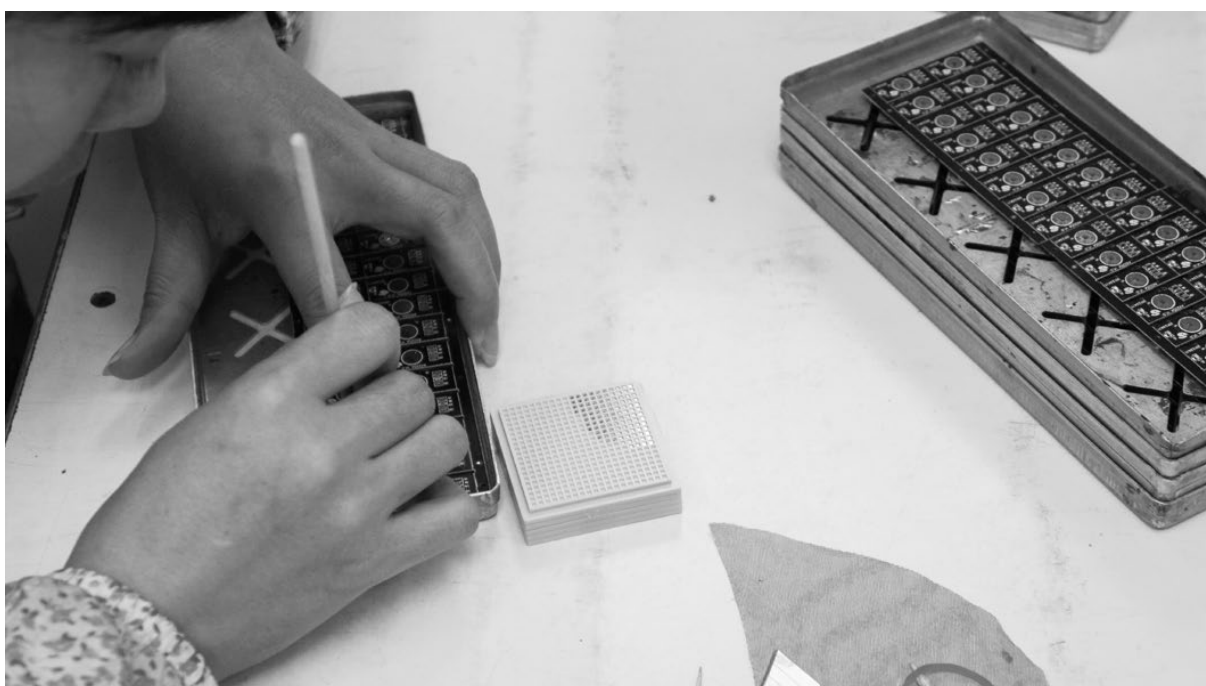


Взгляд через микроскоп на станции микрозондирования. Обратите внимание на иглы, касающиеся квадратных площадок у края поверхности микросхемы флеш-памяти. Каждая площадка имеет сторону, возможно, 100 микрон; диаметр человеческого волоса около 70 микрон.

Интересно, что микросхемы, которые я видел, совершенно точно не тестировались в чистой комнате. Рабочие брали микросхемы пинцетами и ручными вакуумными держателями и вручную устанавливали зондовые карты в свои приспособления.

Ручная установка микросхем на PCB

После того как микросхемы были проверены на функциональность, их вручную устанавливали на PCB USB-флешек. Это не необычная практика; каждое ориентированное на низкую стоимость предприятие проволочного монтажа, которое я посещал, полагается на ручную установку голых кристаллов.



Контроллерная IC устанавливается на панель PCB для USB-флешек. Крошечные голые кристаллы находятся справа, в вафельной кассете.



Более широкий вид рабочего места установки кристаллов.

Женщина, за которой я наблюдал во время установки голых кристаллов, использовала похожий на палочку для еды инструмент, сделанный из вручную обрезанного бамбука. Я до сих пор не понял точно, как работает этот процесс, но, насколько я могу судить, бамбуковые палочки имеют ровно такую поверхностную энергию, чтобы прилипнуть к кремниевому кристаллу, так что кремний держится на кончике бамбукового стержня. На голые платы заранее нанесена точка клея, поэтому, когда оператор касается кристаллом клея, поверхностное натяжение клея стягивает кристалл с бамбуковой палочки.

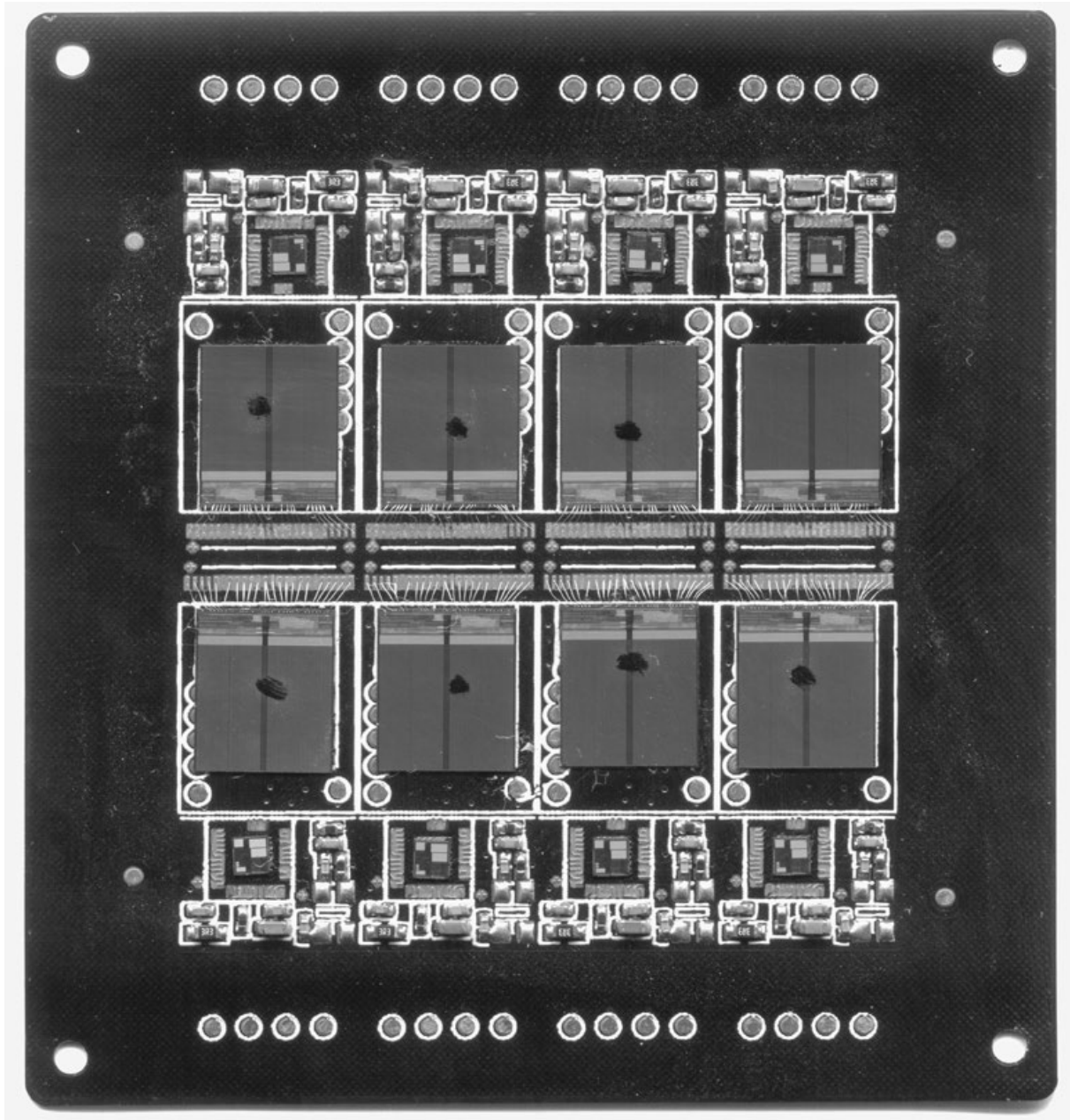
Думать о том, что микросхемы внутри моей USB-флешки обрабатывались модифицированными палочками для еды, - довольно психоделично.

Соединение микросхем с РСВ

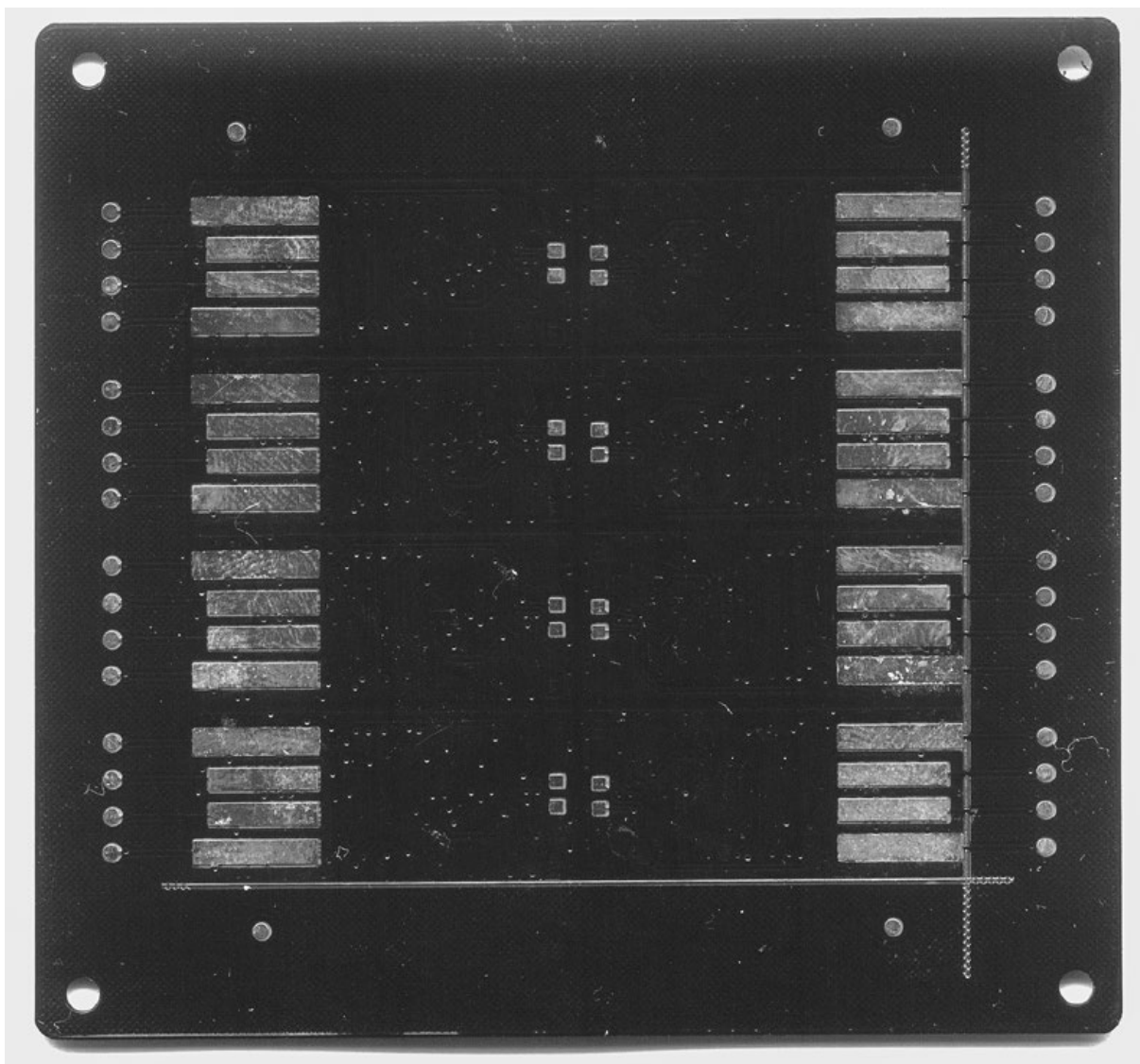
После установки микросхем на РСВ их соединяли с платой проводниками на автоматизированной машине проволочного монтажа, которая использует компьютерное распознавание изображений, чтобы находить расположение контактных площадок (это одна из причин, по которым фабрики могут позволить себе ручную установку кристаллов). Проволочный монтаж - процесс, который соединяет интегральную схему с ее корпусом, и автоматическая машина соединяла провода с IC с безумной скоростью, все это время вращая печатную плату. Пока я наблюдал этот процесс, оператору пришлось вручную снять и заменить неправильно присоединенный провод, а затем снова заправить провод в машину. Учитывая, что эти провода тоньше волоса, а контактные площадки на корпусе и IC микроскопические, это было немалым достижением ручной ловкости.

Крупный план плат USB-флешек

Так же как фабрика Arduino использовала панели, содержащие несколько плат Leonardo, фабрика USB-флешек использовала панели по восемь USB-флешек каждая. Каждая флешка на панели состояла из микросхемы флеш-памяти и контроллерной IC, которая выполняла мост между USB и сырой флеш-памятью, нетривиальную задачу, включающую, среди прочего, управление картами плохих блоков и коррекцию ошибок. Контроллер, вероятно, был CPU класса 8051, работавшим на нескольких десятках MHz.



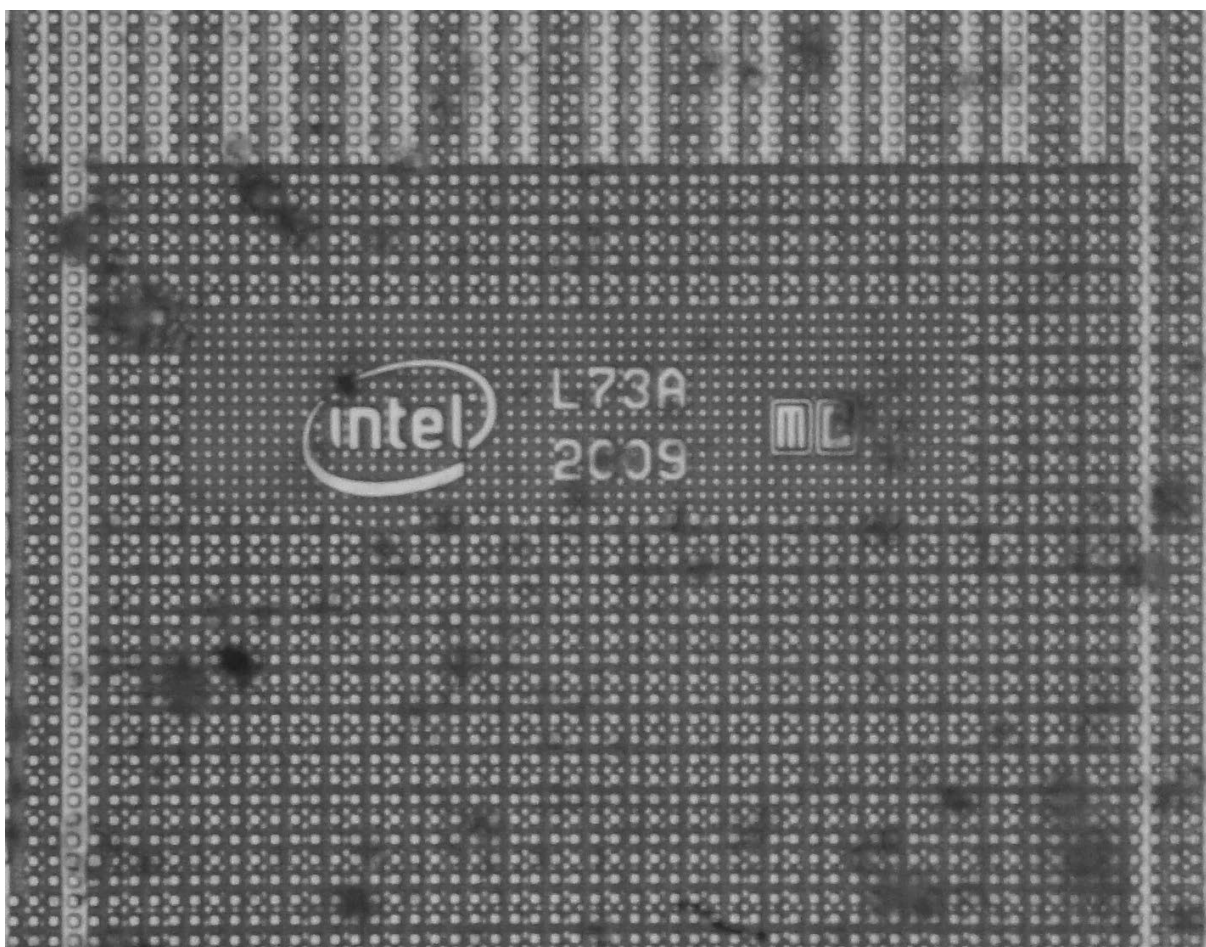
Частично соединенная, но уже полностью укомплектованная кристаллами PCB, которую владелец фабрики подарил мне как сувенир от визита. Некоторые проволочные соединения были раздавлены при перевозке.



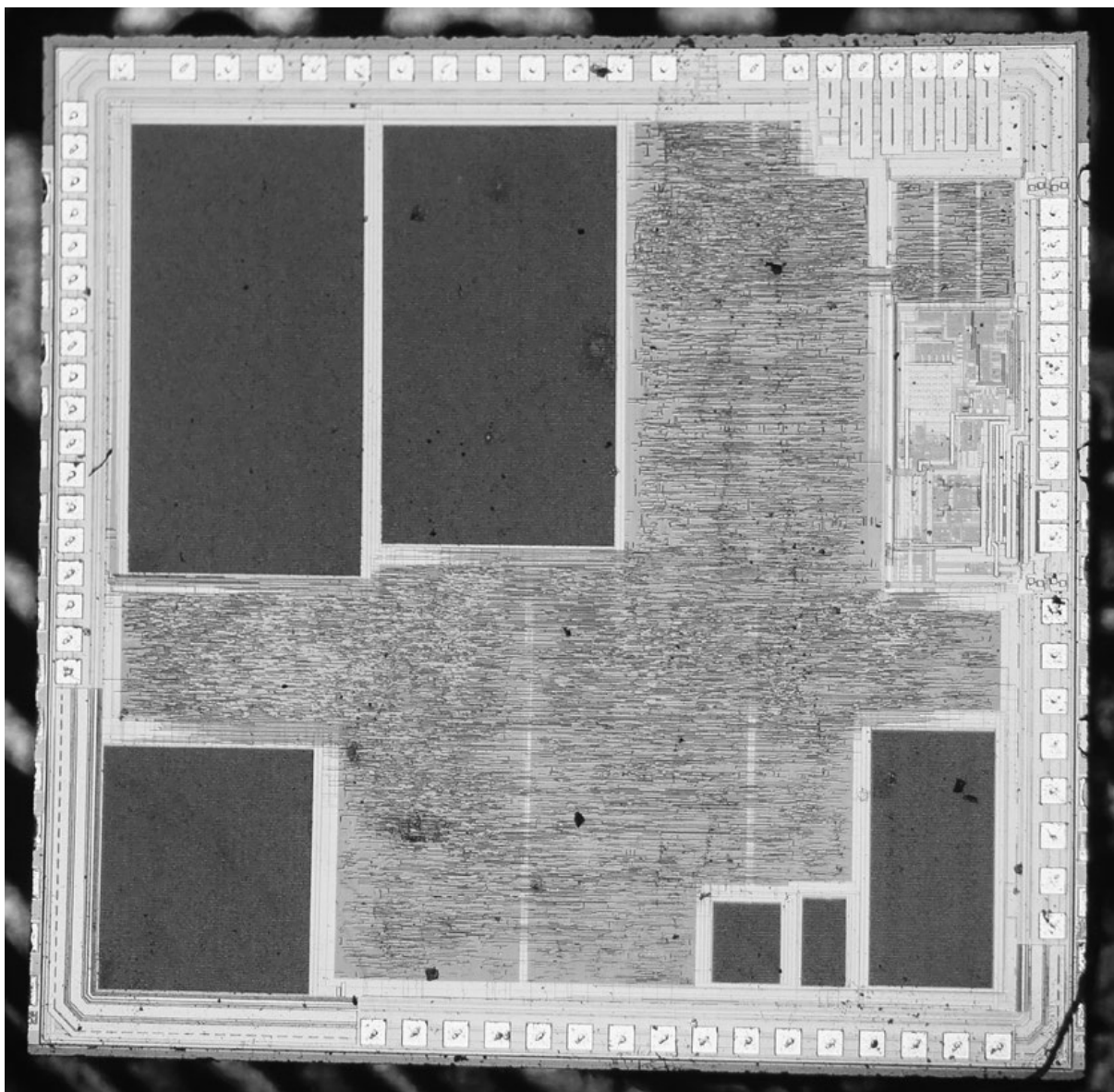
Обратная сторона этой же РСВ.



Интересно, что вся сборка USB-флешки гибкая до инкапсуляции.



Маркировка кристалла микросхемы флеш-памяти. По-видимому, она сделана Intel.



Снимок кристалла контроллерной микросхемы, которая находилась внутри USB-флешек.

После того как панели были соединены проводниками и протестированы, их заливали эпоксидной смолой, а затем разрезали на отдельные части, готовые к продаже.

Но достаточно о производстве электроники; дальше я хочу показать вам фабричный цех другого типа.

История двух застежек-молний

У моего друга Chris "Akiba" Wang биография похожа на мою, только в молодые годы он был куда круче: в девяностые он был танцором у таких исполнителей, как LL Cool J и Run DMC. Пройдя этап работы в крупных полупроводниковых компаниях, он в итоге уволился и последовал своей страсти - проектировать и производить собственные аппаратные проекты. Эксперт по маломощным беспроводным сетям короткого радиуса действия (он был соавтором книги о Bluetooth Low Energy и продает вариант Arduino + 802.15.4 под названием "Freakduino"), теперь он консультирует организации вроде United Nations и Keio University, руководит FreakLabs и сотрудничает с разными танцевальными коллективами, например Wrecking Crew, чтобы предоставлять уникальные и убедительные световые решения для сценических шоу.

Мне посчастливилось познакомиться Akiba с большим районом Шэньчжэня во время поездки со студентами MIT Media Lab в 2013 году - той самой поездки, в которой мы посетили фабрику USB-флешек. С тех пор он исследует этот район все глубже и глубже. Поскольку его работа охватывает дисциплины перформанса, носимых устройств и электроники, его сеть фабрик довольно сильно отличается от моей, поэтому я всегда рад возможности узнать больше о его мире.

В январе 2015 года Akiba взял меня на фабрику застежек-молний своего друга. Я был очень рад экскурсии: каким бы скромным ни был продукт, я всегда узнаю что-то новое, посещая его фабрику. Эта фабрика сильно отличалась и от Arduino, и от предприятий USB-флешек. Сотрудников было намного меньше, и это был высокоавтоматизированный, вертикально интегрированный производитель. Чтобы дать вам представление, что это значит: это предприятие превращало металлические слитки, опилки и рис в детали застежек-молний.



Примерно 1 тонна слитков, состоящих из сплава 93 процентов цинка и 7 процентов алюминия.



Прессованные гранулы из опилок, используемые как топливо для плавильной печи слитков.



Рис, используемый для кормления рабочих.



Готовые сборки язычка и бегунка молнии.

Давайте посмотрим на одну сторону того, как этот процесс реально работает.

Полностью автоматизированный процесс

Между тремя входными материалами и выходным продуктом находилась полностью автоматизированная линия литья под давлением для создания язычков и бегунков молний, набор барабанов и вибрирующих чаш (или, как я люблю их называть, «виброчаш») для высвобождения и полировки молний, а также набор машин для удаления заусенцев и соединения каждого язычка с его бегунком. Думаю, я насчитал на предприятии меньше дюжины сотрудников, и предполагаю, что их мощность значительно превышает миллион застежек-молний в месяц.

Меня заворожали виброчаши[1], которые собирали молнии. Виброчаш было две: одна с язычками, другая с бегунками. И бегунки, и язычки попадали на движущийся рельс, и, когда я смотрел на эти чудеса в работе, казалось, будто бегунки и язычки сами магически выстраиваются в правильной ориентации. Каждый попадал на свой рельс, а в конце линии их прессовали вместе в знакомую форму застежки-молнии, и все это происходило в одной полностью автоматической машине.

Когда я опустил руку в чашу, я обнаружил, что там нет мешалки, вызывающей движение; я просто почувствовал сильную вибрацию. Я расслабил руку и обнаружил, что она начала двигаться вместе со всеми другими предметами в чаше. Вся чаша вибрировала смещенным образом, так что предметы внутри стремились двигаться по кругу. Это выталкивало язычки и бегунки на набор рельсов, форма которых использовала асимметрию объектов, чтобы только те детали, которые попали на рельс в правильной ориентации, могли пройти к следующему этапу.

Полуавтоматизированный процесс

Несмотря на высокий уровень автоматизации на этой фабрике, многие рабочие, которых я видел, выполняли одну операцию. Они подавали язычки для другого типа застежки-молнии в устройство, соединенное с другой вибrotchашей, содержащей бегунки, а устройство соединяло бегунки и язычки.

Конечно, я спросил: «Почему у одних молний процесс сборки полностью автоматизирован, а у других полуавтоматический?»

Ответ, как оказалось, очень тонок, и все сводится к форме.



Обратите внимание на различие между этими двумя язычками, указанное стрелками.

Один крошечный выступ, едва видимый, был разницей между полной автоматизацией и необходимостью, чтобы человек соединял миллионы бегунков и язычков. Чтобы понять почему, давайте рассмотрим один критический шаг в работе вибrotchаша. Рабочий любезно остановил вибrotchашу, отвечавшую за сортировку язычков в правильную ориентацию для полностью автоматического процесса, чтобы я мог сфотографировать ключевой шаг.



Язычки проходят через вибrotchашу.

Когда язычки обходили рельс, их ориентация была случайной: одни смотрели вправо, другие влево. Но операция соединения должна вставлять бегунок только в меньшее из двух отверстий. Этот крошечный выступ позволял гравитации заставить все язычки висеть в одном направлении, когда они падали на рельс влево.

В полуавтоматизированной конструкции молнии этого выступа нет; в результате конструкция слишком симметрична, чтобы вибrotchаша могла выравнивать язычок. Я спросил владельца фабрики, сэкономит ли добавление крошечного выступа этот труд, и он сказал, что безусловно.

В этот момент мне показалось ослепительно очевидным, что у всех молний должен быть этот крошечный выступ, но проектировщик молнии не хотел его иметь. Хотя такой выступ очень мал, потребители могут ощущать тонкие бугорки, и некоторые воспринимают это как дефект дизайна. В результате проектировщик настоял на идеально гладком язычке, который, соответственно, не имел особенности, позволяющей легко и надежно обеспечить автоматическую ориентацию.

Ирония дефицита и спроса

Мне хочется представить, что большинство людей, посмотрев пару минут, как человек соединяет язычки с бегунками, были бы вполне готовы потерпеть крошечный бугорок на кончике своей молнии, чтобы избавить другого человека от судьбы вручную выравнивать язычки с бегунками по восемь часов в день. В качестве альтернативы, полагаю, инженер мог бы потратить бесчисленные часы на попытки разработать более сложный метод выравнивания язычков и бегунков, но здесь есть две проблемы:

- Клиент молнии, вероятно, не стал бы платить за эту работу.
- Вероятно, в чистом итоге дешевле платить неквалифицированному труду за ручное выполнение сортировки.

Этот владелец фабрики молний уже автоматизировал все остальное на предприятии, так что я думаю, они тоже долго и тщательно размышляли над этой проблемой. Моя догадка в том, что роботов дорого строить и обслуживать; люди самовоспроизводятся и в основном сами себя поддерживают. Помните третий входной материал фабрики - рис? Запасные части любого робота должны быть дешевле риса, чтобы робот заслужил место на полу этой фабрики.

В реальности, однако, объяснять эту идею конечным клиентам слишком трудно; фактически на рынке происходит прямо противоположное. Сборка гладких молний требует дополнительного труда, поэтому молнии стоят дороже; следовательно, они чаще попадают в продукты высокого класса. Это дополнительно укрепляет представление о том, что действительно гладкие молнии без крошечного язычка на них должны быть результатом контроля качества и внимания к деталям.

Мой мир полон таких маленьких разочарований. Например, большинство клиентов воспринимает пластики с зеркальной отделкой как более качественные, чем пластики с сатиновой отделкой. Между структурными характеристиками этих двух пластиков нет функционального различия, но изготовление чего-то с зеркальной отделкой требует намного больше усилий. Инструменты для литья под давлением нужно кропотливо и тщательно полировать, а на каждом шаге на фабрике рабочие должны носить белые перчатки. Горы пластика бракуются из-за волосковых дефектов, а на зеркальные поверхности накладываются дополнительные пленки пластика, чтобы защитить их при перевозке.

И что пользователи делают первым делом после всех этих усилий и всех этих отходов? Они оставляют свои грязные отпечатки пальцев по всей зеркальной отделке. В течение минуты после извлечения продукта из коробки все эти усилия сводятся на нет. Или, что еще хуже, пользователь оставляет защитную пленку на месте, что в итоге дает косметический эффект хуже, чем сатиновая отделка.

Сравните это с пластиком с сатиновой отделкой. Сатиновые поверхности не требуют защитных пленок, с ними проще обращаться рабочим и пользователям, они служат дольше и дают гораздо лучший выход годных изделий. В руках пользователя они скрывают мелкие царапины, отпечатки пальцев и пылинки. Можно утверждать, что сатиновая отделка дает лучший долгосрочный опыт использования, чем зеркальная.

Но зеркальная отделка, конечно, красиво выглядит на фотографиях и витринных дисплеях!

Примечания

[1] Я честно не знаю, как они называются, так что да, я продолжу называть их именно так.

Глава 3. Цех

Предыдущие две главы были наполнены историями о моем личном опыте: как я учился, ошибался и рос вместе с производственной экосистемой в районе Большого Шэньчжэня. В январе 2013 года, после того как я уже освоился, MIT Media Lab попросила меня начать наставлять аспирантов по цепочке поставок и производству, и я провел для них тур по Шэньчжэню (тот самый тур, во время которого я познакомился с Акибой и посетил фабрику USB-флешек). Эта глава - попытка сжать все, чему я учил на протяжении нескольких недель, в пару десятков страниц.

Задачи и компромиссы в малосерийном производстве отличаются от задач хорошо финансируемых корпоративных проектов, где прототипирование идет в масштабе тысяч единиц. Я понял это со временем, но не у каждого есть шесть лет, чтобы пройти через все ошибки новичка, спотыкаясь на каждом шагу. Если вы уже находитесь в быстро движущемся технологическом стартапе, у вас, вероятно, вообще нет роскоши что-либо исследовать. Уроки этой главы применимы к любому, кто хочет довести аппаратный продукт от первоначального прототипа до умеренных объемов (возможно, сотен тысяч единиц). Воспринимайте это резюме как общий ориентир, а не как подробную дорожную карту. Дьявол всегда в деталях, и одна из приятных сторон создания новых, инновационных аппаратных продуктов в том, что новые и интересные задачи никогда не заканчиваются.

Как составить ведомость материалов

Большинство мейкеров, пытающихся быстро увеличить выпуск, понимают, что единственный практический путь вперед - передать производство на аутсорсинг. Если бы только аутсорсинг был таким же простым, как схема + деньги = продукт!

Работаете ли вы со сборочным цехом за углом или отправляете работу в Китай, ясная и полная ведомость материалов (BOM) - первый шаг к передаче производства подрядчику. Каждое-каждое допущение, которое вы делаете относительно своей печатной платы, вплоть до цвета паяльной маски, должно быть однозначно прописано, чтобы третья сторона могла точно воспроизвести ваш проект. Отсутствующая или неполная документация - главная причина задержек производства, дефектов и превышения бюджета.

Простая BOM для велосипедного сигнального фонаря

Для примера предположим, что вы провели успешную кампанию на Kickstarter для велосипедного сигнального фонаря. В нем есть схема, использующая таймер 555, чтобы мигать небольшим массивом светодиодов. После отличной маркетинговой кампании нужно за несколько месяцев выполнить несколько сотен заказов.

Поначалу BOM для велосипедного фонаря, автоматически сгенерированная инструментом проектирования вроде Altium, могла бы выглядеть так:

Количество	Комментарий	Позиционное обозначение
1	0.1 uF	C1
1	10 uF	C2
3	белый LED	D1, D2, D3
1	2N3904	Q1

Количество	Комментарий	Позиционное обозначение
1	100	R1
2	20k	R2, R4
1	1k	R3
1	таймер 555	U1

Очень базовая BOM велосипедного сигнального фонаря.

Этой BOM, вместе со схемой, вероятно, достаточно, чтобы любой выпускник американской программы по электротехнике воспроизвел прототип, но для расчета производственной стоимости она далеко не адекватна. Эта версия BOM охватывает только электронику. Полная BOM для мигающего LED-фонаря также должна включать PCB, батарею, детали пластикового корпуса, линзу, винты, любую маркировку (например, серийный номер), руководство и упаковку (например, пластиковый пакет плюс картонную коробку). Ей также может понадобиться мастер-короб для совместной отправки нескольких LED-фонарей, потому что один LED-фонарь в индивидуальной коробке слишком мал, чтобы отправлять его отдельно. Хотя картонные коробки дешевы, они не бесплатны, и если их не заказать вовремя, складской запас будет стоять на погрузочной площадке, пока не доставят мастер-короб для финальной упаковки перед отгрузкой.

Также отсутствует следующая ключевая информация:

- Одобренный производитель для каждого компонента
- Допуск, состав материала и спецификация напряжения для пассивных компонентов
- Информация о типе корпуса для всех деталей
- Расширенные номера деталей, специфичные для каждого производителя

Рассмотрим каждый из недостающих пунктов подробнее.

Одобрённые производители

Настоящая фабрика потребует, чтобы вы предоставили список одобренных поставщиков (AVL), где указан допустимый производитель или производители для каждой детали на PCB. Производитель - это не дистрибьютор, а компания, которая действительно изготавливает деталь. Конденсатор, например, может быть сделан TDK, Murata, Taiyo Yuden, AVX, Panasonic, Samsung и так далее. Меня до сих пор удивляет, сколько BOM, которые я проверял, указывают DigiKey, Mouser, Avnet или какого-нибудь другого дистрибьютора в качестве производителя детали.

Может показаться глупым придираться к тому, кто делает конденсатор, но совершенно точно бывают ситуации, где изготовитель компонента имеет значение - даже для скромного конденсатора. Например, слепая замена фильтрующих конденсаторов в импульсном стабилизаторе, даже если заменитель имеет ту же номинальную емкость и напряжение, может привести к нестабильной работе и даже к тому, что платы загорятся.

Разумеется, некоторые детали в устройстве могут быть по-настоящему нечувствительны к производителю; в таком случае в графе AVL я ставил бы в BOM «любой/открыто» (any/open). Это особенно верно для деталей вроде подтягивающих резисторов. Такая отметка приглашает фабрику предложить от вашего имени своего предпочтительного поставщика.

Допуск, состав и спецификация напряжения

Для пассивных компонентов, помеченных «любой/открыто», всегда следует указывать следующие ключевые параметры, чтобы гарантировать закупку правильной детали:

- Для резисторов укажите как минимум допуск и мощность. Углеродный резистор 1 kOhm, с допуском 1 процент и мощностью 1/4 W - совсем иной зверь, чем проволочный резистор 1 kOhm, с допуском 5 процентов и мощностью 1 W!
- Для конденсаторов укажите как минимум допуск, номинальное напряжение и тип диэлектрика. Для специальных применений также указывайте отдельные параметры, такие как ESR или допустимый пульсирующий ток. Электролитический конденсатор 10 uF, с допуском 10 процентов, рассчитанный на 50 V, имеет на высоких частотах совершенно иные характеристики по сравнению с керамическим конденсатором 10 uF, с допуском 20 процентов, рассчитанным на 16 V.

Дроссели достаточно специализированы, и я вообще не рекомендую когда-либо помечать их в BOM как «любой/открыто». Для силовых дросселей базовые параметры, которые нужно указать, - это состав сердечника, сопротивление постоянному току, насыщение, нагрев и ток, но, в отличие от резисторов и конденсаторов, у дросселей нет стандарта корпуса. Кроме того, важные параметры, такие как экранирование и заливка компаундом, которые могут существенно влиять на работу схемы, часто неявно зашиты в номер детали; поэтому лучше полностью специфицировать дроссель. То же самое относится к RF-дросселям.

Форм-фактор электронного компонента

Всегда полностью указывайте форм-фактор, или тип корпуса, компонента. Плохо заданные или недоопределенные параметры корпуса могут привести к ошибкам сборки. Помимо базовых параметров вроде кода корпуса Electronic Industries Alliance (EIA) или JEDEC Solid State Technology Association (то есть 0402, 0805, TSSOP и так далее), при создании BOM учитывайте следующую информацию о корпусах:

Корпуса для поверхностного монтажа. Высота компонента может различаться, особенно для корпусов крупнее 1206 или для дросселей. Обращайте внимание, не вставляется ли плата в тесный корпус.

Корпуса для сквозного монтажа. Всегда указывайте шаг выводов и высоту компонента.

Для IC в целом старайтесь также указывать распространенное название, соответствующее корпусу, а не только внутренний код производителя. Например, код корпуса Texas Instruments типа "DW" соответствует корпусу SOIC. Такая проверка согласованности помогает защититься от ошибок.

Расширенные номера деталей

Проектировщики часто думают о компонентах сокращенными номерами деталей. Отличный пример - 7404. Почтенный 7404 - это шестиканальный инвертор, и он служит уже десятилетиями. Благодаря его повсеместности 7404 может использоваться среди инженеров-проектировщиков как общий термин для инвертора.

Однако при переходе к производству необходимо указывать такие сведения, как тип корпуса, производитель и логическое семейство. Полный номер детали для конкретного шестиканального инвертора может быть 74VHCT04AMTC; он задает инвертор производства Fairchild Semiconductor, из серии VHCT, в корпусе TSSOP, поставляемый в тубах. Дополнительные символы очень важны,

потому что небольшие вариации могут привести к большим проблемам: например, к расчету цены и заказу устройства в неправильной упаковке, после чего вы застрянете с катушкой непригодных деталей, или к тонким проблемам надежности.

Например, в контроллере робототехники, который я спроектировал (кодовое имя Kovan), я столкнулся с проблемой из-за ошибочной подстановки VHC в номере детали вместо компонента логического семейства VHCT. Использование детали VHC переключило пороги входа инвертора с TTL на совместимые с CMOS-логикой, и в результате некоторые устройства асимметрично реагировали на входные сигналы. К счастью, я поймал эту проблему до наращивания производства. Во всех остальных устройствах использовали правильную деталь, и я избежал огромного объема потенциальной переделки - или, что хуже, возвратов от недовольных клиентов. К счастью, единственной ценой ошибки стала переделка нескольких прототипов, которые я проверял перед производством.

Вот еще один пример того, как пропуск нескольких символов в номере детали может стоить тысячи долларов. Полностью заданный номер детали для импульсного стабилизатора LM3670 может быть LM3670MFX-3.3/NOPB. Если /NOPB опустить, номер детали все равно остается действительным и доступным для заказа - но эта версия использует свинцовый припой. Это может быть катастрофой для продуктов, экспортируемых в регион, который требует соответствия RoHS (то есть, помимо прочего, отсутствия свинца), например в Европейский союз.

X в номере детали - еще одна, более тонкая проблема. Номера деталей с X поставляются в катушках по 3 000 штук, а номера без X - в катушках по 1 000 штук. Хотя многие фабрики зададут вопрос об отсутствии /NOPB, поскольку обычно собирают документацию RoHS по мере закупки деталей, они редко отмечают количество в катушке как проблему.

Но количество в катушке должно вас волновать. Если вы планируете собрать только 1 000 продуктов, включение X в номер детали означает, что у вас останется 2 000 лишних LM3670. И да, платить за излишек придется вам, поскольку ваша BOM указала именно этот номер детали. Существует множество обоснованных причин заказывать детали с избытком, поэтому фабрики редко будут подвергать сомнению такое решение.

С другой стороны, детали, заказанные партиями по 1 000 единиц, немного дороже за штуку, чем детали, заказанные партиями по 3 000. Поэтому, если с ростом объемов вы оставите X за пределами номера, в итоге будете платить за деталь больше, чем нужно. В любом случае фабрика рассчитает вашу BOM ровно так, как она задана, и если ваши спецификаторы количества неверны, вы можете переплачивать - или, что хуже, терять деньги.

Итог? Важны каждая цифра и каждый символ, а невнимательность к деталям может стоить реальных денег!

BOM велосипедного сигнального фонаря: повторный взгляд

Имея в виду эти четыре пункта, посмотрите, как могла бы выглядеть правильная, полностью специфицированная BOM для примера с велосипедным сигнальным фонарем.

Кол-во	Значение	Корпус	Обозначение	AVL1	AVL1 P/N	MOQ	Срок поставки
1	0,1 мкФ, керамический, 25 В, 10%, X5R	0402	C1	Taiyo Yuden	TMK105BJ104KV-F	10000	8 нед.
1	10 мкФ, керамический, 16 В, 10%, X5R	1206	C2	TDK	C3216X5R1C106K(085AB)	2000	12 нед.

Кол-во	Значение	Корпус	Обозначение	AVL1	AVL1 P/N	MOQ	Срок поставки
3	белый LED, прозрачная линза	T-1 3/4	D1, D2, D3	Lumex	SSL-LX5093UWC/G	3000	12 нед.
1	2N3904	SOT-223	Q1	ON Semiconductor	PZT3904T1G	1000	6 нед.
1	100 Ом, 1/2 Вт, 5%	2010	R1	Panasonic	ERJ-12SF100U	5000	8 нед.
2	20k, 1/16 W, 1%	0402	R2, R4	any/open		10000	8 нед.
1	1k, 1/16 W, 5%	0402	R3	any/open		10000	8 нед.
1	NE555D	SOIC-8	U1	TI	NE555D	1000	4 нед.
1	PCB, FR4, 1,6 мм +/- 10%, зеленая паяльная маска, HASL, белая шелкография, 5 см x 8 см	PCB	TBD	FLASHYLIGHT_GERBER S_V1.ZIP		1000	4 нед.
1	ABS-пластик, нижняя часть корпуса, сатиновая отделка, без свинца, черный	TBD	FLASHYLIGHT_BOT_V 1.STEP			1000	16 нед. / 4 нед.
1	ABS-пластик, верхняя часть корпуса, сатиновая отделка, без свинца, черный	TBD	FLASHYLIGHT_TOP_V 1.STEP			1000	16 нед. / 4 нед.
1	поликарбонатный пластик, линза, зеркальная отделка, без свинца, прозрачный	TBD	FLASHYLIGHT_LENS_V1.STEP			1000	16 нед. / 4 нед.
4	винт M2x4, полукруглая головка Phillips, само нарезающий, 5 мм		any/open			4000	со склада

Кол-во	Значение	Корпус	Обозначение	AVL1	AVL1 P/N	MOQ	Срок поставки
1	контактная колодка батареи 9 В, красный и черный провода 15 см 26 AWG (выводы 5 мм)		Kaweei	CBS-150		5000	1 нед.
1	инструкция, лист А4, черно-белая, печать с двух сторон		any/open	flashylite_manual_v2.ai		1000	3 нед.
1	прозрачный PE-пакет 10 см x 12 см		any/open			1000	1 нед.
1	этикетка со штрихкодом, серийным номером и кодом даты, CODE39 5 мм x 15 мм		any/open	barcode_sample_v1.pdf		1000	1 нед.
1	картонная коробка 6 см x 6 см x 10 см, натуральный цвет, картон 50 lb		any/open	см. приложенный образец коробки		1000	1 нед.
0.02	транспортная коробка 60 см x 40 см x 20 см		any/open			100	1 нед.

Улучшенная BOM велосипедного сигнального фонаря.

Между BOM, которую любой инженер мог бы использовать для создания прототипа, как первая показанная мной BOM велосипедного сигнального фонаря, и такой BOM, которую любая фабрика могла бы использовать для массового производства продукта, есть большая разница. В частности, обратите внимание на столбцы MOQ (минимальный объем заказа) и Lead Time (срок поставки). Эти столбцы не имеют значения, когда вы собираете малосерийные прототипы, потому что обычно покупаете детали у дистрибьюторов, у которых мало ограничений по MOQ и есть складской запас для доставки на следующий день. Однако при масштабировании до производства вы экономите много денег, убирая накладные расходы дистрибьютора и покупая через оптовые каналы. В оптовых каналах MOQ и сроки поставки имеют значение.

Хорошая новость в том, что фабрика укажет MOQ и срок поставки как часть процесса расчета цены. Но вам будет полезно отслеживать эти параметры с самого начала. Если MOQ конкретного компонента очень велик, фабрике, возможно, придется купить огромное количество лишних деталей, что увеличит эффективную цену проекта. Если срок поставки детали очень длинный, вам, возможно, стоит рассмотреть переработку конструкции под деталь с более коротким сроком поставки. Использование деталей с более короткими сроками поставки не только экономит время, но и улучшает денежный поток: никто не хочет связывать деньги в компонентах с длинным сроком поставки за четыре месяца до выручки от продаж.

Эта BOM также включает несколько неэлектронных позиций - коробку, этикетку со штрихкодом и так далее, - которых не было бы в BOM инженерного прототипа. Эти разрозненные мелочи легко забыть, но отсутствие руководства пользователя в первоначальной BOM часто обнаруживается только тогда, когда финальный образец открывают для утверждения, что приводит к спешной попытке в последний момент поместить руководство в конечный продукт. Многие продукты задерживались просто потому, что руководство пользователя или оформление коробки не были закончены и утверждены вовремя, и ужасно неприятно, когда складской запас на сто тысяч долларов простаивает на складе из-за отсутствия листка бумаги.

Помимо правильной BOM, еще одна хорошая практика - предоставить фабрике эталонные образцы вашего продукта вместе с CAD-файлами. Эти рабочие прототипы позволяют фабрике принимать более разумные решения по поводу любых неоднозначностей в отправленной вами BOM. Ручная пайка еще одного экземпляра специально для фабрики может казаться раздражающей, но, по моему мнению, несколько часов пайки лучше недели обмена письмами с фабрикой.

Примечание. Когда вы строите бизнес-модель, детали и упаковка все равно не являются единственными затратами, которые нужно учитывать. Даже эта подробная BOM не перечисляет маржу фабрики, труд на сборку, финальную упаковку, доставку, пошлины и так далее. Я обсуждаю эти "мягкие затраты" в разделе "Выбор (и поддержание) партнера" на странице 107.

Планирование изменений и борьба с ними

Разумеется, даже если ваша конструкция идеальна, а BOM безупречна, проект все равно может потребовать изменений, если поставщики объявят о конце жизненного цикла (EOL), то есть прекратят выпуск выбранных вами компонентов. И давайте честно: всегда есть шанс, что ваши проектные допущения тоже не переживут контакта с настоящими потребителями.

Прежде чем переступить порог производства, формализуйте вместе с фабрикой процесс внесения изменений в проект. Хорошая практика - использовать письменные, формальные заявки на инженерные изменения (ECO), чтобы сообщать фабрике о любых изменениях после первоначального расчета цены. Как минимум шаблон ECO должен включать:

- Подробности по каждой измененной детали и краткое объяснение, почему изменение нужно
- Уникальный номер ревизии, чтобы в дальнейшем удобно ссылаться на изменение
- Метод фиксации того, что фабрика получила документы ECO

С ECO нужно быть обстоятельным, а не полагаться на неформальные письма, иначе закупщики на вашей фабрике могут купить неправильную деталь. Хуже того, фабрика может установить неправильную деталь, и целые партии вашего продукта придется списывать или переделывать. Даже после устранения проблемы вместе с инженерами фабрики я все равно оформляю формальный ECO и отправляю его производственному персоналу, чтобы формализовать выводы. Я ненавижу бумажную работу не меньше любого другого инженера, но в производстве одна маленькая ошибка может стоить десятки тысяч долларов, и эта мысль держит меня дисциплинированным в отношении ECO.

На следующей странице приведен настоящий ECO, который я выпустил и который в итоге сэкономил мне время и деньги.

Обратите внимание на дату в этом ECO: 27 февраля 2014 года. Этот ECO был выпущен прямо перед китайским Новым годом, когда фабрики уходят на каникулы на пару недель. После праздников внутри фабрик происходит значительная текучка неквалифицированной рабочей силы,

и поэтому появляется много возможностей для того, чтобы производственные задания потерялись и были забыты. Опасаясь, что ЕСО пропустят, я посоветовался с менеджерами после того, как фабрика возобновила производство, чтобы убедиться, что ЕСО не забыли. Они заверили меня, что он применен, но я все еще смутно тревожился, поэтому попросил фотографии печатной платы для подтверждения. Разумеется, в первой производственной партии изменения из моего ЕСО не оказалось.

Благодаря подробному ЕСО фабрика без труда признала свою ошибку, исправила весь производственный тираж и оплатила переделку. Но если бы я отправил распоряжение об изменении коротким письмом, не ссылаясь на конкретные партии или производственные задания, могло бы остаться достаточно неоднозначности, чтобы фабрика избежала оплаты переделки. Фабрика могла бы утверждать, что, как она поняла, я имел в виду применить изменение к будущему производственному тиражу, или просто могла бы отрицать получение подтвержденного распоряжения, поскольку электронные письма - довольно неформальный способ общения. В любом случае несколько минут документирования сэкономили дни переговоров и сотни долларов платы за переделку.

ENGINEERING CHANGE ORDER

ECO: [REDACTED]
[REDACTED]
[REDACTED]

Date: 27 February 2014

Sutajio Ko-Usagi Pte LTD
bunnie@[REDACTED].com

ECO number: 0001 version: 2
Project: [REDACTED]
Subassembly: [REDACTED] sensor and microcontroller [REDACTED]
Reference PO: PO-0018 and PO-0016

Background

Per request by engineer [REDACTED], pull-ups on inputs to the microcontroller and trigger sticker are to be modified to enhance flexibility and better target user use-cases.

On the microcontroller, R2, R3, and R4 (all 22M, 5%) shall be omitted, to allow the inputs to be used in applications that bar the presence of a pull-up.

On the trigger, R16 shall be changed from 10k, 1% to 22M, 5%, to allow for resistive-touch style sensing of the input pin.

CHANGE ORDER DETAILS

ORIGINAL		NEW		Comments
Designator	Value	Designator	Value	
R2	22M, 5% 0603	R2	DNP	BOM change only
R3	22M, 5% 0603	R3	DNP	BOM change only
R4	22M, 5% 0603	R4	DNP	BOM change only
R16	10k, 1% 0603	R16	22M, 5% 0603	BOM change only

MATERIAL DISPOSITION

No extra material needs to be ordered to execute this change.

Excess material resulting from this change shall be held by [REDACTED] and applied to future builds. No expected change to PO or cost for assembly.

Version history

version 2 – changed 0805 to 0603 for part footprints, was a typo.

Пример настоящего ECO, использованного в производстве. Благодаря формальному процессу документирования производственная путаница, связанная с этим ECO, была разрешена в мою пользу.

Текст изображения: ЗАЯВКА НА ИНЖЕНЕРНОЕ ИЗМЕНЕНИЕ - инженерное распоряжение об изменении. Дата: 27 февраля 2014. номер ECO: 0001, версия: 2. проект: замазано. узел: замазано, сенсор и микроконтроллер, замазано. связанные заказы PO: PO-0018 и PO-0016. обоснование: по просьбе инженера подтяжки на входах микроконтроллера и наклейка-триггер должны быть изменены, чтобы повысить гибкость и лучше соответствовать целевым пользовательским сценариям. На микроконтроллере R2, R3 и R4 (все 22M, 5%) должны быть исключены, чтобы входы можно было использовать в приложениях, где есть подтяжка. В наклейке-триггере R16 должен быть

изменен с 10k, 1% на 22M, 5%, чтобы разрешить резистивно-сенсорное распознавание входного вывода. детали изменения: R2, R3, R4 меняются с 22M, 5% 0603 на DNP, комментарий только изменение BOM; R16 меняется с 10k, 1% 0603 на 22M, 5% 0603, комментарий только изменение BOM. распоряжение материалами: для выполнения этого изменения не нужно заказывать дополнительный материал; избыточный материал, возникший в результате изменения, будет удержан замазанной стороной и применен к будущим сборкам; ожидаемых изменений РО или стоимости сборки нет. история версий: версия 2 - 0805 изменено на 0603 для посадочных мест деталей; была опечатка.

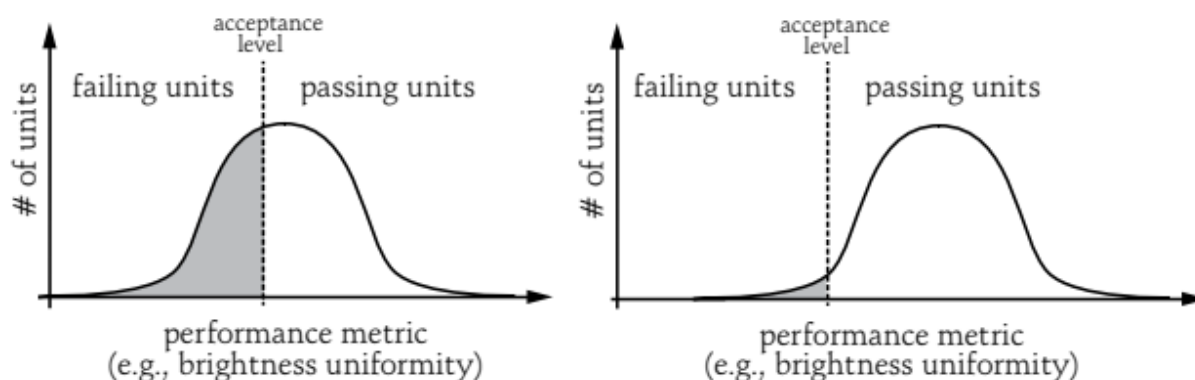
Оптимизация процесса: проектирование для производства

Пока вы проектируете финальный продукт и собираете BOM, важно также учитывать выход годных, то есть число исправных изделий, получающихся из производственного процесса. Для многих инженеров выход годных - скучная тема, но для предпринимателей успех или неудача отчасти будут определяться тем, добьются ли они разумного выхода. К счастью, вы можете помочь выходу годных, проектируя с учетом этого фактора.

Зачем DFM?

В отличие от ПО, каждая копия физического товара имеет небольшие несовершенства. Иногда несовершенства компенсируют друг друга; иногда они складываются и ухудшают характеристики. По мере роста объема производства какая-то доля продукта всегда оказывается непригодной для продажи. В устойчивой конструкции доля отказов может быть настолько мала, что функциональные тесты можно упростить, что ведет к дальнейшему снижению затрат. Напротив, конструкции, чувствительные к допускам компонентов, требуют обширного тестирования и приводят к серьезным потерям выхода годных. Переделка дефектных изделий влечет дополнительные затраты на труд и детали, в итоге размывая прибыль.

Таким образом, переработка конструкции для повышения устойчивости к нормальным производственным допускам - одна из главных задач перехода от инженерного стола к массовому производству. Этот процесс называется проектирование под производство (DFM).



Left, before DFM, almost half the units are not meeting the acceptance level and are therefore failing. Right, after DFM, the acceptance level is the same, but the average performance is improved, leading to most units passing.

Слева, до DFM, почти половина изделий не достигает уровня приемки и поэтому считается браком. Справа, после DFM, уровень приемки тот же, но среднее значение характеристики улучшено, и поэтому проходит большинство изделий.

Чтобы понять важность DFM, рассмотрите эти графики. Каждый изображает колоколообразную кривую, то есть предполагаемое статистическое распределение конкретного параметра. Ось x - интересующий параметр, а ось y - число произведенных изделий, попавших в данное значение параметра. Например, на графике яркости тысяч LED ось x была бы яркостью, а ось y - числом LED, достигающих заданной яркости. Положение колоколообразной кривой относительно критериев «пройдено/провалено» определяет итоговый выход годных изделий на производстве.

На правой кривой большинство LED достаточно яркие, и большая часть производственного запаса пригодна к отгрузке. На левой кривой, возможно, проходит 40 процентов LED. Учитывая, что большинство аппаратных компаний работают с валовой маржой примерно 30-50 процентов, списание 40 процентов материала означало бы конец бизнеса. В такой ситуации единственные жизнеспособные варианты - потратить время и силы на переделку LED, пока они не пройдут, или снизить требование к характеристике. Продукт не будет настолько качественным, как хотелось, но по крайней мере бизнес сможет продолжать работу.

Допуски, которые нужно учитывать

Цель DFM - гарантировать, что ваш продукт всегда проходит приемку и что вы никогда не оказываетесь перед неприятным выбором: снизить маржу, понизить стандарты качества или выйти из бизнеса. Но при применении DFM есть несколько аспектов компонентов, о которых стоит подумать.

Электронные допуски

Допуски пассивных компонентов - самые очевидные допуски, под которые нужно проектировать. Если истинное значение резистора может быть ± 5 процентов от его маркированного значения, убедитесь, что остальная часть вашей схемы справится с крайними случаями.

Параметры активных компонентов из технического описания - такие как коэффициент усиления по току (h_{FE}) для биполярных транзисторов, пороговое напряжение (V_t) для полевых транзисторов (FET) и прямое напряжение смещения (V_f) для светодиодов - также могут широко варьироваться. Всегда читайте техническое описание и следите за параметрами с большим разрывом между минимальным и максимальным значениями, разрывом, который часто называют разбросом min-max. Например, min-max для h_{FE} у Fairchild 2N3904 находится в диапазоне от 40 до 300, а V_f у сверхъяркого светодиода Kingbright - между 2 и 2.5 V.

Если оставить номинальное рабочее напряжение в стороне, максимальный номинал напряжения компонента особенно важен для конденсаторов и входных цепей. Я стараюсь использовать конденсаторы, рассчитанные на удвоенное номинальное напряжение; например, где возможно, я использую конденсаторы 10 V для шин 5 V и конденсаторы 6.3 V для шин 3.3 V. Чтобы понять почему, рассмотрите диэлектрики керамических конденсаторов: их емкость уменьшается с ростом напряжения. В дизайнах, работающих близко к максимальному напряжению керамического конденсатора, рабочая емкость этого компонента будет на отрицательном краю диапазона допуска. Кроме того, входные цепи (любая часть схемы, в которую пользователь может что-то подключить) подвергаются жестким электростатическим разрядам и другим переходным перегрузкам, поэтому уделяйте особое внимание номиналам конденсаторов там, чтобы добиться желаемой надежности.

Наконец, после того как у вас сложилось хорошее представление о компонентах, которые вы будете использовать, при проектировании РСВ внимательно следите за шириной дорожек и вариациями стека слоев. Они повлияют на системы, которые требуют согласованного импеданса или работают с большими токами.

Механические допуски

Электронные допуски, однако, не конец ваших забот; механические допуски тоже важны. Ни РСВ, ни корпуса не выйдут точно правильного размера, поэтому проектируйте корпус с некоторым зазором. Если конструкция корпуса имеет нулевой допуск к размерам РСВ, в половине случаев фабрика будет силой вставлять РСВ в корпуса, когда либо РСВ отрезана чуть крупнее, либо корпус вышел чуть меньше. Это может вызвать непреднамеренное механическое повреждение схемы или корпуса.

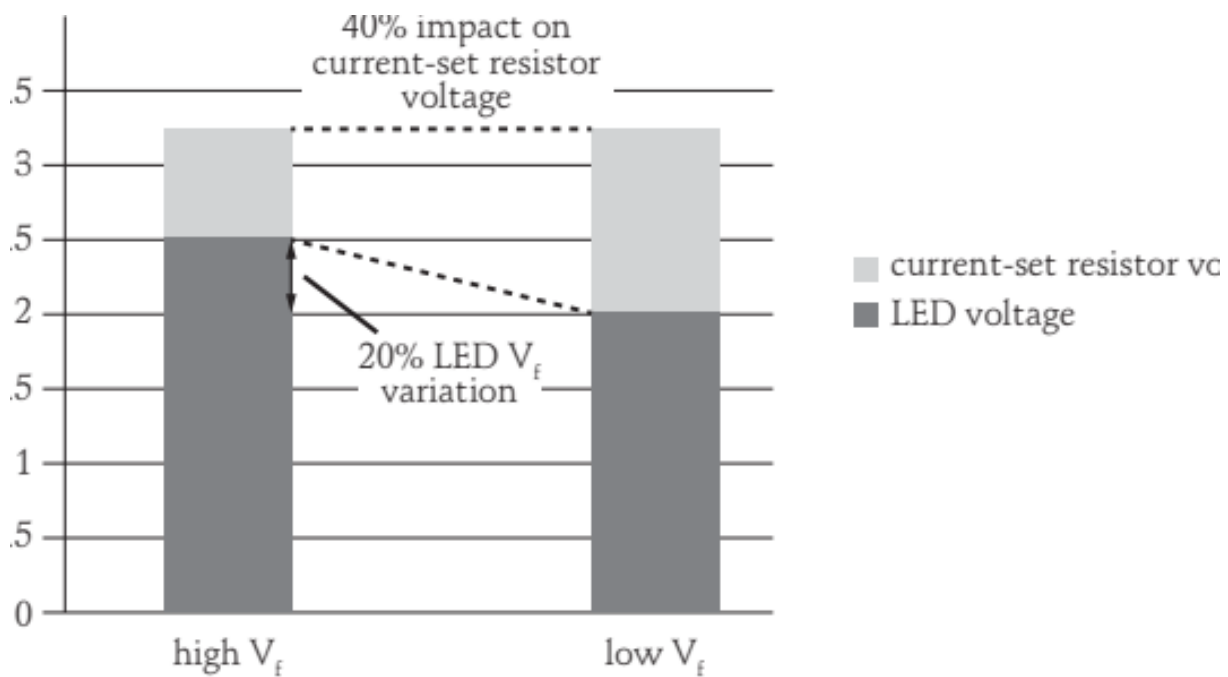
И не забывайте о косметических дефектах! Любой производимый продукт подвержен небольшим дефектам, таким как пыль, застрявшая в пластике, мелкие царапины, утяжины и потертости. Важно заранее согласовать с фабрикой критерии приемки таких дефектов. Например, вы можете сказать фабрике, что изделие может считаться "good", если на нем не больше двух точечных дефектов крупнее 0.2 mm, нет царапины длиннее 0.3 mm и так далее. У большинства фабрик есть определенная система, которую они приняли для описания и обеспечения соблюдения таких стандартов. Если вы обсудите эти параметры заранее, фабрика сможет выстроить производственный процесс так, чтобы избегать таких дефектов, вместо более дорогой альтернативы: собрать дополнительные изделия и выбросить те, что не соответствуют критериям, введенным уже на позднем этапе.

Разумеется, избегание дефектов не бесплатно. Чтобы ваш продукт был дешевле, избегайте глянцевых покрытий и подумайте об использовании матовых или текстурированных отделок, которые естественным образом скрывают дефекты.

Следование DFM помогает вашей прибыли

Чтобы представить DFM в реальном сценарии, вернемся к примеру с велосипедным сигнальным мигающим фонарем из раздела "Как составить ведомость материалов" на странице 74. Допустим, конструкция прототипа требует массив из трех LED, включенных параллельно, каждый со своим резистором для задания тока. Прямое напряжение смещения, или V_f , у LED при заданной яркости может различаться между устройствами примерно на 20 процентов; в этом случае разброс идет от 2.0 до 2.5 V.

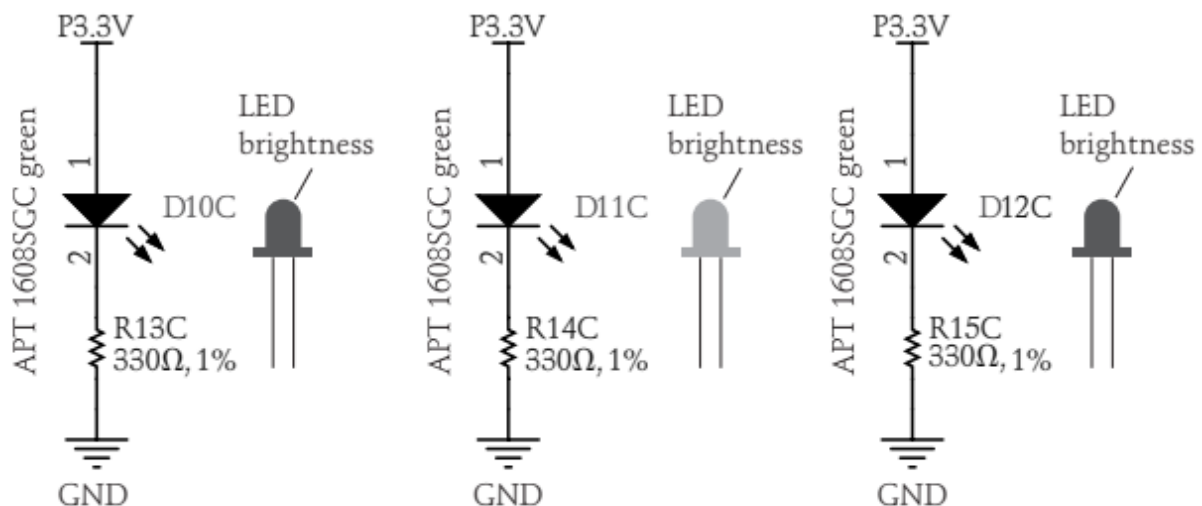
Схема, которая ограничивает ток через LED резисторами и называется резистивным ограничением тока, усилит эту вариацию. Это происходит потому, что эффективная схема сбрасывала бы меньшую часть напряжения на токоограничивающем резисторе, оставляя параметр, задающий ток (падение напряжения на резисторе), более чувствительным к вариации V_f . Поскольку яркость LED пропорциональна не напряжению, а току, протекающему через него, задание яркости LED с помощью резистивного ограничения тока может вызвать резкие несоответствия в яркости LED.



Comparing high V_f and low V_f corners

Сравнение крайних случаев высокого V_f и низкого V_f .

В этом примере 20-процентная вариация V_f LED (от 2.0 V до 2.5 V, согласно спецификации производителя LED) приводит к 40-процентному изменению напряжения на резисторе задания тока при фиксированном питании 3.3 V. Это вызовет 40-процентное изменение тока, протекающего через LED. Поскольку яркость прямо пропорциональна току, изменение проявляется как вариация воспринимаемой яркости между отдельными LED величиной до 40 процентов. Такой дизайн может хорошо работать большую часть времени; проблема станет выраженной только тогда, когда светодиод с высоким V_f окажется рядом со светодиодом с низким V_f .



Setting current for individual LEDs using resistors can lead to dramatic variations in brightness.

Задание тока для отдельных LED с помощью резисторов может привести к резким вариациям яркости.

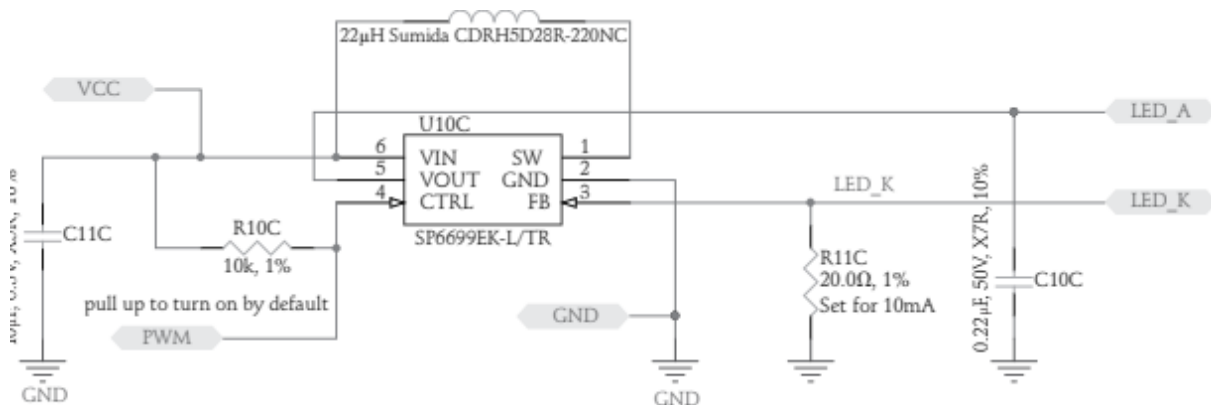
Один или два экземпляра, подготовленные на лабораторном столе во время разработки, могли выглядеть отлично, но в производстве у значимой доли могут оказаться настолько серьезные проблемы равномерности яркости, что устройства придется браковать. Поскольку большинству крупных аппаратных бизнесов приходится выживать на тонкой марже, потеря даже 10 процентов готовых товаров из-за дефектов - ужасный результат.

Один временный вариант - переделывать забракованные изделия. Фабрика может выявить LED, который в массиве слишком тусклый или слишком яркий, и заменить его на тот, что лучше соответствует соседям. Но такая переделка повысит затраты и приведет к неожиданному и неприятному счету в самый последний момент производственного цикла. Наивные проектировщики могут быть склонны обвинить фабрику в плохом качестве и спорить о том, кто должен нести расходы, но лучше заблаговременно избегать таких проблем, подвергая каждый проект проверке DFM и используя небольшой пилотный запуск, чтобы проверить выход годных перед тем, как запускать крупную партию.

Стоимость потерь из-за выхода годных количественно показывает, сколько денег стоит потратить на дополнительную схему, чтобы компенсировать нормальную вариативность компонентов. Например, продукт с себестоимостью проданных товаров (COGS) в 10 долларов и выходом 80 процентов годных изделий имеет эффективную стоимость одного продаваемого изделия 12.50 доллара, как рассчитано по этой формуле:

$$\text{Эффективная стоимость} = \text{COGS} \times \text{всего произведено} / \text{число годных изделий}$$

Увеличение COGS на 2.50 доллара для улучшения выхода годных до 100 процентов позволило бы вам выйти в ноль. Но, используя ту же формулу, трата дополнительного 1 доллара в COGS для улучшения выхода годных до 99 процентов фактически улучшила бы итоговую прибыль на 1.38 доллара.



A circuit to set the current on three LEDs, created by applying DFM

Схема задания тока на трех LED, созданная с применением DFM.

В случае велосипедного сигнального фонаря этот доллар можно было бы потратить на IC повышающего стабилизатора с обратной связью по току, например SP6699EK-L/TR, что позволило бы соединить LED последовательно, а не параллельно. Дизайн был бы намного сложнее и дороже, чем при использовании отдельных резисторов, но он гарантировал бы, что через каждый LED течет согласованный, одинаковый ток, поскольку все три LED включены в последовательную цепь с петлей обратной связи фиксированного тока. Это практически устранило бы вариацию яркости. Хотя стоимость повышающего стабилизатора намного выше, чем цент, потраченный на три токоограничивающих LED, улучшение производственного выхода годных с лихвой окупает дополнительные затраты на компоненты. Фактически этот прием - стандартная практика для приложений, которым нужна хорошая равномерность яркости от LED, например в подсветках

LCD-панелей. Типичная подсветка мобильного телефона использует около дюжины LED, но благодаря таким схемам вы никогда не видите светлых или темных пятен, несмотря на большие вариации V_f между составляющими LED.

Продукт позади вашего продукта

Наряду с работой с допусками, еще одна часто забываемая проектная обязанность - тестовая программа. Фабрика может обнаружить только те проблемы, которые ей поручено искать. Поэтому каждую функцию продукта необходимо тестировать, какой бы тривиальной она ни была. Например, в устройстве chumby у каждой обращенной к пользователю функции был явный фабричный тест, включая LCD, сенсорный экран, звук, микрофон, все порты расширения (USB, аудио), батарею, кнопки, ручки и так далее. Я следил за тем, чтобы тестировались даже самые простые кнопки. Хотя есть искушение пропустить тестирование таких простых компонентов, я гарантирую: все, что не протестировано, приведет к возвратам.

Я люблю называть фабричный тестер "продуктом позади вашего продукта". Потому что в некоторых случаях фабричный тестер сложнее и труднее в инженерном отношении, чем продукт, который вы пытаетесь продать. Это особенно верно для простых продуктов.

Реальная тестовая программа

В качестве примера рассмотрите эту наклейка с микроконтроллером от Chibitronics, проект, который я подробно обсуждаю в главе 8.

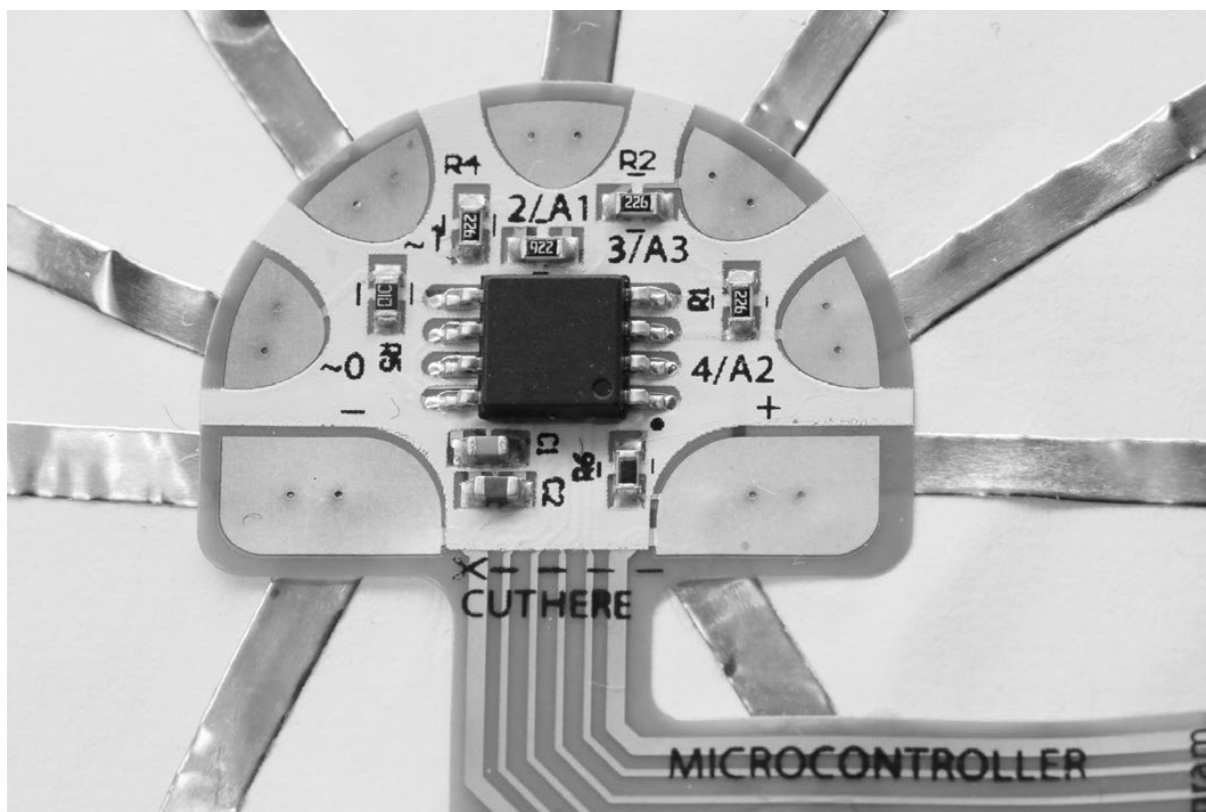


Схема микроконтроллера - на наклейке.

Эта схема очень проста: она состоит всего лишь из 8-битного AVR-микроконтроллера и горстки резисторов и конденсаторов. (Это также тот же продукт, о котором говорилось в примере ECO на странице 84.) Мы с моим соавтором примерно два дня набрасывали в Adobe Illustrator, прежде чем вывели финальную форму для этого продукта. Затем мы потратили около дня в Altium на проектирование схемы и около недели на код в Arduino IDE, чтобы создать его прошивку. В целом процесс разработки занял около двух недель. Для производства микроконтроллер объединяется с набором датчиков, способных обрабатывать звук, свет и касание, и в результате тестовая программа запускается на всех четырех одновременно.



Тестовая машина для наклейка с микроконтроллером Chibitronics.

Показанная тестовая оснастка состоит из 32-битного ARM-компьютера под Linux с графическим интерфейсом, выводимым на HDMI-монитор. За этим стоит FPGA, немного адаптерной электроники для создания аналоговых сигналов при тестировании и механическая сборка с пружинных контактов для контакта с наклейкой. Если разложить процесс проектирования этой оснастки на составные части, мы потратили:

- Несколько дней на проектирование в Altium
- Неделю на программирование FPGA в Xilinx ISE
- Пару недель на взлом Linux-драйверов
- Пару плотных месяцев на C++, чтобы создать интеграционный рамка на Qt
- Пару дней в SolidWorks, чтобы создать механический аппарат, удерживающий все вместе

В сумме создание тестера для наклейка с микроконтроллером заняло больше двух месяцев, по сравнению с двумя неделями на создание самого продукта.

Зачем проходить через все эти усилия? Потому что время - деньги, а обработка дефектов и возвратов стоит дорого. Тестер может обработать одну плату менее чем за 30 секунд; и за эти 30 секунд тестер должен запрограммировать два микроконтроллера; проверить датчики света, звука и касания; и подтвердить работу как при 5 V, так и при 3 V. Ручной тест всех этих операций мог бы занять несколько минут квалифицированного труда и был бы менее надежным. Благодаря этому тестеру мы получили ноль возвратов из-за дефектного материала. Кроме того, графический интерфейс на тестере позволяет фабрике очень легко определить, в какой именно точке схемы происходит отказ, что облегчает быструю переделку любого несовершенного материала.

Рекомендации по созданию тестовой программы

Как правило, для каждого продукта, который вы создаете, вы на самом деле создаете два связанных продукта: один для конечного пользователя и тест для фабрики. Во многих отношениях тест для фабрики должен быть таким же удобным для пользователя и защищенным от ошибок, как и сам продукт; в конце концов, тесты запускают не инженеры-электрики. Но связанную тестовую систему будет намного быстрее и легче построить, если в потребительский продукт заранее заложены адекватные возможности тестирования.

И нет, не отдавайте тестовую программу фабрике на аутсорсинг, даже если фабрика предлагает такую услугу. Фабрика часто не поймет ваш проектный замысел, поэтому ее тестовые программы будут либо неэффективны, либо будут проверять неправильное поведение. У фабрик также есть стимул пропускать как можно больше материала как можно быстрее, поэтому их тестовые программы обычно примитивны и недостаточны.

Вот несколько рекомендаций, которым стоит следовать при проектировании собственной программы:

Стремитесь к 100-процентному покрытию функций. Не упускайте простые или второстепенные функции, такие как индикаторные светодиоды или внутренний датчик напряжения. Когда я составляю список тестов, я использую подход «снаружи/изнутри». Сначала посмотрите на продукт снаружи: перечислите все способы, которыми потребитель может с ним взаимодействовать. Охватывает ли ваша тестовая программа каждую поверхность взаимодействия, пусть даже поверхностно? Зажжен ли каждый светодиод, нажата ли каждая кнопка, стимулирован ли каждый датчик и затронута ли каждое устройство памяти? Подтвержден ли каждый пункт в ваших маркетинговых материалах? Обещать «RF-чувствительность мирового класса» - не то же самое, что просто рекламировать наличие радиомодуля. Затем подумайте о внутренностях: по схеме посмотрите на каждый порт и рассмотрите ключевые внутренние узлы, которые стоит мониторить. Если в продукте есть микроконтроллер, проверьте, какие драйверы загружены, чтобы перекрестно проверить список тестов, и убедитесь, что никакие компоненты не забыты.

Минимизируйте дополнительные усилия по настройке. Оптимизируйте объем времени, необходимого для настройки теста для каждой единицы. Это часто делается через оснастки, использующие пружинные контакты или заранее выровненные массивы разъемов. Тест, требующий от оператора вручную прозондировать мультиметром дюжину контрольных точек или

вставить дюжину разъемов, отнимает время и склонен к ошибкам. Большинство фабрик в Китае могут помочь спроектировать оснастку за номинальную плату, но проектирование оснастки проще и эффективнее, если сам дизайн уже включает адекватные контрольные точки.

Автоматизируйте процедуру теста в линейный поток. Идеальный тест запускается одним нажатием кнопки и выдает результат «пройдено» или «провалено». На практике всегда есть точки остановки, требующие вмешательства оператора, но старайтесь не требовать слишком многого. Например, не требуйте от оператора вводить или выбирать SSID из списка во время каждого теста Wi-Fi-подключения. Вместо этого зафиксируйте SSID тестовой цели и жестко пропишите это значение в тестовом сценарии, чтобы цикл подключения был автоматическим.

Используйте значки и цвета, а не текст, для коммуникации с операторами. Грамотность оператора на конкретном языке не гарантирована.

Используйте журналы аудита. Записывайте результаты тестов, сопоставленные с серийными номерами устройств, включив сканер штрихкодов в тестовую оснастку. Альтернативно заставьте устройство печатать талон с уникальным кодом времени или локально сохранять журнал аудита, чтобы доказать, какие единицы прошли тест. Логи помогут вам понять, что пошло не так, когда потребитель возвращает отказавший продукт, и позволят быстро проверить, что все продукты были протестированы. После восьмичасовой смены тестирования оператор может ошибиться, например случайно положить дефектную единицу в корзину «годные». Возможность проверить, что каждый отгруженный продукт был подвергнут полному тесту и прошел его, поможет выявить и изолировать такие проблемы.

Предусмотрите простой механизм обновления. Как и в любой программе, в тестовых программах бывают ошибки. Тесты также должны развиваться по мере того, как ваш продукт патчится и обновляется. Имейте механизм обновлять и исправлять тестовые программы без личного визита на фабрику. Многие мои тестовые оснастки могут «связываться с домом» через VPN, и я могу зайти по SSH прямо в оснастку, чтобы исправить ошибки. Даже моя самая простая оснастка использует ноутбук с Linux (или эквивалент) в качестве ядра. Отчасти это потому, что Linux проще обновлять и поддерживать, чем заказной микроконтроллер, требующий специального адаптера для обновлений прошивки.

Эти рекомендации легко реализовать, если ваш продукт спроектирован с учетом тестируемости. Большинство продуктов, которые я проектирую, работают под Linux, и я использую процессор внутри самого продукта, чтобы запускать большинство тестов и помогать управлять тестовым пользовательским интерфейсом. Для продуктов, у которых нет поверхностей пользовательского взаимодействия, можно использовать Android-телефон или ноутбук, подключенный через Wi-Fi или последовательный порт, чтобы отрисовывать тестовый пользовательский интерфейс.

Тестирование и валидация

Производственные тесты предназначены для проверки ошибок сборки, а не параметрических вариаций или проблем дизайна. Если тест отбраковывает устройства из-за нормальных параметрических вариаций компонентов, либо покупайте компоненты лучше, либо переделывайте дизайн.

Для потребительских продуктов вам не нужно запускать пятиминутный комплексный тест RAM на каждой единице. В теории ваш продукт должен быть спроектирован достаточно хорошо, чтобы, если все спаяно правильно, RAM выполняла свою работу. Быстрого теста, проверяющего, что нет застрявших или оборванных адресных выводов, часто достаточно. Поставщики брендовых микросхем обычно имеют очень низкую дефектность, поэтому вы проверяете не кремний; скорее, вы проверяете паяные соединения и разъемы, а также ищите отсутствующие или перепутанные

компоненты. (Но если вы покупаете чипы-клоны или небрендовые, перемаркированные или частично протестированные устройства, чтобы сократить затраты, я рекомендую сделать мини-программу валидации для этих компонентов.)

Валидация переключателя

Чтобы показать различие между производственным тестированием и валидацией, посмотрим, как и то и другое может работать для переключателя.

Производственный тест для переключателя может просто попросить оператора несколько раз нажать переключатель и проверить, что ощущение правильное, а электрический контакт устанавливается через простой цифровой индикатор. Валидационный тест, напротив, может включать выбор нескольких устройств случайным образом, измерение контактного сопротивления переключателя мультиметром с точностью до пяти значащих цифр (также называемым мультиметром с пятью разрядами), выдерживание устройств при повышенной влажности и температуре в течение пары дней, а затем помещение устройств в автоматизированную оснастку, которая циклически переключает переключатели 10 000 раз. Наконец, вы можете снова измерить контактное сопротивление переключателя мультиметром с пятью разрядами и отметить любую деградацию контактного сопротивления в замкнутом состоянии.

Очевидно, такой уровень валидации невозможно выполнять на каждом произведенном устройстве. Скорее, программа валидации оценивает работу переключателя на протяжении ожидаемого срока службы продукта. Производственный тест, с другой стороны, просто убеждается, что переключатель собран правильно.

Примечание. Хорошая практика - повторно запускать валидационные тесты на паре случайно выбранных изделий из каждых нескольких тысяч произведенных изделий. Есть формулы и таблицы, которые можно использовать, чтобы вычислить, какой объем выборки нужен для достижения определенного уровня качества; просто поищите онлайн "таблица валидационных тестов производства".

Но сколько тестирования достаточно? Один порог тестирования можно вывести через аргумент стоимости. Каждый дополнительный запуск теста влечет затраты на оборудование, инженерные затраты и переменную стоимость времени тестирования. В результате тестирование подчиняется убывающей отдаче: в какой-то момент дешевле просто принять возврат продукта, чем тестировать больше. Естественно, планка тестирования намного выше для медицинского или промышленного оборудования, потому что ответственность, связанная с неисправным оборудованием, тоже намного выше. Аналогично сувенирный продукт, предназначенный для бесплатной раздачи, может требовать гораздо меньше тестирования.

Проектирование тестового приспособления

Последняя мысль: всегда применяйте основательную инженерию к проектированию тестовой оснастки. Когда я работал над chumby 8, возникла проблема: адаптер 50-контактного плоского гибкого кабеля демонстрировал случайные отказы из-за холодных паяных соединений. Я попросил фабрику построить тест для проверки адаптеров. Их решение состояло в том, чтобы повесить LED на каждый вывод адаптера, подать тестовое напряжение на одну сторону кабеля и смотреть, какие LED не загораются на другой стороне. Холодные паяные соединения не были просто открытыми или закрытыми; некоторые просто имели высокое сопротивление. Тока хватало, чтобы зажечь LED, но сопротивления также хватало, чтобы вызвать отказ в дизайне.

Фабрика предложила купить 50 мультиметров и подключить их к каждому выводу, чтобы вручную проверять сопротивление, что было бы дорого и подвержено ошибкам. Неразумно ожидать, что оператор будет смотреть на 50 дисплеев сотни раз в день и сможет надежно находить числа вне спецификации. Вместо этого я решил соединить контакты через адаптер последовательной цепочки и использовать один мультиметр для проверки суммарного сопротивления этой последовательной цепочки. Соединив контакты последовательно, я мог проверить все 50 соединений одним числовым измерением, в отличие от субъективного наблюдения яркости LED.

Как показывает этот случай, есть хорошие и плохие способы реализовать даже такой простой тест, как проверка холодных паяных соединений на кабельном адаптере. Все более сложные компоненты требуют все более тонких тестов, и есть реальная ценность в использовании инженерных навыков для создания эффективных, но защищенных от ошибок тестов.

Поиск баланса в промышленном дизайне

Даже если ваш продукт проходит все валидационные тесты с блеском, он все равно может оказаться неуспешным, если потребители его не хотят. Помните: внешность продает. С точностью примерно до фактора два производительность CPU или объем RAM в коробке менее важны для типичного потребителя, чем то, как устройство выглядит. Устройства Apple требуют солидную премию отчасти из-за своего гладкого промышленного дизайна, и многие продуктовые дизайнеры стремятся в своих продуктах повторить успех сэра Джонатана Айва, директора по дизайну Apple.

В промышленном дизайне, процессе проектирования того, как продукт будет выглядеть, до того как он будет реально сделан, существует много школ мысли. Одна школа призывает образ монашеского дизайнера, который создает красивую, чистую концепцию, и производственных инженеров, которые портят чистоту дизайна, когда подстраивают его под функциональность. Другая школа призывает прагматичного дизайнера, который тесно работает с производственными инженерами, чтобы выковать суровые компромиссы и получить недорогой дизайн с высоким выходом годных.

По моему опыту, ни одна крайность не убедительна. Монашеский подход часто приводит к непроизводимому продукту, который либо поздно выходит на рынок, либо дорог в производстве. Прагматичный подход часто приводит к продукту, который выглядит и ощущается настолько дешевым, что потребителям трудно приписать ему значимую ценность. Настоящий трюк - понять, как найти баланс между ними, и начинается он с того, чтобы попасть на фабрику и понять, как все делается. Вот пара примеров того, что я узнал о том, как разные фабричные процессы влияют на этот баланс, на примере Chumby и Arduino.

Отделка chumby One

Подгонка и отделка трудны, и потому они становятся отличительными точками во внешнем виде продукта. Когда я работал в Chumby, мы хотели, чтобы финальный продукт имел минималистичную, честную отделку. (Честные отделки демонстрируют естественные свойства используемых систем материалов и избегают применения красок и декалей.) Минималистичные дизайны очень трудно производить, потому что при меньшем количестве функций даже крошечные дефекты бросаются в глаза. Честные отделки тоже могут быть трудными, поскольку все заусенцы, литники, утяжины, линии сварки, царапины и линии течения, которые являются жизненными фактами производства, оказываются обнажены перед потребителем. В результате эта школа дизайна требует хорошо сделанной производственной оснастки, которую постоянно проверяют и обслуживают на протяжении производства.

Если у вас нет достаточно глубоких карманов, чтобы инвестировать в новое оборудование и возможности от имени вашей фабрики (то есть если вы не компания Fortune 500), первый шаг - изучить доступный словарь. Словарь дизайна определяется возможностями фабрики или фабрик, производящих товары: какие материалы вы можете получить, какая finish возможна, какие допуски достижимы и какие технологии крепления существуют. Все это сильно зависит от процессов, доступных вашей фабрике.

Поэтому я считаю, что личное посещение фабрики на раннем этапе процесса дизайна приводит к лучшему дизайну. После визита на фабрику вы отбросите часть словаря дизайна, но также обнаружите новый словарь. Инженеры, которые работают на фабрике изо дня в день, развивают процессные инновации, способные открыть новые возможности дизайна, которые вы не обнаружите, если не приедете.

Chumby One - конкретный пример влияния, которое производственные процессы могут оказывать на результат дизайна. В исходной концепт-графике вокруг переднего края был добавлен голубой акцент, чтобы напоминать облачко реплики, как в комиксах. Идея была в том, что chumby будет подписывать ваш мир фрагментами из интернета.



Готовый блок chumby One.

Но нанести голубую окантовку по приподнятой поверхности было очень трудно. Первая фабрика использовала краску, потому что передний край был недостаточно плоским, чтобы шелкография была вариантом. Тампопечать (процесс, при котором чернила переносятся с силиконовой подушки на объект) может работать с изогнутыми поверхностями, но выравнивание выступа на chumby One было недостаточно хорошим, и малейшее растекание чернил через край выглядело сбоку ужасно. Декали и наклейки также не могли обеспечить нужное нам выравнивание. В итоге был вырезан

небольшой канал для удержания краски, и фабрика создала акцент с помощью трафарета и аэрозольной краски.

Выход годных был ужасным. В некоторых партиях более 40 процентов корпусов chumby One выбрасывались из-за ошибок покраски. К счастью, пластик дешев, поэтому выбрасывание каждого второго корпуса после покраски имело итоговое влияние на себестоимость примерно \$0.35.



Два chumby One с плохой покраской.



Еще один chumby One с плохой покраской: перелив краски, слишком тонкий слой, недокрас.

В середине производства мы начали выпускать chumby One на резервной производственной площадке. У второй фабрики было другое оборудование для формования пластика, и, в отличие от первой фабрики, эта площадка могла делать двухкомпонентные формы. Двухкомпонентная форма требует вдвое большего числа инструментов по сравнению с однокомпонентной формой для литья под давлением, но позволяет отливать под давлением два разных цвета или даже два разных материала в одной и той же форме. На новой фабрике для тонкой голубой полоски мы попробовали двухкомпонентный процесс вместо покраски.

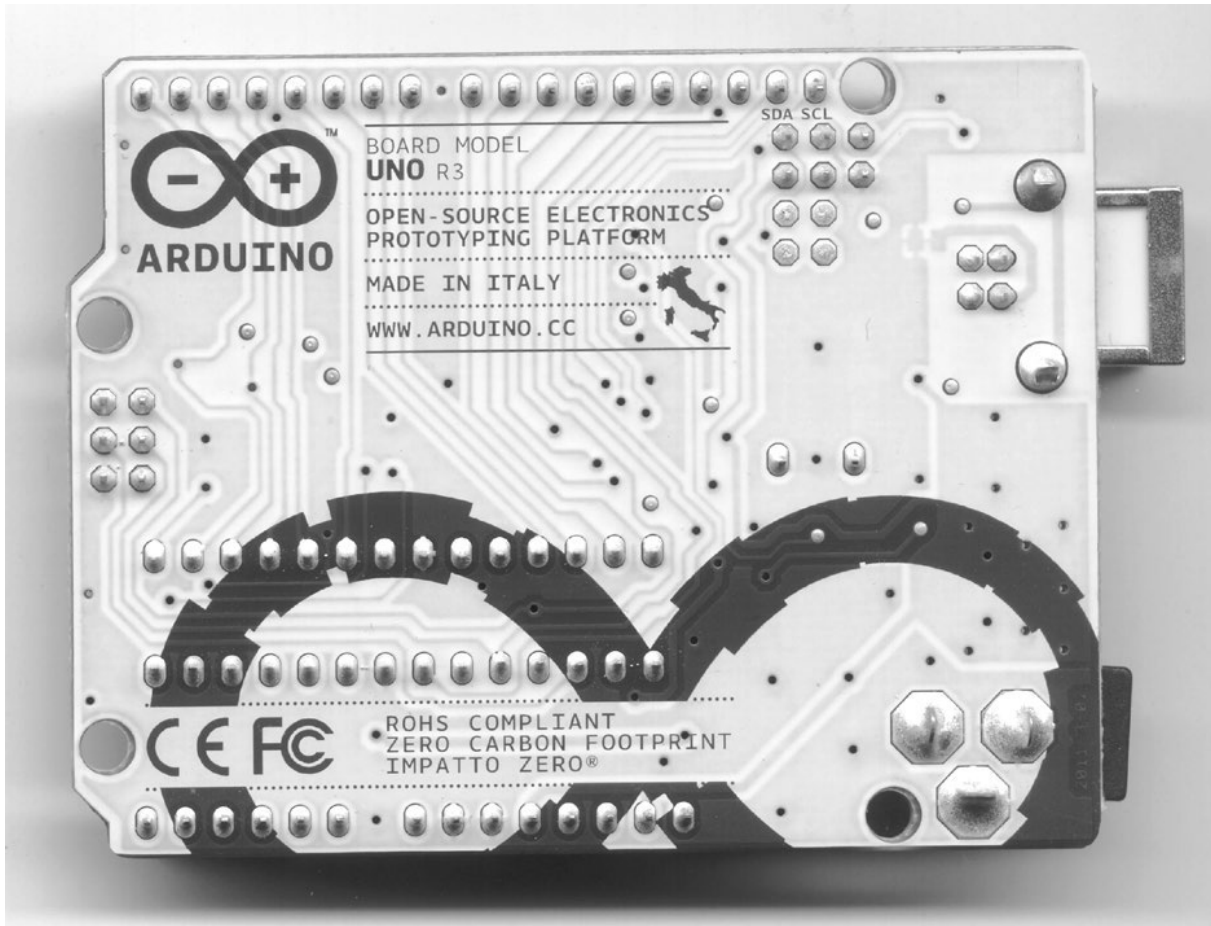


Идеальный выступ chumby One после двухкомпонентного литья.

Результаты были поразительными. Каждое изделие сходило с линии с четкой голубой линией, а отсутствие краски означало более чистую, более честную отделку. Но стоимость одного корпуса подскочила до \$0.94 за штуку при более дорогом процессе, несмотря на 100-процентный выход годных изделий. Было бы дешевле выбрасывать больше половины покрашенных корпусов, но даже лучшие покрашенные корпуса не могли сравниться с качеством отделки, которое давала двухкомпонентная пресс-форма.

Шелкография Arduino Uno

Еще один отличный пример того, как настройка фабричного процесса может улучшить внешний вид продукта, - материнская плата Arduino. Великолепно детализированная графика на обратной стороне, с контуром Италии и очень мелкими буквами, - не шелкография. Фабрика, которая делает эти платы, на самом деле наносит два слоя паяльной маски: один голубой и один белый.



Нижняя сторона Arduino Uno R3.

Когда платы Arduino производятся, паяльная маска наносится через фотолитографический процесс, который я описал в разделе "Где рождаются Arduino" на странице 44. Этот процесс дает графику с гораздо лучшим разрешением, стабильностью и выравниванием, чем шелкография. А поскольку внешний вид Arduino - это ее печатная плата, эта графика придает продукту отличительный, высококачественный облик, который трудно скопировать обычными методами обработки.

Таким образом, возможности фабричного процесса (будь то покраска против двухкомпонентного литья или двойная паяльная маска против шелкографии) могут реально влиять на воспринимаемое качество продукта без огромного влияния на стоимость. Однако фабрика может не осознавать полный потенциал своих процессов, и пока дизайнер не взаимодействует с площадкой напрямую, ваш продукт тоже не может задействовать этот потенциал.

К сожалению, многие дизайнеры не посещают фабрику, пока что-нибудь не пойдет не так. К этому моменту инструменты уже вырезаны, и даже если вы обнаружите классный процесс, который мог бы решить все ваши проблемы, часто уже слишком поздно.

Мой процесс дизайна

Дизайн - чрезвычайно личная деятельность, и поэтому каждый дизайнер выработает свой собственный процесс. Однако если вам нужна рамка для разработки своего процесса, вот общий процесс, который я мог бы использовать для разработки продукта при жестком стартап-бюджете:

1. Начните со скетчбука. Определите душу и идентичность дизайна и выберите систему материалов и словарь, которые подходят вашей концепции. Но не влюбляйтесь в них, потому что им, возможно, придется измениться.
2. Разбейте дизайн по системам материалов и определите фабрику, способную производить каждую систему материалов.
3. Посетите площадку и отметьте, что реально идет по производственным линиям. Не делайте никаких допущений на основе единичных образцов из демонстрационной комнаты. Практика ведет к совершенству, и от операторов до инженеров работники фабрики выполняют процедуры, которые делают ежедневно, намного лучше, чем редкую возможность процесса, которой они пользуются нечасто.
4. Переоцените свой дизайн на основе нового понимания того, что возможно на фабрике, и итерируйте. Вернитесь к шагу 1, если небольших корректировок недостаточно. Это стадия, на которой проще всего идти на компромиссы, не жертвуя чистотой вашего дизайна.
5. Набросайте детали дизайна. Выберите скользящие поверхности, линии разъема формы, где части корпуса защелкиваются друг с другом, отделки, системы крепления и так далее, исходя из того, что фабрика умеет лучше всего.
6. Передайте переработанный чертеж фабрике и работайте с ней, чтобы финализировать детали вроде уклонов, крепежных поверхностей, внутренних ребер и так далее.
7. Валидируйте дизайн с помощью 3D-печати и обширных проверок 3D-модели.
8. Определите функции, склонные к ошибкам допуска, и подрежьте начальную производственную оснастку так, чтобы допуск благоприятствовал модификациям, которые помогут вам минимизировать дорогостоящие изменения инструмента. Например, рассмотрите литье под давлением, где стальной инструмент является негативом пластика, который он формует. Удалять сталь из инструмента (добавлять пластик) проще, чем добавлять сталь (убирать пластик), поэтому нацельте первый пробный отлив на использование большего количества стали в критических размерах, а не слишком малого. Кнопка - один из механизмов, выигрывающих от такой настройки: точно предсказать по CAD или 3D-печатам, как будет ощущаться кнопка, трудно, и доведение тактильного ощущения обычно требует небольшой подрезки инструмента.

Разумеется, этот процесс - не набор жестких правил, которым нужно следовать. Возможно, вам придется добавлять или повторять шаги на основе своего опыта с вашей фабрикой, но если вы выбрали хорошую фабрику, это должно быть хорошей отправной точкой.

Выбор (и поддержание) партнера

Точно так же, как палочки из «Гарри Поттера», хорошая фабрика выбирает вас не меньше, чем вы выбираете ее, поэтому забудьте термин «поставщик» и замените его на «партнер». Если вы делаете все правильно, вы не просто отдаете фабрике указания; должен быть откровенный диалог о связанных компромиссах и о том, как можно улучшить производственный процесс. Это единственный способ получить наилучший возможный продукт.

Здоровые отношения с фабрикой также могут привести к лучшим условиям оплаты, что улучшает ваш денежный поток. В некоторых случаях фабричный кредит может напрямую заменить привлечение венчурного капитала, получение кредитов или финансирование через Kickstarter. В

результате я отношусь к хорошим фабрикам с тем же уважением, что и к инвесторам и партнерам в бизнесе. Чтобы дать представление, что это значит, вот несколько советов о том, как выбирать фабрику и работать с ней.

Советы по формированию отношений с фабрикой

Во-первых, выберите фабрику правильного размера для вашего продукта. Если вы работаете с фабрикой, которая слишком велика, вы рискуете потеряться в бюрократии и быть вытолкнутым с производственной линии более крупными клиентами в критические моменты. Работайте с фабрикой слишком маленькой - и она не сможет предоставить нужные вам услуги. Как правило, я выбираю самую большую площадку, где могу регулярно получать прямой доступ к lao ban (боссу фабрики), потому что если вы не можете поговорить с боссом, вы никто. Хороший знак, если lao ban присутствует на первой встрече, чтобы провести для вас экскурсию, и за обедом задает проницательные вопросы о вашем бизнесе.

Во-вторых, следуйте поговорке "солнечный свет - лучшее дезинфицирующее средство". Если фабрика не дает коммерческое предложение с открытой BOM, где явно раскрыта стоимость каждого компонента, процесса и маржи, я не буду с ней работать. Обсуждения снижения стоимости не могут работать без прозрачности, потому что иначе слишком много мест, где можно спрятать затраты. Аналогично, если обсуждения стоимости превращаются в игру "ударь крота", где сниженные затраты по одной строке необъяснимо всплывают в другой, бегите.

Последний совет относится прежде всего к стартапам. На ранних стадиях все знают, что ваши денежные запасы конечны. Даже если вы только что закрыли большой раунд финансирования, приходить на фабрику с важным видом и мешками денег - неустойчивый подход. Умные фабрики знают, что ваши денежные запасы ограничены, и если самая большая ценность, которую вы предлагаете фабрике, - это груды денег, ваша ценность ограничена; в лучшем случае она по-настоящему окупится только через годы, когда продукт будет отгружаться в больших объемах. Поэтому полезно пытаться приносить фабрике ценность неденежными способами.

Как бы глупо это ни звучало, быть приятным и конструктивным человеком очень помогает завоевать расположение вашей площадки. Производство - высокострессовый бизнес с низкой маржей, и всем на площадке весь день приходится иметь дело с трудными проблемами. Я обнаружил, что получаю лучшее обслуживание - даже лучше, чем клиенты с более глубокими карманами, - если отношусь к своим фабрикам так, как относился бы к приятельски настроенному знакомому, а не как к рабскому труду или простому субподрядчику. Ошибки случаются, и способность превратить плохую ситуацию в опыт обучения принесет вам пользу в день, когда вы совершите глупую (и, возможно, дорогую) ошибку.

Советы по коммерческим предложениям

Помимо открытости, знайте: если коммерческое предложение кажется слишком хорошим, чтобы быть правдой, часто так и есть. Когда договариваетесь с фабрикой о ценах, отступите на шаг и проверьте, имеет ли предложение смысл. Фабрики, теряющие деньги на сделке, не остановятся ни перед чем, чтобы вернуть их, и многие производственные страшные истории коренятся в нездоровых структурах затрат. Первый приоритет фабрики - выживание, даже если это означает подмешивание дефектных изделий в партии для повышения маржи или назначение начинающих инженеров на буксующий проект, чтобы лучше монетизировать опытных инженеров на более прибыльных клиентах.

Оценивая коммерческое предложение, убедитесь, что оно включает следующее:

- Цена каждой детали
- Избыточный материал для задания из-за минимальных объемов заказа (MOQ)
- Затраты на труд
- Накладные расходы фабрики
- Единовременные инженерные расходы (NRE)

Рассмотрим некоторые из этих пунктов подробнее.

Следить за излишками

Излишек - результат того, что я называю проблемой "сосисок и булочек". Сосиски продаются упаковками по 10, а булочки упаковками по 8. Если вы не купите 40 порций, у вас останутся лишние булочки или сосиски.

Точно так же многие компоненты поставляются только катушками по 3 000 штук. Сборка 10 000 штук требует 4 катушки, всего 12 000 штук, оставляя 2 000 лишних деталей. Фабрики могут покупать детали в нарезанной ленте или частичных катушках, но стоимость одной детали в нарезанной ленте намного выше, поскольку риск излишков материала переносится на дистрибьютора.

Однако излишек не всегда плох: его можно включить в будущие тиражи продукта. Пока ваш продукт поддерживает приличный темп производства, излишний запас компонентов должен регулярно превращаться в наличность. Однако в какой-то момент производство закончится или приостановится, и счет за излишек придет, ухудшив денежный поток. Если смета не содержит столбца излишков, фабрика может взять с вас плату за полную катушку, но оставить излишек для собственных целей; именно отсюда берется много товаров серого рынка в Шэньчжэне. Она также может просто прислать неожиданный счет за него позже, и он часто приходит в самый неподходящий момент - выручка от продукта уже прекратилась, а счета продолжают поступать. В любом случае лучше заранее знать полную бизнес-модель "от колыбели до могилы".

Выяснение затрат на труд

Затраты на труд дьявольски трудно оценивать, но хорошая новость в том, что для высокотехнологичных сборок труд обычно составляет малую долю общей стоимости. Стоимость труда при сборке небольших объемов прямолинейной платы с 200 деталями может составлять около \$2 или \$3 в Китае, тогда как стоимость сборки в Соединенных Штатах ближе к \$20 или \$30. Даже если цены на труд в Китае за ночь удвоятся, а в Соединенных Штатах уменьшатся вдвое, Китай все еще может быть конкурентоспособен.

Это контрастирует с товарами меньшей стоимости, уходящими из Китая (например, текстилем), где базовая стоимость сырья уже низка, поэтому затраты на труд составляют значительную часть финальной стоимости продукта. Обычно я не спорю особо о затратах на труд, поскольку итогом экономии на труде часто становится снижение качества, а слишком сильное давление на затраты на труд может заставить фабрику ухудшить качество жизни работников, урезав льготы.

Накладные расходы фабрики

Переговоры о фабричной марже - тоже в некоторой степени искусство, и жестких правил нет. Я дам здесь ориентиры, но из правила всегда есть исключения, и каждая фабрика может предложить вам особые условия в зависимости от обстоятельств. В конечном счете при рассмотрении фабричного предложения важно смотреть на общую картину и использовать здравый смысл.

То, что считается справедливой маржей для фабрики, зависит от того, сколько ценности она добавляет вашему продукту, и от объема производства. Определение "маржи" также различается в зависимости от площадки. Некоторые площадки включают брак, накладные расходы на обработку и даже расходы на исследования и разработку в маржу, тогда как другие могут выделять это отдельными строками.

В целом маржа находится в диапазоне от однозначных до низких двузначных процентов, в зависимости от объема, добавленной ценности и сложности проекта. Для производственных партий с очень малым количеством (менее 1 000 штук) вам также могут начислить плату за линию за партию. Этот сбор частично покрывает стоимость настройки сборочной линии только для того, чтобы разобрать ее через пару часов. Пропускная способность линии может быть очень высокой, производя сотни и тысячи изделий в день, но ее настройка тоже занимает дни.

Единовременные инженерные затраты

Расходы NRE - это разовые платежи, необходимые для настройки производственного тиража, такие как трафареты, программирование SMT, оснастка и тестовое оборудование. Обратите внимание, что повторное использование тестового оборудования между клиентами считается плохой практикой; если мультиметр требуется как часть производственного теста, не удивляйтесь, если счет за мультиметр добавят к NRE. У клиентов резко различаются стандарты обслуживания и использования тестового оборудования, поэтому хорошие фабрики с этим не рискуют.

Разные советы

То, с кем вы можете говорить, и насколько открыта фабрика в отношении затрат, конечно, являются ключевыми вопросами, но с опытом вы узнаете гораздо больше о работе с фабриками, что не попадает ни в какую конкретную категорию. В завершение - еще несколько важных моментов, которые стоит держать в голове при выборе фабрики.

Брак и выход годных

В идеале вы платили бы фабрике только за годные, доставленные изделия, а фабрика несла бы бремя дефектных изделий. Это дает фабрике стимул поддерживать высокое качество производства, потому что каждый процент дефектности съедает ее маржу. Но если в вашем дизайне есть изъян или его слишком трудно собрать, а дефектность высока, фабрика может начать отгружать изделия более низкого качества как отчаянную меру для достижения целей по производству и марже. Она также может начать продавать дефектные товары на сером рынке, чтобы вернуть затраты, что позже приведет к проблемам с репутацией бренда.

Чтобы избежать таких ситуаций, заранее договоритесь с фабрикой о том, как обрабатывать бракованные изделия или исключительную потерю выхода годных. Это может включать, например, отдельную строку "брак" внутри коммерческого предложения, чтобы явно учитывать дефектность.

Заказывайте больше изделий, чем подтвержденный спрос

Несмотря на все усилия всех участников, ошибки будут случаться, клиенты будут получать плохие устройства, и вам понадобятся дополнительные рабочие единицы для возвратов и обменов. Заказать 1 000 штук, чтобы выполнить Kickstarter-кампанию на 1 000 штук, означает, что если клиенты захотят вернуть или обменять устройства, сломанные при доставке, все, что вы сможете сделать, - вернуть деньги. Просто непрактично запускать фабрику, чтобы сделать дюжину замен.

Как общее правило, я заказываю запас на несколько процентов сверх числа изделий, которые нужно доставить клиентам, чтобы иметь резерв для обработки возвратов и обменов. Изделия, которые не понадобятся для возвратов, можно превратить в демонстрационные образцы во временное пользование или подарки для развития бизнеса, чтобы разогнать следующий набор заказов!

Доставка стоит денег

Следите за затратами на доставку. Эти сборы обычно не встроены в фабричное коммерческое предложение, но они влияют на вашу итоговую прибыль, особенно для малосерийных продуктов. Доставка FedEx - отличный способ сэкономить время, но она также очень дорога. Курьерские сборы легко могут смыть прибыль с небольшого проекта, поэтому управляйте этими затратами.

Примечание. Курьерские службы предлагают скидки частым отправителям, но вам нужно позвонить, чтобы договориться о специальных тарифах.

Учитывайте импортные пошлины

На компоненты, импортируемые в Китай без импортной лицензии, начисляется обязательная пошлина примерно 20 процентов от их стоимости. Общее правило для Китая: облагается пошлиной при импорте, не облагается при экспорте. Если что-то случайно отправлено через границу в Гонконг, ожидайте, что за возврат этого в Китай тоже придется заплатить пошлину.

Найдите таможенного брокера, чтобы искать способы экономить деньги; например, некоторые брокеры могут добиться налогообложения товаров по весу, а не по стоимости, что для микроэлектроники обычно выгодно. Я не разобрался во всех таможенных правилах, поскольку они, кажется, постоянно меняются. Каждый месяц, кажется, появляется новое правило, штраф, исключительный сбор или тариф, с которым нужно иметь дело. Также есть множество теневых способов завозить товары в Китай, но я лучше сплю ночью, зная, что делаю все возможное, чтобы соблюдать каждое правило.

Коммерческие предложения не включают пошлины, потому что фабрики по умолчанию предполагают, что у вас будет импортная лицензия. Импортные лицензии позволяют беспошлинно импортировать товары. Но они стоят несколько тысяч долларов, оформляются неделями и не оставляют места для гибкости, поскольку привязаны к точной BOM продукта. Небольшие заявки на инженерные изменения могут сделать импортную лицензию недействительной. Я знаю случаи, когда таможенники считали количество развязывающих конденсаторов на PCB, и если оно не совпадало с числом в лицензии, начислялся штраф, а лицензия аннулировалась. Даже отклонения в материале, используемом для облицовки декоративной коробки, могут аннулировать лицензию. Короче говоря, эта схема импортной лицензии благоприятствует крупносерийным продуктам и наказывает малосерийных производителей, поэтому ступайте осторожно.

Заключительные мысли

Поездка в Китай ради производства явно подходит не всем. Особенно если вы находитесь в Соединенных Штатах, накладные расходы на курьерские доставки, поездки, пошлины и ночные конференц-звонки быстро накапливаются. Как правило, небольшой американской компании часто лучше собирать РСВ в Соединенных Штатах при объемах ниже 1 000 единиц, и вы не начнете видеть ясные преимущества, пока объемы не достигнут, возможно, 5 000-10 000 единиц.

Эта математика смещается в пользу Китая, когда в игру вступают процессы вроде литья под давлением и сборки шасси, благодаря экспертизе китайских фабрик в этих трудоемких процессах. Точка безубыточности также может быть намного ниже, если вы живете в Китае или рядом с ним, поскольку курьерские расходы, поездки и влияние часового пояса составляют лишь малую долю того, чем они были бы из Соединенных Штатов. Это усиливается тем фактом, что местные эффективнее используют компонентную экосистему в Китае, что приводит к дальнейшему снижению затрат по сравнению с дизайном, произведенным только с использованием американских деталей.

С другой стороны, физически крупные узлы или системы, построенные с использованием множества компонентов, облагаемых пошлиной, может быть дешевле собирать внутри страны, поскольку они экономят на доставке и тарифах. В конце концов, сохраняйте открытость и старайтесь рассмотреть все возможные вторичные затраты и преимущества местного и зарубежного производства, прежде чем решать, где размещать производство.

Часть 2. Мыслить иначе: интеллектуальная собственность в Китае

Глава 4. Инновации гункай

Если термин "интеллектуальная собственность" звучит для вас как оксюморон, вы не одиноки. Если я дам вам яблоко и скажу: "Это ваше яблоко", смысл довольно ясен. Вы можете делать с этим яблоком что хотите: можете съесть его, продать или даже использовать семена, чтобы посадить яблоню и получить больше яблок, которые затем сможете продать или использовать, чтобы кормить свою семью. Но если я передам вам телефон и скажу: "Этот Apple iPhone ваш", вы владеете совокупностью атомов в вашей руке, но у вас крайне ограниченные права на программное обеспечение, патенты и товарные знаки - интеллектуальную собственность, - связанные с этим телефоном. В отличие от фрукта, вы не можете взять то, что внутри вашего iPhone, и использовать это знание как семя, чтобы сделать больше iPhone.

В Китае, однако, интеллектуальная собственность работает совсем иначе. Там вы могли бы (и люди так делают) использовать телефон как семя для собственных оригинальных работ. Два опыта, которые у меня были в Китае, открыли мне глаза на то, что не существует одного истинного пути обращения с интеллектуальной собственностью.

Я разбил экран своего телефона, и это было отлично

Моя первая история начинается, как многие мои приключения, с выхода из такси у пограничного пункта Futian по пути в Китай. Это был май 2014 года, и я направлялся в Шэньчжэнь, чтобы утрясти производственные планы для ноутбука Novena с открытым железом, о котором я подробнее расскажу в главе 7. Когда я выходил из такси, моя рука зацепилась за рюкзак, отправив телефон кувыркаться к бетонному тротуару. Когда телефон ударился о землю, я услышал сухой стук разбивающегося сенсорного экрана.

В мире нет лучшего места, чтобы разбить экран телефона, чем пограничный переход в Шэньчжэнь. В течение часа мне поставили новый экран умелыми руками в Hua Qiang Bei всего за \$25 - включая детали и работу.

Изначально я планировал заменить экран сам. Телефон все еще работал, поэтому я поспешно зашел на iFixit за подробностями о том, как заменить экран, а затем рванул в Hua Qiang Bei покупать запасные части и инструменты. Ларек, куда я пришел, назвал мне цену около \$120 за новый экран, но затем владелица магазина выхватила телефон из моих рук и запустила встроенную программу самотестирования, набрав #0# в интерфейсе набора номера.

Она подтвердила, что на моем OLED-дисплее нет битых пикселей и что дигитайзер все еще работоспособен, только треснул. Затем она предложила купить мой сломанный модуль OLED-дисплея и дигитайзера, но только если ее магазин сможет заменить мне экран. Я сказал, что это меня устроит, если я смогу посмотреть, чтобы убедиться, что они не заменят никакие другие детали.

Разумеется, у них с этим не было проблем. За 20 минут они разобрали мой телефон, сняли сломанный модуль, удалили клей с корпуса телефона, заменили клей, установили в телефон «новый» (предположительно восстановленный) модуль и собрали все обратно. Процесс включал фен для волос (использованный как термофен), обильное количество очистителя контактов (использованного для размягчения клея) и очень длинный ноготь большого пальца (вместо пластиковой лопатки или медиатора). К сожалению, я не мог фотографировать процесс, потому что устройство, которым я сделал бы снимки, лежало передо мной в разобранном виде.

Такова сила переработки и ремонта. Вместо того чтобы платить \$120 за экран и выбрасывать функциональный кусок электроники, я просто оплатил стоимость замены разбитого стекла. Я предполагал, что стекло на дигитайзере неотделимо от OLED, но, очевидно, эти умные ребята в Hua Qiang Bei нашли эффективный способ перерабатывать эти детали. В конце концов, основная часть стоимости модуля была в OLED-дисплее. Электроника сенсора тачскрина, которая также была привита к модулю, не пострадала при падении. Зачем тратить впустую совершенно хорошие детали?

Итак, экран моего телефона был разбит всего один час, и его починили дешевле, чем стоила бы доставка запасных частей в Сингапур (мою страну проживания). Такие опыты заставляют меня думать: почему подобных услуг нет в каждой стране? Что делает Шэньчжэнь настолько уникальным, что вы можете перейти от разбитого экрана к починенному телефону за полчаса и за сумму намного меньше стоимости месячного телефонного счета? Множество факторов вносят вклад в это явление, и большинство из них можно проследить к группе людей, называемой шаньчжай.

Шаньчжай как предприниматели

Шаньчжай в Китае изначально прославились как производители подделок продуктов вроде iPhone, поэтому популярная пресса исторически отмахивалась от них как просто от "баронов копирования". Но я думаю, что у них может быть нечто общее с командами вроде Хьюлеттом и Паккардом или Джобсом и Возняком в те времена, когда они работали в гаражах.

Кто такие шаньчжай?

Чтобы понять, почему я так думаю, полезно понять культурный контекст слова shanzhai. Shanzhai (山寨) происходит от китайских слов «горная крепость», но буквальный перевод немного вводит в заблуждение. Английский термин fortress вызывает ассоциацию с большим укрепленным сооружением или твердыней, возможно, рисуя в воображении башни замка и рвы. С другой стороны, его денотативное значение говорит, что это просто укрепленное место, и это ближе к исходному китайскому значению, которое относится к чему-то вроде пещеры или партизанского убежища.

В современном контексте shanzhai - это историческая аллюзия на людей, которые жили в таких убежищах, вроде Song Jiang и его 108 разбойников, группы вне закона, живших в XII веке. Один мой друг описал Song Jiang как нечто вроде Робин Гуда, встретившего Че Гевару. Он был мятежником и солдатом удачи, но при этом самоотверженным и добрым к нуждающимся. Рассказ популярен и сегодня; мой отец мгновенно узнал его, когда я спросил его об этом.

Современные шаньчжай-инноваторы - мятежные, индивидуалистичные, подпольные и самоуполномоченные - совсем как Song Jiang. Они мятежные в том смысле, что их прославляют за продукты-клоны. Они индивидуалистичны в том смысле, что испытывают нутряную неприязнь к крупным компаниям. Многие шаньчжай - бывшие сотрудники крупных компаний, как американских, так и азиатских, которые ушли, потому что были разочарованы неэффективностью своих работодателей. Они подпольные в том смысле, что как только шаньчжай "легализуется" и начинает вести бизнес напрямую через традиционные розничные каналы, он больше не принадлежит к братству шаньчжай. Они самоуполномоченные в том смысле, что это повсеместно крошечные операции, бутстрэпнутые на минимальном капитале, и их установка такова: "Если ты можешь это сделать, значит, и я тоже могу".

По оценкам, в 2009 году в Шэньчжэне работало 300 шаньчжай-организаций. Шаньчжайские лавки варьируются от пары человек до нескольких сотен сотрудников. Одни специализируются на процессах вроде изготовления оснастки, проектирования PCB, сборки PCB или внешней кастомизации мобильных телефонов, тогда как другие имеют более широкие возможности.

Поскольку shanzhai малы, им приходится быть эффективными, чтобы максимизировать выпуск. Одна мастерская численностью менее 250 сотрудников может выдавать более 200 000 мобильных телефонов в месяц при широком ассортименте моделей, иногда производя тиражи всего по несколько сотен штук. В совокупности shanzhai в районе Шэньчжэня в 2009 году производили, по оценкам, 20 миллионов телефонов в месяц. Это экономика, приближающаяся к миллиарду долларов в месяц. Большинство этих телефонов продаются на рынки третьего мира и развивающиеся рынки, такие как Индия, Африка, Россия и Юго-Восточная Азия.

Больше чем копии

Важно, что продуктовый портфель шаньчжай включает не только телефоны-клоны. Они инновировать и играют с дизайнами, чтобы делать также оригинальные продукты. Эти оригинальные телефоны интегрируют чудаческие функции вроде 7.1-стереозвука, двух SIM-карт, функционального держателя сигарет, объектива с большим зумом или встроенного UV LED для обнаружения фальшивых денег.

Shanzhai делают с аппаратурой то, что веб-мэшапы делают с мэтшапами. Мобильные телефоны, которые одновременно являются игрушечными Ferrari, и комбинации часы-телефон (с камерой!) - хорошие примеры: они не копируют какую-то одну идею, а смешивают IP из множества источников, создавая новую разнородную композицию, так что исходный материал все еще отчетливо узнаваем в конечном продукте. Как и многие веб-мэшапы, результат может казаться бессмысленным для массового рынка (как телефон-Ferrari), но быть чрезвычайно актуальным для узкой ниши длинного хвоста. В некотором смысле некоторые продукты shanzhai просто опережают свое время; часы-телефоны, которые я видел, например, предшествовали умным часам на несколько лет.



Сверху: передняя сторона телефона, сделанного похожим на пачку сигарет.



Сверху: задняя сторона телефона, сделанного похожим на пачку сигарет.



Снизу слева: умные часы на Android, которые, в отличие от Apple Watch, включают в часы телефон, способный совершать вызовы.



Снизу справа: разработанный shanzhai "baby iPhone" под Android, показанный рядом с Apple iPhone 6 для масштаба.

Правила IP, поддерживаемые сообществом

Шаньчжай также используют концепцию, называемую открытой BOM: когда один шаньчжай строит что-то новое, он делится ведомостью материалов и другими проектными документами с остальными. Если продукт основан на существующем продукте, любые сделанные им улучшения также распространяются. Эти правила контролируются сарафанным радио внутри сообщества до такой степени, что если кого-то ловят на обмане, его изгоняют из шаньчжайской экосистемы.

В Китае эта система воспринимается очень позитивно. Например, однажды я слышал, как местный сказал, что это здорово, что shanzhai могут не только клонировать iPhone, но и улучшить оригинал, дав батарею-клон, заменяемую пользователем. Закон США назвал бы такую деятельность незаконной и нарушающей права, но, учитывая плодородную культуру мэшапов в интернете, я не могу не задаваться вопросом, не может ли аппаратный мэшап быть чем-то хорошим. В Соединенных Штатах определенно есть восприятие, что если что-то странное происходит в Китае, оно должно быть плохим. Это предубеждение отбрасывает длинную тень на объективную оценку культурного явления, которое со временем может стать очень актуальным для Соединенных Штатов.

В некотором смысле шаньчжай - братья классического западного представления о хакерах-предпринимателях, но с отчетливо китайским поворотом. Моя любимая личная история о

шаньчжай - о человеке, которому принадлежит трехэтажный дом, которому я чрезвычайно завидую. Его спальня находится наверху, средний этаж - полноценная линия SMT-производства, а нижний этаж - розничная торговая точка для продуктов, произведенных этажом выше и спроектированных в его спальне. Вот это вертикально интегрированная цепочка поставок! Владение такой инфраструктурой определенно изменило бы то, как я инновирую. Я мог бы экономить на производственных затратах, сокращать время прототипирования и агрессивно оборачивать запасы, тем самым снижая требования к капиталу в запасах. А если бы мой магазин находился в городской локации с большим трафиком, я также мог бы убрать минимальную розничную маржу в 20-50 процентов, обычно требуемую американскими ритейлерами.

У меня есть теория, что когда объем знаний и масштаб рынков в Шэньчжэне достигнут критической массы, китайцы перестанут быть просто рабочими или копировальщиками. Они возьмут свои судьбы под контроль и в конечном счете станут лидерами инноваций. Эти истории о shanzhai и их мешапах - лишь верхушка айсберга с потенциалом изменить способ ведения бизнеса - возможно, не в Соединенных Штатах, но определенно на том огромном, неосвоенном рынке, который часто называют «остальным миром».

Телефон за \$12

Гибридные мобильные телефоны демонстрируют инновации шаньчжай и их готовность экспериментировать. Но, несмотря на все навороты, эти телефоны вполне доступны. Тогда один вопрос, который вы можете задать: насколько дешево можно сделать телефон?

Короткая прогулка к северо-восточному углу электронного района Hua Qiang Bei приводит вас в Mingtong Digital Mall. Это четырехэтажный лабиринт, набитый крошечными лавками, которые торгуют всеми видами причудливых телефонов с функциями, полезными в экономиках, где нет инфраструктуры стабильного электричества или кабельных сетей. Например, некоторые телефоны могут работать месяц благодаря комически огромным батареям. У других есть аналоговые ТВ-тюнеры, встроенные ручные зарядки и несколько пользовательских профилей, позволяющих семье или небольшой деревне делить один телефон.

Во время визита в район Hua Qiang Bei в 2013 году я заплатил \$12 за полноценный телефон с четырехдиапазонным GSM, Bluetooth, воспроизведением MP3, OLED-дисплеем и клавиатурой для управления интерфейсом. По сравнению со смартфоном это ничто, но он полезен, если вы выходите куда-то и беспокоитесь, что ваш основной телефон намокнет или будет украден. А для пары миллиардов людей это может быть единственный телефон, который они могут себе позволить.

Имейте в виду: это цена без контракта. В странах, где операторам разрешено блокировать телефоны, таких как Соединенные Штаты, телефоны часто отдаются бесплатно или продаются покупателям за долю своей стоимости в обмен на абонентский контракт, часто стоящий в несколько раз больше цены телефона. Тот факт, что я заплатил \$12 через прилавок за телефон без контракта, без промоакции, разблокированный, новый в коробке, с зарядным устройством, защитным силиконовым чехлом и кабелем, означает, что производственная стоимость телефона должна быть где-то ниже розничной цены \$12. Иначе изготовитель телефона терял бы деньги. По слухам, его стоимость была ниже \$10.



Мой простой, но функциональный телефон за \$12.

Это действительно удивительная цена. Это примерно цена большой сырной пиццы Domino's или приличного бокала вина в городском ресторане США. Он даже дешевле по сравнению с Arduino Uno. Признаю, сравнение немного несправедливое, но сделайте мне одолжение и взгляните на характеристики обоих, показанные в таблице 1.

Таблица 1: сравнение телефона за \$12 с Arduino

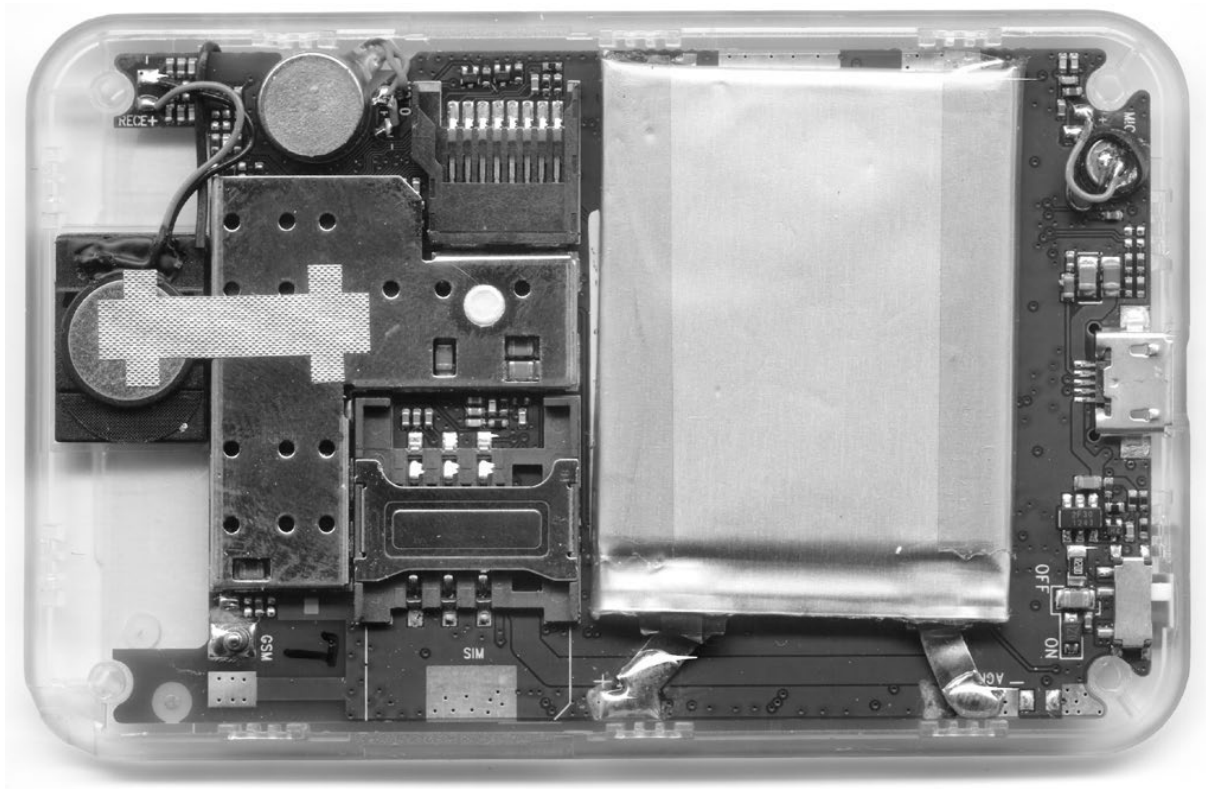
Спец	Этот телефон	Arduino Uno
Price	\$12	\$29
CPU speed	260 MHz, 32-bit	16 MHz, 8-bit

Спец	Этот телефон	Arduino Uno
RAM	8 MiB	2.5 kiB
Interfaces	USB, microSD, SIM	USB
Wireless	Quadband GSM, Bluetooth	-
Power	LiPo battery, includes adapter	External, no adapter
Display	Two-color OLED	-

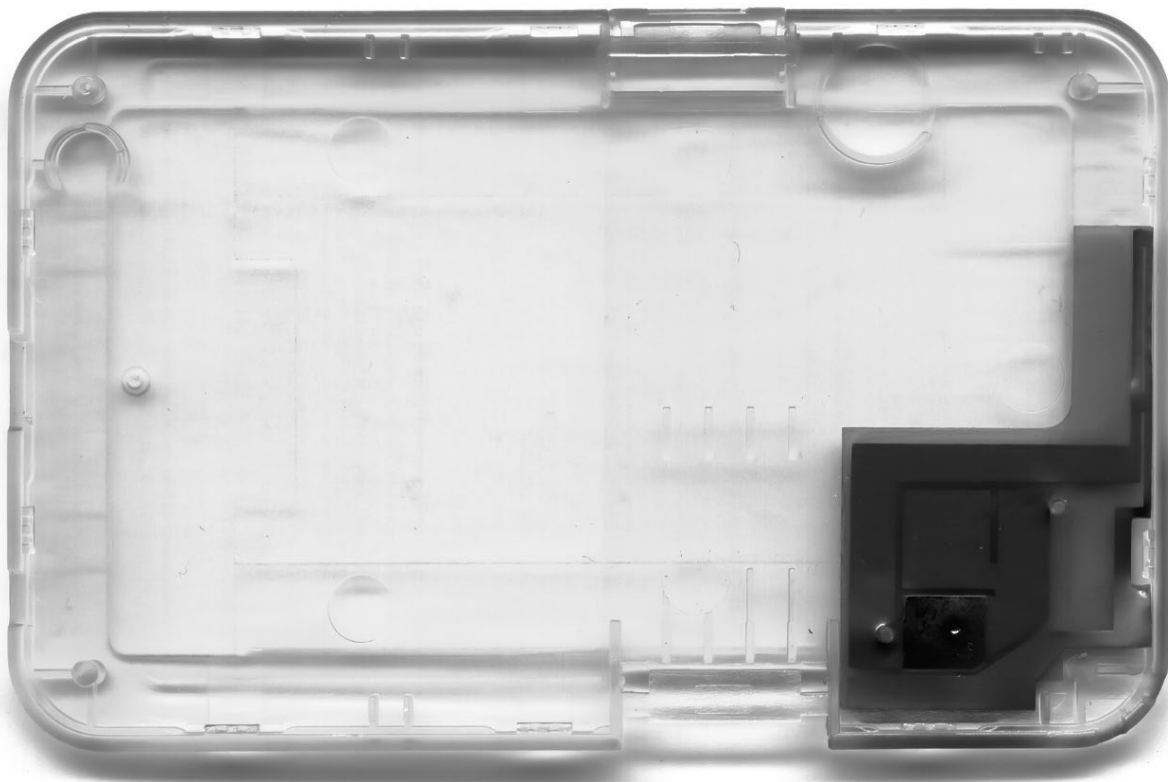
Как возможно, что этот телефон имеет лучшие характеристики, чем Arduino, и стоит меньше половины цены? У меня нет ответов, но я пытаюсь их узнать. Разбор телефона дал несколько подсказок.

Внутри телефона за \$12

Во-первых, в этом телефоне нет винтов. Весь корпус защелкивается.

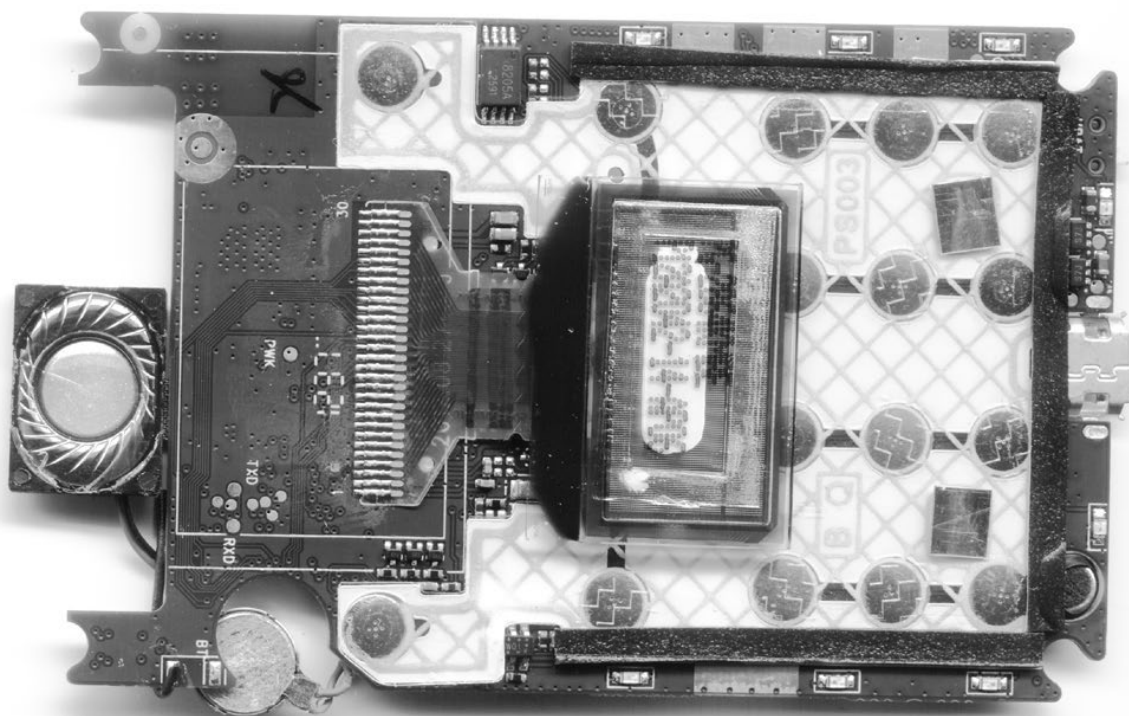


Задняя часть телефона после снятия крышки.

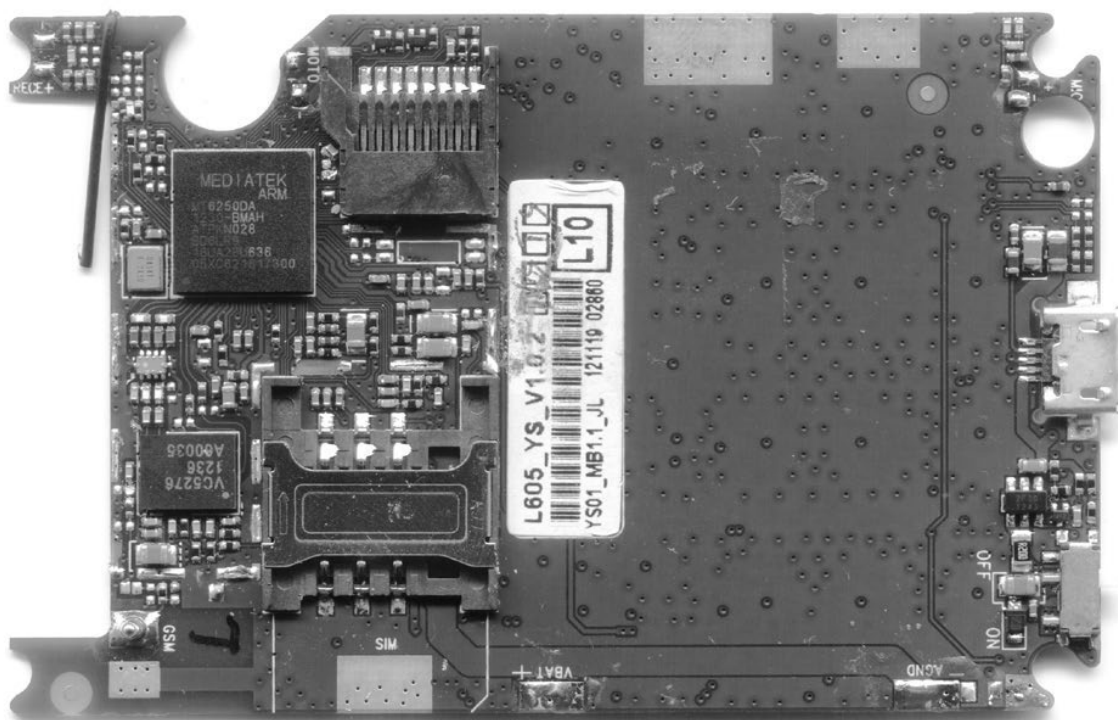


Снятая задняя крышка телефона.

Внутри почти нет разъемы. Для доставки и хранения вы можете щелкнуть переключателя, чтобы жестко отключить батарею. Насколько я могу судить, у батареи также нет вторичной схемы защиты. Тем не менее телефон имеет асскутрементс вроде подсвеченной клавиатуры и декоративных огней по краю.

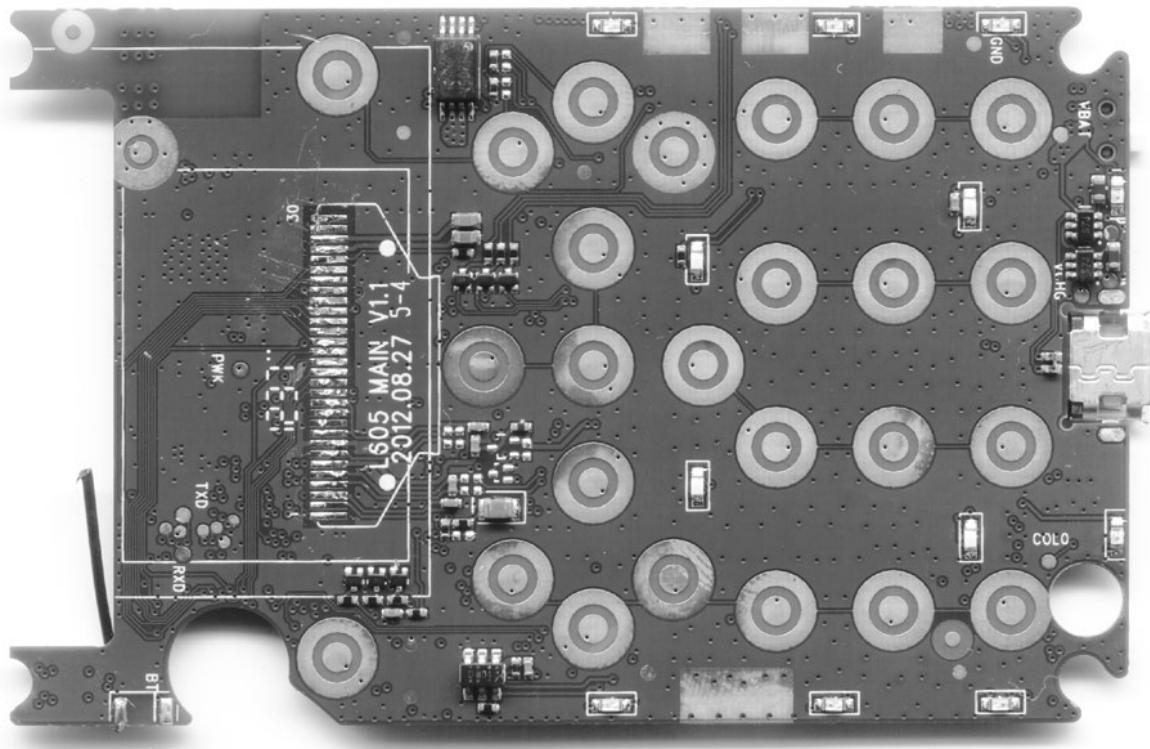


Все, от дисплея до батареи, припаяно прямо к плате. По всей этой PCB разбросаны маленькие декоративные светодиоды.



Bluetooth-антенна - это короткий кусок провода внизу слева.

Электроника состоит всего из двух основных IC: MediaTek MT6250DA и Vanchip VC5276. По слухам, MT6250 в больших объемах продается дешевле \$2. Я смог неподтвержденно проверить цену, купив пару штук в нарезанной ленте у розничного брокера примерно по \$2.10 за штуку.[1] Это лучше лучшей цены, которую мне когда-либо удавалось получить на ATmega тех типов, что используются в Arduino. При такой ценовой конкуренции западные фирмы судятся, чтобы защитить позиции: Vanchip вступила в небольшую юридическую стычку с RF Micro, а MediaTek сама была объектом нескольких исков.



Две MediaTek MT6250 IC.

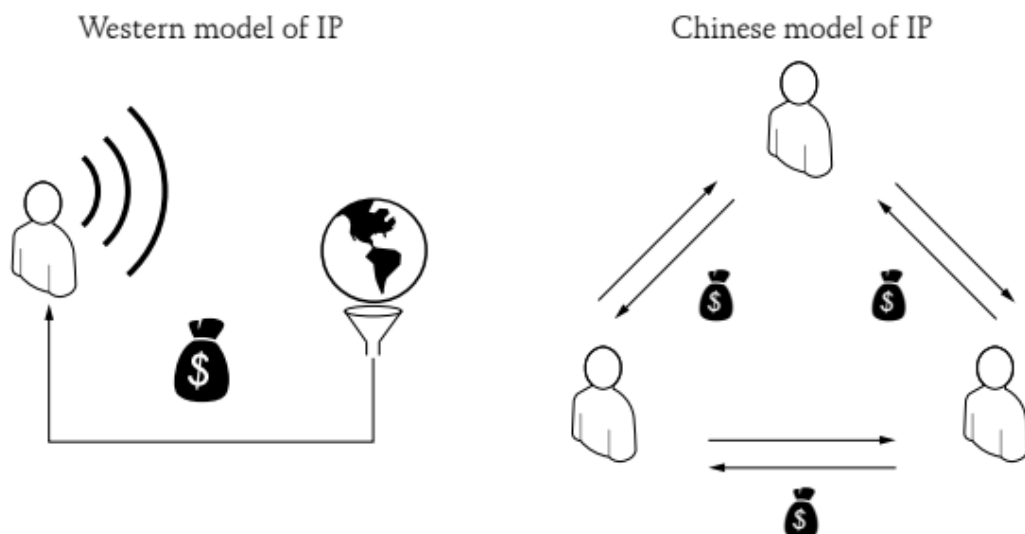
Разумеется, вы не можете просто позвонить в MediaTek и купить эти чипы. Невероятно трудно взаимодействовать с ними официальным путем, чтобы разработать устройство. Однако если вы немного знаете китайский и правильные сайты, вы можете скачать схемы, топологии плат и программные утилиты для чего-то похожего на этот телефон, возможно, с некоторыми другими деталями... "бесплатно". Бесплатно здесь в кавычках, потому что вы могли бы получить исходный код, но не однозначное юридическое право его использовать, поскольку исходный код распространялся без явного юридического согласия правообладателей. Но любой, кого такие юридические ограничения не волнуют или кто с ними незнаком, мог бы собирать версии этого телефона с минимальными вложениями. Это ощущается как открытый исходный код, но это не он: это другой тип открытой экосистемы.

Представляем гункай

Добро пожаловать на Галапагосы китайского "открытого" исходного кода. Я называю это gongkaì (□□), китайской транслитерацией английского open применительно к открытому исходному коду. Для открытого исходного кода есть буквальный перевод на китайский (kāiyuán), но единственное сходство между практиками gongkaì и западными практиками открытого исходного кода в том, что и те и другие позволяют вам скачать исходный код; юридические и культурные рамки, которые делают такое совместное использование возможным, не могли бы быть более разными. Это как конвергентная эволюция, где два вида могут проявлять похожие признаки, но гены и происхождение совершенно разные.

Gongkaì означает, что документы, защищенные авторским правом и иногда помеченные как «конфиденциально» и «проприетарно», становятся публично известны и открыто распространяются, но не обязательно согласно букве закона. Такое копирование - не односторонний поток ценности, каким оно было бы в случае скопированных фильмов или музыки.

Скорее, эти документы - база знаний, нужная для создания телефона на чипах правообладателя, а распространение документов стимулирует продажи этих чипов. В конечном счете правообладатели и копирующие получают взаимную выгоду.



Comparing IP models. On the left, the Western "broadcast" model, with a single owner who controls and disseminates IP and is paid by society. On the right, the Chinese "network" model, where IP trades hands like a commodity, and payment is often in-kind or as favors.

Сравнение модели IP. Слева западная "вещательная" модель, с одним владельцем, который контролирует и распространяет IP и получает плату от общества. Справа китайская "сетевая" модель, где IP переходит из рук в руки как товар, а оплата часто происходит натурой или в виде услуг.

Эти серые отношения между компаниями и предпринимателями - лишь одно проявление гораздо более широкого культурного разрыва между Востоком и Западом. У Запада "вещательный" взгляд на IP и собственность: хорошие идеи и инновации приписываются четко заданному набору авторов или изобретателей, а общество платит им роялти за их инициативу и хорошие работы. У Китая "сетевой" взгляд на IP и собственность: человек достигает дальнего обзора, необходимого для создания хороших идей и инноваций, стоя на плечах других, и люди обменивают эти идеи как услуги. В системе с таким свободным отношением к IP обмен с сетью необходим, потому что завтра ваш друг может стоять на ваших плечах, а вы будете обращаться к нему за услугами.

На Западе, однако, верховенство права позволяет накапливать IP на протяжении долгого времени, создавая непроницаемые монопольные позиции. Это хорошо для ребят наверху, но тяжело для новичков, вызывая ситуацию вроде современного западного рынка сотовых телефонов. Компании вроде Apple и Google строят потрясающие телефоны выдающегося качества, а стартапы могут лишь надеяться построить "accessory" для их экосистемы.

Я просматривал бизнес-планы более чем 100 аппаратных стартапов, и фундамент большинства из них - переоцененные чипсеты, выпущенные по устаревшим техпроцессам. Я не исключение из этого правила; Novena использует процессор Freescale (теперь NXP после поглощения) i.MX6, который не был ни самым дешевым, ни самым быстрым чипом на рынке, когда я проектировал ноутбук. Но у него есть два критически важных качества: любой может свободно скачать почти полную документацию на него, и любой может купить его на Digi-Key.

Скудные документация и поставки для передовой технологии заставляют западных аппаратных предпринимателей смотреть прежде всего на Arduino, Beaglebone и Raspberry Pi как на отправные точки для своих хороших идей. Китайские entrepreneurs, с другой стороны, выдают новые телефоны почти пугающим темпом.



Каждый объект на этом изображении - телефон.

Модели телефонов меняются сезонно. Предприниматели постоянно экспериментируют, интегрируя в телефоны чудачковатые функции, такие как зажигалки, сверхбольшие батарейные блоки (чтобы заряжать второй телефон), огромные кнопки (для людей с нарушением зрения), кнопки только для звонка домой (чтобы давать детям на случай чрезвычайных ситуаций), форм-факторы часов и так далее. Это работает потому, что небольшие команды инженеров могут получить полные пакеты проектной документации для работающих телефонов - включая корпус, плату и прошивку, - что позволяет им ответить дизайн и сосредоточиться только на изменении тех частей, которые им действительно важны.

Как аппаратный инженер, я хочу этого.

Я хочу иметь возможность ответвлять существующие дизайны сотовых телефонов. Я увидел телефон за \$12, и мне тоже захотелось использовать 364 MHz 32-битный микроконтроллер с мегабайтами встроенной RAM и десятками периферийных устройств, который стоит \$3 при покупке единичными партиями. Микроконтроллер ATmega в Arduino Uno, 16 MHz 8-битный микроконтроллер с несколькими килобайтами RAM и россыпью периферии, бледнеет в сравнении, но стоит вдвое больше, \$6.

От гункай к открытому исходному коду

Итак, я решил продвинуть свое изучение телефона на шаг дальше разборки и попытаться сделать собственную версию - в стиле shanzhai, но с западной точки зрения. Так мы с Sean "xobs" Cross начали проект, который назвали Fernvale. Sean был моим напарником по приключениям в десятках проектов с тех пор, как мы впервые встретились в Chumby, где я распознал его талант инженера по прошивке, когда он показал мне, как в свободное время портировал Quake на chumby. Sean всегда жил в собственном ритме. Родившись в Германии у американских родителей, он изучал когнитивные науки в колледже, а до работы в Chumby провел шесть месяцев, странствуя по Новой Зеландии и Австралии в поисках приключений и работы. В Chumby его было легко заметить благодаря хвосту и килту (на самом деле Utilikilt).

После того как Chumby ушла с рынка, мы с Sean оказались на берегах Сингапура, где я начал бутиковую фирму аппаратного консалтинга под названием Sutajio Ko-Usagi: это bunniestudios, переведенное на японский, а затем романизированное английскими символами. Виртуозные способности Sean к программированию стали отличным дополнением к моим навыкам аппаратного проектирования, и с тех пор мы завершили несколько значительных проектов с открытым исходным кодом.

Сначала мы решили, что должны хотя бы попробовать официальный путь и напрямую спросить производителей чипов, что потребуется, чтобы получить надлежащий комплект для встраиваемой разработки (EDK) с западной лицензией для чипов, используемых в этих шаньчжайских телефонах. Наши запросы встретили холодный прием. Мне сказали, что объемы для нашего маленького эксперимента слишком малы, или что нам придется заключить соглашения о минимальном объеме закупок, подкрепленные неподъемным денежным депозитом в сотни тысяч долларов.

Даже для людей, которые проходят через все эти бюрократические круги, EDK не включают весь справочный материал, которым пользуются китайцы. Технические описания неполные, и вас принуждают использовать проприетарные порты ОС от компаний. Получается ситуация, где те, кто действует честно, оказываются в хвосте. Могли ли мы найти способ выйти вперед и при этом играть по правилам?

У инженеров тоже есть права

Таким образом, у Fernvale было две половины: техническая задача реверс-инжиниринга и переинжиниринга телефона и юридическая задача создания общей методологии для переноса gongkai-IP в западную экосистему. Техническую задачу я перескажу в главе 9, а в оставшейся части этой главы сосредоточусь на юридической.

После некоторого исследования правовой базы и связанных с ней трудностей я счел, что нашел путь перенести часть IP из gongkai в настоящий открытый исходный код. Однако я должен дать оговорку: я не юрист. Я расскажу вам о своих убеждениях, но не воспринимайте их как юридическую консультацию.[2]

Моя базовая идея с Fernvale состояла в том, чтобы пользоваться правом на реверс-инжиниринг осторожно и осознанно, повышая вероятность того, что, если дело дойдет до столкновения, суды согласятся с моими действиями. Но я также чувствую, что уклоняться от реверс-инжиниринга только потому, что он спорен, - опасный путь: чтобы иметь права, ими нужно пользоваться. Если бы женщины не голосовали, а чернокожие сидели в задней части автобуса из страха перед спорами, Соединенные Штаты все еще были бы сегрегированы и лишены всеобщего избирательного права. Хотя реверс-инжиниринг - тривиальный вопрос по сравнению с расовым равенством и всеобщим

избирательным правом, прецедент ясен: чтобы иметь права, нужно быть достаточно смелым и заявлять о них.

Работа с патентами и другими законами

Открытый исходный код имеет две широкие категории IP-вопросов, с которыми нужно иметь дело: патенты и авторские права. Патенты создают сложные вопросы, и, кажется, самый практичный подход - по сути отложить этот вопрос. Например, насколько я знаю, никто не проверяет свои коммиты Linux на нарушение патентов перед отправкой в основной проект, и фактически у многих корпораций есть похожие политики на инженерном уровне.

Почему? Определение, какие патенты применимы и нарушает ли их продукт, требует огромного объема ресурсов. Даже потратив эти ресурсы, вы не можете быть уверены на 100 процентов. Кроме того, очень близкое знакомство с корпусом патентов усиливает возможность того, что любое нарушение будет умышленным, тем самым утраивая ущерб. Наконец, даже не ясно, где находится ответственность за нарушение, особенно в контексте открытого исходного кода.

Таким образом, мы с Sean сделали все возможное, чтобы Fernvare не нарушал чужие права, но не могли быть на 100 процентов уверены, что никто не заявит о нарушении. Однако мы применили к нашей работе лицензию, включающую пункт-"ядовитую пилюлю" для патентообладателей, которые могли бы попытаться судиться. Такие "ядовитые пилюли" делают всю совокупность работ с открытым исходным кодом недоступной для любой стороны, которая подает иск о нарушении прав против какой-либо другой стороны.[3]

С авторскими правами вопрос тоже чрезвычайно сложен. У проекта Coders' Rights от Electronic Frontier Foundation (EFF) есть Reverse Engineering FAQ[4], который стоит прочесть, если вы действительно хотите углубиться в эти вопросы. Если суммировать, суды установили, что реверс-инжиниринг для понимания идей, встроенных в код, и достижения совместимости является добросовестным использованием. В результате любой, вероятно, имеет право изучить IP в стиле gongkai, понять ее, создать новую работу и применить к ней западную лицензию открытой интеллектуальной собственности.

Однако прежде чем разбираться с вопросами авторского права для Fernvare, мне нужно было убедиться, что мы не столкнемся с другими законами, которые могли бы помешать нашим правам на добросовестное использование. Во-первых, есть Digital Millennium Copyright Act (DMCA). DMCA делает обход любого шифрования, предназначенного для защиты авторских прав, в основном незаконным, разрешая лишь несколько еще недостаточно проверенных исключений. Поскольку ни один из файлов или бинарников, которые мы с Sean скачали, не был зашифрован и не имел доступа, контролируемого какой-либо технологической мерой, нам не нужно было выполнять обход. Нет обхода - нет проблемы с DMCA.

Все файлы, которые мы получили, нашлись через поиски, ведущие к публичным серверам, поэтому проблем с Computer Fraud and Abuse Act (CFAA) не возникало. Ни одно из устройств, которые мы использовали в работе, не поставлялось с оберточными, кликовыми или другими лицензионными соглашениями конечного пользователя (EULA), условиями использования или иными соглашениями, которые могли бы лишить нас прав.

Работа с авторскими правами

Отложив в сторону вопросы DMCA, CFAA и EULA, мы наконец смогли обратиться к главному вопросу: что делать с авторскими правами.

Краеугольный камень нашей методологии опирался на решения, несколько раз вынесенные судами, где говорилось, что факты не охраняются авторским правом. Например, судья O'Connor написала следующее в *Feist Publications, Inc. v. Rural Telephone Service Co., Inc.* (449 U.S. 340, 345, 349 (1991)): [5]

Здравый смысл говорит нам, что 100 неохраняемых авторским правом фактов не меняют магически свой статус, когда собраны вместе в одном месте. . . . Ключ к разрешению напряжения состоит в понимании, почему факты не охраняются авторским правом: непереносимое условие авторского права - оригинальность.

И:

Несмотря на действующее авторское право, последующий составитель остается свободен использовать факты, содержащиеся в чужой публикации, для подготовки конкурирующей работы, пока эта работа не имеет того же отбора и расположения материала.

Согласно этому подходу, любой имеет право извлекать факты из проприетарной документации и тщательно заново выражать эти факты в собственной структуре и формулировках. Точно так же, как факты "номер телефона John Doe - 555-1212" и "адрес John Doe - 10 Main St." не защищаются авторским правом, факты вроде "базовый адрес контроллера прерываний - 0xA0060000" и "бит 1 управляет отчетами о состоянии LCD" тоже не защищаются авторским правом. Мы с Sean извлекли такие факты из технических описаний и заново выразили их в наших собственных заголовочных файлах, где, как законные владельцы вновь созданного авторского выражения, применили подходящую лицензию открытого исходного кода по нашему выбору.

Создание языка программирования

Но ситуация дополнительно осложнялась аппаратными блоками, по которым у нас не было абсолютно никакой документации. В некоторых случаях мы даже не могли узнать из технического описания, что означают регистры блока или как работают эти блоки. Для этих блоков мы изолировали и извлекли код, отвечающий за инициализацию их состояния. Затем мы свели этот код к списку пар "адрес-данные" и выразили его на специальном скриптовом языке, который назвали *scriptic*. Мы изобрели собственный язык, чтобы избежать неосознанного плагиата: слишком легко прочитать один кусок кода и по памяти написать почти такой же. Преобразуя код в новый язык, мы были вынуждены рассматривать представленные факты и выражать их в оригинальной компоновке.

Scriptic - это в основном набор ассемблерных макросов, и синтаксис очень прост. Вот пример *scriptic*-скрипта:

```
#include "scriptic.h"
#include "fernvale-pll.h"
sc_new "set_plls", 1, 0, 0
    sc_write16 0, 0, PLL_CTRL_CON2
    sc_write16 0, 0, PLL_CTRL_CON3
    sc_write16 0, 0, PLL_CTRL_CON0
    sc_usleep 1
    sc_write16 1, 1, PLL_CTRL_UPLL_CON0
    sc_write16 0x1840, 0, PLL_CTRL_EPLL_CON0
    sc_write16 0x100, 0x100, PLL_CTRL_EPLL_CON1
    sc_write16 1, 0, PLL_CTRL_MDDS_CON0
```

```

sc_write16 1, 1, PLL_CTRL_MPLL_CON0
sc_usleep 1
sc_write16 1, 0, PLL_CTRL_EDDS_CON0
sc_write16 1, 1, PLL_CTRL_EPLL_CON0
sc_usleep 1
sc_write16 0x4000, 0x4000, PLL_CTRL_CLK_CONDB
sc_usleep 1
sc_write32 0x8048, 0, PLL_CTRL_CLK_CONDC
/* Run the SPI clock at 104 MHz */
sc_write32 0xd002, 0, PLL_CTRL_CLK_CONDH
sc_write32 0xb6a0, 0, PLL_CTRL_CLK_CONDC
sc_end

```

Этот скрипт инициализирует Phase Locked Loop (PLL, схему для генерации тактовых сигналов) на целевом чипе Fernvale, MediaTek MT6260. Для сравнения, вот первые несколько строк фрагмента кода, из которого был выведен этот scriptic-код:

```

// enable HW mode TOPSM control and clock CG of PLL control
*PLL_PLL_CON2 = 0x0000; // 0xA0170048, bit 12, 10 and 8 set to 0
// to enable TOPSM control
// bit 4, 2 and 0 set to 0 to enable
// clock CG of PLL control
*PLL_PLL_CON3 = 0x0000; // 0xA017004C, bit 12 set to 0 to enable
// TOPSM control

// enable delay control
*PLL_PL LTD_CON0 = 0x0000; // 0x A0170700, bit 0 set to 0 to
// enable delay control
// wait for 3us for TOPSM and delay (HW) control signal stable
for(i = 0 ; i < loop_1us*3 ; i++);
// enable and reset UPLL
reg_val = *PLL_UPLL_CON0;
reg_val |= 0x0001;
*PLL_UPLL_CON0 = reg_val; // 0xA0170140, bit 0 set to 1 to
// enable UPLL and
// generate reset of UPLL

```

Исходный код на самом деле продолжается страницы и страницы, и даже этот фрагмент окружен условными операторами, которые мы отбросили, поскольку они не имели отношения к правильной инициализации PLL.

Знание наших прав, набор документации, из которого можно было извлекать факты, и scriptic были инструментами в нашем арсенале. С ними мы с Sean вывели достаточно функциональности для проекта Fernvale, чтобы в конце концов загрузить небольшую BSD-лицензированную операционную систему реального времени (RTOS), известную как NuttX, работающую на нашем собственном нестандартном железе. Технические подробности того, как мы это сделали, я расскажу в главе 9.

Заключительные мысли

Права атрофируются и уступают место конкурирующим интересам, если ими активно не пользоваться. Мы с Sean сделали Fernvale, потому что считаем необходимым пользоваться нашими правами на добросовестное использование, реверс-инжиниринг и создание совместимых решений с открытым исходным кодом. На протяжении десятилетий инженеры сидели в стороне и смотрели, как все более широкие патентные и авторско-правовые законы ограничивают их свободу учиться и изобретать. Мне грустно, что детские технические эксперименты, которые сформировали меня как инженера, больше не являются законным вариантом для следующего поколения инженеров.

Рост shanzhai и их удивительных возможностей - это тревожный звонок. Я вижу в этом доказательство, что либеральная среда интеллектуальной собственности стимулирует инновации, особенно на низовом уровне. Если больше инженеров узнают свои права на добросовестное

использование и будут пользоваться ими энергично и сознательно, возможно, это сможет подтолкнуть более крупную и очень нужную реформу системы патентов и авторского права. Наш проект Fernvale, надеюсь, всего лишь указатель, ведущий к гораздо более крупным усилиям по наведению мостов между сообществами gongkai и открытого исходного кода.

Способность выбирать позитивные аспекты gongkai и переносить их в западную IP-экосистему - важный инструмент. Верховенство права имеет свое место, и у чрезмерно разрешительной системы есть собственные проблемы. Следующая глава исследует некоторые негативные последствия чрезмерно разрешительной IP-экосистемы: подделки и контрафактные товары.

Примечания

[1] Нет, я не буду выступать посредником по продаже этих микросхем для вас.

[2] Я часто задавался вопросом, почему необходима оговорка "I am not a lawyer". Мне объяснили, что даже видимость предоставления юридической консультации без такой оговорки может сделать меня виновным в юридической практике без надлежащей лицензии. Меня также могли бы привлечь к ответственности за неудачные решения людей, которые истолковали мои мнения как юридическую консультацию.

[3] Конкретно раздел 3 Apache 2.0 гласит: "Предоставление патентной лицензии. . . . Если Вы инициируете патентный спор против какой-либо стороны (включая встречный или перекрестный иск), утверждая, что Работа или Вклад, включенный в Работу, представляет прямое или косвенное нарушение патента, то любые патентные лицензии, предоставленные Вам по этой Лицензии для этой Работы, прекращаются с даты подачи такого иска."

[4] <https://www.eff.org/issues/coders/reverse-engineering-faq/>

[5] См. также Sony Computer Entertainment, Inc. v. Connectix Corp., 203 F. 3d 596, 606 (9th Cir. 2000) и Sega Enterprises Ltd. v. Accolade, Inc., 977 F.2d 1510, 1522-23 (9th Cir. 1992).

Глава 5. Поддельные товары

Система gongkaі поощряет удивительный объем инноваций в Китае, а shanzhai могут делать интересные оригинальные продукты, вроде мобильных телефонов, которые я показал вам в главе 4. Тем не менее Китай действительно производит множество поддельных электронных товаров, и это далеко не только копии iPhone. Умелые фальсификаторы могут производить поддельные интегральные схемы, включая microSD-карты и даже FPGA.

Хорошо выполненные поддельные чипы

Например, в 2007 году (когда я все еще работал с Chumby) я столкнулся с несколькими поддельными чипами, выполненными настолько хорошо, что без расследования не мог быть уверен, что они фальшивые.

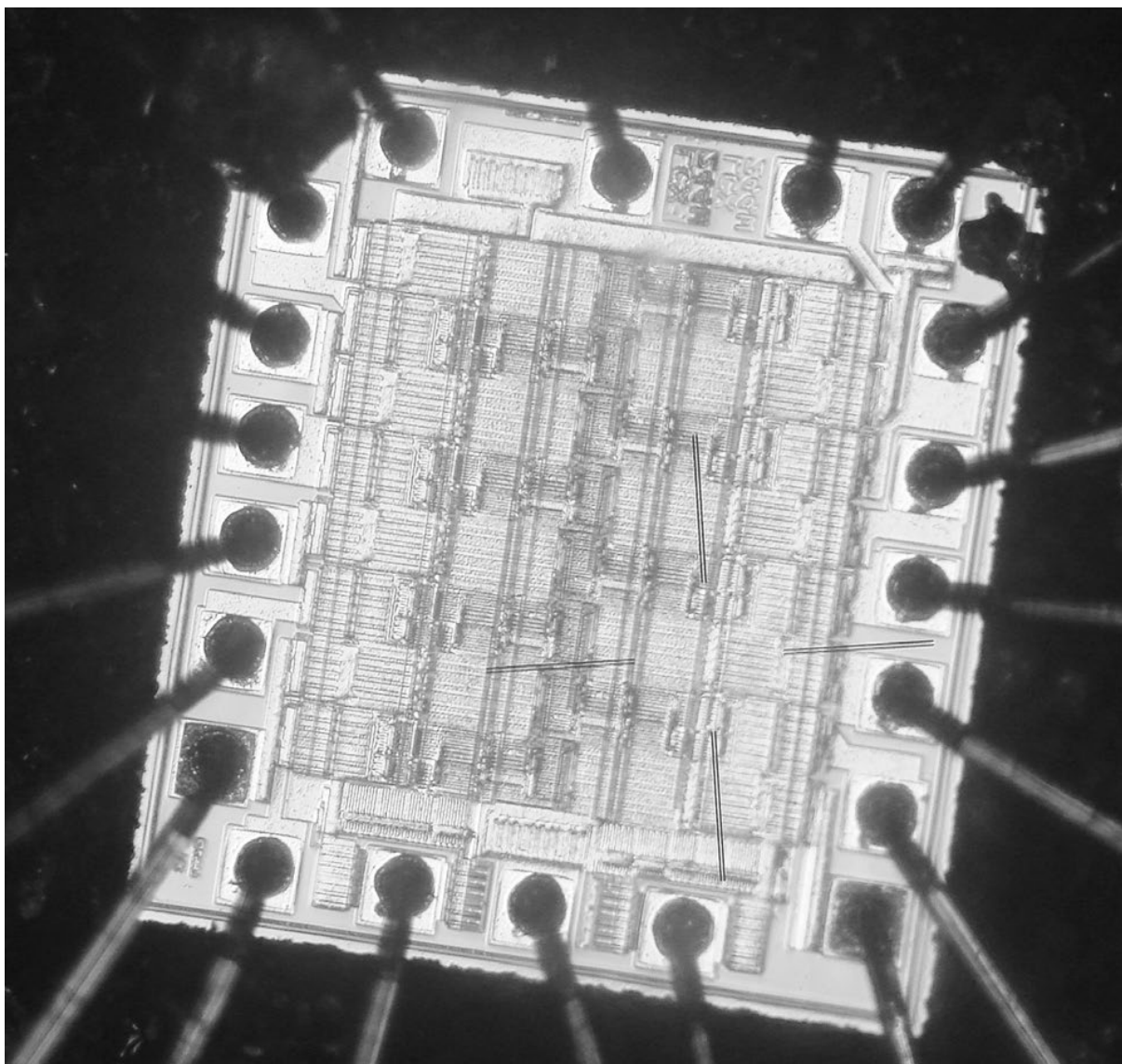


Два подозрительных экземпляра чипов из азиатского источника.

Согласно маркировке, чипы были ST19CF68 производства STMicroelectronics; в техническом описании он описан как "CMOS MCU Based Safeguard Smartcard I/O with Modular Arithmetic Processor". ST19CF68 обычно продаются уже упакованными в формате смарт-карты (например, чип

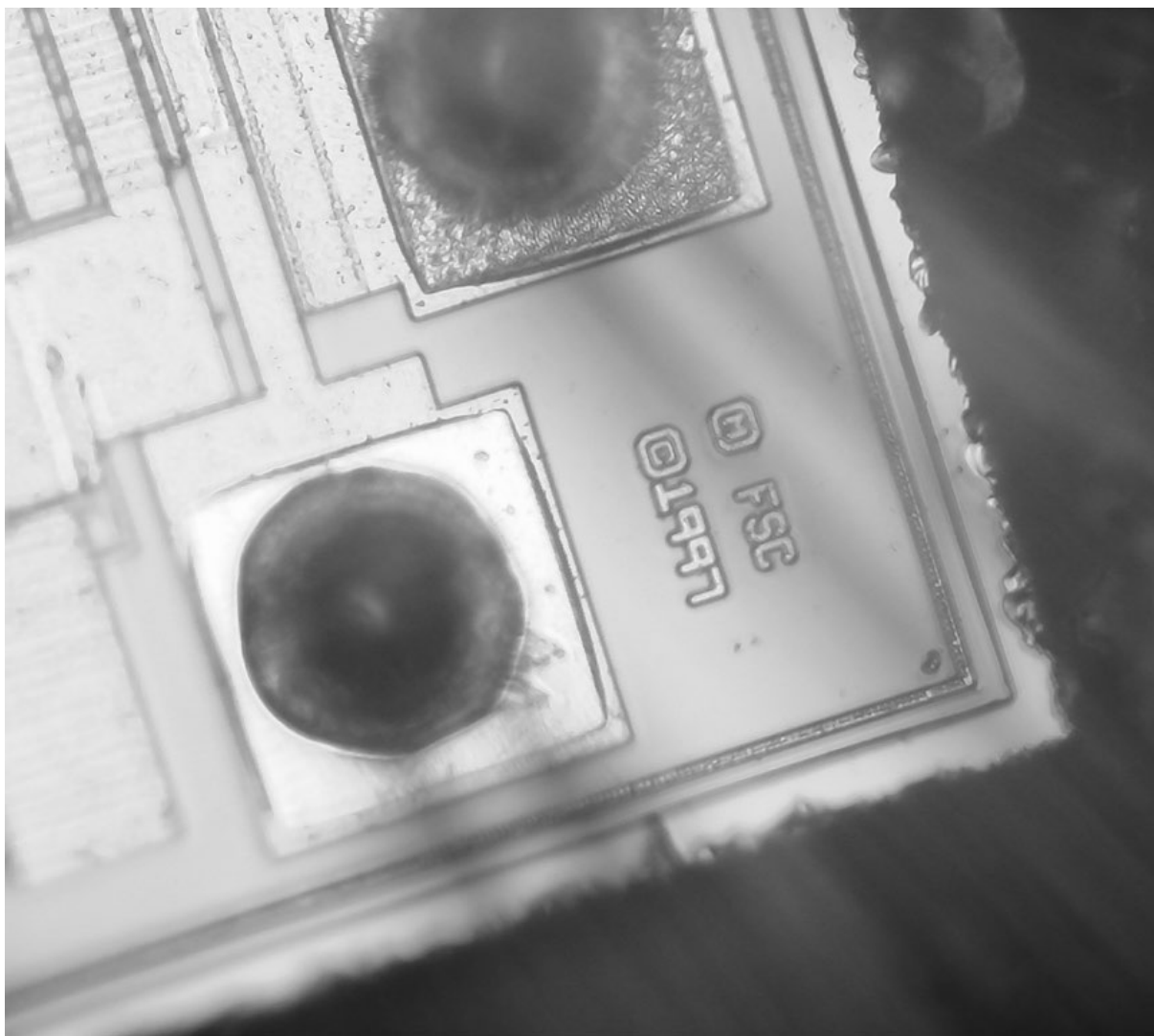
на лицевой стороне кредитной карты) или в виде нарезанной пластины (кремниевая пластина, нарезанная на отдельные чипы, но без какой-либо другой упаковки вокруг него), но, что любопытно, эти устройства были в корпусах SOIC-20. Чтобы выяснить причину странного выбора корпуса, я растворил черную эпоксидную заливку с верхней части одного чипа, снял крышку и осмотрел кремний внутри под микроскопом.

Кристалл внутри корпуса был слишком мал и прост для сложного микроконтроллера (MCU), соответствующего описанию ST19CF68. Узор из золотистых прямоугольников, покрывавших чип плиткой, был слишком грубым; я мог различить отдельные транзисторы при малом увеличении оптическим микроскопом. Размер этих элементов называется технологической нормой чипа. Технологическая норма smartcard обычно отставала бы от передовых CPU максимум на три или четыре поколения, из-за чего транзисторы было бы очень трудно различить даже при самых высоких уровнях увеличения.



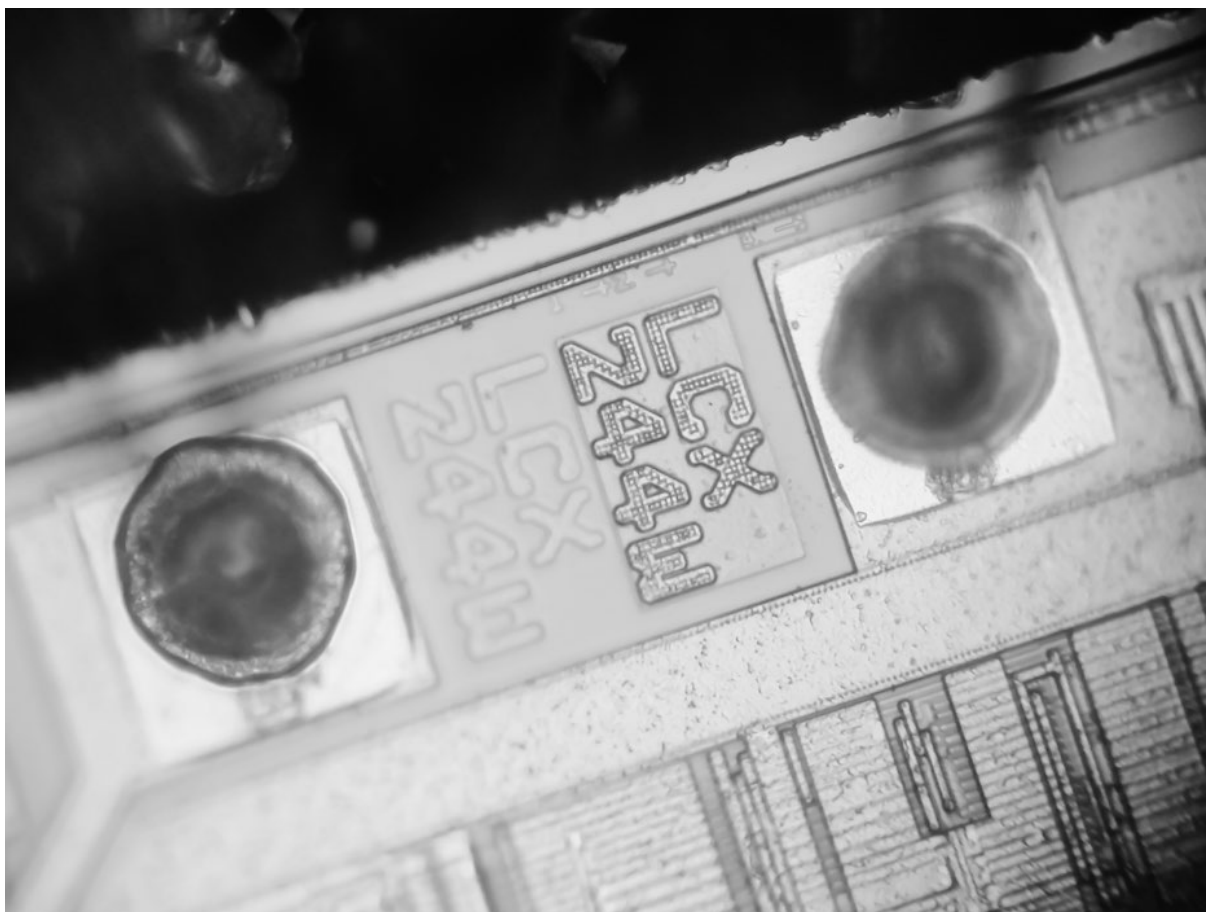
Silicon внутри поддельного ST19CF68.

Наряду с неожиданно грубой технологической нормой возникал вопрос: почему у этой детали было 20 контактных площадок и 20 выводов, когда, согласно техническому описанию, у нее должно быть только 8 площадок? Небольшое увеличение кристалла выявило несколько интересных деталей.



Производитель чипа и дата авторского права.

В конце концов чип был сделан не STMicroelectronics! Метка на кремнии говорила FSC, указывая, что он был сделан Fairchild Semiconductor. Разумеется, затем мне пришлось проверить и маркировку детали на кремнии.



Обнаружение настоящего номера детали.

Кристалл внутри этого чипа оказался Fairchild 74LCX244, то есть "Low Voltage Buffer/Line Driver with 5V Tolerant Inputs and Outputs". 74LCX244 - гораздо более дешевый кусок кремния, чем ST19CF68, который якобы содержал корпус.

Конечно, несовпадающее число выводов было подозрительным, но производители, как известно, помещали чипы в более крупные корпуса, особенно во время ранних выпусков чипа, до того как он был оптимизирован по размеру. Что действительно меня зацепило, так это убедительное качество корпуса и маркировки.

Обычно перемаркированные или поддельные чипы выглядят дешевле, чем этот. Исходные чипы шлифуют или закрашивают, чтобы удалить прежнюю маркировку, а новая маркировка обычно наносится краской через трафарет.

Но на этих чипах вообще не было следов перемаркировки. Маркировка была качества первого заводского выпуска: кто-то получил немаркированные заготовки чипа 74LCX244 и запрограммировал производственный лазерный гравер, чтобы нанести высококачественную поддельную маркировку на в остальном нетронутый корпус. Они даже идеально попали в пропорции логотипа ST.



Крупный план внешней стороны поддельного ST19CF68.

Разница в качестве между перемаркированным чипом и маркировкой первого выпуска похожа на разницу между краской из баллончика, использованной, чтобы скрыть царапину на машине, и оригинальной заводской покраской. У этого чипа определенно был вид "новой машины".

Это открытие оставило у меня множество вопросов без ответа. Как кто-то получил немаркированный кремний Fairchild? Был ли этот человек инсайдером, или Fairchild небрежно выбросила немаркированные отбракованные чипы, не измельчив их и не отрезав выводы, так что их можно было вытащить из мусорного контейнера и перепродать? Машина для лазерной маркировки, использованная для создания этих надписей, тоже не была дешевым настольным гравером; это должен был быть мощный растровый гравер, и макет был выполнен безупречно.

Мне до сих пор трудно поверить, что эти поддельные чипы были сделаны и проданы, но, возможно, не стоит удивляться. Я видел дерзкую перемаркировку модулей памяти DIMM (двухрядных модулей памяти, используемых в персональных компьютерах) на SEG Electronics Market, и многие фальсификаторы на рынке открыто демонстрируют свой арсенал из профессиональных термотрансферных принтеров этикеток и заготовок голографических наклеек.

Если подделки такого качества станут более распространенными, они могут создать проблему для цепочки поставок. Очевидно, тот, кто сделал поддельный ST19CF68, может подделать практически любой чип, и такие подделки постепенно появляются на рынке США. Перепродавцы, особенно дистрибьюторы, специализирующиеся на покупке избыточных складских остатков производителя, неявно доверяют маркировке на чипе.

Я не думаю, что производители чипов будут добавлять меры защиты от подделок в маркировку чипов, но качество этих подделок определенно насторожило меня, когда я их обнаружил, и настораживает до сих пор. Не все подделки замечают до того, как их используют, а поддельные компоненты создают проблемы в любом проекте, где они появляются.

Поддельные чипы в военном оборудовании США

Поддельные чипы могут быть особенно проблематичны, когда они попадают в военные проекты. У американских военных есть уникальная проблема: это один из самых больших и богатых покупателей действительно старых деталей, потому что срок службы военных разработок измеряется десятилетиями. Как и со всем остальным, чем старше деталь, тем труднее ее найти, и иногда подрядчики продают подделки. Например, отчет слушаний Сената 2011 года показал, что некоторые детали, использованные в P-8 Poseidon (самолете, который ВМС США заказали у Boeing), были, как выразилась статья сайта Defense Tech, «плохо восстановлены», что вызвало отказ ключевой системы.

Правительство США попыталось снизить количество подделок в своей цепочке поставок с помощью поправки 1092 к National Defense Authorization Act for Fiscal Year 2012 (H.R. 1540). Поправка была хорошо намеренной, но ошибочной нормой, описывающей меры для снижения распространенности поддельных чипов в военной цепочке поставок США.

Еще до того как поправка 1092 была вынесена на обсуждение, Defense Authorization Act получил критику за положение, разрешающее американским военным бессрочно задерживать граждан США без суда. Он также, довольно иронично, требует оценки федерального долга США перед Китаем как потенциального «риска национальной безопасности» (section 1225 of H.R. 1540).

Согласно поправке о борьбе с подделками, нарушители при первом же нарушении могут получить штраф \$5 млн и 20-летний тюремный срок для физических лиц или штраф \$15 млн для корпораций - наказание, сравнимое с наказанием за торговлю кокаином.[1] Хотя поправка явно определяет подделку так, что включает восстановленные детали, представленные как новые, формулировка, к сожалению, расплывчата в вопросе о том, должны ли вы сознательно торговать такими товарами, чтобы также нести ответственность по столь суровому наказанию.

Если бы вы взяли грязную, но законно отчеканенную монету и вымыли ее так, чтобы она выглядела как новая, никто не обвинил бы вас в подделке. Однако эта поправка устанавливает 20-летнее наказание и штраф \$5 млн не только за подделку чипов, предназначенных для военного применения, но потенциально и за непреднамеренное распространение восстановленных чипов, которые вы предположительно купили как новые. К сожалению, во многих случаях электронная деталь может использоваться годами без признаков внешнего износа.

Поправка также содержит положение о создании «программы проверки»:

(b) Проверка импортируемых электронных деталей -

(1) . . . Министр внутренней безопасности должен создать программу усиленной проверки Службой таможенного и пограничного контроля США электронных деталей, импортируемых из любой страны, которую министр обороны признал значительным источником поддельных электронных деталей . . .

Проверять фрукты и овощи при ввозе в страну на вредителей и другие проблемы разумно, но требовать от таможенников стать экспертами по обнаружению поддельных электронных компонентов кажется ошибочным. Возлагать на поставщиков обязанность обнаруживать подделки при таких высоких штрафах за промах тоже ошибочно, учитывая, насколько легко фальсификаторам создавать высококачественные подделки.

Типы поддельных деталей

Чтобы лучше понять масштаб проблемы подделки чипов, посмотрим, как создаются подделки. Поддельные чипы, которые я видел, попадают в следующие широкие категории.

Внешнее подражание

Самые тривиальные поддельные чипы - это просто пустые пластиковые корпуса с правдоподобной верхней маркировкой или перемаркированные детали, которые имеют с настоящими деталями только внешние признаки. Например, простой чип транзисторно-транзисторной логики (transistor-transistor logic, TTL) может быть помещен в такой же корпус, с такой же маркировкой, как дорогой микроконтроллер.

Я считаю внешнее подражание тривиальным, потому что подделки, произведенные таким способом, легко обнаружить в заводском тесте. В худшем случае вам продают смесь, состоящую в основном из настоящих деталей с несколькими подделками, подмешанными так, что проверки одной детали из тубы или катушки недостаточно, чтобы поймать проблему. Но большинство продуктов проходят стопроцентное тестирование на уровне системы, поэтому обычно проблема обнаруживается до того, как что-либо покинет фабрику.

Восстановленные детали

Подделки, строго говоря, вообще не обязаны быть фальшивыми. Восстановленные детали - это настоящие чипы, выпаянные из электронных отходов и переработанные так, чтобы выглядеть новыми. Их очень трудно заметить, поскольку чип действительно настоящий, а умелый восстановитель может производить поразительно новые на вид чипы, которые только изотопный или элементный анализ смог бы идентифицировать как использованные.

Эта категория также включает детали, которые "новые" в том смысле, что их никогда не припаивали к плате, но они хранились неправильно, возможно, во влажной среде. Такие чипы следовало бы списать в брак, но иногда их кладут в свежий фольгированный пакет с более новым кодом даты и продают как новые.

Детали, перемаркированные по классу

Фальсификаторы иногда перемаркируют настоящие детали, которые никогда не использовались (и поэтому могут считаться новыми), как лучшую версию в остальном идентичной детали. Классический пример - шлифовка и перемаркировка CPU под более высокий скоростной класс или, более тривиально, маркировка деталей, содержащих свинец, как RoHS-совместимых.

Но повторная сортировка может быть более изощренной. Поставщики могут провести реверс-инжиниринг и перепрограммировать fuse-коды внутри перемаркированного чипа так, что электронные записи чипа действительно совпадут с поддельной маркировкой сверху. Также известно, что поставщики взламывают прошивку флеш-накопителей так, чтобы операционная система хоста воспринимала небольшой объем памяти как гораздо больший. Такие взломы заходят даже настолько далеко, что зацикливают адреса памяти, чтобы записи за пределами реальной емкости устройства выглядели успешными; чтобы обнаружить проблему, тогда требуется долгое полное чтение с проверкой записанных данных.

Детали из неучтенных смен

Некоторые подделки создаются на точно той же фабрике, что и настоящие детали; их запускают очень поздно ночью недобросовестные сотрудники без разрешения производителя, и они никогда не попадают в официальные книги учета. Такие незарегистрированные производственные прогоны называются «призрачными сменами» (ghost shifts). Это похоже на сотрудника монетного двора, чеканящего дополнительные монеты после смены. Детали из таких ночных смен часто получают код партии, идентичный законному выпуску, но определенные этапы тестирования пропускаются.

«Призрачные смены» часто используют пограничный материал, оставшийся от настоящего продукта, который обычно был бы утилизирован, но был перехвачен по дороге к измельчителю. В результате маркировка и характеристики материала часто выглядят совершенно подлинными. Такие подделки могут быть крайне трудны для обнаружения.

Заводской брак

Заводской брак и прототипные партии можно извлечь из кучи отходов за небольшую взятку, снабдить подлинной маркировкой и перепродать как новые. Чтобы избежать обнаружения, работники часто заменяют пригодный для спасения брак физически идентичными пустышками, тем самым срывая попытки проверить путь отходов. Эта практика замены пригодного брака фиктивными подделками помогает поддерживать рынок тривиальных подделок типа "внешнее подражание". Само существование отрасли, которая поставляет низкокачественные подделки для обхода проверок, иначе запрещающих бы высококачественные подделки, дает представление о том, насколько изощренной и зрелой стала индустрия подделок.

Неудачный поиск второго поставщика

Вторичное производство у альтернативного поставщика (second-sourcing) - стандартная отраслевая практика, когда конкуренты создают совместимые по выводам замены для популярных продуктов, чтобы подстегнуть ценовую конкуренцию и укрепить цепочку поставок на случай событий вроде природных катастроф. Эта практика становится проблемой, когда худшие детали перемаркируют логотипами премиальных брендов.

Дорогие, но функционально простые дискретные аналоговые чипы, такие как стабилизаторы питания, особенно уязвимы к этой проблеме. Премиальные стабилизаторы питания под американскими брендами иногда стоят в 10 раз дороже, чем совместимые замены под азиатскими брендами. Однако азиатские детали печально известны неровным качеством, срезанными углами и плохими параметрами. Очевидно, у фальсификаторов есть широкая возможность заработать много денег, покупая немаркированные чипы у второго производителя и нанося на них правдоподобную маркировку премиальных американских брендов. В некоторых случаях нет недорогих или быстрых тестов для обнаружения таких подделок, кроме снятия корпуса с чипа и сравнения топологии масок и поперечных срезов, как я сделал для ST19CF68.

Подделки и военные разработки США

Разнообразие доступных методов подделки в сочетании с тем, что многие массовые детали имеют производственные циклы всего в несколько лет, создает большую проблему для организаций вроде американских военных, где срок жизни разработок часто измеряется десятилетиями. Это похоже на просьбу к кому-то сегодня собрать материнскую плату NeXTcube[2], используя только гарантированно новые детали, без бывших в употреблении или восстановленных компонентов. Я не думаю, что это возможно.

Безвыходность этой ситуации иногда может делать военных подрядчиков соучастниками потребления поддельных деталей, с плохими последствиями. В случае P-8 Poseidon люди быстро начали указывать пальцами на Китай, но плохая работа по восстановлению, вероятно, обнаруживается простым визуальным осмотром. Возможно, часть проблемы была в том, что субподрядчик небрежно проверял входящие запасы, а возможно, сознательно смотрел в другую сторону. Если эти детали были последними в своем роде в мире, что еще можно было сделать?

Моя догадка в том, что запасы любого дистрибьютора в бизнесе подержанной электроники уже наводнены невыявленными подделками. Помните, ловят только плохие подделки, а корпуса чипов не проектировались с учетом защиты от фальсификации. Хотя все детали серого рынка подозрительны, это не обязательно плохо.

Серые рынки играют важную роль в экосистеме электроники; пользоваться ими - просчитанный, но иногда неизбежный риск. Фактически многие торговцы на сером рынке очень прямо говорят, что их товары восстановлены. Многие даже вывешивают на своих прилавках таблички, сообщающие об этом. Однако эти таблички написаны на китайском. В таком случае чья это вина - продавца, продающего восстановленные товары, или покупателя, не умеющего прочитать табличку?

Меры против подделок

Ситуация с поддельными чипами - бардак, но некоторые простые меры могли бы ее исправить.

Физические идентификаторы

Встраивание мер защиты от подделок в чипы, одобренные для военного применения, - один вариант. Для чипов шириной больше 1 см уникальный двумерный штрихкод можно было бы выгравировать лазером на оборудовании, довольно обычном для предприятий по корпусированию чипов. Несмотря на крошечную площадь, коды сопровождались бы гарантией стопроцентной уникальности. Такие методы эффективны в биотехнологии, где системы вроде Matrix 2D отслеживают одноразовые пробирки с образцами в биологических лабораториях.

Еще одно потенциальное решение - смешать УФ-краситель с эпоксидной смолой компонента, чтобы он менял флуоресцентные свойства при воздействии температур пайки оплавлением, то есть устойчивого набора четко определенных температур, при которых плавится припой. Это делает невозможным восстановление чипа до "нового" состояния после того, как он был впервые припаян. Если краситель распределен по всему телу корпуса, его будет невозможно удалить одной только поверхностной шлифовкой.

Изменение обращения с электронными отходами

Более эффективное управление электронными отходами тоже облегчило бы проблему подделок. Электронные отходы массово разбирают ради использованных деталей. Чипы серии MSM, грубо выпаянные из плат, - мозги многих Android-смартфонов, сделанные Qualcomm и продаваемые под брендом Snapdragon, - можно купить на вес, примерно по 10 центов за чип. Фальсификаторы очищают чипы, выполняют реболлинг (то есть добавляют новые шарики припоя для корпусов ball-grid array, BGA) и иногда перемаркируют их, помещают в ленты и катушки и продают как совершенно новые, получая наценку в 10 раз выше исходной закупочной цены. Одна партия восстановленных чипов может принести тысячи долларов, делая эту практику убедительным источником дохода для квалифицированных работников, которые иначе зарабатывали бы \$200 в месяц на фабрике, делая ровно то же самое.[3] Фабрикам обычно разрешено извлекать чипы с дефектных плат или из потребительских возвратов, которые нельзя отремонтировать.

Если бы United States перестали отправлять электронные отходы за океан на утилизацию или хотя бы измельчали детали перед отправкой, предложение для рынков восстановленных чипов уменьшилось бы. Внутренняя переработка электронных отходов также создала бы больше рабочих мест, ресурс не менее ценный, чем золото.

С другой стороны, я считаю переработку на уровне компонентов в долгосрочной перспективе довольно хорошей для окружающей среды и человеческой экосистемы. Большинство электронных деталей будут прекрасно работать еще много лет после того, как потребитель выбросил устройство, а развивающиеся экономики создают жадные до технологий рынки, которые не могут позволить себе новые детали, купленные на первичном рынке.

Резерв подлинных деталей

Последний вариант обеспечить надежность критически важного военного оборудования мог бы состоять в создании стратегического резерва деталей. Производственный выпуск военных самолетов ограничен, возможно, сотнями единиц - это небольшой объем по сравнению с производственными тиражами потребительской электроники. Я представляю, что пожизненный спрос на деталь, включая замены, ограничивается десятками тысяч единиц. Физически такой резерв деталей тогда не является неподъемным: 10 000 чипов поместятся в большую коробку из-под обуви.

С финансовой точки зрения, я оцениваю, что покупка резерва исходных запасных компонентов для критически важных систем авионики добавила бы только долю процента к стоимости самолета. Это могло бы даже привести к долгосрочной экономии, поскольку производители могут достичь большей эффективности масштаба, если сразу запустят одну большую партию.

Очевидно, меры против подделок были бы невероятно полезны и в гражданских проектах. Я сочувствую каждому, кому приходится иметь дело с поддельными деталями, поскольку сам несколько раз обжигался. Вот история об особенно неприятной проблеме, с которой я столкнулся во время работы над chumby One.

Поддельные microSD-карты

В декабре 2009 года, в середине производственного тиража chumby One, я начал криминалистическое расследование, чтобы выяснить правду о некоторых подозрительных картах памяти Kingston. Фабрика позвонила и сказала, что выход годных при SMT резко упал на одной партии chumby One, поэтому я поехал туда посмотреть, что можно сделать, чтобы исправить проблему. После осмотра и проверки нескольких chumby One я понял, что все отказавшие устройства имели карты microSD Kingston из конкретного кода партии. Я попросил фабрику снять всю партию microSD-карт с линии и переделать устройства, в которые эти карты были установлены. После замены карт выход годных вернулся к норме.

История должна была закончиться на этом. В такой ситуации я обычно получил бы разрешение на возврат товара (RMA) от производителя для дефектных деталей, обменял бы партию на рабочие детали и двинулся дальше. Но у меня было две проблемы. Во-первых, Kingston не хотела принимать карты обратно, потому что мы их запрограммировали. Во-вторых, дефектных карт было много (около 1000 всего, а chumby уже сильно отставала по невыполненным заказам), и карты памяти недешевы. Такой тип карты памяти тогда стоил примерно 4 или 5 долларов, оставляя несколько тысяч долларов лома, если бы мы не смогли их обменять. Chumby не могла позволить себе просто списать несколько тысяч долларов, поэтому я включил режим расследования.

Видимые отличия

Неровная внешняя маркировка была первым подозрительным признаком, который я заметил у дефектных карт Kingston.



Нестандартная microSD-карта (слева) и обычная карта (справа). Стрелки и кружки показывают подозрительные различия.

Самое странное физическое отличие состояло в том, что код партии на нестандартной карте был нанесен через тот же трафарет, что и основной логотип. Наносить код партии шелкографией не необычно, но обычно производитель не использует один и тот же трафарет для кода партии и логотипа. Должно быть некоторое различие в цвете, шрифте или выравнивании кода партии относительно остального текста. Вся партия нестандартных карт также имела один и тот же код партии (N0214-001.A00LF). Обычно код партии меняется хотя бы каждые несколько сотен карт. Сравните нестандартную карту с обычной, на которой маркировка нанесена лазером. Коды партии обычных карт менялись с каждым лотком из 96 штук.

Вторая странная особенность была тоньше и, возможно, не решающей: неровность в логотипе microSD. Брендоры поставщики вроде Kingston очень придирчивы к точности своих логотипов: у карт SanDisk буква D разорвана, но карты Kingston, продаваемые в United States, почти повсеместно используют сплошную D.

Исследование карт

Странности во внешней маркировке были только началом. Когда я прочитал электронные ID-данные двух карт (проверяя записи /sys в Linux), вот что я нашел в нестандартной карте:

```
cid:41343253443247422000000960400049
csd:002600325b5a83a9e6bbff8016800095
date:00/2000
fwrev:0x0
hwrev:0x2
manfid:0x000041
name:SD2GB
oemid:0x3432
scr:0225000000000000
serial:0x0000960
```

А вот что я нашел в обычной карте:

```
cid:02544d5341303247049c62cae60099dd
csd:002e00325b5aa3a9ffffff800a80003b
date:09/2009
fwrev:0x4
hwrev:0x0
manfid:0x000002
name:SA02G
oemid:0x544d
scr:0225800001000000
serial:0x9c62cae6
```

Во-первых, обратите внимание на код даты на нестандартной карте. Даты считаются как смещение от 00/2000 в поле CID, поэтому значение 00/2000 означает, что производитель не потрудился назначить дату. Кроме того, в 2000 году 2GB microSD-карт даже не существовало. Серийный номер на дефектной карте тоже очень низкий: в десятичном виде 0x960 - это 2 400. Другие карты в нестандартной партии имели столь же низкие серийные номера, в сотнях или тысячах.

Для популярного продукта вроде microSD-карты шанс получить самые первые единицы с фабрики довольно мал. Например, серийный номер обычной карты - 0x9C62CAE6 в шестнадцатеричном виде, или 2 623 720 166 в десятичном, что гораздо более правдоподобно. Очень низкие серийные номера, как и очень низкие MAC ID-адреса, являются характерными признаками «призрачной смены».

Наконец, идентификатор производителя на нестандартной карте - 0x41 (заглавная A в ASCII), который я не узнал.[4] Номер идентификатора OEM-производителя (OEMID) был 0x3432 - ASCII 42, что на единицу больше шестнадцатеричного значения идентификатора производителя. Идентификаторы производителей обычно являются ASCII-символом, заданным

шестнадцатеричным значением, а не самими шестнадцатеричными числами. Путаница между hex и ASCII - возможный признак того, что «призрачную смену», изготовившую эти карты, запускал человек, который не понимал смысл полей.

Были ли microSD-карты подлинными?

Вооружившись этими доказательствами, Chumby обратилась к дистрибьютору Kingston в Китае и торговому представителю Kingston в США. Мы спросили, были ли карты подлинными и, если да, почему коды сериализации были нестандартными. Через некоторое время Kingston поклялась, что карты подлинные, а не подделки, но изменила свою позицию по обмену карт. Компания забрала запрограммированные карты и дала нам новые, без дальнейших вопросов.

Однако Kingston так и не сказала, почему ID-номера карт были нестандартными. Я знаю, Chumby была мелкой рыбешкой по сравнению с мировыми Nokia, но компании все же должны отвечать на базовые вопросы о контроле качества, даже мелким клиентам. Однажды мне случайно отправили старую версию детали Quintic, и как только я смог доказать проблему, я получил обслуживание мирового уровня от Quintic. Компания дала мне подробное объяснение и немедленно оплатила полный обмен деталей. Это был образцовый сервис, и я хвалю и настойчиво рекомендую Quintic за него. Kingston, с другой стороны, не подала пример, которому стоит следовать.

Обычно я бы исключил Kingston как поставщика, но я был настойчив. Было тревожно, что известный и устоявшийся бренд готов поддерживать столь нестандартные компоненты. Кто мог сказать, что SanDisk или Samsung не сделали бы то же самое? Падение цен в то время сильно ударило по поставщикам флеш-памяти, и как мелкий клиент я мог быть использован любой из этих компаний как приемник пограничного материала, чтобы улучшить их прибыльность. Учитывая относительно высокую стоимость microSD-карт, мне нужны были правила входного контроля качества (IQC) для инспекций, по которым можно было принимать или отклонять поставки от поставщиков памяти на основе заданных стандартов качества. Чтобы разработать эти правила, я продолжил копать правду об этих картах.

Дальнейшее криминалистическое исследование

Сначала я собрал много образцов microSD-карт. Я хотел собрать и обычные, и нестандартные карты в реальной среде, поэтому отправился в район Hua Qiang Bei и побродил по тамошним серым рынкам. Я купил 10 карт памяти у мелких продавцов по ценам от 30 до 50 RMB (\$4.40-\$7.30 USD).

Покупать нестандартные карты было интересно. Поговорив с парой десятков продавцов, я узнал, что Kingston как бренд был слаб в Китае для microSD-карт. SanDisk занималась маркетингом намного активнее, поэтому карты SanDisk было гораздо проще найти на открытом рынке, а качество серых SanDisk-карт было довольно стабильным.

Мелкие продавцы также совершенно беззастенчиво продавали хорошо сделанные подделки. У них были голые карты, свободно лежавшие в лотках витрины. Фотографии на странице 11 в главе 1 показывают, как выглядит прилавок продавца SD-карт. Как только я согласился на цену и подтвердил покупку карты, продавец бросил свободную карту в "настоящую" розничную упаковку Kingston, чудесным образом извлек сертификат - с голограммой, серийными номерами и URL kingston.com для проверки покупки - и прямо у меня на глазах наклеил сертификат на заднюю сторону розничной упаковки.



Свежекупленная microSD-карта Kingston. Она была совсем как новая!

Один продавец особенно меня заинтересовал. Там буквально были мама, папа и маленький ребенок, сидевшие в небольшом киоске на рынке мобильных телефонов. Они были заняты тем, что вставляли десятки не-Kingston карт в розничную упаковку Kingston. Они вовсе не хотели продавать мне, но я был настойчив. Эта карта заинтересовала меня особенно, потому что у нее тоже был логотип с разорванной D, но не было маркировки Kingston. Предыдущая фотография показывает эту карту и упаковку, в которой она была; карта является образцом 4 в следующем разделе, где можно увидеть подробный анализ семи разных microSD-карт из моей поездки за покупками.

Сбор данных

После сбора образцов я считал их ID-информацию карт, проверяя записи /sys в Linux, а затем снял корпуса, растворив их азотной кислотой. Как видно на фотографиях в таблице 2, моя техника снятия корпуса была довольно грубой. Большая часть повреждений карт возникла от удаления растворенного герметика ацетоном и ватной палочкой. Мне пришлось действовать немного жестко, что не пошло на пользу соединительным проводам. Но для моих целей этого было достаточно.

Вот вся базовая информация, которую я извлек из этих карт:

Образец 1. Нестандартная карта, с которой началось все это расследование. Она была куплена через официального дистрибьютора Kingston в Китае, и, насколько мне известно, ни одна такая карта не была отправлена конечным клиентам Chumby. MID = 0x000041, OEMID = 0x3432, serial = 0x960, name = SD2GB.

Образец 2. Обычная карта, которую я купил у того же официального дистрибьютора Kingston в Китае, где купил образец 1. Она была типичной для microSD-карт, фактически отправленных в первой партии chumby Ones. MID = 0x000002, OEMID = 0x544D, serial = 0x9C62CAE6, name = SA02G.

Образец 3. Карта Kingston, купленная через крупную розничную сеть США. MID = 0x000002, OEMID = 0x544D, serial = 0xA6EDFA97, name = SD02G. Обратите внимание, что MID и OEMID идентичны образцу 2, но не образцу 1.

Образец 4. Карта не Kingston, которую я видел вставленной в упаковку с маркировкой Kingston; куплена на открытом рынке в Шэньчжэне. MID = 0x000012, OEMID = 0x3456, serial = 0x253, name = MS. Обратите внимание на низкий серийный номер.

Образец 5. Устройство от более надежного розничного продавца на рынке Шэньчжэня. Я купил его, потому что у него была маркировка XXX.A00LF, как у моей исходной нестандартной карты. MID = 0x000027, OEMID = 0x5048, serial = 0x7CA01E9C, name = SD2GB.

Образец 6. Карта SanDisk, купленная на открытом рынке в сомнительном магазине, которым управляла бойкая девушка, непрерывно курившая и не отрывавшаяся от сообщений. На самом деле я приобрел в общей сложности три карты SanDisk из разных сомнительных источников, но все они проверились с одной и той же CID-информацией, поэтому я вскрыл только одну. MID = 0x000003, OEMID = 0x5344, serial = 0x114E933D, name = SU02G.

Образец 7. Карта Samsung, которую я купил у оптового дистрибьютора Samsung. Я не сканировал ее перед вскрытием корпуса, и снаружи у карты фактически не было маркировки (она была пустой, только с лазерной меткой на обратной стороне), поэтому я ее не фотографировал. По одному внешнему виду она была самой подозрительной из всей группы, но оказалась одной из самых качественно сделанных. Нельзя судить книгу по обложке! MID = 0x00001B, OEMID = 0x534D, serial = 0xB1FE8A54, name = 00000.

Данных много, и у меня была серьезная работа: вывести из всего этого какой-то полезный вывод.

Примечание. Интересно, что одна SanDisk-карта из трех в образце 6 оказалась использованной и только быстро отформатированной. С помощью программы восстановления я нашел DLLs, WAVs, карты и сертификаты VeriSign, принадлежащие Navione's Careland GPS. Когда-нибудь я приобрету много восстановленных microSD-карт и соберу с них интересные данные.

Таблица 2: разбивка всех карт, собранных для расследования.

Table 2: A Breakdown of All the Cards Collected for the Investigation

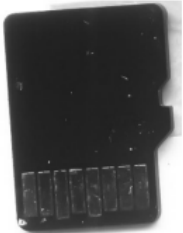
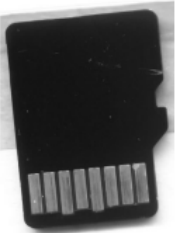
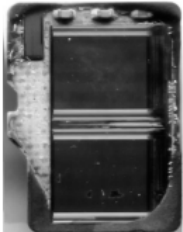
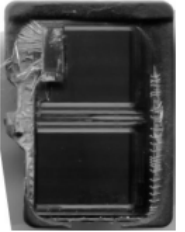
	Sample 1: Original Kingston card from authorized Kingston distro	Sample 2: Normal Kingston card from authorized Kingston distro	Sample 3: US retail Kingston card
Front marking			
Back marking			
Decapsulated			
Controller die marking		 	
Flash die marking		SanDisk/Toshiba flash	

Таблица 2, samples 1-3: исходная карта Kingston от официального дистрибьютора Kingston; обычная карта Kingston от официального дистрибьютора Kingston; карта Kingston из розничной сети США.

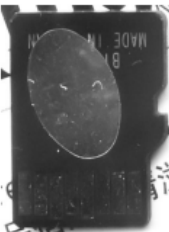
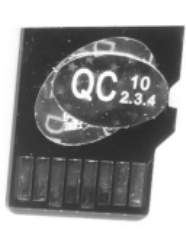
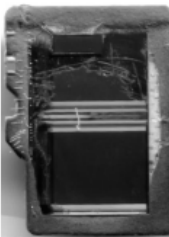
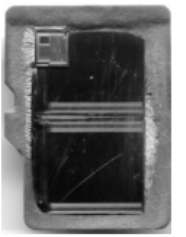
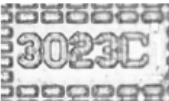
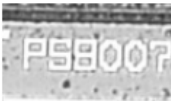
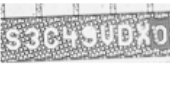
Sample 4: Fake card bought from Shenzhen market	Sample 5: Questionably authentic Kingston card bought from Shenzhen market	Sample 6: SanDisk card bought from Shenzhen market	Sample 7: Samsung card bought from authorized Samsung distro
			Photo unavailable
			Photo unavailable
			
			
SanDisk/Toshiba flash		SanDisk/Toshiba flash	

Таблица 2, samples 4-7: поддельная карта, купленная на рынке Шэньчжэня; карта Kingston сомнительной подлинности, купленная на рынке Шэньчжэня; карта SanDisk, купленная на рынке Шэньчжэня; карта Samsung от официального дистрибьютора Samsung.

Итоги наблюдений

Вот самые интересные общие выводы, которые я сделал из своего обзора:

- "Обычные" карты Kingston (Samples 2 and 3) были изготовлены Toshiba, как указывают маркировки flash-кристаллов и их OEMID. В ASCII 0x544D - это TM, предположительно от Toshiba Memory. Эти карты используют контроллеры Toshiba и микросхемы памяти Toshiba и, похоже, имеют хорошее качество. К счастью, именно они были единственными, отправленными клиентам Chumby.
- Нестандартная карта (образец 1) использовала тот же чип контроллера, что и откровенная подделка (образец 4), которую я купил на рынке. И нестандартная Kingston, и поддельная Kingston имели низкие серийные номера и странную ID-информацию. Обе эти карты проявляли ненормальное поведение при определенных обстоятельствах. Я все еще колеблюсь называть нестандартную карту Kingston подделкой, потому что это очень серьезное обвинение, но ее конструкция была похожа на другую карту явно сомнительного качества, что заставляет меня сомневаться в выборе официальных производственных партнеров Kingston.
- Нестандартная карта - единственная карта в группе, которая не использует конструкцию многослойного CSP (stacked CSP). Вместо этого она использует бондинг «бок о бок» - то есть микроконтроллер и чип памяти просто размещены рядом друг с другом. В многослойном CSP микроконтроллер помещают поверх чипа памяти. Это значительно сложнее, чем размещение бок о бок, потому что с чипов сначала нужно сточить инертный материал задней стороны, чтобы общая высота стопки поместилась внутри такого тонкого корпуса. Несмотря на сложность, укладка чипов друг на друга популярна, потому что позволяет поставщикам уместить больше кремния на той же площади.
- Единственные две фабрики чипов памяти в этом наборе образцов - Toshiba/SanDisk и Samsung. SanDisk и Toshiba совместно владеют фабрикой, которая делает их чипы памяти.
- NAND-кристалл Samsung, самая дорогая часть microSD-карты, примерно на 17 процентов больше кристаллов Toshiba/SanDisk. Это значит, что microSD-карты Samsung естественным образом должны иметь чуть более высокую цену, чем карты Toshiba/SanDisk. Однако Samsung может компенсировать это возможностью поместить тот же голый кристалл, который обычно втискивают внутрь корпуса microSD, в корпус TSOP (thin small outline package), пригодный для машинной сборки на уровне платы. Если спрос на microSD-карты падает, Samsung может засунуть избыток голых кристаллов в корпуса TSOP и продать их третьим сторонам, которые делают обычную машинную сборку чипов. Плюс у Samsung также нет посредника вроде Kingston, который съедает маржу.

Я знал (как и многие другие в производстве), что Kingston не является производителем полупроводников в том смысле, что не владеет фабриками, но это исследование намекало, что Kingston не занималась и собственной оригинальной разработкой. Я надеялся хотя бы найти внутри карт Kingston контроллерный чип под брендом Kingston, даже если сам чип был изготовлен сторонней фабрикой. Я также ожидал увидеть, что Kingston закупает чипы памяти у более широкого круга компаний. Возможность балансировать цепочку поставок и меньше зависеть от одного крупного конкурента по чипам была бы заметным преимуществом для клиентов и дала бы Kingston переговорную силу для получения лучшей цены, доступную далеко не всем. Но каждая карта Kingston, которую я купил, имела внутри чип памяти SanDisk/Toshiba. Единственная "добавленная ценность", которую я увидел, была в выборе чипа контроллера.

Как ни странно, из всех поставщиков Kingston предложила Chumby лучшие сроки поставки и цены, несмотря на то что SanDisk и Samsung делают весь свой кремний сами и потому имеют более низкие внутренние затраты. Это сказало мне, что Kingston, должно быть, имеет очень низкую маржу на своих microSD-картах, что могло бы объяснить, почему нестандартные карты нашли путь в ее цепочку поставок. Kingston, вероятно, также охотнее разговаривает с небольшими клиентами вроде

Chumby, потому что как каналный бренд не может конкурировать с OEM вроде SanDisk или Samsung за крупнейшие контракты с компаниями уровня Nokia и Apple.

Итак, нестандартная microSD-карта, которую я вытащил с производственной линии chumby One, возможно, не была подделкой, но все равно была порождением китайской экосистемы перемаркировки. Kingston - скорее торговец канала, чем поставщик технологии, и, вероятно, SanDisk и Toshiba видят ее как буфер спроса для своего производственного выпуска. Я также не удивился бы, если SanDisk/Toshiba продавали Kingston неидеальные детали, оставляя лучшее из партии себе. Поэтому я ожидал бы, что карты Kingston имеют чуть больше дефектных секторов, но благодаря магии коррекции ошибок и запасных секторов этот факт скрыт от конечных пользователей.

В результате Kingston играет важную роль в стабилизации цен на microSD-карты и улучшении маржи фабрик. Но потенциальный конфликт интересов кажется ошеломляющим, и мне до сих пор очень любопытно, как возникла эта экосистема. Покупать значительный объем технологии конкурента у фабрики конкурента и при этом продавать по конкурентной цене для меня противоречит интуиции, и, возможно, моей величайшей ошибкой в расследовании той нестандартной microSD-карты было то, что я ожидал чего-то другого.

Поддельные FPGA

Любой, кто достаточно долго занимался производством в Китае, имеет больше одной истории о нарушениях в цепочке поставок. Вот еще одна из моих любимых историй, которая подчеркивает некоторые ключевые стимулы, побуждающие участников обманывать.

Проблема белого экрана

Это был март 2013 года, и я завершал первый серийный производственный прогон специализированной платы контроллера робототехники под кодовым именем Kovan.[5] В конце любого производственного прогона я всегда просматриваю список проблем, возникших в производстве, чтобы определить области для улучшения. Производство - это сизифова борьба за совершенство: в каждом прогоне есть несколько изделий, которые просто приходится списать, а разница между прибылью и убытком заключается в том, насколько хорошо вы можете управлять уровнем брака.

В этом прогоне одна конкретная проблема, названная "проблемой белого экрана" по самому очевидному симптому, была доминирующей. Около 4 процентов всего выпуска проявляли эту проблему, составляя почти 80 процентов отказов устройств. Я попросил фабрику отправить мне несколько образцов отказавших устройств, чтобы проанализировать их подробнее.

Как я часто обнаруживал при анализе отказавших устройств, самый очевидный симптом проблемы был лишь косвенно связан с первопричиной. LCD-экран выглядел белым на этих устройствах, потому что FPGA не конфигурировалась. FPGA, сокращение от Field Programmable Gate Array (программируемая пользователем вентильная матрица), по сути является массивом логики и памяти, встроенным в плотную сеть проводников, который можно настроить во время запуска на определенное поведение. Поведение FPGA обычно описывается на высокоуровневом языке, напоминающем язык программирования вроде C (например, Verilog) или Ada (например, VHDL), который затем компилируется в конфигурационный битовый поток.

FPGA очень удобны для реализации чувствительных ко времени аппаратных интерфейсов, которые программному обеспечению было бы трудно эмулировать. В этом конкретном приложении FPGA управляла всем: от моторов до датчиков и даже LCD. Когда FPGA не конфигурировалась, LCD не получал сигналы синхронизации и данных, из-за чего показывал пустой белый экран вместо ожидаемых заводских тестовых шаблонов.

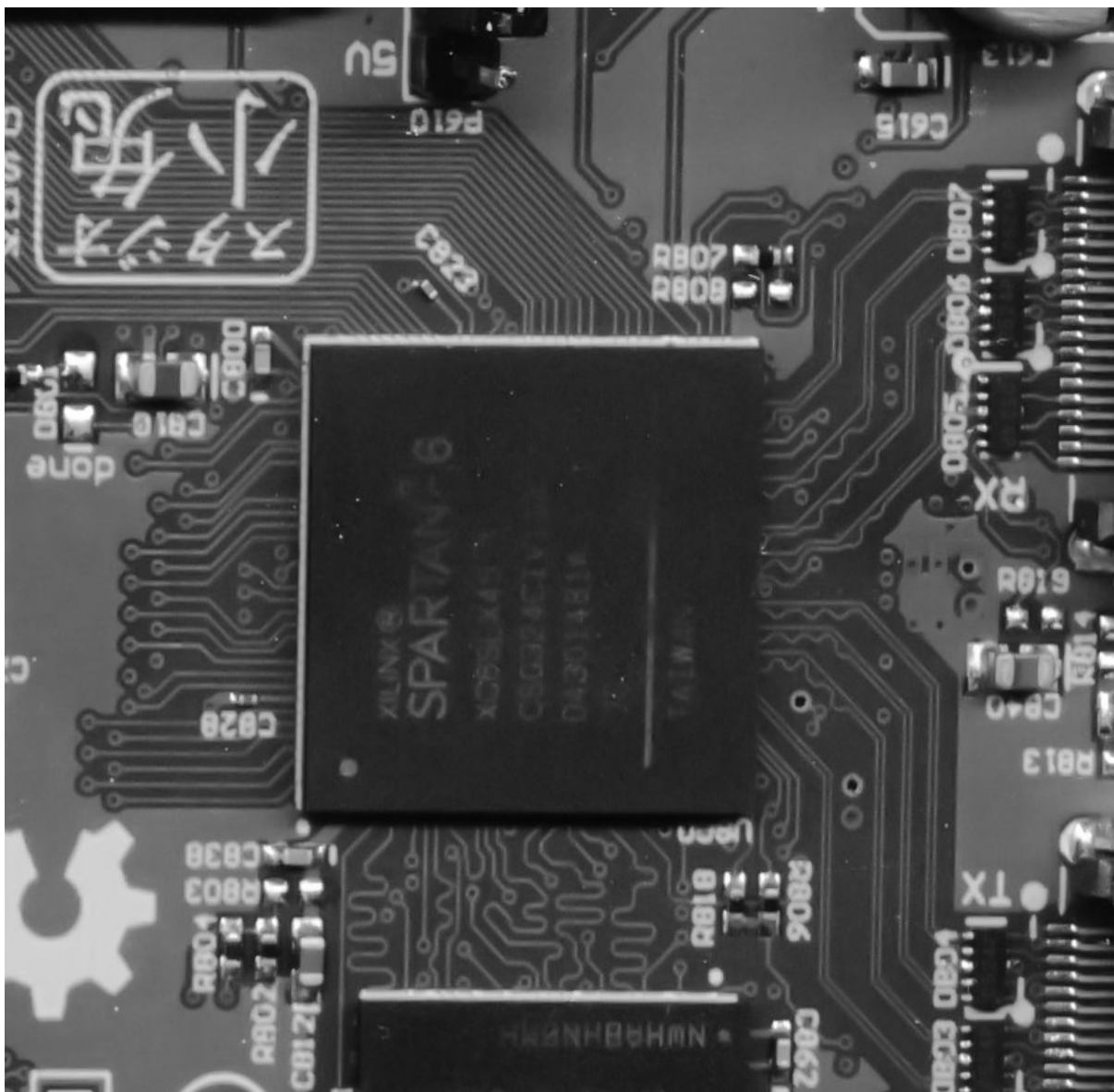
Отказ FPGA был серьезной проблемой. Во-первых, FPGA был с большим отрывом самой дорогой частью на плате, около \$11 за чип. Я также беспокоился, что эта проблема может указывать на более глубокую проблему разработки. Возможно, стабилизаторы питания FPGA были нестабильны, или, может быть, была проблема с последовательностью загрузки, усугублявшая граничный случай в таймингах конфигурации, который мог проявиться в "хороших" производственных устройствах по мере их старения. Ситуация явно требовала более глубокого расследования.

Неверные ID-коды

Я подключил отладочную консоль, разобрался в проблеме и обнаружил, что отказ был связан с тем, что FPGA не отвечала правильным ID-кодом. ID-код проверяется запросами по тестовой шине доступа, известной как JTAG. Большинство пользователей не проверяют ID FPGA перед программированием, но мы встроили проверку ID-кода в Kovan, потому что позволяли клиентам указывать, какую емкость FPGA они хотят использовать для конкретной производственной партии. Некоторые приложения более требовательны, тогда как другие более чувствительны к цене. В результате у клиента мог быть смешанный запас FPGA, и мы хотели обнаруживать и защищать аппаратуру от случайного несовпадения между битовым потоком и FPGA.

Но это была единая производственная партия, и теоретически все FPGA должны были быть одинаковыми. Как же тогда FPGA вообще мог сообщать несовпадающий ID-код? Я некоторое время чесал голову и подозревал ошибку в нашей реализации JTAG, пока не проверил сообщенный ID-код. Это был знакомый код - но для кремния, маркированного как "инженерные образцы" от Xilinx, поставщика, который делает эти FPGA. Инженерные образцы - это предсерийные устройства, продаваемые Xilinx; у них есть небольшие известные дефекты, но они достаточно функциональны для большинства приложений, до такой степени, что большинство клиентов не увидели бы разницы, кроме ID-кода.

Я внимательнее посмотрел на PCB и впервые заметил, что в поверхность FPGA был лазером вытравлен маленький белый прямоугольник. Прямоугольник находился прямо под номером детали, там, где обычно ставится обозначение "ES" для инженерного образца. Кто-то стер эти буквы и продал нам инженерные образцы как полноценные серийные детали!



Для сравнения: FPGA того же типа, с которой не проводили вмешательства.

Проблема совершенно явно была проблемой цепочки поставок, а не проблемой разработки. Кто-то в цепочке брал ES-кремний, стирал буквы и подмешивал его к законным деталям в доле около 3-5 процентов. Обычно Xilinx требовала бы, чтобы весь ES-кремний в запасах дистрибьютора списывался, когда становятся доступны серийные детали, но ES-детали были почти полностью функциональны, до такой степени, что большинство приложений не пострадало бы. Серийный битовый поток без проблем загружался бы в ES-деталь, и никто не узнал бы разницы. Единственный способ отличить их друг от друга - делать проверку ID-кода, что, как я отмечал ранее, нетипично.

Таким образом, подмешивание ES-кремния в производственные партии, скорее всего, осталось бы незамеченным. Подмешивать ES-детали в доле 3-5 процентов тоже было очень хитро: низкая доля подмешивания делает такие замены очень трудными для выявления без стопроцентной предварительной проверки деталей. Даже в производстве, если ES-кремний пограничный, было бы безумно трудно установить первопричину проблемы из-за ее редкости.

Фактически существует корреляция между сложностью производства и использованием FPGA. Обычно, если ваша разработка требует FPGA, вы расширяете границы сразу на нескольких направлениях, поэтому уровень брака в несколько процентов ожидаем. Маржа на аппаратуре с FPGA также часто достаточно высока, чтобы клиент просто принял 4 процента отказов. Значит, кто бы это ни сделал, он точно знал, что делал; это были почти безрисковые деньги.

Наконец, важно отметить, что большинство поставщиков в цепочке поставок выживают на однозначной марже, поэтому найти дополнительные 3-5 процентов "бесплатных денег" на самой дорогой детали платы фактически удваивает прибыльность. Это дает очень сильный стимул обманывать, особенно если вы думаете, что вас не поймут.

Решение

Решение этой проблемы было довольно интересным. Я встретился с менеджерами и гендиректором AQS, контрактного производителя, которому было поручено производство Kovan, рассказал им о проблеме и показал накопленные доказательства. Когда моя презентация закончилась, гендиректор не стал указывать пальцем на вышестоящих поставщиков или партнеров. Вместо этого он сразу посмотрел своим сотрудникам в глаза и спросил: "Кто-нибудь из вас это сделал?" Он лучше всех в комнате понимал, что любой отдельный закупщик или менеджер мог фактически удвоить свой доход за этот месяц, если бы сумел повернуть этот обман и не попасться.

Иными словами, по-настоящему примечательная часть этой ситуации заключается в том, насколько редко происходит проблема, с которой я столкнулся, учитывая ставки и то, как трудно такие проблемы ловить. И хотя у меня есть несколько хороших историй для бара о подделках в цепочке поставок, помните, что я также отгрузил сотни тысяч экземпляров хорошего продукта. Большинство людей, с которыми я работал в Китае, - трудолюбивые, честные люди, которые отказываются от легких возможностей обмануть меня и получить прибыль. Важно не обобщать целое по плохим поступкам немногих.

В конце концов поставщик, продавший нам чипы, не признал вину, но заменил все перемаркированные детали за свой счет. Мы все равно должны были оплатить труд по замене чипов и повторной сертификации плат. Это примерно самый близкий вариант к мирному решению в Китае, когда вы не гигант вроде Apple или Foxconn. Я отправил сообщение в штаб-квартиру Xilinx о возможном недобросовестном поведении одного из их авторизованных поставщиков, но в конце концов я мелкий клиент, а подмена деталей могла произойти буквально в любом месте цепочки поставок. Даже курьер, доставлявший упаковки, мог сделать замену.

Для Xilinx не стоило бы затрат в людях, отношениях и внимании расследовать проблему и вычислять одного плохого участника среди буквально сотен возможных подозреваемых. Но мне хочется представить, что хотя бы служебная записка была разослана, и тот, кто подменял детали на ES, достаточно испугался и остановился.

Заключительные мысли

В конце концов либеральная IP-экосистема имеет преимущества и недостатки. Как инженер и дизайнер, я предпочитаю быть в экосистеме, где идеи доступны, даже если это значит, что мне нужно быть настороже из-за периодических проблем с поддельными товарами. Иначе говоря, фундаментальная предпосылка вирусного распространения - возможность делать копии. Взрыв интереса к аппаратным стартапам отчасти обязан высококонкурентной производственной экосистеме, которая могла процветать только в культуре "продукт важнее патента".

Западные люди, которые приезжают в Китай без понимания принципов *gongkai* и *guanxi*[6], часто чувствуют, будто их обманывают. Но как только вы понимаете правила и учитесь использовать их в своих интересах, вы больше не будете чувствовать, что игра настроена против вас.

В американской системе IP честь имеет малую экономическую ценность, а закон превалирует над честью. Например, патентный троллинг - совершенно законный и очень прибыльный способ зарабатывать на жизнь. В китайской системе, однако, репутация может превалировать над законом. Это открывает дверь для коррупции, но также передает сообществу контроль за соблюдением социальных и моральных ценностей, создавая рыночную ценность чести, особенно в местных тесно связанных сообществах.

Разумеется, подход к зарабатыванию денег через закрытие идей и продажу прав на них явно несовместим с либеральной IP-экосистемой. К счастью, представление о том, что идеи являются общественным достоянием, хорошо сочетается с моей философией открытого исходного кода. В следующей части книги я подробнее расскажу о моем опыте создания открытого железа и бизнесов, укорененных в этих принципах.

Примечания

[1] См. Sec 2320 (b) на <https://www.govtrack.us/congress/bills/112/hr1540/text>.

[2] Помните такой? NeXTcube был компьютером, выпущенным в 1990 году компанией NeXT Стива Джобса.

[3] Это был уровень зарплаты в середине 2000-х; из-за инфляции зарплат с тех пор он вырос примерно до \$1 000 в месяц, но восстановление чипов все еще более прибыльно.

[4] Публикация JEDEC N. 106AA перечисляет все ID-коды производителей SD-карт, и 0x41 там не было.

[5] Kovan - открытое железо; подробнее о нем можно прочитать и скачать исходники на Kosagi wiki: http://www.kosagi.com/w/index.php?title=Kovan_Main_Page.

[6] Guanxi (关系) - традиционная социальная сеть, глубоко встроенная в китайскую культуру. Как современные социальные сети, она имеет понятия подписчиков, лайков/дизлайков, кармы и модераторов. Guanxi предшествует современной правовой системе и может быть эффективнее гражданского кодекса для разрешения или предотвращения самых разных споров. Guanxi также необходима для заключения новых сделок и построения отношений.

Часть 3. Что открытая аппаратура значит для меня

Глава 6. История Chumby

Одним из моих самых ранних проектов открытого железа был chumby - устройство для доставки контента по Wi-Fi, которое привело меня в Китай, чтобы в 2007 году настроить мою первую цепочку поставок.[1] Работа над chumby была для меня лично захватывающей по двум причинам.

Во-первых, у меня была возможность создать продукт, который мог бы в какой-то небольшой мере улучшить жизнь людей. Пользователи, которые постоянно онлайн, ведут блоги и полагаются на IM, чтобы поддерживать связь, могли использовать chumby, чтобы упростить свое общение. В то же время chumby был шансом создать по-настоящему открытую платформу, позволявшую хакерам разбирать, изменять и переделывать ее так, как им хотелось.

Платформа, дружелюбная к хакерам

Хакеры испытывают ненасытное желание расширять, изменять, настраивать и использовать потребительские продукты не по назначению, чтобы обнаруживать непредусмотренную функциональность. В Chumby мы надеялись, что хакеры разберутся, как работает устройство, и преобразуют его во что-то, чего мы никогда не представляли, поэтому мы спроектировали chumby максимально открытым для любого, кто хотел его взломать или переделать. Мы учитывали не только хакеров программного обеспечения с открытым исходным кодом, но также аппаратных хакеров, художников и ремесленников - то есть людей, умелых и увлеченных некомпьютерными вещами вроде обработки металла, шитья или столярного дела. Чтобы поощрить и поддержать хакеров chumby, мы сделали исходный код, схемы, разводки плат, ведомость материалов, плоские выкройки и 3D CAD-базы пластиковых деталей свободно доступными. Все это все еще можно найти на chumby wiki (<http://wiki.chumby.com/>).



Оригинальный мягкий chumby.



Оригинальный мягкий chumby, вид сзади.

Идея состояла в том, чтобы дать хакерам возможность перейти от точечных взломов непонятной аппаратуры к проектам, которыми они могли бы поделиться почти с кем угодно. Например, представьте, что вы добавляете манжету для измерения давления к chumby и отдаете chumby своей бабушке. Теперь вы можете проверить здоровье бабушки, а она может смотреть фотографии внуков, пока ей измеряют давление. Но представьте тот же сценарий с роутером WRT-54G вместо chumby. Конечно, вы можете добавить манжету для измерения давления и к WRT-54G (фактически архитектурно он довольно похож на chumby), но попробуйте научить бабушку, как его настроить и использовать. Иными словами, мы чувствовали, что превращение chumby в простой продукт позволит хакерам делать собственные проекты более пригодными к использованию и понятными для менее технических людей в их жизни.

Открытость chumby имела и другие преимущества для хакеров. На этот раз представьте, что ваш термостат находится немного дальше от места, где вы на самом деле хотите регулировать температуру. Вы могли бы решить эту проблему за выходные, добавив датчик температуры к chumby. Платформа Chumby имеет Wi-Fi, а я построил пакет датчиков для хакеров под это устройство, поэтому проект потребовал бы минимальной грязной аппаратной работы: вы бы просто модифицировали два chumby (один с датчиком температуры и один с интерфейсом к термостату) и включили на обоих пакет датчиков. Такое устройство не только помогло бы поддерживать правильную температуру в гостиной, но и сообщало бы последние новости и помогало отслеживать любимые телепередачи.

Вишенка на торте состояла в том, что вы также были бы свободны публиковать свои модификации и даже перепродавать модифицированные chumby с такими custom-возможностями. Другие могли бы извлечь пользу из вашей работы, а вы могли бы немного заработать. Если говорить легче: оригинальный корпус chumby был сделан из ткани, так что вы могли даже изменить его под свой интерьер!

Оригинальный дизайн chumby, теперь называемый chumby classic, впервые показали на FOO Camp в 2006 году, а в продажу он поступил в 2008 году. К сожалению, полноценный запуск chumby classic пришлось на середину худшего экономического спада со времен Великой депрессии. Его милый, уютный форм-фактор стоил столько, что многие потребители просто не могли себе его позволить, поэтому я сделал то, что сделал бы любой предприниматель во время рецессии: сократил масштаб.

Эволюция chumby

Вскоре после того, как Lehman Brothers в 2008 году подала заявление о защите от банкротства по главе 11 Кодекса США о банкротстве, мы начали работу над продуктом, который мог бы отвечать новой экономической реальности. Когда я рисовал первые эскизы на салфетке для продукта, который позже мы назвали chumby One, фондовый рынок находился в свободном падении и терял по несколько сотен пунктов в день. С учетом этого ключевой целью было снижение стоимости. Я внимательно и жестко посмотрел на весь дизайн, чтобы построить более дешевый и быстрый продукт, который лучше подходил бы рынку. Мы хотели, чтобы chumby One привлек новых покупателей, но сохранил лояльность нашей существующей потребительской базы, и хотели выпустить его к Рождеству 2009 года.

К счастью, инженер по применению из Freescale (позже приобретенной NXP) связался со мной насчет нового, необычайно недорогого CPU (i.MX233), который Freescale планировала выпустить в 2009 году. Он выглядел многообещающим и подходящим для chumby, поэтому я набросал несколько черновых рендеров и просчитал несколько вариантов себестоимости. На CES в январе 2009 года мы показали новый дизайн нескольким потенциальным клиентам, чтобы получить обратную связь по функциям и цене. Работа неторопливо шла до марта, и после китайского Нового года я построил первую плату-прототип.

Примечание. Одна действительно классная вещь в i.MX233 состоит в том, что он имеет встроенные регуляторы питания, и это не просто линейные регуляторы: это импульсные регуляторы. Но это не просто какие угодно импульсные регуляторы; они получают три напряжения, используя всего один индуктор! Насколько это круто? Я должен отдать должное человеку, который спроектировал эту систему.

Примерно в мае мы наняли промышленного дизайнера, чтобы он сделал несколько эскизов, и к июню у нас был почти финальный промышленный дизайн. Тогда же мы сделали первые 3D-печатные прототипы, но не могли позволить себе подрядчика по механическому проектированию. Мне пришлось изучить SolidWorks и самому выполнить механическую интеграцию для 3D-прототипа. Поскольку мне нравится учиться новым вещам, этот опыт оказался весьма полезным.

В июле мы подписали заказ на стальную оснастку, а к августу у нас были первые отливки пластика. Сентябрь я провел за доработкой и отладкой дизайна, а октябрь - за дополнительным тестированием, доводкой и запуском массового производства. К ноябрю 2009 года первая партия chumby One находилась на высоте 35 000 футов над Тихим океаном по пути в LAX.



Готовый chumby One.

Chumby One продавался в рознице примерно за половину цены chumby classic и имел больше функций, например FM-радио и поддержку перезаряжаемой литий-ионной батареи - функцию, о которой пользователи мягкого кожаного chumby classic часто просили. Первые реакции на батарею в chumby One стали интересным исследованием потребительской психологии. По какой-то причине, хотя chumby One был меньше и легче, чем chumby classic, и делал ровно те же вещи, люди не чувствовали, что у него должна быть перезаряжаемая батарея. У них не было внутреннего желания взять chumby One и носить его с собой. Это просто показывает, насколько сильно форм-фактор влияет на потребительское восприятие функции!

В любом случае покупателям все эти варианты определенно нравились, но для меня они не были самыми значимыми новыми функциями.

Более открытое для хакинга устройство

Что действительно меня воодушевляло в chumby One, так это то, что его было намного легче взламывать, чем chumby classic. В chumby classic мы использовали припаянный SLC NAND-чип; это было выгодно по стоимости, но сильно усложняло разработку. Разработчикам приходилось напрямую сталкиваться со всеми неприятными особенностями NAND-флеш-памяти, включая плохие блоки и коррекцию ошибок, а если система не загружалась правильно, вариантов восстановления было мало. В chumby One мы решили эти проблемы, храня прошивку на карте microSD.

Если вам случится заполучить chumby One, вы заметите, что не можете заменить microSD-карту снаружи. Мы сделали такой выбор, чтобы не дать обычным пользователям вытащить microSD-карту и потом гадать, почему устройство не загружается. Но если открутить и снять заднюю панель (никаких клеевых пломб, в отличие от chumby classic), microSD-карта легко доступна. Благодаря этому ключевому изменению хакерам не приходилось беспокоиться о том, что они окирпичат свои chumby. Если кто-то испортил прошивку, он мог просто вытащить microSD-карту, смонтировать ее на своей машине разработки и записать новый образ.

Мы также решили сделать microSD-карту chumby One устройством с управляемой NAND-памятью, чтобы можно было напрямую разместить на ней ext3, популярную стандартную файловую систему Linux. Root-раздел все равно монтировался на фабрике только для чтения, чтобы предотвратить случайное повреждение, но управляемая NAND-память превращала повторное монтирование root-раздела в режиме чтения-записи и изменение Linux-системы в тривиальную операцию. Мы сознательно сделали образ ОС так, чтобы он использовал лишь малую часть общей емкости microSD-карты, оставляя хакерам больше гигабайта дополнительного пространства для загрузки собственных приложений и библиотек. Помните, что в то время гигабайт был большим делом.

В аппаратуре то, что хорошо для хакеров, хорошо и для разработчиков. Гибкость, которую мы добавили для хакеров, позволила нам добавить в ОС массу отличных функций. Например, chumby One поддерживал некоторые 3G-модемы и мог раздавать Wi-Fi как точка доступа через эти 3G-модемы. Это фактически превращало устройство в роутер 3G-в-Wi-Fi, что я находил чрезвычайно полезным в поездках, когда нужно было создать Wi-Fi-точку доступа для других устройств. Сначала мы не выводили эту функцию на уровень массового пользователя, но знали, что мы (или кто угодно еще - в конце концов, это был открытый проект) могли бы обернуть ее графическим интерфейсом и сделать более удобной для пользователя, если людям это понравится. А если подключить USB-клавиатуру к chumby One, он автоматически открывал консольную оболочку, в которую можно было печатать. Это удобно в моменты, когда нельзя подключиться по SSH, например при отладке сетевых скриптов.

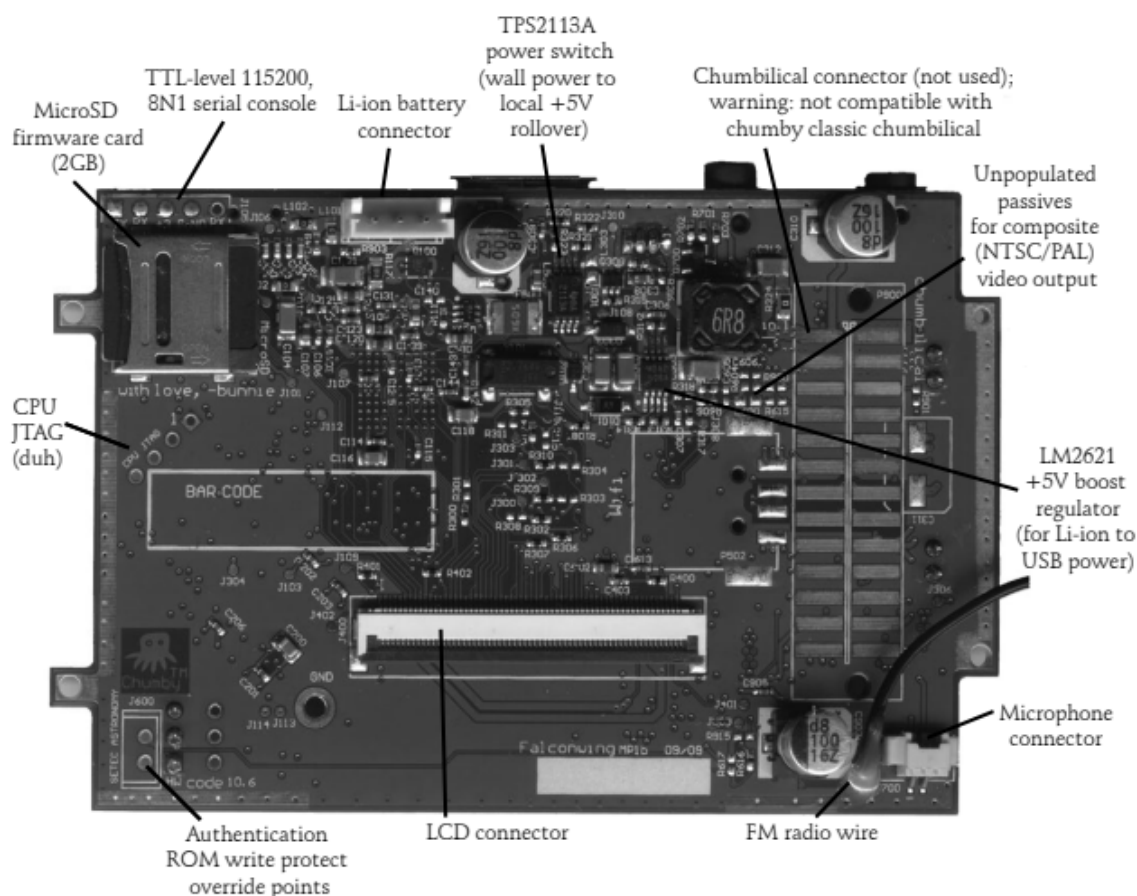
Аппаратура без секретов

Как и с chumby classic, мы сделали дизайн chumby One настолько открытым, насколько возможно. Мы выложили онлайн-схемы, Gerber-файлы и исходный код под GPL. На следующем рисунке можно увидеть предсерийную пилотную плату chumby One. Серийная плата была в основном идентичной, с несколькими небольшими доработками для лучшей совместимости со SMT-машинами, которые мы использовали в Китае.

В частности, обратите внимание на пару тестовых точек на плате с меткой SETEC ASTRONOMY, в нижнем левом углу фотографии обратной стороны материнской платы. Эти точки можно было использовать, чтобы обойти защиту от записи в аутентификационной ROM chumby One и стереть ключи, которые Chumby использовала для аутентификации устройства. Я не могу придумать

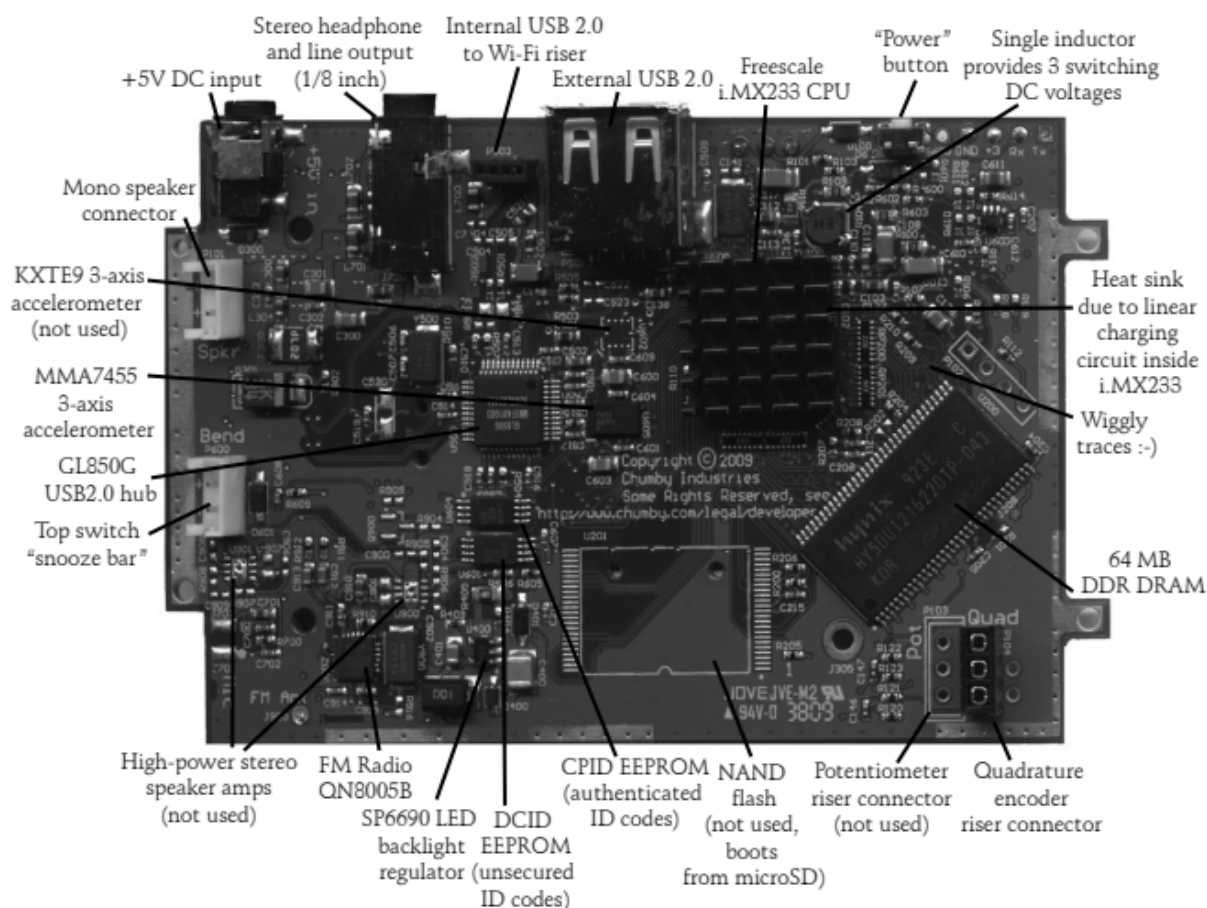
реальной причины делать это, но добавил их из принципа: аппаратура, которой вы владеете, не должна хранить от вас секреты. Если вам не нравится, что на устройстве есть зашифрованные коды доступа, вы должны иметь возможность их уничтожить. В случае chumby One это означало, что у вас больше не было бы кодов для получения виджетов с серверов Chumby, но, эй, это же ваше железо. Когда железо действительно ваше, вы можете лишиться гарантии и делать с ним что хотите. Конечно, мы также опубликовали протокол безопасности, который chumby One использовал для получения виджетов.

Я также спроектировал материнскую плату chumby One с монтажными отверстиями и особенностями, чтобы ее можно было установить обратно в chumby classic. Хотя Chumby никогда не планировала ставить платы chumby One в корпуса chumby classic - сшитая вручную итальянская кожа была просто слишком дорогой, и существовала пара технических проблем интеграции, - я думал, что бесстрашные хакеры оценят возможность сделать это самостоятельно.



The chumby One mainboard (back)

Материнская плата chumby One (обратная сторона). На изображении отмечены: microSD-карта с прошивкой (2 ГБ), последовательная консоль TTL-уровня 115200 8N1, разъем Li-ion-батареи, разъем Chumbilical с входом 5 В, два USB-порта и линии ввода-вывода общего назначения, LCD-разъем с выходом повышающего преобразователя 5 В, проволочная антенна FM-радио, разъем микрофона, точки обхода защиты от записи аутентификационной ROM, повышающий стабилизатор +5 В, выключатель питания и CPU JTAG.



The chumby One mainboard (front)

Материнская плата chumby One (передняя сторона). На изображении отмечены: вход +5 В DC, стереовыход для наушников/линейный выход, внешний USB 2.0, внутренний USB 2.0 к Wi-Fi-переходнику, CPU Freescale i.MX233, кнопка питания, один дроссель для трех импульсных DC-напряжений, радиатор, 64 МБ DDR DRAM, квадратурный энкодер, плата-переходник потенциометра, NAND-флеш не используется - загрузка идет с microSD, CPID EEPROM, DCID EEPROM, стабилизатор LED-подсветки SP6690, FM-радио QN8005B, мощные усилители стереодинамиков, хаб GL850G, MMA7455, KXTE9, разъем монодинамика, верхняя кнопка/snooze-планка и извилистые дорожки.

Я продолжал работать над улучшением линейки chumby несколько лет, но в конце концов захотел больше времени для личных проектов и перерыва от предпринимательства.

Конец Chumby и новые приключения

В апреле 2012 года Chumby в том виде, в каком мир ее знал, подошла к концу. У нас закончились деньги, а у инвесторов закончилось терпение. Я уже незаметно покинул компанию в январе; это был хороший забег, но для меня также пришло время двигаться дальше. Услышав новости, мой хороший друг Phil Torrone из Make: обратился ко мне за интервью, и я был рад согласиться. Вы можете прочитать полное интервью онлайн.[2] но я включил сюда выдержки, которые могут быть полезны, если вы воодушевлены идеей войти в аппаратный бизнес.

Phil: Как вы оказались вовлечены в Chumby? И какова была ваша роль в компании?

bunnie: Изначально я был советником компании, консультантом, которого привлекли, чтобы разобраться с несколькими частями аппаратной стратегии. У нас были еженедельные ужины, где мы говорили о том, каким мог бы быть продукт. В конце концов я настолько воодушевился продуктом, что в свободное время просто собрал начальный прототип материнской платы. Примерно в то же время мой начальник в предыдущей компании сильно меня раздражал (он читал мне лекцию о важности быть на своем стуле каждое утро к 9 утра, полностью игнорируя тот факт, что накануне я работал до полуночи), поэтому я уволился на месте и присоединился к команде основателей Chumby.

Моя роль в компании изначально называлась VP of Hardware, что звучит внушительно. Но когда аппаратная организация состоит ровно из одного человека, вы одновременно и мастер пайки, и уборщик. Теперь, вспоминая это, я понимаю, что команда сильно рискнула со мной. В то время у меня не было опыта управления цепочкой поставок, и я никогда не был в Китае. Они сделали прыжок веры и дали мне возможность со всем разобраться. Я очень ценю, что они дали мне столько свободы учиться прямо в работе.

Phil: Что было лучшей частью создания chumby?

bunnie: В создании chumby было так много замечательных вещей. Думаю, в целом одной из лучших частей было то, что мне пришлось разобраться во всем от концепции до дистрибуции. Это означало, что я смог увидеть каждую часть процесса своими глазами: промышленный дизайн, электронный дизайн, оснастку, цепочку поставок, розницу и обратную логистику. В продукт входит так много всего, и удовлетворить это любопытство о том, как вещи производятся, было прекрасно.

Другая вещь, которую я действительно ценю в создании chumby, - все замечательные люди, с которыми мне довелось работать и познакомиться по пути. Я завел много друзей, и у меня было очень много превосходных mentors.

И наконец, думаю, лучшая часть создания chumby - на самом деле не само создание. Это видеть, как люди им пользуются, как люди наслаждаются устройством и ценят его. Улыбка на лице пользователя - высшая награда.

Phil: Можете рассказать о создании устройства от начала до конца, от идеи до фабрики и полок магазинов?

bunnie: Одна из лучших частей создания устройства от начала до конца состоит в том, что у вас есть совершенно неограниченный набор инструментов для решения текущих проблем. Вы можете решать бизнес-проблемы через разводку платы и наоборот. Например, был вопрос о том, как мы могли бы уникально и гибко брендировать устройства так, чтобы это позволяло менять лицевые панели: защелкнуть панель NFL и получать футбольные счета, защелкнуть панель Bloomberg и получать финансовые новости и так далее. Это тема, которую можно было бы разбирать на десятках совещаний. Но как единственный аппаратный специалист я знал, что встраивание EEPROM стоит всего 0,20 доллара, и пока все остальные обсуждали возможные решения на общем совещании, я открыл инструмент проектирования плат, добавил на плату восьмивыводную EEPROM, поставил подходящий разъем и успел спроектировать все решение к моменту, когда были распределены задачи. Мне фактически потребовалось больше времени, чтобы убедить их, что работа сделана, чем чтобы ее сделать.

Думаю, в итоге я освоил многие навыки, необходимые для создания продукта от начала до конца, потому что передавать требования другим людям очень трудно. Вопрос всегда был в том, будет ли быстрее сделать это самому или объяснить кому-то другому, дождаться, пока он сделает, а затем, возможно, объяснять снова и просить изменить. Это одна из причин, почему я изучил механическое проектирование; промышленный дизайн и пластиковая оснастка часто оказываются самым долгим этапом в расписании многих потребительских продуктов, и способность эффективно общаться с командой инженеров-механиков на их языке была важна, чтобы сделать работу правильно.

Phil: Какие были сложности с розничными продажами?

bunnie: Розница и дистрибуция были самыми трудными задачами. Вот несколько трудностей, с которыми я столкнулся:

Работа с закупщиками розничных сетей. Физические ритейлеры нанимают команды закупщиков, которым поручено монетизировать место на полке. Они думают о продуктах в терминах выручки с полочного места и на самом деле не видят ничего за пределами этого. Это резко высвечивает любые улучшения, которые вы хотите добавить в продукт и которые также повышают стоимость продукта. Закупщики склонны смотреть на ваш продукт как на столько-то граммов пластика и столько-то проводов. Они умножают эти числа на биржевую цену сырья, чтобы сформировать ожидания о том, сколько они заплатят за место на полке. Можно заключить сделки получше, но объяснение закупщику ценности вашего продукта требует больших усилий. К сожалению, текучка среди персонала закупщиков может быть довольно высокой, так что вы можете месяцами заключать сделку, а потом обнаружить, что человек, с которым вы работали, ушел из организации.

Маржа. Каждый участник цепочки поставок протягивает руку: дистрибьютор, продавец и фабрика. Помимо этого, нужно учитывать фонды развития рынка и другие скрытые расходы. В конце концов розничная цена продукта примерно втрое больше вашей стоимости BOM. Это означает, что добавление детали за 0,50 доллара превращается во влияние на розничную цену в 1,50 доллара.

Это усугубляется тем, что цены квантуются в "магические" числа (например, \$19.99, \$49.99 или \$99.99), в которые нужно попасть. Вы просто не ставите MSRP продукта на \$127.45. Если продукт продается в рознице выше \$99, психологически его помещают в одну корзину с продуктами за \$149 или \$199. Когда стоимость BOM вашего продукта приближается к одной из таких точек квантования, вы будете долго раздумывать, стоит ли \$0.50 того, чтобы улучшить, скажем, динамики. Либо это небольшое увеличение стоимости выйдет из вашей собственной маржи, либо вы рискуете подтолкнуть продукт в более высокий ценовой уровень.

Денежный поток. Ритейлеры печально известны тем, что плохо платят вовремя. Вы можете договориться об оплате через 60 дней, но часто вам не платят через 90 или даже 120 дней. Если ваш продукт не распродается так, чтобы ритейлеру пришлось разместить у вас еще один заказ (и тогда у вас появляется некоторый рычаг, чтобы собрать просроченный платеж), вас растянут. Это можно частично смягчить финансовыми инструментами вроде факторингового страхования. Страховые компании продадут страховку на что угодно, включая страхование от того, что ритейлеры не заплатят вовремя или станут неплатежеспособными до того, как смогут вам заплатить.

Обратная логистика и возвраты. Многие ритейлеры предлагают гарантии возврата без лишних вопросов. Это отлично для клиента, но угадайте, кто обслуживает эти возвраты? Ритейлер перекладывает ответственность обратно на предпринимателя! Отчасти поэтому сроки оплаты могут быть очень плохими: ритейлеры удерживают деньги, чтобы вернуть их клиентам при возвратах. После обработки возвратов вам приходится выяснять, как забрать возвращенный материал с их разгрузочной площадки и вернуть его на предприятие, где можно восстановить устройства. Обычно большинство возвращенных устройств не являются дефектными. Они просто не соответствовали ожиданиям клиента, или клиент пожалел о покупке, сделанной импульсивно. В остальном рабочим устройствам обычно не хватает аксессуаров или они косметически повреждены, поэтому для восстановления нужна обширная доработка.

Контракты. Розничные продавцы дадут вам стандартный контракт, полный условий, которые почти при любых обстоятельствах очень сильно благоприятствуют им. Иногда контракты могут подвергать вас обязательствам, которые вы никак не можете надеяться покрыть. Например, я видел формулировку, согласно которой, если связанный контентный сайт был недоступен дольше указанного времени, вы могли бы нести ответственность за неопределенный ущерб репутации бренда розничного продавца, продающего ваши товары. Такие открытые обязательства

неприемлемы, и их исключение из договора может занять месяцы. Другие обременительные условия включают штрафы за поздние отгрузки или штрафы за дефектные устройства. Процесс переговоров по контракту очень отвлекает высшее руководство и может серьезно тормозить организацию.

Phil: Получили ли вы какие-нибудь патенты? Как они работают в мире открытого исходного кода?

bunnie: Да, мне действительно выдали несколько патентов за время моей работы в Chumby. Патенты - очень естественный способ защищать аппаратные идеи. Так же как лицензии FOSS [свободное ПО с открытым исходным кодом] вроде GPL [GNU General Public License] и BSD [Berkeley Software Distribution] опираются на авторское право как источник силы, лицензии открытой аппаратуры могут аналогично опираться на патенты.

Когда мы начинали, не существовало лицензии, которая решала бы патентный вопрос, поэтому chumby создала собственную разновидность лицензии открытого исходного кода. По сути это была автоматическая перекрестная лицензия с пользователями, которые создавали производные работы. Те, кто использовал наш исходный код, получали лицензию на патенты при условии, что любые патенты, выданные на производную работу, также должны были автоматически лицензироваться обратно нам.

В лицензии было еще несколько ограничений, которые не были "по-настоящему" открытыми, например условие, что производная работа должна как минимум давать пользователям возможность запускать сетевые сервисы chumby в конкурирующем продукте (добровольная контрольная точка во время загрузки). Был также пункт "спросите нас, если хотите производить", где говорилось, что производные работы, идущие в массовое производство, должны получить дополнительное разрешение от Chumby. Мы добавили это главным образом, чтобы создать контрольную точку для проверки совместимости с серверами, а также обеспечить соблюдение правил товарных знаков и брендинга. То, что этот пункт был спрятан в лицензии, означало, что лицензию нельзя было назвать лицензией открытого исходного кода, потому что Chumby всегда могла сказать "нет", хотя на практике она никогда этого не делала. Однако ситуация действительно подчеркивает постоянную борьбу в открытой аппаратуре: как решать вопросы товарных знаков и совместимости во все более сложной и разнообразной экосистеме.

Кроме того, права на патенты, которые я создал в Chumby, все переданы инвесторам. Вероятно, они будут проданы тому, кто предложит самую высокую цену, которым вполне может оказаться патентный тролль. Я считал бы такой исход печальным, но это реальность, которую я должен принять. Инвесторы имеют право исследовать все законные пути, чтобы вернуть свои инвестиции. Однако в идеальном мире я бы выкупил права по доступной цене, лицензировал их сообществу открытого исходного кода и попытался создать практический прецедент для того, как обращаться с патентами в сообществе открытого исходного кода.

Phil: Есть ли у вас совет для мейкера, который рассматривает привлечение венчурного финансирования? Есть ли отличия, если он делает открытую аппаратуру?

bunnie: Думаю, венчурное финансирование подходит только для ускорения определенных видов роста. Оно не очень хорошо для ранних исследований и разработки или для бизнесов с медленными, но устойчивыми моделями роста.

Аппаратная модель радикально отличается от модели программного обеспечения. ПО по своей природе масштабируемо. Вы можете получить 100 000 пользователей за ночь. Монетизировать пользовательскую базу в ПО сложнее, но большинство программных проектов начинают с масштабирования, а уже потом беспокоятся о деньгах.

Поскольку в аппаратуре приходится иметь дело с реальными атомами, чтобы добраться до пользователя, масштабируемость ограничена тем темпом, с которым вы можете экономично и надежно собрать физические устройства и отправить их клиенту. С другой стороны, в аппаратуре

есть очень естественный способ монетизации: маржа, которую вы берете с каждой проданной единицы. Деньги приходят раньше и чаще, но темп роста ограничен упрямой реальностью вроде законов физики и доступности сырья и квалифицированного труда для сборки устройств. Заметные исключения из этого правила - концепции вроде Square Reader. Аппаратура Square была спроектирована настолько умно и дешево, что ее стоимость, возможно, была ниже стоимости привлечения клиента другими способами (например, печатной рекламой и почтовыми кампаниями), что делало донгл достаточно дешевым, чтобы просто раздавать его.

Поэтому в аппаратуре сначала спросите вот что: каков ваш канал распространения и насколько трудно доставить продукт конечным пользователям? В конечном счете пропускная способность этого канала и транзакционные издержки ограничивают темп роста вашей идеи. Вам также нужно учитывать послепродажные издержки вроде возвратов и расходов на поддержку клиентов. Вы удивитесь, сколько звонков в поддержку получите от людей, которые забыли включить ваш продукт в розетку.

Если у вас потрясающий канал распространения, крепкая маркетинговая кампания и клиенты стоят в очереди за дверью, возможно, венчурное финансирование вам подходит. Но типичный мейкер начнет с продаж онлайн, возможно, в небольших бутиковых магазинах. Время, нужное, чтобы превратить капитал в выручку, поначалу будет измеряться месяцами, и это жестокий цикл для финансирования через венчурный капитал. Все деньги, застрявшие в цепочке поставок, не добавляют вам никакой ценности, но ради этих денег вы обменяли большую часть собственности в компании.

Обычно я рекомендовал бы мейкеру сначала попытаться финансировать исследования и разработку из собственного кармана или с помощью очень дружелюбного ангельского займа. Когда у вас есть прототип и надежный план производства, разумнее взять долг для финансирования небольших партий сборок, чтобы вы никогда не были чрезмерно растянуты и строили свой рынок шаг за шагом. Каждый раз, когда вы оборачиваете запасы, вы должны возвращаться с большим количеством наличности, которую можно вложить в создание еще большего запаса.

Такой подход навязывает хорошую дисциплину. Он поможет вам сосредоточиться на оптимизации цепочки поставок, чтобы обороты запасов происходили быстрее. Лучшие аппаратные компании оборачивают запасы за считанные дни. Если вы увеличиваете базу капитала на 20 процентов с каждым оборотом запасов, нужно всего четыре оборота, чтобы удвоить деньги: \$100 превращаются в \$120, затем в \$144, затем в \$172, а на четвертом обороте дают \$207. В этом магия сложных процентов.

Если вы можете делать полный оборот запасов раз в восемь недель и сохранять 20-процентный темп роста с каждым оборотом, вы вырастите бизнес более чем на 300 процентов за один год. Конечно, рынки никогда не бывают настолько идеальными и предсказуемыми, но вы можете играть соотношением времени оборота и маржи, доступной для роста бизнеса. Бизнесы с более высокой маржей могут дольше оборачивать запасы и все равно сохранять приемлемый темп роста.

Такое самофинансирование - это много тяжелой работы, но в конце концов вам принадлежит каждый заработанный цент, потому что у вас нет инвесторов. Истории громкого успеха для этой модели не такие большие, как, скажем, Instagram или Google, но если вы делаете все правильно, вы сохраняете контроль, и ваша работа с большей вероятностью окупится. На самом деле многие успешные китайские бизнесы по производству аппаратуры выросли главным образом на самофинансировании.

Phil: Что вы думаете о Kickstarter как источнике финансирования?

bunnie: Я не думаю, что финансировать ранние исследования и разработку через Kickstarter или другие краудфандинговые платформы - хорошая идея, из-за жестких обязательств, которые приходится рано брать перед клиентами. Kickstarter - замечательное явление, но к сбору средств

там тоже нужно относиться осторожно. В некоторой степени Kickstarter - крайний случай "глупых денег". Клиенты верят в вашу идею и приходят рано, а вы должны эту идею воплотить. Краудсорсинг ваших денег также означает, что вы фактически краудсорсили свой совет директоров. Но дорога к разработке продукта никогда не бывает гладкой. В результате деньги Kickstarter могут слишком рано связать вас обязательствами, из которых вы не сможете выйти.

Думаю, Kickstarter может быть лучшим решением, чем венчурный капитал, но использовать его следует только после того, как идея достаточно созрела и вы в первую очередь ищете способ профинансировать производство лучше, чем через венчурные деньги или банковский кредит. На самом деле, если учесть издержки при выводе денег с Kickstarter, банковский кредит под несколько процентов годовых может оказаться выгоднее. Но, конечно, банковский кредит не дает той же видимости, маркетинга и потенциала роста, что краудфандинговая платформа.

Phil: Когда вы консультируете компании, что вы чаще всего советуете основателям?

bunnie: Выпускай продукт или умри! Особенно если вы приняли венчурное финансирование. В тот момент, когда венчурные деньги появляются на вашем балансе, у вас загорается фитиль фиксированной длины. Если фитиль догорит, а вы не создали существенной ценности, взорвется бомба, которая ударит по вашей оценке. Если вы привлекли миллион долларов и планируете потратить его за год, каждый день «стоит» вам \$4 000. Я использую это как критерий ценности при принятии решений: если \$30 платы за ускорение могут сократить критический путь проекта на один день, деньги потрачены хорошо. Это также часть причины, по которой я жил по «китайскому времени», пока chumby был в производстве, хотя находился в Калифорнии. Не спать до 4 или 5 утра каждую ночь, чтобы обмениваться письмами с фабрикой и сокращать критический путь проекта, сэкономило дни в расписании, а вместе с ними - десятки тысяч долларов.

Перед лицом принципа «выпускай продукт или умирай» не стремитесь выпустить идеальный продукт. Выпустить достаточно хороший продукт важнее, чем поздно выпустить отличный, особенно в потребительской электронике или любом похожем сезонном бизнесе. В потребительской электронике до 90 процентов вашего бизнеса может приходиться на четвертый квартал. Если вы пропустите Рождество, у вас не будет выручки следующие три квартала; пропустить Рождество - это как добавить к финансовому плану лишний год сжигания денег. Еще хуже, что в течение этого года ваши конкуренты продолжают улучшаться.

Chumby пострадала именно от этого. Мы впервые показали альфа-версию устройства в августе 2006 года, но пропустили Рождество 2007 года. Наш мягкий подключенный будильник вышел только сразу после Рождества, в феврале 2008 года.

Рассмотрите некоторые мировые события, которые произошли примерно в эти даты: iPhone вышел в июне 2007 года, а мировая экономика рухнула в октябре 2008 года. Уже было достаточно плохо, что нам пришлось пережить почти полный год, с февраля 2008 до Рождества 2008, сжигая венчурные деньги, чтобы оставаться на плаву. Но когда экономика провалилась, исчез и аппетит к подарку в рождественский чулок за \$200. У нас было слишком много запасов, и нам пришлось бороться за выживание.

Если память меня не подводит, мы могли бы отгрузить продукт к Рождеству 2007 года. Он просто не был бы настолько отполированным и ему не хватало бы некоторых функций. Но, возможно, он был бы достаточно хорош. Оглядываясь назад, у iPhone в 2007 году было намного меньше импульса, чем в 2008-м, и мы, вероятно, могли бы распродать большой объем запасов. С другой стороны, возможно, понимание того, что iPhone, его приложения и потрясающий тачскрин сделают подключенный будильник устаревшим, заставило нас усомниться в нашей стратегии и отложить запуск, чтобы усилить функции вроде интеграции потоковой музыки.

В любом случае урок для меня достаточно ясен: выпускай продукт или умирай!

Второй совет, который я дал бы аппаратным компаниям, - целиться выше с ценой. Практически невозможно поднять цену, если вы начали слишком низко, и ничто так не заставляет людей покупать, как распродажа.

Аппаратные стартапы, которые в основном продают онлайн, испытывают соблазн установить цену как можно ниже, чтобы разогнать шум и улучшить начальные продажи. Соблазн продавать ваше устройство стоимостью \$35 за \$49 напрямую онлайн огромен. В конце концов это примерно 28-процентная маржа (если ваш BOM не учитывает мягкие издержки). Это отлично до тех пор, пока вы не исчезли с главной страницы Engadget и ваши продажи не рухнули.

Работа с розничным продавцом может принести больше и более стабильных продаж, но сначала розничный продавец попытается купить ваш продукт у вас за 40-60 процентов от рекомендованной розничной цены. Это означает, что он хотел бы купить продукт за 49 долларов и продать его за 99. Если вы уже продали кучу устройств по 49 долларов, розничный продавец никак не сможет продавать его за 99. Чтобы попасть в розницу, вам пришлось бы продавать ваш 35-долларовый продукт розничному продавцу за 25 долларов, чтобы он мог продавать его по вашей уже установленной цене 49 долларов. Даже если вы добьетесь такого радикального снижения затрат, вы все равно останетесь без прибыли!

Продажа вашего устройства стоимостью \$35 за \$99 поначалу может привлечь меньше клиентов, но начальные маржи были бы впечатляющими, и у вас было бы пространство, чтобы включить ритейлера или проводить собственные распродажи для привлечения большего числа клиентов. Это часть причины, почему MSRP так высоки. Ритейлеры также любят использовать распродажи, чтобы устройства двигались, и устройство за \$99, уцененное до \$69, ощущается как умная покупка. Но при \$69 ритейлер получает только 29-процентную маржу.

Слишком низкое прицеливание по цене фактически лишает вас возможности использовать розницу как возможный канал распространения, и одновременно вы теряете возможность самому проводить распродажи и акции. Акции важны, потому что вирусный маркетинг может поставить вас перед клиентом в лучшем случае один или два раза. Так что когда вы вкладываете сердце и душу в свой продукт, назначайте цену так, будто действительно имеете это в виду.

Phil: Если бы вы могли сделать это заново, как бы вы изменили аппаратуру chumby? Программное обеспечение? То, как chumby был сделан?

bunnie: Ну, как показывает мой предыдущий ответ, я бы гораздо сильнее сосредоточился на своевременной отгрузке, возможно, ценой выбрасывания некоторых функций.

Более неочевидная вещь, которую я усвоил, состоит в том, что аксессуары и упаковка могут потребовать больше времени на разработку, чем сам продукт. Мягкий chumby classic поставлялся с замечательным набором льняных и микрофибровых мешочков и резиновых подвесок. Всего мы разработали больше дюжины подвесок. Был также нестандартный адаптер питания, брендированные ленты, подарочные коробки, брендированная тонкая бумага... Мне даже пришлось итеративно дорабатывать аппаратный дизайн и заказывать пресс-форму для литья, чтобы улучшить способ крепления подвесок к устройству. Я провел как минимум четыре месяца, плотно занимаясь аксессуарами и упаковкой продукта. Наше сообщество поклонников было в восторге от внимания к деталям, и это помогло подстегнуть продажи.

Но задним числом я думаю: могли ли мы поступить лучше, отказавшись от деталей и отгрузив до Рождества? Одно из самых мучительных осознаний, которые должны сделать малые компании, - они не Apple. Apple тратит больше миллиарда долларов в год на оснастку. Инструмент для литья под давлением может стоить около \$40 000 и требовать два-три месяца на изготовление; Apple известна тем, что строит пять или шесть одновременно, а затем выбрасывает все, кроме одного, чтобы оценить несколько подходов к дизайну. Для Apple выбросить \$200 000 на оснастку, чтобы сэкономить два месяца выхода на рынок, - мелочь. Но для стартапа, который привлеч миллион

долларов, это немыслимо. У Apple также сотни сотрудников; у стартапа всего несколько участников, которые делают все. Точность и доводка продуктов Apple имеют огромную стоимость, недостижимую для стартапов.

Я не хочу сказать, что дизайн не важен. Он все еще абсолютно критически важный элемент продукта, а хороший дизайн и внимание к деталям позволяют стартапу брать за продукт больше и отличаться от конкурентов. Apple очень высоко подняла планку дизайна и пользовательского опыта, и пользователи будут судить ваш продукт соответственно. Но важно помнить, что ваш настоящий ориентир для сравнения - другие стартапы, а не Apple. Если ваш главный конкурент - Apple, вам либо нужен миллиард долларов наличными для инвестиций в дизайн продукта, либо нужно переосмыслить стратегию.

Это ведет к другой вещи, которую я, вероятно, изменил бы. Пивот крайне важен для стартапа. Стартап должен быть способен бегать кругами вокруг больших компаний. Культурно Chumby просто оказалось трудно быть достаточно гибкой, чтобы адаптироваться к быстро меняющемуся технологическому ландшафту.

Конечно, задним числом все кажется очевидным. Многое мы могли бы сделать иначе, но когда я думаю обо всех ранних решениях, которые мы приняли, и о том, как мы к ним пришли (резистивный сенсорный экран, отсутствие встроенной батареи, использование Flash как основной платформы), я не вижу, как мы могли бы тогда принять какие-либо иные решения, основанные на фактах.

Но это действительно показывает изъян рассуждений, основанных на фактах. Engineers любят принимать решения на основе доступных данных и надежных моделей будущего. Но я думаю, что настоящие визионеры либо не знают достаточно, либо обладают чистой убежденностью и смелостью, чтобы видеть за пределами фактов и делать дальний бросок. Вероятно, это немного и то, и другое. Риск также означает, что в этом участвует немного удачи.

У меня определенно есть близорукость, вызванная фактами. Мой недавний фокус на операционной эффективности, графиках и управлении рисками подорвал мою способность иметь творческие и дерзкие видения. Я фактически беру годовой перерыв в предпринимательстве, чтобы немного выдохнуть и попытаться заново обнаружить и развить творческие части себя, которые атрофировались за последние пару лет.

Phil: Теперь, когда вы прошли полный цикл венчурно финансируемой компании, которая делает аппаратуру, какие предложения у вас есть по структуре компании - от людей до местоположения и общей организации?

bunnie: Структура действительно зависит от типа продукта, который вы пытаетесь создать. У аппаратуры есть много разных специализаций (например, потребительская, медицинская и промышленная) и рынков (например, дорогие бутиковые продукты, любительские изделия и устройства массового рынка). Во всех них есть хороший бизнес-потенциал, но ваши местоположение, фокус и состав команды нужно настраивать под ваш продукт и то, что дает вам конкурентное преимущество. В Chumby аппаратура была лишь входным билетом для приложений, работающих у вас дома, поэтому это мгновенно стало гонкой на понижение цены. Аппаратная часть компании должна была работать бережливо (помните, у Chumby был один аппаратный инженер и один операционный директор), и с первого дня ей была нужна стратегия, ориентированная на Китай.

В целом, если вы можете выдержать аппаратный стартап через бутстрэппинг, это того стоит. Широкий диапазон аппаратных продуктов можно сначала бутстрэпить, а затем масштабировать через Kickstarter, долговое финансирование или венчурное финансирование. Например, MakerBot разработал и отгрузил свой 3D-принтер полностью на ангельские деньги до закрытия раунда венчурного финансирования. Bre Pettis, один из сооснователей, однажды упомянул, что месяц они жили только на лапше быстрого приготовления.

Любая аппаратная компания, которая прошла фазу идеи и входит в фазу масштабирования, должна быть предельно сосредоточена на операциях и денежном потоке. Поддерживать парадигму производства под заказ критически важно, но трудно: ключевой показатель для любой аппаратной компании, малой или крупной, – как быстро вы можете превращать запасы в деньги. У уравнения две половины. Одна – оптимизировать цепочку поставок и сокращать сроки поставки, чтобы вам не нужно было сидеть на большом запасе, но вы могли быстро удовлетворять новые заказы. Другая – оптимизировать управление наличностью, чтобы вы могли быстро выставлять счета клиентам, одновременно растягивая кредитные линии насколько возможно. Это многомерная оптимизационная задача, от которой без правильных людей может взорваться голова, поэтому ваша команда должна включать сильного операционного директора и кого-то, кто разбирается в полукзотических финансовых инструментах вроде страхования факторинга, обеспеченных кредитных линий и торговых контрактов.

Способность эффективно выходить на Китай на раннем этапе дает вашему стартапу подрывное преимущество (трудно игнорировать преимущество на порядок, которое Китай имеет над Соединенными Штатами в затратах на сборку), но работа с Китаем действительно несет огромные затраты и риски для организации. Это может быть не для всех, особенно с первого дня.

Я переехал в Сингапур, чтобы быть ближе к Китаю, потому что знал: мне никогда не уйти от китайской экосистемы. Китай так крепко удерживает производство аппаратуры, и я думаю, что потребуются десятилетия, прежде чем он потеряет свое преимущество. Такое географическое разнообразие также означает, что любой эффективный аппаратный стартап должен уметь эффективно работать с распределенной командой.

Phil: Что дальше для bunnie? Что вам больше всего хочется делать дальше?

bunnie: Вот это вопрос для меня! Я на самом деле не знаю, что дальше. Как я отметил раньше в интервью, я беру год перерыва, чтобы заниматься тем, что не связано напрямую с предпринимательством. Мои текущие приоритеты: во-первых, получать удовольствие от работы; во-вторых, не потерять слишком много денег; и, в-третьих, сделать что-то хорошее для сообщества через сочетание хактивизма, волонтерской работы и методологии открытого исходного кода. Я надеюсь, что в этот год соберу кусочки своей души, которые потерял по пути, найду несколько новых и заново выучу ценность магии в моей жизни. Я также уделяю немало внимания себе: набираю форму, меняю пищевые привычки и сбрасываю вес. Самое крутое устройство, которым вы когда-либо будете владеть, – ваше тело, и если оно не работает хорошо, надежды нет ни на что другое. Когда я закончу свои бесцельные странствия, надеюсь, у меня будет лучшее представление о том, что дальше!

Пересматривая это интервью для этой книги, я немного усмехнулся про себя. К тому моменту год, который я взял как перерыв, превратился в четыре года. Несколько обеспокоенных знакомых спрашивали меня: "Когда ты уже прекратишь свой кризис среднего возраста и получишь настоящую карьеру?" Но задним числом не возвращаться в корпоративный мир было лучшим решением, которое я когда-либо принимал.

Я действительно живу намного скромнее, чем когда у меня была венчурная и корпоративная поддержка, но у меня намного больше независимости. Это был выбор между «золотыми наручниками» и креслом Аегон или рюкзаком и интересной точкой на горизонте. Я все еще работаю над собиранием кусочков своей души и все еще медленно заново учусь ценностям очарования и удивления. Но по крайней мере у меня есть свобода размышлять о ценностях, отличных от богатства моих инвестировавших акционеров. К счастью, у меня были некоторые успехи в пересмотре пищевых привычек и уровня физической формы; настройка собственного тела была мучительным годом учета калорий, больных мышц и рук в мозолях, но это окупилось сторицей. Моя мать говорила мне, что без здоровья у тебя ничего нет; она абсолютно права. Если у вас нет выносливости работать, трудно превращать возможности в результаты. Если повезет, мое

здоровье продержится, и в будущем у меня будет еще много историй, которыми я смогу с вами поделиться.

Почему лучшие дни открытой аппаратуры еще впереди

Одним из самых критически важных результатов моего года поиска себя стало осознание, что лучшие дни открытого железа все еще впереди. Как я размышлял в интервью с Phil, Chumby не провалилась из-за своей модели открытого железа. В худшем случае эта модель имела мало отношения к потребительской привлекательности продукта; в лучшем - она была хорошей темой для разговора. Нигде в этом интервью я не жаловался на падение продаж в ответ на дешевые клоны, появившиеся на рынке из-за наших либеральных политик открытого исходного кода.

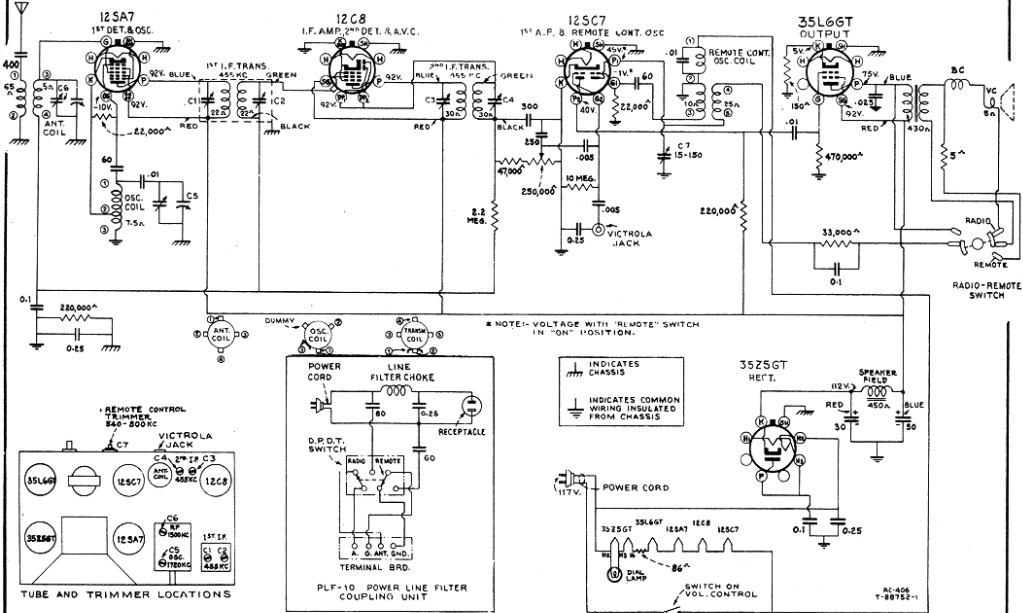
Скорее, одной из наших самых больших проблем была неспособность поспевать за законом Мура. У Chumby как стартапа просто не было ресурсов, чтобы держать темп. На крупную ревизию платформы уходило два-три года, и к этому моменту эта ревизия уже была устаревшей. Моя докторская диссертация[3] была сосредоточена на законе Мура и его влиянии на компьютерную архитектуру. Самые мощные компьютеры - потомки процессора, спроектированного в 1970-х (Intel 8085), производные которого до сих пор используются как мозги мини-печи. Почему? Потому что запуск существующего кода на CPU с обратной совместимостью почти всегда был быстрее, чем перенос старого кода на новую микроархитектуру. С учетом этого факта в диссертации я спроектировал микроархитектуру, которую никто не мог бы реализовать в то время, но которая могла бы быть оптимальной для компьютера, который можно будет построить через 10-15 лет. У небольшой команды исследователей было бы достаточно времени, чтобы разработать инфраструктуру, необходимую для нового компьютера, который будет актуален в день, когда его наконец включат. В конце 1990-х я потратил несколько месяцев, изучая основы закона Мура и пытаясь понять, где он истончается, а где держится крепко. В то время самым сильным ограничением была скорость света, поэтому моя диссертация вращалась вокруг архитектурных приемов для сокращения задержек связи.

В 2011 году, примерно через десятилетие после моего выпуска и как раз около конца Chumby, у меня появилась возможность выступить с визионерским keynote-докладом на Open Hardware Summit. Я решил пересмотреть свои конспекты из колледжа и посмотреть, остался ли у закона Мура еще один десяток лет. Не остался, и это имеет глубокие последствия для будущего открытой аппаратуры. Этот раздел - адаптация записи в блоге, которую я написал в 2011 году, делаясь своими мыслями; к счастью, здесь, в 2016 году, мне пока не пришлось отказываться ни от одного из утверждений, которые я сделал тогда.

Откуда мы пришли: от открытого к закрытому

Открытая аппаратура - нишевая отрасль, и определенные тенденции заставили аппаратную индустрию благоприятствовать крупным закрытым компаниям за счет малых или индивидуальных инноваторов. Однако, глядя на 20-30 лет вперед, я вижу фундаментальный сдвиг тенденций, который может склонить баланс сил в пользу инноваций, а не масштаба.

Как я сказал в предисловии к этой части: в начале аппаратура была открытой. Ранние потребительские электронные продукты, такие как ламповые радиоприемники, часто поставлялись с пользовательскими руководствами, содержащими полные схемы, список запасных частей и инструкции по обслуживанию. В 1980-х компьютеры часто поставлялись со схемами. Например, Apple II поставлялся со справочным руководством, включавшим полную схему материнской платы, - артефакт, который, как я считаю, сильно повлиял на мое решение заняться аппаратурой.



Replacement Parts

Insist on genuine factory-tested parts, which are readily identified and may be purchased from authorized dealers.

STOCK No.	DESCRIPTION	STOCK No.	DESCRIPTION
CHASSIS ASSEMBLIES			
13057	Capacitor—60 mmfd.	32969	Socket—Dial lamp socket
12488	Capacitor—250 mmfd.	14278	Socket—Phonograph socket
12952	Capacitor—300 mmfd.	32537	Socket—Tube socket
30433	Capacitor—400 mmfd.	30585	Spring—Drive cord spring
4838	Capacitor—.005 mfd.		
4837	Capacitor—.01 mfd.	33319	Transformer—First i-f transformer
4870	Capacitor—.025 mfd.		
4839	Capacitor—.1 mfd.	32578	Volume control and power switch
12484	Capacitor—.25 mfd.		
33321	Capacitor—Electrolytic, 2 sections 30 mfd. each	POWER LINE FILTER PLF-10	
32572	Coil—Antenna coil	13057	Capacitor—60 mmfd.
33320	Coil—Duplex oscillator coil	12484	Capacitor—.25 mfd.
32962	Coil—Oscillator coil	33492	Coil—Choke coil
33323	Condenser—Trimmer 20-150 mmfd.	33493	Receptacle—Power receptacle
32968	Condenser—2-gang variable tuning	33491	Switch
32634	Cord—Drive cord		
32946	Drum—Condenser drive drum	SPEAKER ASSEMBLIES (39105-2)	
12409	Lead—Antenna lead	32964	Transformer—Output transformer
33322	Resistor—5 ohms, 5 watts		
14871	Resistor—33 ohms, 1/2 watt	MISCELLANEOUS ASSEMBLIES	
13428	Resistor—150 ohms, 1/2 watt	X-639	Cabinet—Ivory finish—Model 5X51 (net)
13998	Resistor—22,000 ohms, 1/2 watt	X-638	Cabinet—Walnut finish—Model 5X5W (net)
12454	Resistor—33,000 ohms, 1/2 watt	32942	Dial—Glass dial scale
12412	Resistor—47,000 ohms, 1/2 watt	33317	Fastener—Push fastener to hold cabinet back
12264	Resistor—220,000 ohms, 1/2 watt	33306	Knob—Black tuning knob—Model 5X51
12285	Resistor—470,000 ohms, 1/2 watt	32447	Knob—Ivory knob—Model 5X5W
12679	Resistor—2.2 meg., 1/2 watt	32943	Nut—Speed nut to hold dial
13601	Resistor—10 meg., 1/2 watt	31646	Spring—Knob retaining spring
32946	Shaft—Tuning knob shaft and bushing		

Additional Replacement Parts:

Stock No.

- 32946 Drum—Condenser drive drum and indicator
- 11765 Lamp—Dial lamp, Mazda No. 51
- 33324 Switch—"Remote" switch
- 32967 Transformer—Second I-F trans.
- 84569 Speaker—Complete—less transformer

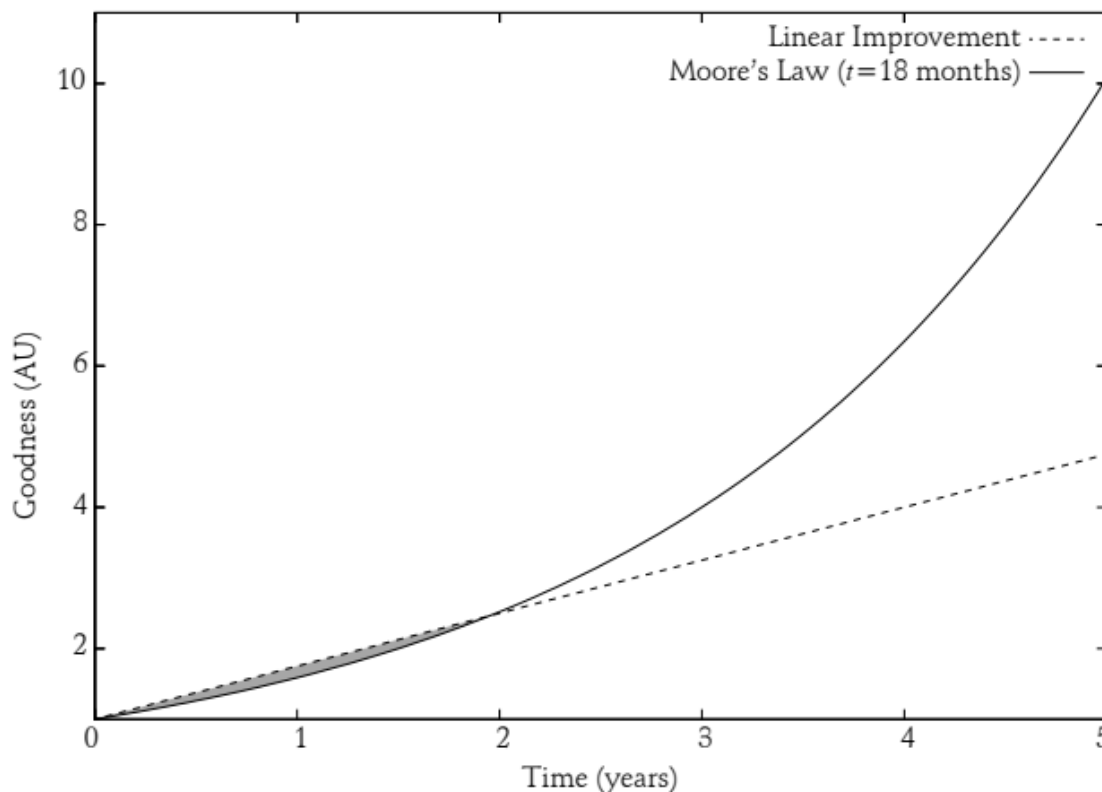
Схема вакуумного радиоприемника.

Но современные пользовательские руководства лишены такой глубины информации. Самая сложная схема, которую я видел в руководстве пользователя Mac Pro, объясняет, как сидеть за компьютером: держать "бедра слегка наклоненными", "плечи расслабленными" и так далее.

Что случилось? Электроника просто стала слишком трудной и сложной? Напротив, улучшать электронику стало слишком легко: темп закона Мура оказался слишком высоким, чтобы малые инноваторы успевали за ним.

Где мы сейчас: «сидеть и ждать» против «изобретать»

Рассмотрим этот снимок закона Мура, который утверждает, что условная "мощность" технологии (выберите практически любую метрику: производительность, плотность транзисторов, цена за квант и т. д.) удваивается каждые 18 месяцев.



Moore's Law: doubling once every 18 months versus linear improvement of 75 percent

Закон Мура: удвоение раз в 18 месяцев против линейного улучшения на 75 процентов в год. Затененный клин между двумя линиями при $t < 2$ года показывает окно возможностей, где линейное улучшение превышает закон Мура.

Этот график необычен тем, что вертикальная ось линейная. Большинство графиков, изображающих закон Мура, используют логарифмическую вертикальную шкалу, которая сглаживает резкий восходящий тренд кривой в гораздо более безобидно выглядящую прямую линию. Затененная область, напротив, показывает линейное улучшение со временем. Она может соответствовать малому инноватору, который с постоянным, некомпандированным, но достойным темпом 75 процентов в год добавляет или улучшает функции на заданной платформе. Крошечное (почти невидимое) пространство между кривыми показывает рыночную возможность малого инноватора в гонке с законом Мура.

Сопоставление этих двух кривых подчеркивает главную проблему малых инноваторов. Долгое время выгоднее было сидеть и ждать, чем внедрять инновации. Если для удвоения производительности системы нужно два года, вам лучше просто подождать и перейти на новое железо через два года. Гонка против закона Мура - сизифов труд.

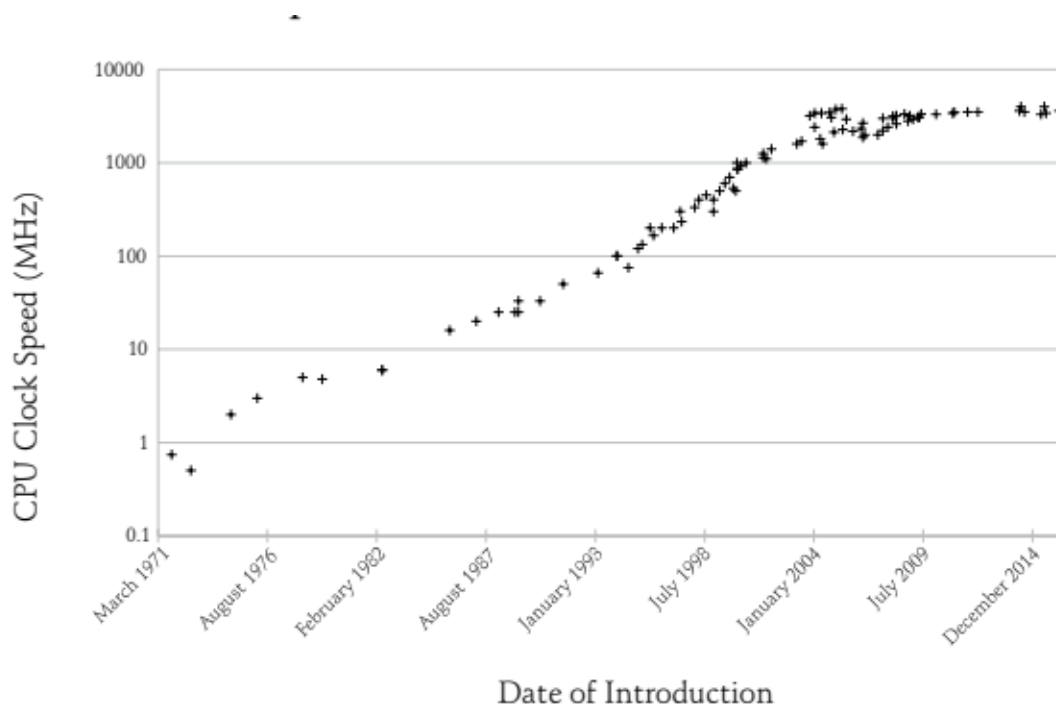
Эта механика экспоненциального роста благоприятствует крупным бизнесам с ресурсами, позволяющими достигать огромного масштаба. Вместо разработки одного продукта за раз конкурентоспособный бизнес должен иметь ресурсы и видение, чтобы разрабатывать три или четыре поколения продуктов одновременно. Достижение глобального рынка в пределах срока жизни одного поколения технологии требует цепочки поставок и канала распространения,

способных делать миллионы единиц в месяц: при продаже 10 000 единиц в месяц достижение "всего" миллиона пользователей, или примерно 1 процента домохозяйств только в Соединенных Штатах, заняло бы восемь лет. И существенно, что небольшой барьер (несколько месяцев времени), созданный закрытием дизайна и принуждением конкурентов заниматься обратным инжинирингом продуктов, может быть преимуществом, особенно на фоне темпа закона Мура.

Таким образом, технологические рынки стали недоступны для малых инноваторов, поскольку отдельным людям трудно поспевать за технологической беговой дорожкой, а большие компании продолжают закрывать свои дизайны, чтобы получить тонкое преимущество над конкурентами. Однако этот тренд меняется.

Куда мы идем: долговечные ноутбуки

Gordon Moore, человек, сформулировавший закон Мура, - один из сооснователей Intel. Закон Мура больше всего известен описанием того, как плотность транзисторов, а следовательно и производительность CPU, будут увеличиваться со временем. Например, рассмотрим этот график тактовой частоты Intel CPU на момент выхода по времени.[4]



CPU clock speed over time. The plateau has held steady since 2014.

Тактовая частота CPU с течением времени. Плато держится с 2014 года.

Обратите внимание на резкое плато, где тактовая частота перестает увеличиваться. В этот момент производители CPU начали использовать многоядерную технологию, чтобы повышать производительность, но это не был выбор. CPU достигли физических пределов, которые не позволяли практически масштабировать тактовую частоту, главным образом связанных с масштабированием мощности и задержек в проводниках. Плотность транзисторов, а значит и число ядер, продолжает увеличиваться со временем, но темп замедляется. Раньше число транзисторов удваивалось раз в 18 месяцев; затем оно замедлилось до удвоения реже, чем раз в 24 месяца. В конце концов масштабирование плотности транзисторов фактически закончится. Абсолютная конечная точка масштабирования транзисторов - предмет споров, но одно исследование[5] указывает, что масштабирование может остановиться на эффективной длине затвора около 5 нм.

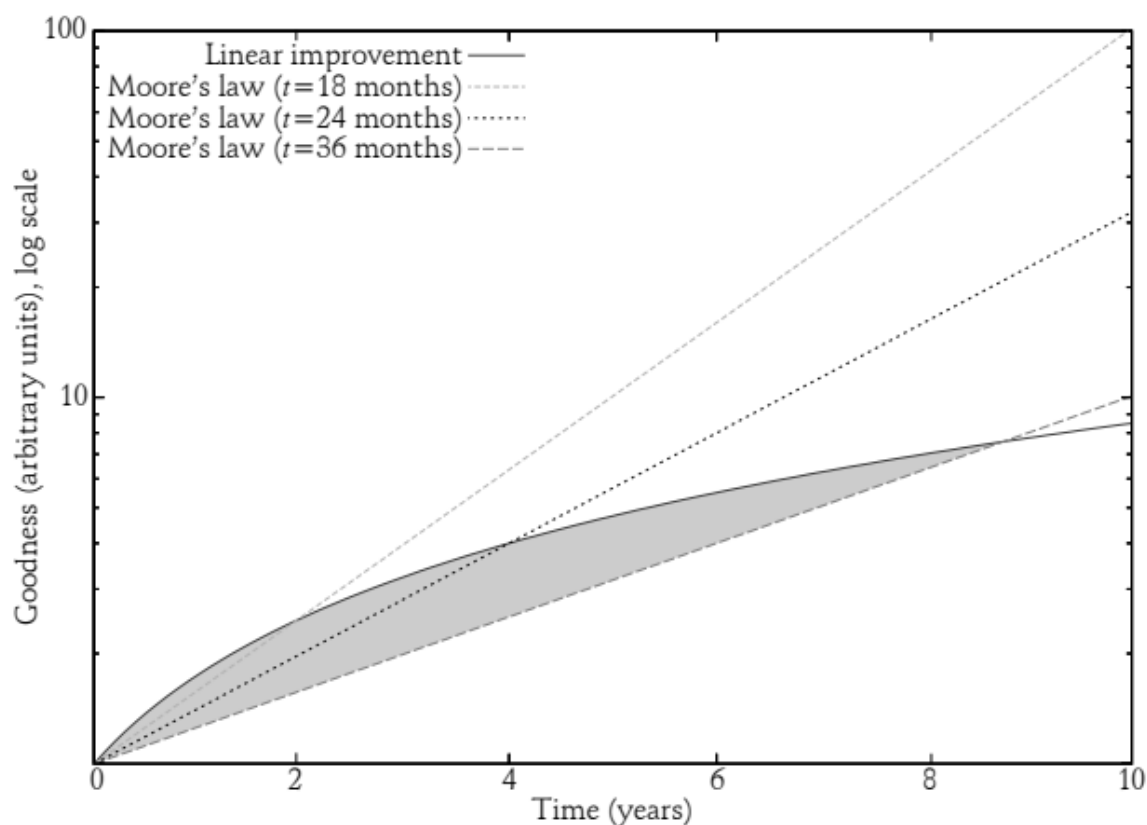
Это примерно расстояние между 10 атомами кремния, так что даже если эта оценка неверна, она не может быть сильно неверной.

Последствия глубоки. Однажды вы не сможете рассчитывать на покупку более быстрого компьютера в следующем году. Ваш телефон не станет меньше или мощнее. А флеш-накопитель, который вы купите в следующем году, будет стоить столько же и хранить столько же битов, сколько тот, который вы купили в этом году. Идея "наследуемого ноутбука" сегодня может звучать нелепо, но когда-нибудь мы можем воспринимать свои компьютеры как дорогие сердцу и полезные семейные реликвии, которые передаем детям как часть нашего наследия.

Возможность для открытой аппаратуры

Этот тренд замедления хорош для малого бизнеса, а также для практик открытого железа. Чтобы понять почему, вернемся к графику закона Мура против линейного улучшения. На этот раз я наложу два новых сценария: удвоение технологии раз в 24 и 36 месяцев.

36 months.



Three different Moore's law scenarios. The shaded sliver between linear

Три разных сценария закона Мура. Затененная полоска между линейным улучшением и сценарием $t=18$ месяцев превращается в большую область возможностей при сценарии $t=36$ месяцев. (Обратите внимание, что вертикальная ось имеет логарифмическую шкалу.)

Область, ограниченная кривой линией и прямой линией внизу, показывает рыночную возможность для линейного улучшения по сравнению с законом Мура. В 36-месячном сценарии у линейного улучшения есть больше восьми лет, прежде чем закон Мура его догонит, а примерно на втором или третьем году появляется точка, где оптимизированное решение явно превосходит закон Мура.

Иными словами, существует настоящее рыночное окно для монетизации инновационных решений в темпе, с которым малые бизнесы могут справиться.

По мере замедления закона Мура появляется также потенциал для большей стандартизации платформ. Создание стандартного планшета или корпуса мобильного телефона со взаимозаменяемыми компонентами может сейчас казаться нелепым, но становится разумным предложением, когда компоненты перестают так сильно уменьшаться и меняться. По мере замедления технологий возникнет сближение между аппаратурой в мобильных телефонах и аппаратурой во встраиваемых CPU-модулях вроде Arduino. Просто посмотрите на Raspberry Pi, представленный в 2012 году. Модели, выпущенные в 2016 году, предлагают четырехъядерный CPU 1.2 ГГц с производительностью, сопоставимой со смартфонами начального уровня того времени.

Создание стабильных, конкурентоспособных по производительности открытых платформ даст малым бизнесам больше возможностей. Конечно, малый бизнес все еще может выбрать закрытость, но в таком случае ему нужно создать вертикальный набор проприетарной инфраструктуры, а размывание фокуса для реализации такого стека может оказаться невыгодным.

В будущем после закона Мура FPGA могут показывать достойные результаты по сравнению со своими жестко реализованными CPU-родственниками, как минимум по двум причинам. Во-первых, гибкая, но регулярная структура FPGA может дать ей более длинную кривую масштабирования, отчасти благодаря способности FPGA перенастраивать схемы вокруг мелкомасштабных колебаний технологических допусков. Во-вторых, дополнительные усилия по оптимизации кода для аппаратного ускорения будут амортизироваться выгоднее, поскольку масштабирование производительности CPU все больше опирается на трудные техники вроде использования параллельных ядер в массовом масштабе. Массивно-многоядерные CPU-архитектуры сильно похожи на крупнозернистые FPGA-архитектуры, предложенные в академических кругах в 1990-х. Выравнивание производительности FPGA и CPU должно сильно облегчить проникновение открытой аппаратуры на глубоком уровне.

Будет подъем ремонтной культуры, поскольку технология станет менее одноразовой и более долговечной. Замена изношенных компьютерных деталей через пять лет после даты покупки не будет казаться такой глупой, когда запасная деталь имеет практически те же характеристики и цену, что старая деталь. Этот подъем ремонтной культуры создаст спрос на схемы и запасные части, что, в свою очередь, облегчит рост открытых экосистем и малого бизнеса.

Лично я с нетерпением жду возвращения ремесленной инженерии, где элегантность, оптимизация и баланс ценятся выше разрастания функций, и где я могу использовать один и тот же инструмент десятилетиями и не выглядеть анахронизмом. Большинство людей смеются, когда слышат, что я держался за Eudora 7 как почтовый клиент до 2012 года, когда перешел на мой нынешний клиент Thunderbird.

Замедление закона Мура уже повлияло на рынки, менее чувствительные к производительности. Рассмотрим подъем Arduino. Ему потребовалось несколько лет, чтобы набрать популярность, при фактически одном и том же железе в основе все это время. К счастью, запросы основного рынка Arduino (физические вычисления, образование и приложения встраиваемого управления) не выросли, позволив платформе оставаться стабильной. Эта стабильность позволила Arduino пустить глубокие корни в процветающем пользовательском сообществе с открытыми и совместимыми стандартами.

При достаточном упорном труде и доле удачи я верю, что экосистемы открытого железа обязательно расцветут. Неизбежное замедление закона Мура может означать проблемы для технологических гигантов, но оно также дает движению открытого железа шанс укорениться и вырастить что-то потенциально очень большое. Чтобы воспользоваться этой возможностью, пионеры открытого железа должны подготовить основу, создавая культуру разрешительных стандартов и обычаев, которая сможет масштабироваться со временем.

Я с нетерпением жду возможности быть частью светлого будущего открытого железа.

Заключительные мысли

Хотя chumby, задуманный в 2006 году, немного опередил свое время, а компания в итоге стала жертвой закона Мура, мои размышления о замедлении темпа закона Мура подтолкнули меня попробовать еще один эксперимент в открытом железе. Следующая глава, о Novena, рассказывает историю моих донкихотских приключений по созданию заказного ноутбука с открытым исходным кодом.

Примечания

[1] Разумеется, я хочу ясно сказать, что я не был единственным человеком за chumby; я работал с целой командой веселых, талантливых людей. Как я упоминал в главе 1, я был всего лишь ведущим аппаратным дизайнером, хотя вещами вокруг ядра Linux я тоже занимался. В то время это было для меня новым, но было очень весело изучать Linux изнутри, от загрузки до остановки!

[2] См. <http://makezine.com/2012/04/30/makes-exclusive-interview-with-andrew-bunnie-huang-the-end-of-chumby-new-adventures/> для полного интервью.

[3] <http://bunniestudios.com/bunnie/phdthesis.pdf>

[4] Данные в основном из https://en.wikipedia.org/wiki/List_of_Intel_microprocessors и https://en.wikipedia.org/wiki/List_of_Intel_Core_i7_microprocessors. Я отслеживаю CPU Intel, потому что исторически они задавали кривую частоты в МГц и поэтому дают наиболее строгую интерпретацию закона Мура.

[5] H. Iwai, "Roadmap for 22nm and Veyond," Microelectronic Engineering 86, no. 7-9 (2009), doi: 10.1016/j.mee.2009.03.129.

Глава 7. Novena: как я строил собственный ноутбук

Был 2012 год, и я был безработным. Мой предыдущий стартап провалился, и я взял год перерыва, чтобы понять, что мне делать дальше. У моего друга hobs (представленного в главе 4) и у меня была традиция, которую мы сохраняем до сих пор: каждую пятницу мы садимся за несколько кружек пива за обедом и болтаем. Во время одного из таких обсуждений "Beer Friday" мы решили построить собственный ноутбук. Я выразил недовольство тем, что меня никогда не нанимали строить продукт, которым я действительно хотел бы пользоваться каждый день. Как инженер-проектировщик вы обычно движимы требованиями рынка, а не собственными эклектичными вкусами. Мы немного поболтали о вещах, которые были бы для нас полезны, и поняли, что благодаря постепенному замедлению закона Мура, возможно, идея построить открытый ноутбук с несколькими странными функциями специально для хакеров не так уж безумна. С этого мы начали любительский проект: построить компьютер только для себя, что-то, чем мы пользовались бы каждый день и что было бы легко расширять и модифицировать, - наш собственный электронный швейцарский нож. Мы дали проекту кодовое имя Novena, по названию станции метро в Сингапуре и латинскому слову "nine".





Дизайн Novena второго поколения, который вышел на Crowd Supply.

Готовая Novena была компьютером с 1.2GHz четырехъядерной ARM-архитектурой Freescale (теперь NXP) i.MX6, тесно связанным с Xilinx FPGA. Она была спроектирована для пользователей, которые хотели модифицировать и расширять свое железо: вся документация и PCB были и остаются открытыми и свободными для скачивания,[1] и мы снабдили ее разнообразными функциями, которые облегчали быстрое прототипирование.

Ноутбук не для слаонервных

Однако по мере того как я говорил о Novena с большим числом людей, я понял, что другим интересно владеть таким ноутбуком, но они, возможно, не хотят (или не знают как) делать собственные печатные платы. В ответ на чрезвычайно положительные отзывы, которые мы получили на публикацию в блоге по этой теме, хобс и я запустили кампанию на Crowd Supply в 2014 году, когда дизайн был стабилен и проверен. Более 1 000 человек поддержали проект; я рад сообщить, что мы выполнили каждое обещание кампании, большинство - в пределах нескольких месяцев от обещанной даты. После закрытия кампании мы решили, что поддержка цепочки поставок для полной конфигурации ноутбука слишком распылит наши ограниченные ресурсы, но мы будем продавать и поддерживать материнскую плату Novena как минимум пять лет с запуска кампании.

Скажем ясно: Novena - машина не для слаонервных. Это проект с открытым исходным кодом, а значит, часть радости (и раздражения) от устройства состоит в том, что оно постоянно улучшается. Возможно, это единственный ноутбук, который когда-либо поставлялся с отверткой. Каждый, кто покупал один из исходных дизайнов, должен был установить батарею и прикрутить LCD-рамку по своему выбору - зеленую или синюю. Динамики поставлялись как набор, чтобы пользователи не были обязаны использовать наш дизайн акустической камеры. Если у кого-то был доступ к 3D-принтеру, он мог сделать и довести собственную акустическую камеру.

Несмотря на все эти DIY-варианты, я не стремился побить какие-либо рекорды низкой цены с Novena. Она была спроектирована как малосерийный, сделанный вручную ноутбук с уникально открытыми компонентами, и стоимость соответствовала дизайну. Мы предложили три уровня:

- Вариант «настольный моноблок» за \$1 195, готовый к использованию с клавиатурой и мышью прямо из коробки, но требующий подключения к сети.
- Вариант «ноутбук» за \$1 995, включавший плату контроллера батареи для мобильных хакеров.
- Уровень «наследуемого ноутбука» за \$5 000, поставлявшийся в великолепном корпусе ручной работы из дерева и алюминия.

В главе 6 я сказал, что по мере замедления закона Мура я предсказываю, что родители будут передавать компьютеры своим детям. Heirloom Novena задумана именно для такого обращения, хотя внутри у нее то же железо, что и у двух других вариантов.

Но эти цены не так уж отличались от цен потребительских ноутбуков высокого класса. Самая большая сложность состояла в том, чтобы понять, как предложить настолько нестандартную и сложную вещь по такой цене и в малых объемах. Мы не стремились возместить затраты на исследования и разработку через кампанию; это невозвратные издержки, поскольку каждый сегодня может свободно скачать исходники и воспользоваться нашим тщательно проверенным дизайном. Наша минимальная цель финансирования в \$250 000 была лишь малой долей того, что обычно требуется, чтобы вернуть инвестиции более чем в миллион долларов, стоящие за разработкой и производством обычного ноутбука; хобс и я справились с этой задачей благодаря сочетанию ноу-хау, уникального дизайна и крепких отношений с нашей цепочкой поставок.

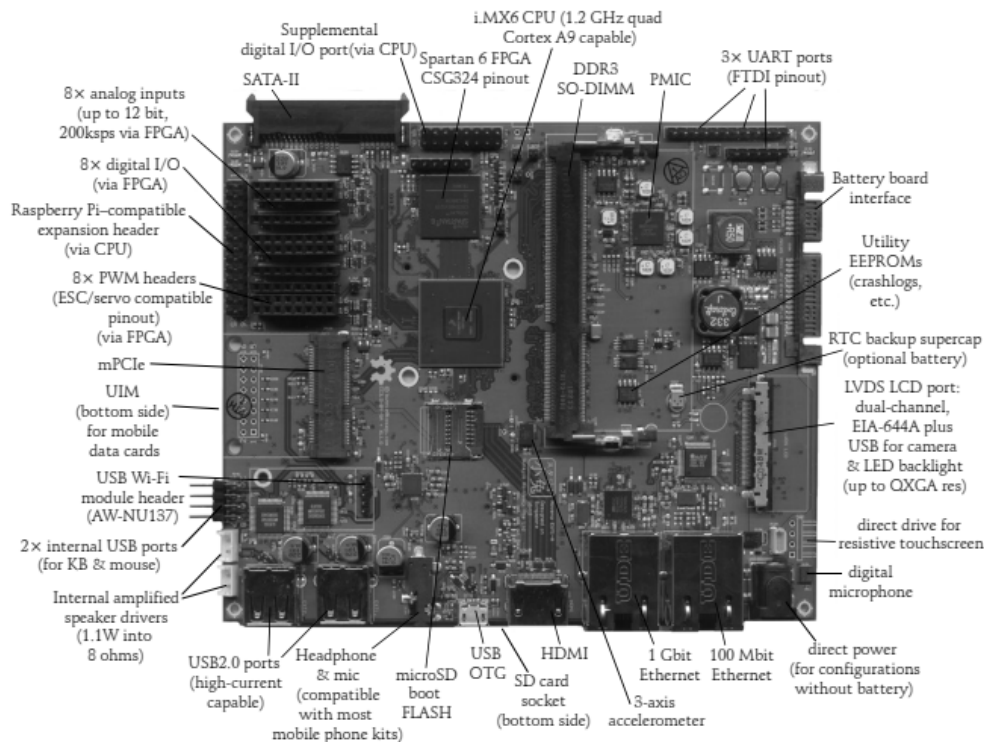
Проектирование ранней Novena

Мы оптимизировали дизайн Novena, чтобы уменьшить требуемое количество дорогой оснастки, но при этом сохранить главную цель: чтобы ее было легко взламывать и модифицировать. Мы потратили полтора года, тщательно разбирая три ревизии РСВА, пока не убедились, что сложный дизайн будет функциональным и пригодным к производству. Мы также оптимизировали некоторые трудные компоненты, такие как LCD и внутренний адаптер DisplayPort, для надежных закупок в малых объемах. Наконец, я потратил несколько месяцев, путешествуя по миру и выстраивая цепочку поставок, которая могла бы поставить этот дизайн (даже в малом объеме) по цене, сравнимой с другими премиальными ноутбуками.

Конечно, вся проектная документация открыта, так что при достаточных навыках и ресурсах вы могли бы собрать Novena с нуля самостоятельно. Я выбрал железо и подкомпоненты так, чтобы сделать это самым практически открытым аппаратным ноутбуком, какой я мог создать с современной технологией. Вы можете скачать технические описания для всех компонентов без NDA, а ключевые периферийные варианты были выбраны так, чтобы можно было собрать полную прошивку из исходников без непрозрачных блоков.

Под капотом

Размеры этой платы примерно 121 mm x 150 mm; она рассчитана так, чтобы удобно помещаться под стандартной ноутбучной клавиатурой (хотя изображение повернуто относительно ориентации установки). Как видно на полных фотографиях ноутбука выше в главе, набор портов находится на правой стороне ноутбука, а не снизу. Толщина платы чуть меньше 14 mm; эту высоту задает толщина Ethernet-разъема. Нижняя часть моего Lenovo T520 чуть меньше 24 mm толщиной, и когда клавиатура и пластик уложены поверх этой платы, основание Novena получается примерно той же толщины.



The earliest Novena motherboard

Самая ранняя материнская плата Novena.

Теперь посмотрим на некоторые возможности материнской платы.

Предварительные возможности

Первая итерация материнской платы Novena использовала CPU Freescale iMX6, для которого есть техническое описание и руководство по программированию без NDA. В следующих списках элементы, отмеченные dagger (†), требуют блоб закрытой прошивки, но система загружается и пригодна к использованию без этого блоба.

Посадочное место CPU, которое мы использовали, могло поддерживать следующие версии iMX6 Quad и DualLite:

- Quad-core Cortex A9 CPU с NEON FPU @ 1.2 GHz
- Vivante GC2000 OpenGL ES2.0 GPU, 200Mtri/s, 1Gpix/s†

Эта версия Novena загружалась с прошивки на microSD. С точки зрения другой внутренней памяти она имела 64-bit DDR3-1066 SO-DIMM, который можно было расширить до 4GB, и жесткий диск SATA-II (3Gbps).

Novena с самого начала также была наполнена внутренними портами и датчиками. Вот основные моменты:

- Разъем Mini PCI-express (mPCIe) для Wi-Fi без бинарных blob-файлов, Bluetooth, мобильных данных и так далее.
- Разъем интерфейса SIM для mPCIe-карт мобильных данных.

- Двухканальный LVDS LCD-разъем с разрешением до QXGA (2 048 x 1 536 px) при 60 Hz и боковым каналом USB 2.0 для камеры со стороны дисплея.
- Контроллер резистивного сенсорного экрана (емкостные сенсорные дисплеи, напротив, обычно идут со встроенным контроллером).
- Внутренние разъемы динамиков 1.1 W, 8 ohm.
- Два внутренних разъема USB2.0 для клавиатуры и мыши или trackpad.
- Цифровой микрофон.
- Трехосевой акселерометр.
- Разъем для опционального Wi-Fi-модуля AW-NU137†.

Мы сделали следующие порты доступными снаружи:

- HDMI
- Устройство чтения SD-карт.
- Разъемы для наушников и микрофона (совместимые с большинством гарнитур мобильных телефонов; они также поддерживали распознавание кнопок на кабеле).
- Два порта USB 2.0 с поддержкой зарядки устройств повышенным током (1.5A).
- Порт 1Gb Ethernet.

И, конечно, поскольку hobs и я делали Novena для себя, мы включили кучу других "веселых" функций, которые, как мы знали, будут отличны для хакеров:

- 100Mb Ethernet (возможность двойного Ethernet позволяет использовать Novena как встроенный пакетный фильтр или маршрутизатор).
- USB On-the-Go (позволяет Novena подменять или фаззить Ethernet, последовательные и другие соединения через USB с помощью gadget-интерфейса к другим USB-хостам).
- Вспомогательная последовательная EEPROM для хранения журналов сбоев и других полезных данных.
- FPGA Spartan-6 в корпусе CSG324 с несколькими интерфейсами к CPU, включая 2 Гбит/с (пиковую) RAM-подобную шину - для майнинга биткоина или чего угодно еще, что вы захотите забросить в FPGA.
- Восемь 12-bit, 200ksps аналоговых входов, управляемых FPGA.
- Восемь цифровых выводов ввода-вывода, управляемых FPGA.
- Восемь PWM-разъемов, управляемых FPGA и совместимых с распиновками любительских ESC и PWM (позволяют прямое сопряжение с различными RC-конфигурациями моторов/сервоприводов и контроллерами квадрокоптеров).
- Разъем расширения, совместимый с Raspberry Pi.
- Тринадцать дополнительных цифровых линий ввода-вывода, управляемых CPU.
- Три внутренних порта UART.

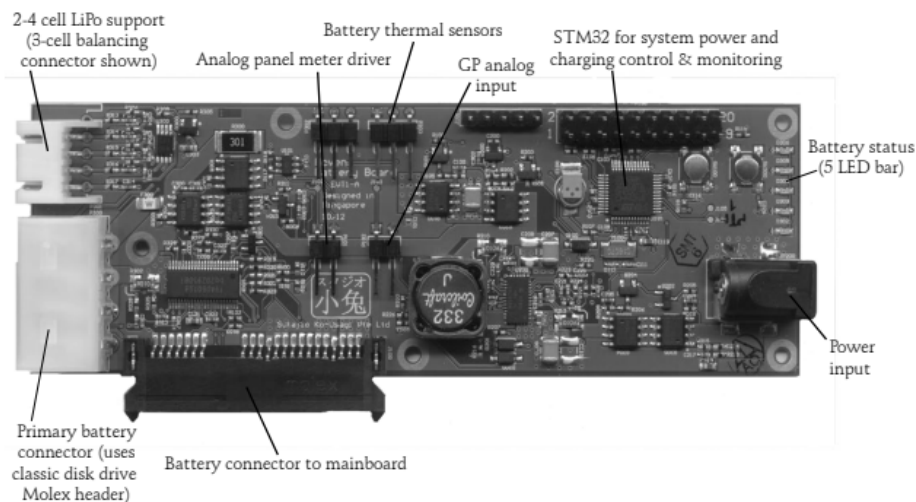
Мы корректировали эти спецификации по мере выхода в производство, сделав самые резкие изменения вокруг разъемов расширения FPGA. Вместо группы разъемов, ориентированных на управление движением, мы решили установить разъем, способный к высоким скоростям передачи данных, который hobs и я затем очень эффективно использовали в будущих проектах с Novena.

Плата батареи

Чтобы дать максимальную гибкость управления питанием, я реализовал функции интерфейса батареи на дочерней плате. Я приспособил дешевый и распространенный разъем в стиле SATA, чтобы передавать питание и управляющие сигналы между основной платой и дочерней платой. Чтобы пользователи случайно не подключили жесткий диск к батарейному порту, я инвертировал пол батарейного SATA-разъема относительно настоящего SATA-II-разъема для накопителей.

Батарейная плата в первой плате Novena была спроектирована для работы с батарейными блоками, используемыми большинством RC-энтузиастов: LiPo-батарейными блоками от 2S1P до 4S1P (то есть от двухъячеечных до четырехъячеечных). RC-батареи хороши, потому что рассчитаны на сверхбыструю зарядку, дешевы и их легко купить. Для разъема батареи со стороны платы я решил использовать разъем Molex, который встречается на классических дисковых накопителях, поскольку они дешевы, распространены и легко собираются простыми инструментами. Я не мог использовать стандартный RC-разъем, потому что подавляющее большинство из них рассчитано на линейное включение, а те немногие, у которых есть крепление на плату, были слишком толстыми или слишком странными для этого применения.

board-side battery plug, I decided to use the Molex connector found on classic disk drives, since they are cheap, common, and easy to assemble with simple tools. I couldn't use a standard RC connector because the vast majority of them are designed for inline use, and the few that have board mounts were too thick or too weird for this application.



The preliminary Novena battery board

Предварительная батарейная плата Novena.

Батарейная плата могла заряжать батареи током свыше 4A; например, зарядка трехъячеечной батареи 45 Wh (4 Ah) занимала около одного часа. Если типичное энергопотребление составляло около 5-6 W, это давало бы семь или восемь часов работы при одном часе зарядки. Конечно, поскольку весь ноутбук был настраиваемым пользователем, типичное энергопотребление было очень трудно оценить. Если пользователь ставил огромный LCD и прожорливый магнитный жесткий диск с кучей периферии, энергопотребление было бы гораздо выше.

hobs предложил еще одну милую функцию, связанную с питанием, которая попала в дизайн. Он подумал, что было бы здорово встроить ретро-аналоговый стрелочный индикатор в упор для ладоней ноутбука, чтобы отображать энергопотребление в реальном времени. Я подумал, что это отличная идея, и заложил это в печатную плату. Конечно, аналоговый индикатор управляется DAC на батарейном микроконтроллере, поэтому его можно было настроить для множества полезных (или не очень полезных) аналоговых показаний, таких как оставшееся время работы, напряжение батареи, температура, время (представленное как аналоговое значение) и так далее.

После того как мы потратили пару месяцев на проверку всех функций (это был длинный список возможностей, который нужно было пройти), мы портировали драйверы и Linux-дистрибутив на плату. Это тоже было немалой задачей, но, к счастью, у меня была умелая помощь hobs, и мы справились.

Корпус

После этого я с нетерпением ждал проектирования корпуса. Для первой ревизии я думал сделать что-то из акрила, вырезанного лазером, отдаленно похожее на планшет, чтобы избежать возни с фрикционной петлей при первой попытке сделать корпус. В итоге я вручную собрал наши первые прототипы корпусов из алюминия и кожи, чтобы проверить ноутбучный сценарий использования Novena. Этот дизайн был грубым; как Cory Doctorow выразился на Boing Boing, он был "gloriously fuggly".[2]





Мне нравится, что мой ноутбук пахнет кожей, когда работает!

Корпус Novena второго поколения, который я показывал выше, более изящный. Первое, что вы, вероятно, заметили в дизайне, - он открывается "неправильным" способом. Эта особенность позволяет использовать Novena как настенный блок, когда экран закрыт. Она также решает крупную проблему, которая была у меня с исходным прототипом-раскладушкой: добраться до железа для взлома было сущим мучением, потому что его блокировала пластина крепления клавиатуры.

В версии, которую мы продавали на Crowd Supply, экран автоматически раскрывается при сдвиге защелки благодаря внутренней газовой пружине. (Novena - не просто открытый ноутбук, это самооткрывающийся ноутбук!) Мы намеренно оставили внутренности обнаженными в этом режиме для легкого доступа, но голые внутренности также ясно показывают, что Novena не для обычных домашних пользователей.

Мы также включили массив монтажных стоек, который назвали Peek array, чтобы облегчить взламываемость. Обычно ноутбуки имеют монтажные точки только для небольшого числа функций, заложенных в исходные чертежи. Но ноутбук для взлома должен вмещать огромное пространство возможной периферии. Вместо того чтобы требовать от пользователей сверлить отверстия или приклеивать вещи внутри корпусов ноутбуков, мы предоставили регулярный массив резьбовых вставок. Это было немного похоже на макетную плату, но для быстрого механического прототипирования. Чтобы помочь определить массив, я консультировался с Nadya Peek, аспиранткой MIT Center for Bits and Atoms и экспертом по цифровому производству, - отсюда название Peek array.

Другая особенность дизайна второго поколения состоит в том, что LCD-рамка сделана из одного простого алюминиевого листа. Это позволяет любому, у кого есть доступ к минимальной мастерской, модифицировать или изготовить собственные рамки; специальная оснастка не требуется. Моя надежда в этом дизайне состояла в том, чтобы сделать добавление ручек и разъемов или замену LCD относительно легкими для хакеров Novena. Чтобы поощрить пользователей экспериментировать, мы отправляли настольную и ноутбучную версии Novena с двумя LCD-рамками, чтобы никому не пришлось беспокоиться, что машина станет непригодной, если одна рамка будет испорчена во время экспериментов.

У большинства ноутбуков клавиатура и мышь прикреплены к корпусу, но Novena имеет отдельные клавиатуру и trackpoint, потому что эта особенность лично была привлекательна для меня. Я всегда хотел дисплей, который можно "повесить" на сиденье перед собой в самолете или автобусе: так намного легче для шеи, и такая компоновка действительно работает лучше, если человек впереди откидывает свое сиденье.

Пока я все еще размышлял, делать ли дизайн-раскладушку или какой-нибудь другой странный внешний дизайн, я также думал попробовать корпус из дерева и латуни. В конце концов, вся идея создания собственного ноутбука состояла в том, чтобы поиграть с новыми идеями! Как упоминалось выше, мы действительно в итоге сделали ограниченный выпуск деревянной Novena, которую называли наследуемым ноутбуком.



Ноутбук Heirloom Novena.

Собственный древесный композит для долговечного ноутбука

Когда основное производство Novena наконец заработало в апреле 2015 года, я провел неделю в Portland, Oregon, работая вместе с Kurt Mottweiler (дизайнером и мастером по дереву, который специализируется на камерах в деревянных корпусах), чтобы закрыть все последние открытые вопросы в устройствах Heirloom. хобс и я, безусловно, гордимся тем, какими получились Heirloom Novena!



Работа с Kurt над ноутбуком Heirloom.

Выращивание Novena

В буквальном смысле Heirloom Novena были "выращены". Деревянные корпуса означали, что важные структурные элементы пришли из деревьев. Сделать каждый ноутбук одинаковым было бы легко, но мы чувствовали, что для заказного продукта куда уместнее сделать каждый ноутбук уникальным, выбирая лучшую древесину и со вкусом подбирая отделку и цвет. В результате нет двух Heirloom-ноутбуков, выглядящих одинаково; каждый уникально красив.



Немного отобранной вручную древесины, ожидающей превращения в корпус Novena.

В Heirloom-ноутбуках также было много науки и инженерии. Для начала Kurt создал уникальный композитный материал, наслаивая пробку, стекловолокно и дерево. Чтобы помочь охарактеризовать новый композит, мы взяли образцы материала в Center for Bits and Atoms, где Nadya Peek и Will Langford оценили характеристики материала. Мы взяли секции древесного композита и провели трехточечный тест на изгиб, используя электромеханическую машину для испытания материалов Instron 4411.

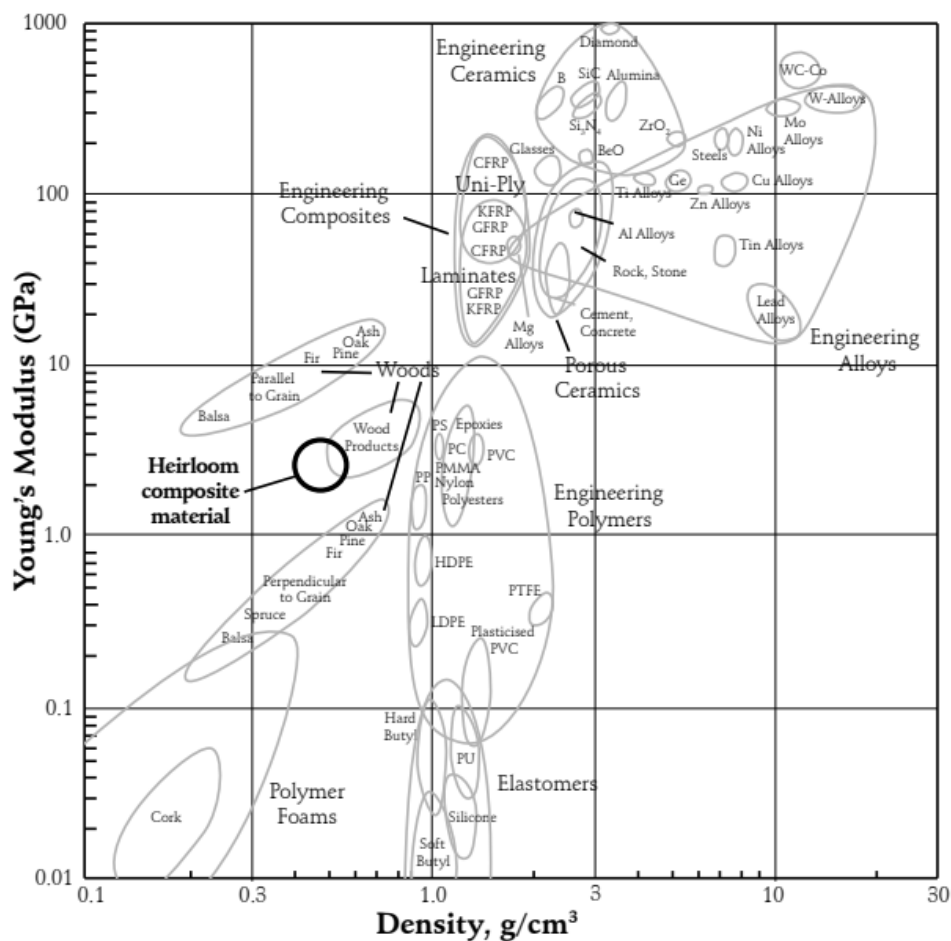


Композитный материал Heirloom, загруженный в испытательную машину.

Детали механической инженерии

Из данных испытаний мы смогли извлечь модуль изгиба (также называемый модулем Юнга) и прочность материала на изгиб. По образованию я не инженер-механик, поэтому термины вроде модуля упругости и удельной прочности в некоторой степени проходят мимо меня. Но Nadya была достаточно добра, чтобы дать мне некоторое понимание. Она указала мне на диаграмму Ashby, на которую, как и на некоторые панели комиксов xkcd, я мог смотреть час и все равно не впитать всю информацию внутри.

terms like *modulus* and *specific strength* kind of go over my head. But Nadya was kind enough to lend me some insight. She pointed me at the Ashby chart, which, as with some xkcd comic panels, I could stare at for an hour and still not absorb all the information contained within.



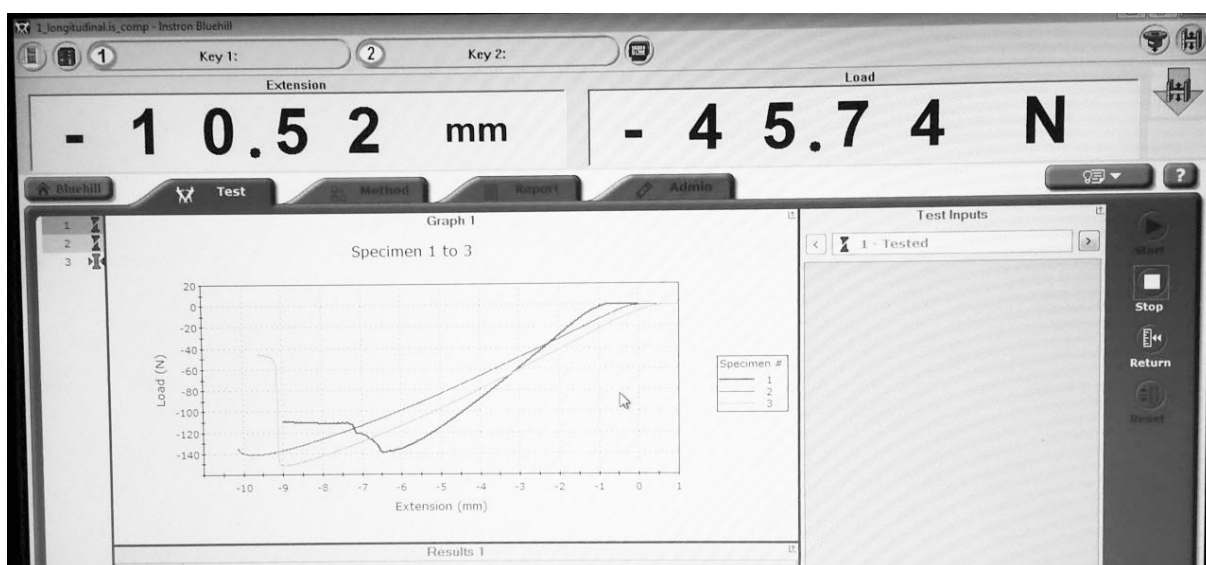
The Ashby chart plots Young's modulus versus density for many materials. The annotated area shows approximately where the Heirloom composite material lands.

Диаграмма Ashby показывает модуль Юнга относительно плотности для многих материалов. Отмеченная область показывает примерно, куда попадает композитный материал Heirloom.

Левый нижний угол диаграммы показывает гибкие, легкие материалы вроде пробки, а правый верхний угол содержит жесткие, тяжелые материалы вроде вольфрама (W). Для корпуса ноутбука мы хотели материал с плотностью пробки, но жесткостью пластика. Древесные продукты занимают

область диаграммы левее пластиков, то есть они менее плотны, но у них есть проблема: они слабы поперек волокон. В зависимости от направления деформации дерево может быть податливым как полиэтилен (материал, используемый для пластиковых пакетов) или жестче поликарбоната (материала, наложенного со стеклом для пуленепробиваемых окон). Композитные материалы хороши тем, что позволяют смешивать характеристики нескольких материалов, чтобы попасть в нужное свойство. В случае ноутбука Heirloom Kurt смешал пробку, стекловолокно и дерево.

Измерения композита Heirloom показывают прочность на изгиб около 33 мегапаскалей и модуль изгиба около 2,2-3,2 гигапаскалей.[3] Плотность материала составляет 0,49 г/см³, то есть примерно половину плотности ABS-пластика, из которого делают кирпичики LEGO. Как показано на диаграмме Ashby, нанесение этих чисел показывает, что композит Heirloom занимает хорошее место левее пластиков и дает компромисс по жесткости в зависимости от направления волокон. И во время испытаний материал не разрушился катастрофически.



Графики зависимости нагрузки от удлинения для композита ноутбука Heirloom, полученные на испытательной машине Instron.

Даже после изгиба за предел пиковой нагрузки композит все еще был в основном целым и продолжал сопротивляться. Этот результат был немного удивительным. Мы ожидали, что материал при отказе переломится надвое, как натуральное дерево. Более того, после того как мы сбросили нагрузку, материал вернулся к исходной форме. Мы согнули композит более чем на 10 mm, но после снятия нагрузки я едва мог сказать, что он проходил испытание. Такая высокая вязкость разрушения и упругость - желательные свойства для корпуса ноутбука.

Конечно, наблюдать, как машина работает с материалом, было весело, но ничто не сравнится с тем, чтобы поддержать его самому. Я все еще помню, как поднял материал, почувствовал, насколько он легкий, хорошо его согнул и удивился его жесткости и прочности.

Изменения в готовом продукте

С момента, когда Novena была успешно профинансирована через краудфандинг, невероятная команда людей работала, чтобы воплотить ее в реальность. С помощью инженеров и менеджеров продукта у нашего производственного партнера AQS корпус Novena перешел от прототипа к пилотному производству всего через четыре месяца после кампании.



Конференц-зал, где мы проводили T1-обзор пластика в Дунгуане, Китай.

Конечно, хobs и я сделали много работы самостоятельно еще до того, как начали краудфандинг, но для создания продукта такой сложности нужно много рук. Мы не смогли бы сделать это без нашей преданной и трудолюбивой команды в AQS. Я уже говорил, что ваша фабрика - ваш партнер, и благодаря отличному партнеру мы смогли сделать это за короткое время.

Конструкция корпуса и проблемы литья под давлением

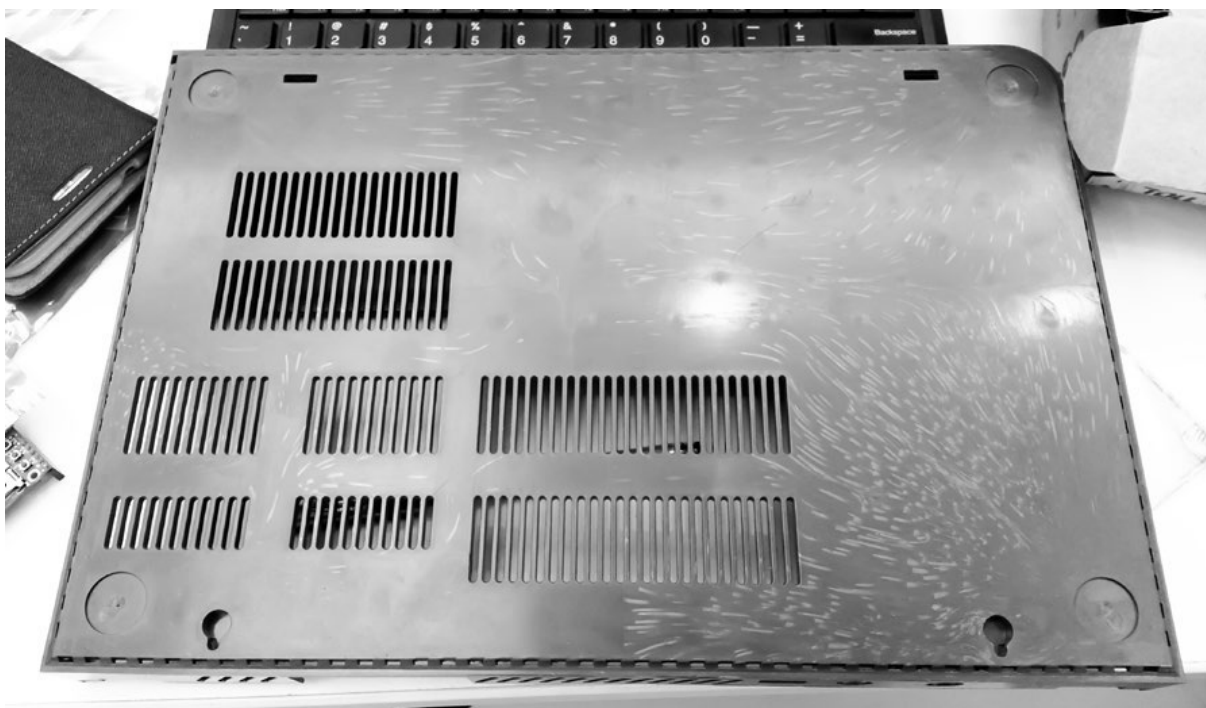
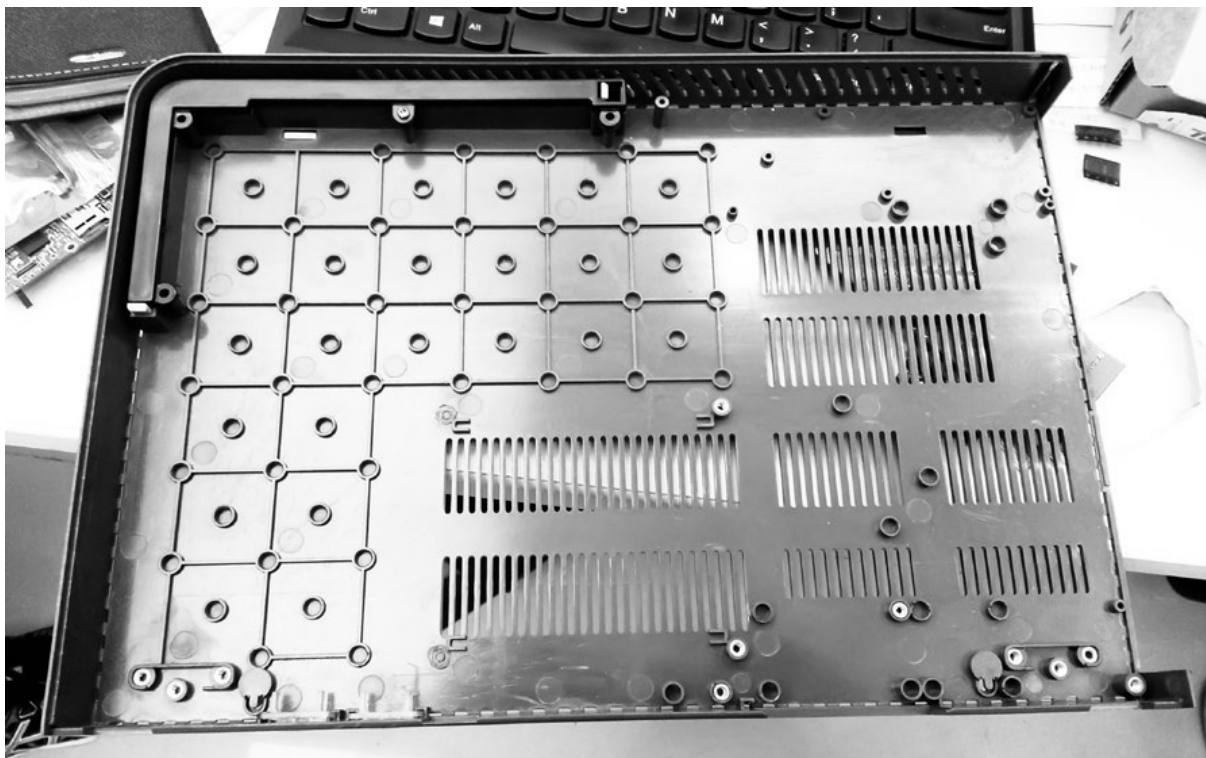
К концу лета 2014 года корпуса Novena, которые мы носили с собой, были сделаны полностью на оборудовании серийного процесса - больше никаких прототипов ручной сборки. Чтобы к этому прийти, мы открыли в общей сложности 10 пресс-форм для литья под давлением; для сравнения, у продукта вроде NeTV или chumby было, возможно, 3 или 4 такие пресс-формы.

Как я кратко описал в главе 1, литье под давлением - процесс, при котором пластик формуется в конечную форму. Горячий жидкий пластик под высоким давлением загоняется в полость из закаленной стали, называемую пресс-формой. Стальная пресс-форма сама по себе шедевр инженерии: это охлаждаемый водой блок весом около тонны, способный выдерживать давления, которые встречаются на дне Mariana Trench, а его внутренние поверхности обработаны с допусками меньше ширины человеческого волоса. Кроме того, пресс-форма содержит часовой механизм движущихся частей с десятками выталкивающих штифтов, ползунов, подъемников и поверхностей разъема, которые плавно расходятся и сходятся снова на протяжении тысяч циклов. Удивительно, что инструменты такой сложности и точности можно изготовить за пару месяцев.

При таком количестве движущихся частей неудивительно, что формы потребовали нескольких итераций доводки, чтобы стать абсолютно идеальными. На жаргоне оснастки эти итерации называются T0, T1, T2 и так далее. Вы в довольно хорошем положении, если можете перейти к серийному производству на T2; к счастью, наши T1-пластики были уже на 99 процентов там, где нужно, что означало легкий путь к серийному производству. У T1 было всего несколько проблем, связанных с следами течения и линиями спая, а также места, где пластик коробился при охлаждении или цеплялся за форму при выталкивании, вызывая деформацию. Это проявлялось

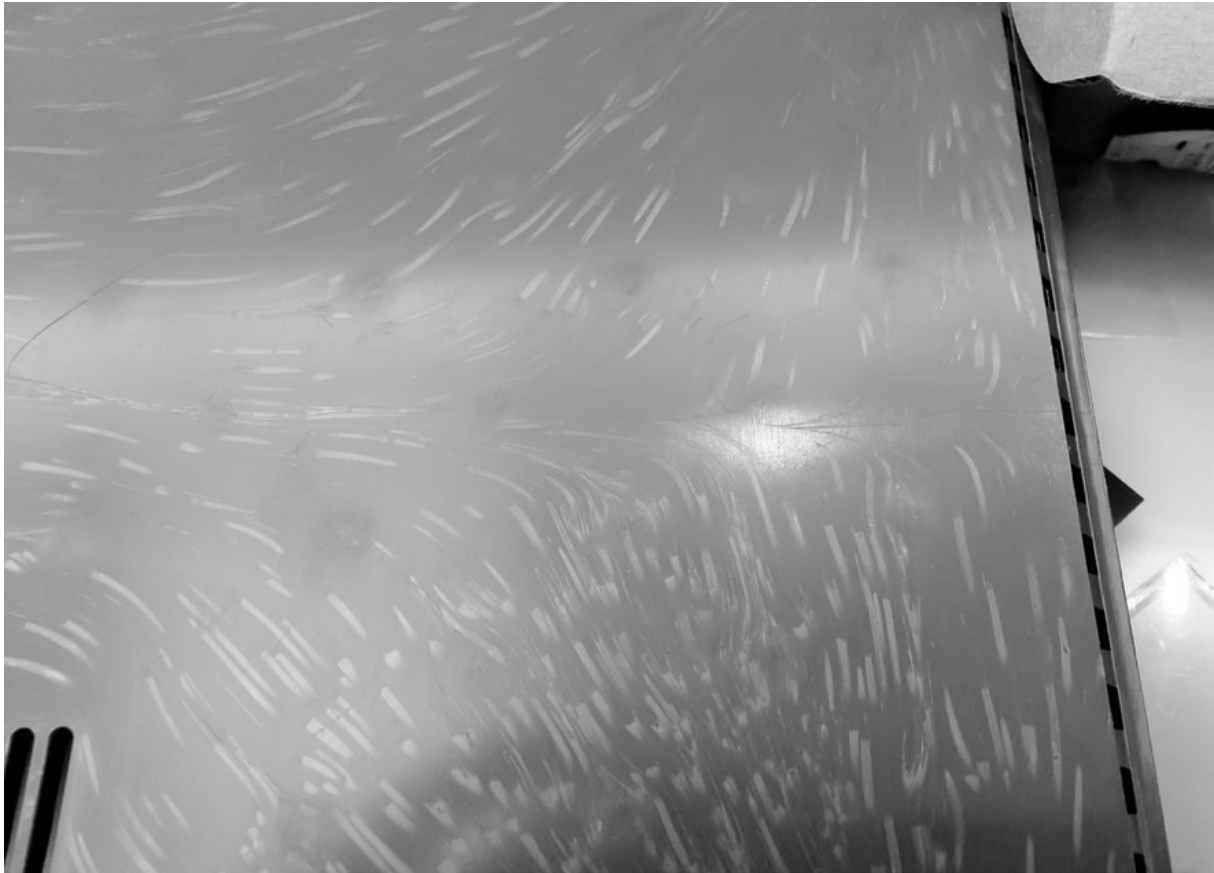
как места, где швы в корпусе были не такими плотными, как мы хотели, и с совсем небольшой настройкой мы были готовы к производству.

Большинство людей видели только продукты готовой оснастки, поэтому я покажу, как выглядит довольно типичный T0 (первая попытка) shot, особенно для большой и сложной формы вроде нижней части корпуса Novena. Такие пробные отливки обычно делают из отходов смолы светлых цветов, которые подчеркивают дефекты. Здесь мы использовали серый пластик, чтобы упростить доводку формы, но финальные устройства имели черные основания.



Несколько T0-отливок основания корпуса Novena. Регулярный массив кругов слева на верхней фотографии формирует основу Reek array. Чтобы сделать массив, резьбовые латунные вставки были вплавлены в круглые стойки после литья под давлением.

В этом куске пластика происходит много всего. Давайте приблизим некоторые артефакты.



Визуальный guide по деформациям в основании корпуса T0.

Круги выделяют набор утяжин, которые появляются, когда на противоположной стороне пластика есть особенно тонкая или толстая особенность. Эти области остывают быстрее или медленнее, чем основная масса пластика, из-за чего они слегка морщатся и создают что-то вроде тени. Утяжины особенно заметны на деталях с зеркальной отделкой. В этом случае утяжины появились потому, что пластик под стойками гаек массива Реек был гораздо тоньше окружающего пластика. Чтобы исправить проблему, мы немного утолщили эту область, уменьшив общий внутренний зазор корпуса на 0.8 мм. Это было возможно, потому что, к счастью, я спроектировал корпус с небольшим дополнительным запасом по зазору.

Прямая стрелка указывает на линию спая. Это область, где поток пластика встречается внутри формы. Когда пластик впрыскивается в полость, он стремится течь из одного или нескольких литников, и там, где расплавленный пластик встречается сам с собой, образуется тонкий шрам. Линии спая часто расположены в точках симметрии между литниками, через которые впрыскивается пластик. В этой форме было четыре литника, расположенных под местами, куда идут резиновые ножки. Литники считаются косметически непривлекательными, поэтому мы разместили их стратегически, чтобы скрыть их расположение.

Белые перистые артефакты, указанные изогнутой стрелкой, - это следы течения. Эти полосы появились потому, что пластик немного слишком быстро остывал внутри формы. Часто проблему можно исправить, настраивая давление впрыска, длительность цикла и температуру. Лучше всего использовать пробные отливки на литьевой машине, чтобы делать эти настройки. Можно менять по одному параметру за раз, отливка за отливкой, пока не найдете оптимальную скорость охлаждения. Этот процесс иногда может занять сотни отливок, создавая как побочный продукт небольшую гору пластикового брака.

Большинство этих грубых дефектов были исправлены к Т1, и к этому моменту пластик выглядел намного ближе к серийному качеству. Мы также смогли начать использовать черный пластик, который обычно скрывает дефекты.

Конечно, все еще оставалось несколько вопросов по подгонке и отделке. Но несмотря на них, корпус ощущался гораздо более прочным, чем прототипы, а механизм газового поршня наконец стал стабильным и действительно плавным.



Основание корпуса Т1 при первоначальном тестировании после переноса рабочего железа в пластик.

Изменения передней рамки

Передняя рамка корпуса Novena (не путать с алюминиевой LCD-рамкой) прошла через некоторые изменения после кампании. Когда мы закрыли финансирование, у нее было два USB-порта, смотрящих наружу, и один переключатель. Novena отгружалась с двумя переключателями, одним USB-портом наружу и одним USB-портом внутрь.

Один переключатель отвечает за питание: он идет напрямую к плате питания и может использоваться для включения и выключения системы даже когда основная плата полностью обесточена. Другой переключатель подключен как пользовательское нажатие клавиши, чтобы облегчить Bluetooth-сопряжение для упрямых клавиатур. Некоторым клавиатурам может потребоваться до полуминуты, чтобы пройти что-то (предположительно связанное с безопасностью), прежде чем подключиться. Есть хаки для обхода этого, но пришлось бы запускать скрипт на хосте. Наша идея состояла в том, что нажатием этой кнопки пользователи могли бы запускать удобный скрипт, чтобы обойти полную нелепость Bluetooth. Этот переключатель также служит кнопкой пробуждения, когда система приостановлена.

Что касается USB-портов, дизайн все еще имел четыре в общей сложности, но конфигурация стала такой:

- Два порта справа, способные отдавать повышенный ток.
- Один порт спереди, способный отдавать стандартный ток.
- Один порт, обращенный к Peek array и способный отдавать стандартный ток.

Иными словами, мы развернули один USB-порт внутрь машины. Поскольку половина удовольствия от Novena - это модификация железа, я решил, что USB-порт внутри будет как минимум так же полезен, как один снаружи.

Для пользователей, которые не будут делать аппаратные модификации, внутренний USB-порт также был бы хорошим местом для подключения маленьких донглов, которые обычно остаются подключенными, например радиоприемопередатчика для клавиатуры. Сначала подключить донгл немного неудобно, но удерживать радиодонгл обращенным внутрь помогает защитить его от повреждений, когда вы бросаете ноутбук в дорожную сумку.

Самодельные динамики

Мы пробовали несколько вариантов динамиков для Novena. Главная идея дизайна состояла в том, чтобы поощрить каждого пользователя выбрать собственный динамик. Некоторые люди действительно слушают музыку на ноутбуке в поездках, но другие полагаются на динамик только для звуков уведомлений и предпочитают использовать наушники для медиа. Физика диктует, что высококачественный звук требует определенного объема и массы. Мы хотели, чтобы пользователи с более расслабленными требованиями к качеству могли вернуть себе пространство и вес, которые требуются более хорошим динамикам.

Kurt Mottweiler выбрал хороший, но очень компактный готовый динамик PUI Audio ASE06008MR-LW150-R для Heirloom. Когда мы обнаружили, что тот же динамик хорошо помещается в Peek array стандартной Novena и имеет приемлемую верность звучания, особенно для своего размера, мы приняли его как стандартное аудиопредложение. Но мы отправляли его с монтажным набором для легкого снятия, чтобы пользователи, которым могло понадобиться вернуть это пространство (или которые хотели поставить более крупные динамики), могли сделать это легко.

Материнская плата PVT2

Материнская плата Novena прошла небольшую ревизию перед массовым производством.

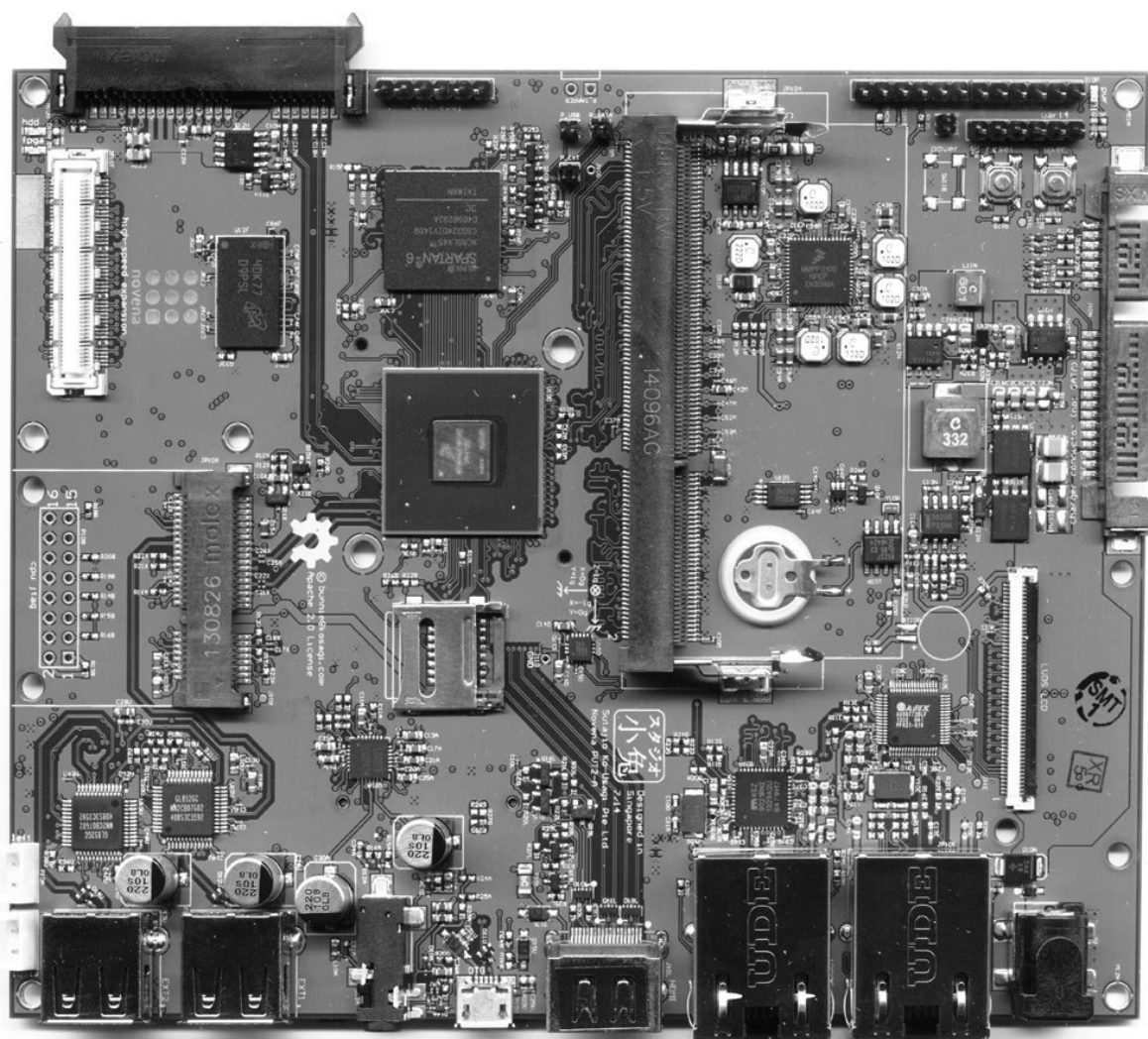
Четвертая и финальная ревизия материнской платы была известна как версия "PVT2".

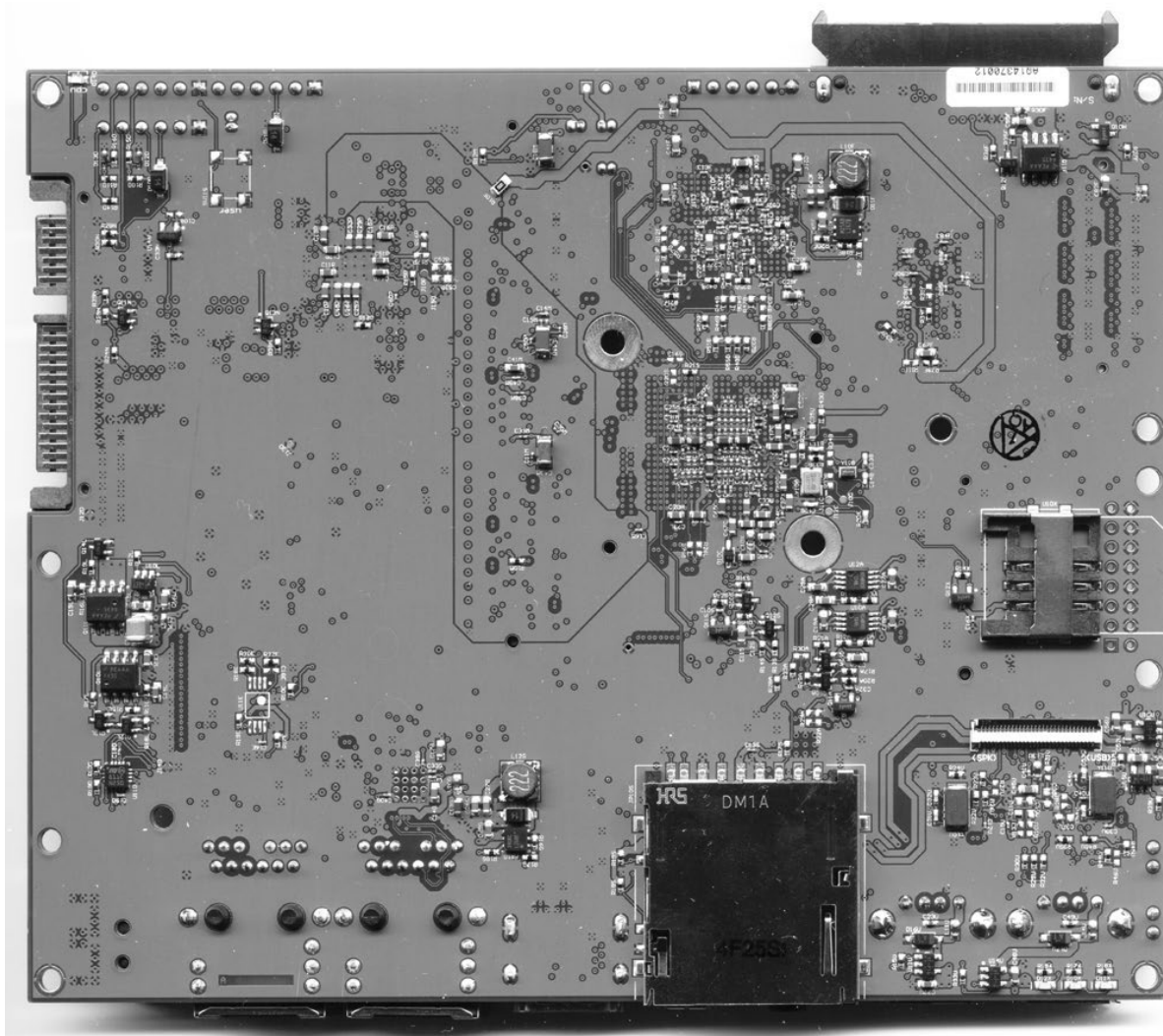
Большинство изменений было сосредоточено на замене или обновлении компонентов, которые рисковали приблизиться к концу жизненного цикла. Два самых значимых конструктивных добавления - внутренний разъем для гибкого печатного шлейфа (FPC) для соединения с кластером передней рамки и отдельный аппаратный модуль часов реального времени (RTC).

Мы добавили внутренний FPC-разъем, чтобы улучшить разводку сигналов от материнской платы к кластеру передней рамки. Нам нужно было провести два USB-порта плюс россыпь GPIO и питание к передней рамке, а исходная схема соединения требовала нескольких кабелей. Обновленный дизайн свел это к одному FPC, чтобы упростить дизайн и повысить надежность.

Мы включили отдельный аппаратный модуль RTC, потому что встроенный RTC i.MX6 работал плохо. CPU просто имел более высокую утечку на RTC, чем указано в техническом описании, и время жизни RTC при выключенной системе измерялось, в лучшем случае, минутами. Мы решили, что продолжать разработку со встроенным RTC несет слишком большой риск, и выбрали внешний отдельный модуль RTC, который, как мы знали, работал. Чтобы повысить совместимость с другими платформами i.MX6, мы выбрали тот же модуль, который использовался Solid-Run Hummingboard, NXP PCF8523T/1.

Также важно отметить, что мы полностью переработали разъем расширения FPGA на второй ревизии материнской платы. Версия материнской платы, показанная в начале этой главы, содержала кластер разъемов, оптимизированных для приложений управления движением. Мы решили, что наша материнская плата слишком велика, чтобы кто-то поместил ее внутрь квадрокоптера, и, возможно, FPGA найдет больше применения как устройство высокоскоростного сбора и обработки данных. Чтобы включить эту функциональность, мы дали FPGA выделенную 256 МБ памяти DDR3 и вывели высокоскоростные дифференциальные сигналы на разъем, способный передавать сигналы со скоростями выше гигабита в секунду. Пользователи все еще могли использовать FPGA для приложений управления движением, но им нужно было подключить простую breakout-плату (вроде GPBB, о которой я расскажу дальше), чтобы развести наши сигналы к форматам разъемов, обычно используемым системами управления движением.

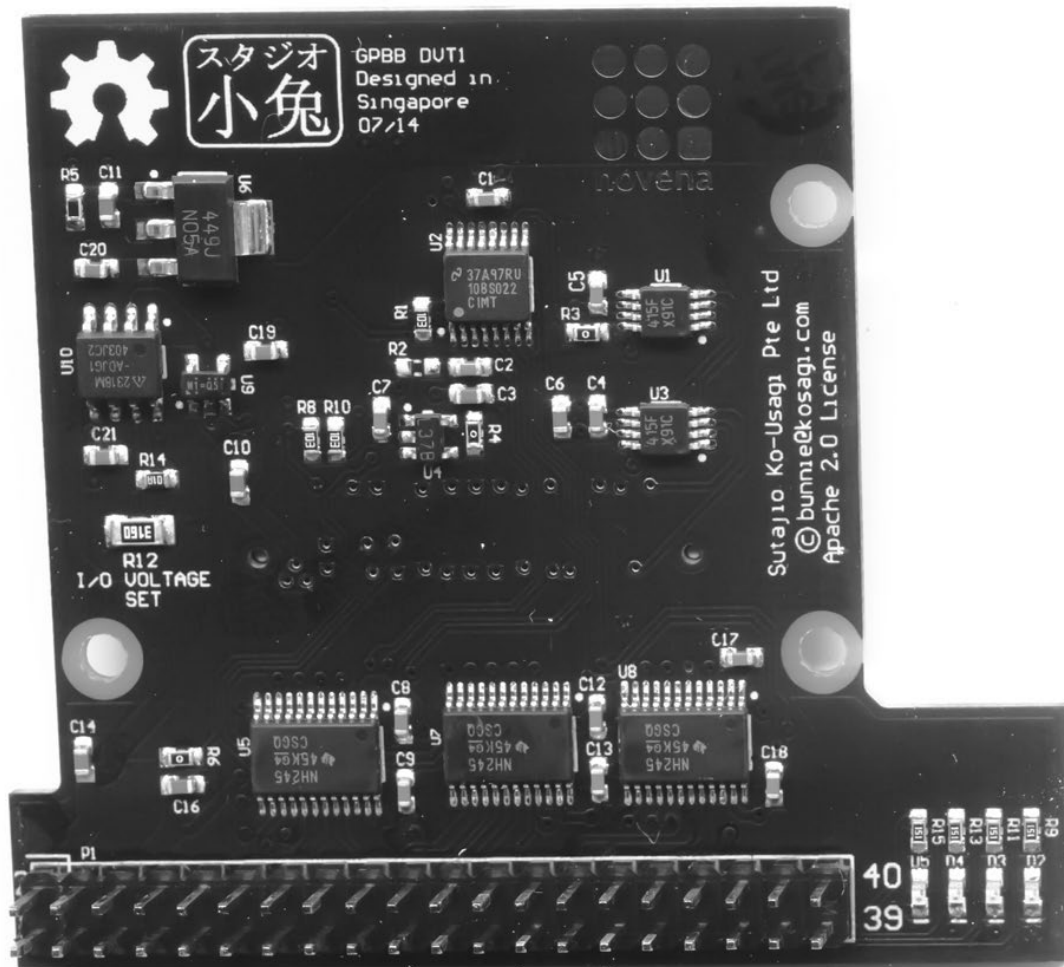




Обновленная материнская плата Novena.

Отладочная плата для начинающих

Одной из наград, которую каждый спонсор получил как благодарность за поддержку нашей кампании, была breakout-плата, которую мы называли GPBB, или General-Purpose Breakout Board. Перепроектирование разъема расширения FPGA на Novena под высокоскоростные приложения также сделало начало работы с устройством гораздо труднее для начинающих хакеров. Из-за ограничений физики высокоскоростные разъемы обычно имеют очень плотное расположение выводов, недружелюбное к начинающим. Мы спроектировали GPBB, чтобы помочь начинающим пользователям работать с FPGA. GPBB преобразует плотный высокоскоростной сигнальный разъем на FPGA в удобный для начинающих 40-контактный разъем с шагом 0,1 дюйма и вдобавок включает несколько светодиодов и аналоговых преобразователей данных.



Финальная серийная GPBB.

Одна растущая трудность для начинающих - тот факт, что закон Мура продолжает снижать допустимый диапазон напряжений цифровых I/O. Новые поколения транзисторов работают при более низких напряжениях, что делает их несовместимыми с почтенными стандартами +5 V, которые использует большинство проектов начального уровня. Например, наша FPGA могла обрабатывать сигналы только до +3.3 V. В результате мы встроили в GPBB преобразователи уровней, которые могли безопасно принимать +5 V и снижать их до уровней +3.3 V, принимаемых FPGA.

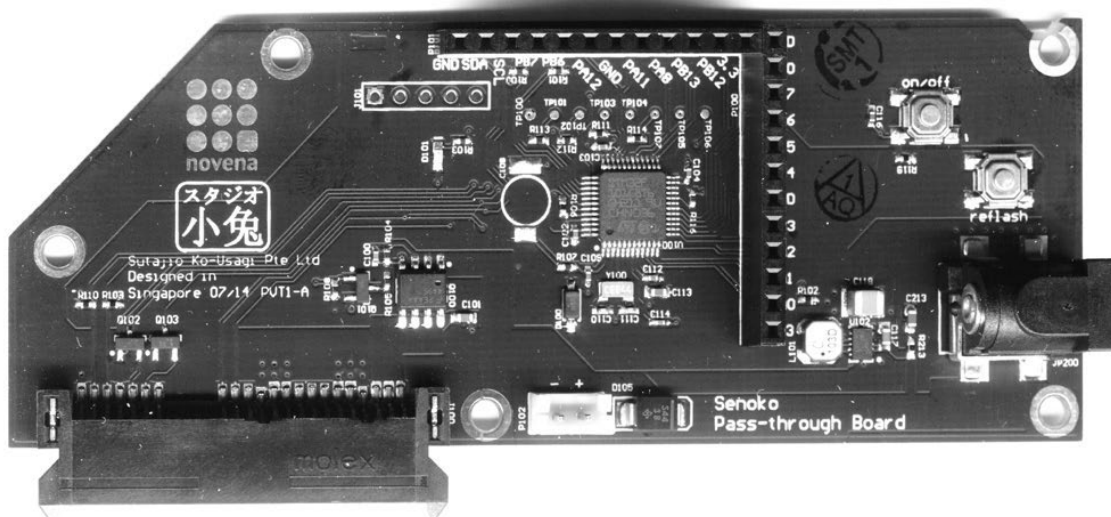
Финальная версия GPBB включала доработку, позволявшую пользователям регулировать I/O voltage вместо фиксации на +5 V. Мы предоставили программную настройку, позволяющую пользователям выбирать, будут ли внешние I/O GPBB по умолчанию 5 V или 3.3 V, и спроектировали плату так, чтобы пользователи могли снизить напряжение до 2.5 V или 1.8 V заменой одного резистора (R12). Я подписал этот резистор "I/O VOLTAGE SET" и сделал его деталью 1206, чтобы новички в пайке могли внести изменение самостоятельно.

Плата сквозного питания для настольной Novena

Уровень "настольный моноблок" изначально включал только настольный корпус, материнскую плату Novena и breakout-плату передней панели. Но эта конфигурация делала управление питанием неудобным, поскольку я проектировал общую систему управления питанием корпуса, предполагая, что будет вспомогательный микроконтроллер, управляющий главным выключателем.

Сложность - дьявол, и заставить программное обеспечение работать даже для единственной конфигурации было уже достаточно трудно. В итоге мы сочли более дешевым добавить новую железку в систему управления питанием настольной версии, чем иметь дело с несколькими конфигурациями кода.

Поэтому настольные системы отгружались с платой сквозного питания. Это была простая сборка PCB, содержащая только контроллер STM32 и силовой переключатель полной батарейной платы. Это позволило нам использовать согласованную общую архитектуру управления питанием и в настольных, и в ноутбук-системах.



Плата сквозного питания настольной версии.

Этот подход был похож на попытку прихлопнуть муху кувалдой - но кувалда стоила столько же, сколько мухобойка. К тому же неудобно таскать с собой и мухобойку, и кувалду. Так что да, мы использовали 32-битный ARM CPU, чтобы прочитать состояние кнопки и переключить GPIO, и да, под всем этим работала полноценная многопоточная операционная система реального времени (ChibiOS).

Это действительно казалось немного глупым. Поэтому мы вывели несколько неиспользуемых GPIO-контактов, сделав Novena еще более пригодной для хакинга. Надеюсь, какой-нибудь изобретательный пользователь найдет применение всей этой неиспользованной мощности!

Проблемы нестандартного батарейного блока

Батарейный блок для Novena определенно был непредсказуемым элементом в стеке проекта. Создание Novena было первым случаем, когда хобс или я сделали систему с батареей такой высокой емкости, и разобраться со всеми правилами доставки, чтобы отправить их клиентам, было отдельной сложностью.

Некоторые страны имеют особенно строгие правила импорта литиевых батарей. В худшем случае нам приходилось отправлять некоторым клиентам ноутбук без батареи внутри, а готовый

батареиный блок от поставщика, специализирующегося на RC-батареях (например, Hobby King), мы отправляли этим клиентам отдельно за свой счет. Они получали ту же батарею, которая была представлена в краудфандинговой кампании, но должны были подключить ее самостоятельно. Это было наше самое безопасное запасное решение, поскольку Hobby King отправляет тысячи батарейных блоков в день по всему миру.

Однако проблемы доставки не остановили нас от разработки нестандартного батарейного блока. Поддерживать постоянный запас батарейных блоков трудно, потому что батареи нужно периодически кондиционировать, поэтому этот батарейный блок получили только спонсоры кампании - при условии, что их страна проживания разрешала его импорт. Мы не могли знать наверняка, пока не попробуем, но мы получили сертификацию UN38.3 для нестандартного батарейного блока. Теоретически эта сертификация позволила бы нам отправлять батареи авиагрузом, но правила перевозки батарей постоянно меняются. Похоже, страны и перевозчики постоянно придумывают новые правила, особенно на фоне всей паранойи вокруг потенциального использования литиевых батарей как зажигательных устройств, а у нас не было ресурсов, чтобы поспевать за духом времени.

Емкость нестандартного блока была заявлена как 5 000 мА·ч, что примерно вдвое больше емкости блока, который мы показывали в краудфандинговой кампании. (На том снаружи было напечатано 3 000 мА·ч, но на практике он выдавал около 2 500 мА·ч.) В реальном тестировании нестандартный блок обеспечивал около шести или семи часов работы с включенным минимальным управлением питанием. Кроме того, поскольку я смог задать спецификацию батареи, я знал, что в нее встроена правильная схема защиты, и знал происхождение ее ячеек, поэтому был уверен в ее долгосрочной производительности и стабильности.

Выбор жесткого диска

Краудфандинговая кампания упоминала предоставление накопителей 240GiB Intel 530 (или эквивалент) и 480GiB Intel 720 для моделей "ноутбук" и "heirloom" соответственно. Мы оставили спецификацию слегка неоднозначной, потому что рынок SSD движется быстро. Мы знали, что лучший накопитель на момент составления спецификации, вероятно, будет отличаться от лучшего накопителя, который мы сможем получить при фактической закупке.

После некоторого исследования мы почувствовали, что лучшие эквивалентные накопители на момент покупки - 240GiB Samsung 840 EVO (для модели ноутбук) и 512GiB Samsung 850 Pro (для Heirloom). xobs и я лично использовали 840 EVO в собственных устройствах несколько месяцев, и он работал превосходно.

Важным показателем для нас было то, насколько хорошо накопители выдерживают неожиданные отключения питания. Отключения случаются довольно часто, например, когда вы занимаетесь разработкой подсистемы управления питанием. Некоторые жесткие диски вполне надежно отказывали (как вам такой оксюморон?) после нескольких неожиданных циклов выключения питания.

Для Heirloom мы использовали серию Samsung 850 PRO. Этот накопитель поставлялся с серьезной гарантией, подходящей для семейной реликвии: 10 лет. Samsung могла заявлять такую высокую надежность, потому что накопитель использовал технологию, которую компания называет V-NAND, а я считаю ее первой настоящей серийной 3D-транзисторной технологией.

Примечание. Intel заявляет, что делает 3D-транзисторы, но это всего лишь маркетинговый шум. Да, область затвора имеет приподнятую топологию поверхности, но вы все равно получаете только один слой устройств. С точки зрения проектирования вы все еще работаете с двумерным графом устройств. Intel стоило придерживаться того, что я считаю

«исходным» (и более описательным, менее вводящим в заблуждение) названием, FinFET, потому что, называя это 3D-транзисторами, я не знаю, как она назовет настоящие трехмерные массивы транзисторов, если когда-нибудь доберется до их создания.

Chipworks, компания патентной поддержки, сделала отличный начальный анализ V-NAND,[4] показывающий, что эта технология - не просто укладка пары транзисторов друг на друга. Stack V-NAND - это 38-слойный активный транзисторный сэндвич, весь в одной точке. Это высший пилотаж технологического процесса. Это Нео, расшифровывающий Матрицу. Это Мэл, стреляющий первым. Это меняет правила игры, и это не varogware. Спонсоры Heirloom получили ноутбуки с более чем 4 трлн таких транзисторов внутри.

Финализация прошивки

Со стороны программного обеспечения следующим шагом было финализировать выбор ядра, загрузчика и дистрибутива, а также решить, что показывать, когда Novena загружается впервые.

Marek Vasut добился поддержки Novena в основной ветке U-Boot (Universal Bootloader), одного из самых популярных загрузчиков с открытым исходным кодом. (Marek - один из сопровождающих U-Boot.) Процесс потребовал удивительно много патчей, отчасти потому что мало ARM-плат поддерживают столько RAM, сколько Novena. С этими патчами Novena имела полную поддержку U-Boot, включая USB и видео.

Мы решили сделать Debian заводским дистрибутивом по умолчанию для Novena и использовали стандартное ядро Linux с добавленными патчами. Все патчи, которые, как мы считали, могли быть полезны другим проектам, были отправлены в основную ветку и будут продолжать туда отправляться. Отправка изменений в основную ветку просто означает, что пакет, являющийся частью производной операционной системы, становится частью дистрибутива, от которого она производна.

Мы сохранили несколько локальных патчей: от специализированных хаков до экспериментальных функций, которые не были готовы к отправке в основную ветку или опирались на функции, которых тогда еще не было в основной ветке. Например, система отображения на ноутбуке очень отличается от того, что обычно видишь на ARM-устройстве. В большинстве ARM-устройств экран фиксирован во время загрузки, и горячая замена дисплеев во время работы невозможна. Как типичный ноутбук, Novena поддерживает два разных дисплея одновременно и позволяет подключить HDMI-монитор без перезагрузки. Поддержка этой функции потребовала локального патча к ядру, поскольку опиралась на функции, которые в то время еще не были отправлены в основную ветку для ARM-платформы.

Наконец, нам нужно было решить, что показывать, когда Novena включается. В Linux совсем не принято иметь экран настройки первого запуска, где вы создаете пользователя, выставляете время и настраиваете сеть. Это обычно для Windows и OS X, которые поставляются предустановленными, но в Linux этим обычно занимается установщик.

Мы разрывались между созданием хорошего опыта в стиле настольной системы и практичного опыта разработчика встраиваемых систем. Настольный сценарий поставлялся бы как чистый лист и предлагал пользователю создать учетную запись через локально подключенные клавиатуру и монитор. Но разработчики встраиваемых систем могут никогда не подключать монитор и вместо этого предпочитать подключение через консоль или SSH; для них имя пользователя, пароль и имя хоста по умолчанию были бы полезнее. В любом случае мы хотели создать одну прошивку, общую для всех платформ, и избежать специальных релизов под конкретную цель.

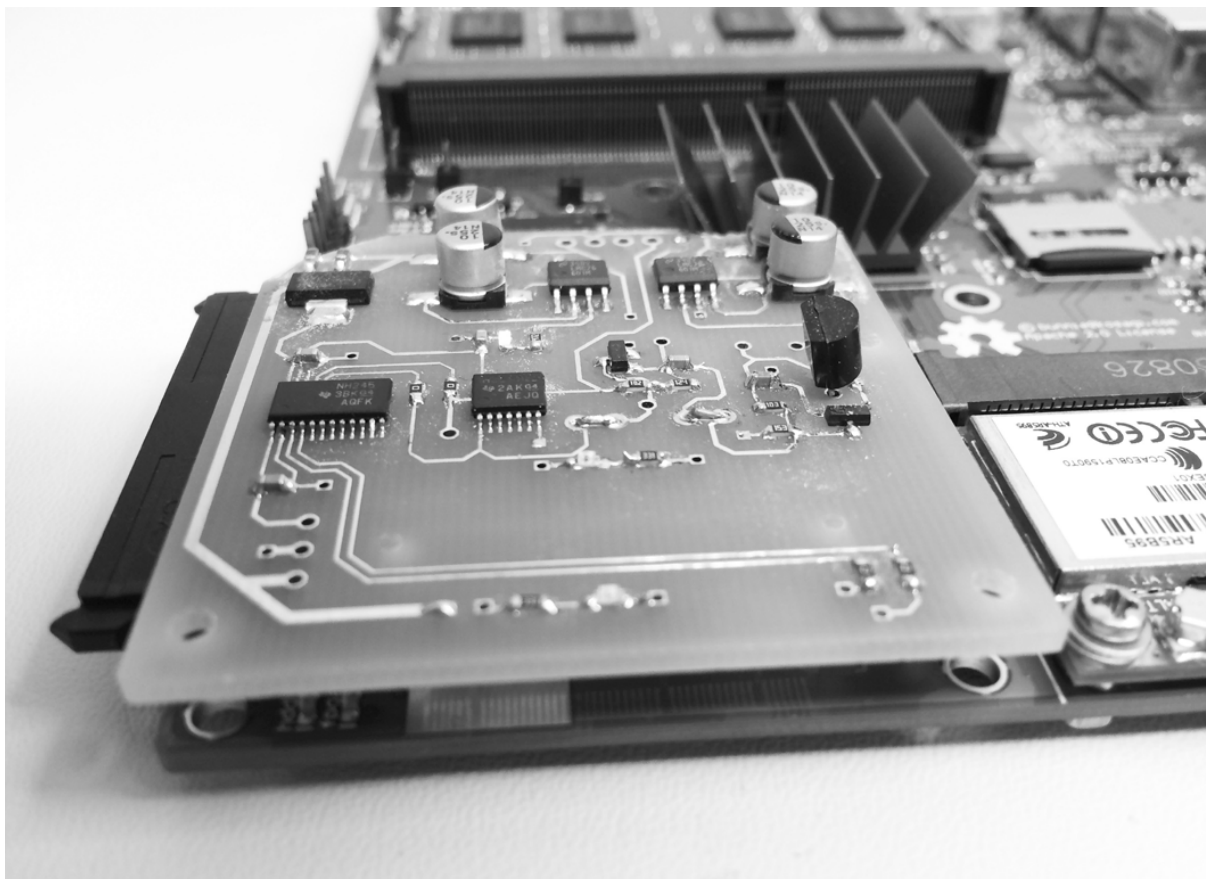
В итоге мы решили создать опыт в стиле настольной системы с выходами для опытных пользователей, чтобы обходить формальности регистрации пользователя. Это дало нам лучшее из двух миров. Это улучшило доступность Novena для начинающих пользователей, но опытные пользователи все еще могли сразу перейти к делу.

Создание сообщества

С самого начала hobs и я строили Novena, чтобы расширить возможности хакеров, поэтому я был рад, что еще до отгрузки у Novena были активные альфа-разработчики. Jon Nettleton и Russell King работали над графикой, Marek Vasut из U-Boot помогал, и пара других групп альфа-пользователей действительно сделали железо для системы.

MyriadRF, сообщество открытого железа и программного обеспечения, ориентированное на беспроводные технологии, создало плату программно-определяемого радио для Novena. Мы купили и интегрировали эти платы с первыми настольными и ноутбучными устройствами, которые отгружались.

Группа CrypTech также начала применять Novena в своих проектах до того, как ноутбук был отгружен. Проект CrypTech разработал аппаратный модуль безопасности с референсным дизайном под лицензиями BSD и CC BY-SA 3.0. Группа хотела создать широко проверенное устройство, спроектированное для криптографии, которое любой мог бы скомпоновать для своего приложения и легко собрать со своей доверенной цепочкой поставок. CrypTech использовала Novena для прототипирования элементов своего дизайна.



Прототип платы расширения CrypTech, подключенный к материнской плате Novena.

Плата расширения, показанная здесь, - прототип источника шума, основанного на лавинном шуме транзистора в середине платы. CtrupTech использует этот шум для генерации энтропии в FPGA Novena. Затем энтропия объединяется с энтропией, генерируемой кольцевыми осцилляторами в FPGA, и перемешивается, например, с помощью SHA-512 для генерации начальных значений. Затем эти начальные значения используются для инициализации потокового шифра ChaCha, что в итоге дает поток криптографически надежных случайных значений. Результат - высокопроизводительный современный сопроцессор генератора случайных чисел.

Заключительные мысли

В качестве заключительного замечания: если есть одна вещь, которую хobs и я усвоили в аппаратном бизнесе, так это то, что нельзя считать цыплят до того, как они вылупились. Хороший прогресс до определенного момента не означал, что у нас будет легкий путь к готовым устройствам. Хотя на момент закрытия сбора средств у нас были полностью функциональные прототипы, все равно потребовались месяцы интенсивных усилий, чтобы доставить сотни устройств конечным пользователям.

Теперь, когда Novena закончила отгрузку, мы продолжаем поддерживать нашу восторженную, но очень терпеливую пользовательскую базу. Это большой объем работы, который ложится главным образом на плечи хobs, но мы отвечаем на вопросы пользователей, отправляем патчи и поддерживаем kernel Novena в актуальном состоянии.

Мы делаем это, хотя больше не получаем новой выручки от продаж Novena. Анализ наших данных продаж после кампании довольно ясно показал, что жизнеспособного пути для аппаратного бизнеса по продаже Novena не было; мы продавали бы в среднем пару устройств в месяц. Хотя мы выполнили минимальные объемы заказов наших поставщиков через исходную краудфандинговую кампанию, было бы очень трудно вовлечь любого из наших поставщиков при объемах меньше пары сотен единиц. Продажа пары устройств в месяц при такой минимальной закупке оставила бы нас с запасом примерно на сто месяцев. Мы были бы должны нашим поставщикам несколько лет. Неспособность расплатиться с поставщиками в течение нескольких лет также известна как банкротство.

Мы, конечно, выполняем исходное обещание поддерживать материнскую плату Novena как минимум пять лет с момента первоначальной кампании финансирования. Мы отложили солидную сумму денег, чтобы обеспечить стабильный запас материнских плат. Наш первоначальный краудфандинговый, а теперь партнер по онлайн-продажам Crowd Supply взял на себя оставшийся запас корпусов и аксессуаров. Благодаря нашей модели открытого железа Crowd Supply может производить и продавать аксессуары для Novena, если появится спрос конечных пользователей.

В итоге я очень рад видеть нежные зеленые ростки новых проектов, стремящихся предложить конечным пользователям лучшие решения ноутбуков с открытым исходным кодом. Вместо того чтобы конкурировать с ними, я думаю, что для Novena уместнее всего уступить дорогу и дать увлеченным новым разработчикам шанс найти свою удачу, продавая собственные решения. В конце концов, мы начали это приключение главным образом чтобы посмотреть, можно ли это сделать. Мы хотели построить крутой инструмент, настроенный под наш повседневный сценарий использования; мы не хотели начинать бизнес по продаже ноутбуков с устойчивым массовым спросом. Если конечное влияние проекта Novena - поднять планку открытого железа и, возможно, даже вдохновить новое поколение проектов на тему ноутбуков, это само по себе будет огромной наградой.

Примечания

[1] Документацию можно найти онлайн в Kosagi wiki: <http://www.kosagi.com/>.

[2] <http://boingboing.net/2014/01/17/building-a-fully-open-transpa.html>

[3] Один мегараскал равен 1 newton (единица силы) на мм²; 1 gigapascal равен 1 kilonewton на мм².

[4] Если вам любопытно, этот анализ можно найти по адресу <https://www.chipworks.com/about-chipworks/overview/blog/second-shoe-drops-%E2%80%93-samsung-v-nand-flash/>.

Глава 8. Chibitronics: создание наклеек-схем

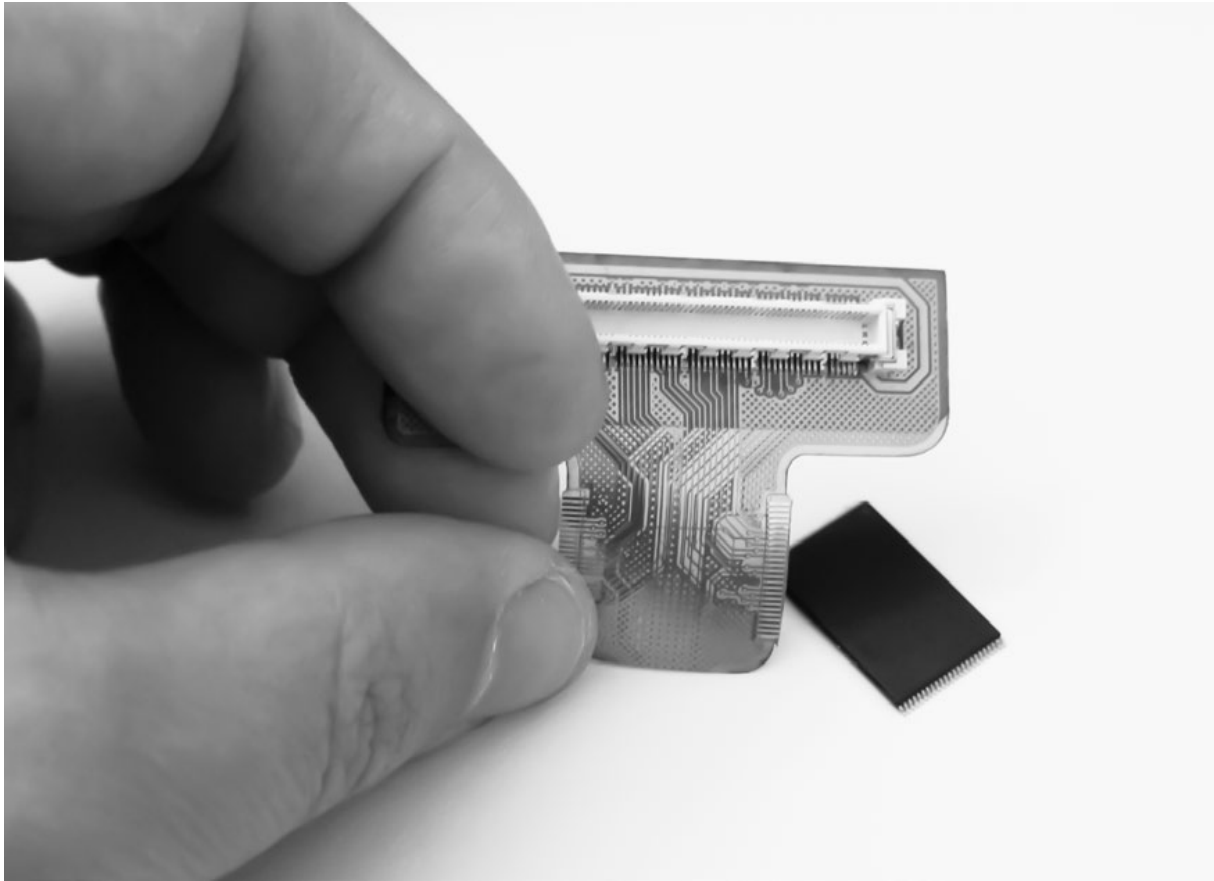
В сегодняшнем мире контрактного производства и поставщиков услуг "под ключ" дизайнеры обычно выбирают из палитры существующих процессов, чтобы разрабатывать продукты. Большинство потребительских электронных устройств - это смесь жестких PCB с SMT-пайкой оплавлением или волновой пайкой сквозного монтажа, литья под давлением из ABS или PC, формовки листового металла и некоторых финишных процессов вроде покраски или гальваники. Эти варианты покрывают весь диапазон полезности, который требуется большинству продуктов. Однако действительно выдающиеся продукты также часто вводят новые материалы или новые производственные процессы.

Разработка таких новых процессов не обязана быть дорогой - пока вы готовы выйти в цех и направлять улучшения самостоятельно. Иными словами, дорогая часть разработки процесса обычно состоит в оплате экспертов, которые разрабатывают и аттестуют процесс, а не столько в оборудовании или материалах.

Чтобы доказать это самому себе, я начал исследовать гибкие схемы как среду проектирования. Вместо жесткой подложки толщиной 1 или 2 миллиметра, состоящей из тканого стекловолокна, пропитанного жесткой эпоксидной смолой, гибкие схемы обычно используют податливую полимерную подложку толщиной всего в доли миллиметра. Polyimide - популярная подложка в гибких схемах благодаря способности выдерживать температуры пайки. Хотя технология гибких схем часто встречается внутри потребительских продуктов (мобильный телефон, вероятно, содержит как минимум полдюжины гибких PCB, соединяющих периферию вроде кнопок, камер и дисплеев с основной платой), эта технология недостаточно представлена в любительских и DIY-продуктах. Но я не думаю, что так должно быть.

У меня было предчувствие, что правильный тип продукта, спроектированный на гибкой схеме, может открыть новые творческие применения, но я не был вполне уверен какие, поэтому решил больше узнать об уникальных преимуществах и трудностях проектирования гибких схем. В рамках проекта, где я исследовал внутренности SD-карт, о котором подробнее расскажу в главе 9, мне нужно было создать адаптер для моей Novena, который позволил бы подслушивать и эмулировать flash-память NAND, находящуюся внутри некоторых типов старых SD-карт. Тонкость и гибкость гибких схем идеально подходили для этой задачи.

Получившийся адаптер был очень тонким; он идеально помещался под TSOP-корпус NAND. Гнущаяся природа платы означала, что я также мог подстроиться под широкий набор форм целевых плат, даже плат гораздо больше типичной SD-карты. Хотя это было полезное применение гибких схем, все равно казалось, что я только царапаю поверхность возможного.



Мой нестандартный гибкий адаптер.

Затем наступил счастливый случай. Работая над проектом SD-карт, я познакомился с Jie Qi, тогда PhD candidate в MIT Media Lab, которая в своем исследовании сочетала бумажное рукоделие и электронику. Она была частью группы студентов MIT Media Lab, которую я водил на экскурсию по Шэньчжэню в январе 2012 года, и примеры ее бумажных схем запустили у меня в голове шестеренки.



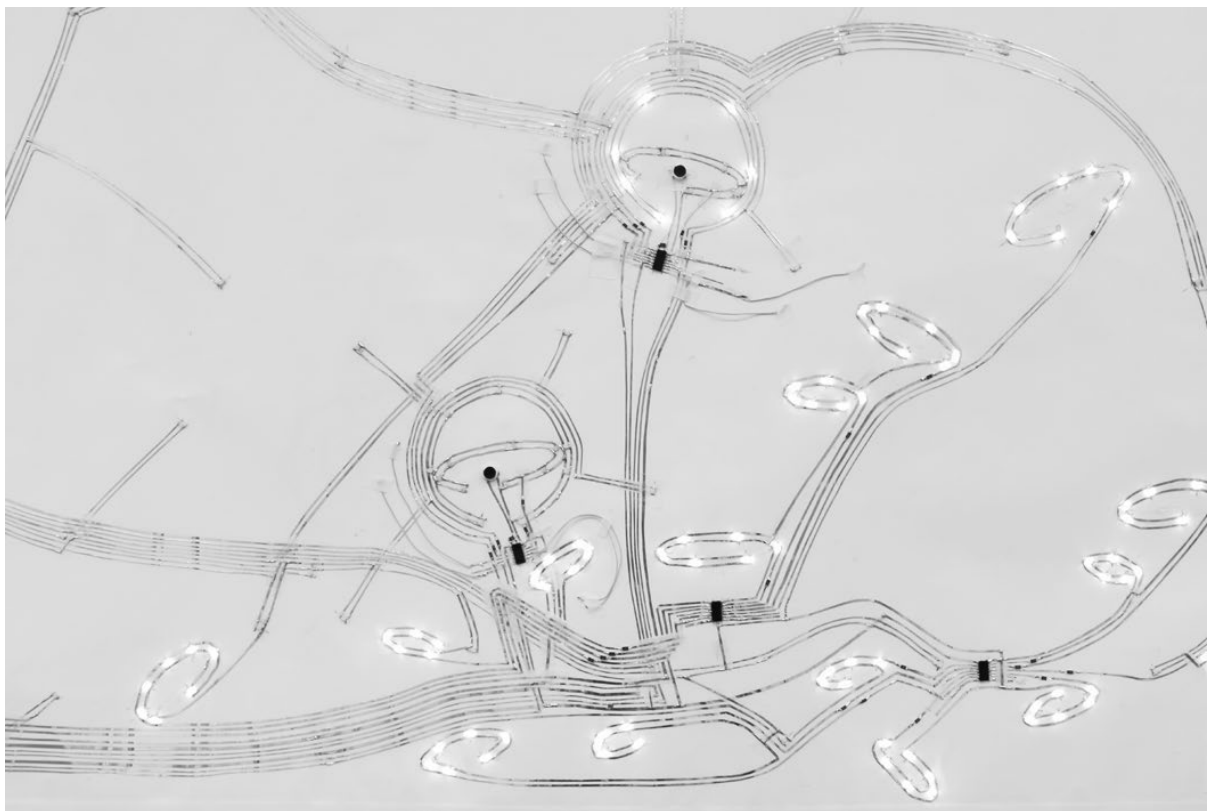
Финальная графика для производства бумажной схмотехники Jie Qi, Pu Gong Ying Tu.



Крупный план цветов.



Картина отгибается, открывая схему.

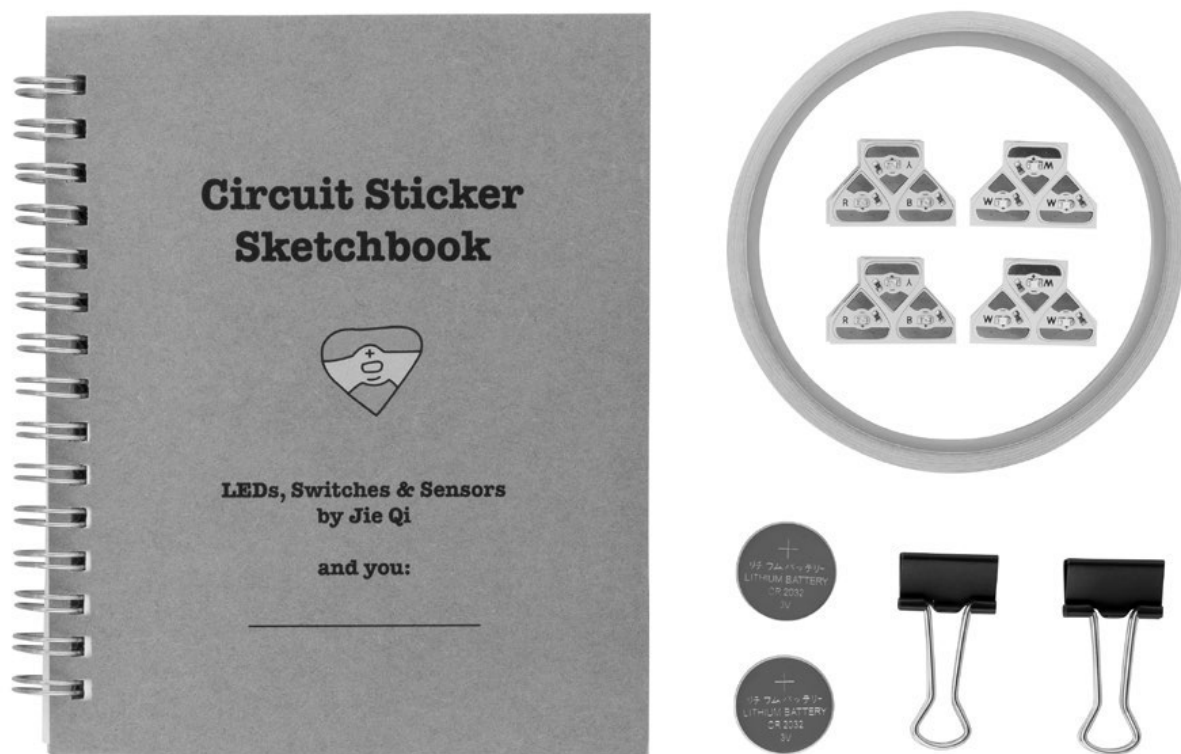


Схемы цветов внутри Pu Gong Ying Tu.

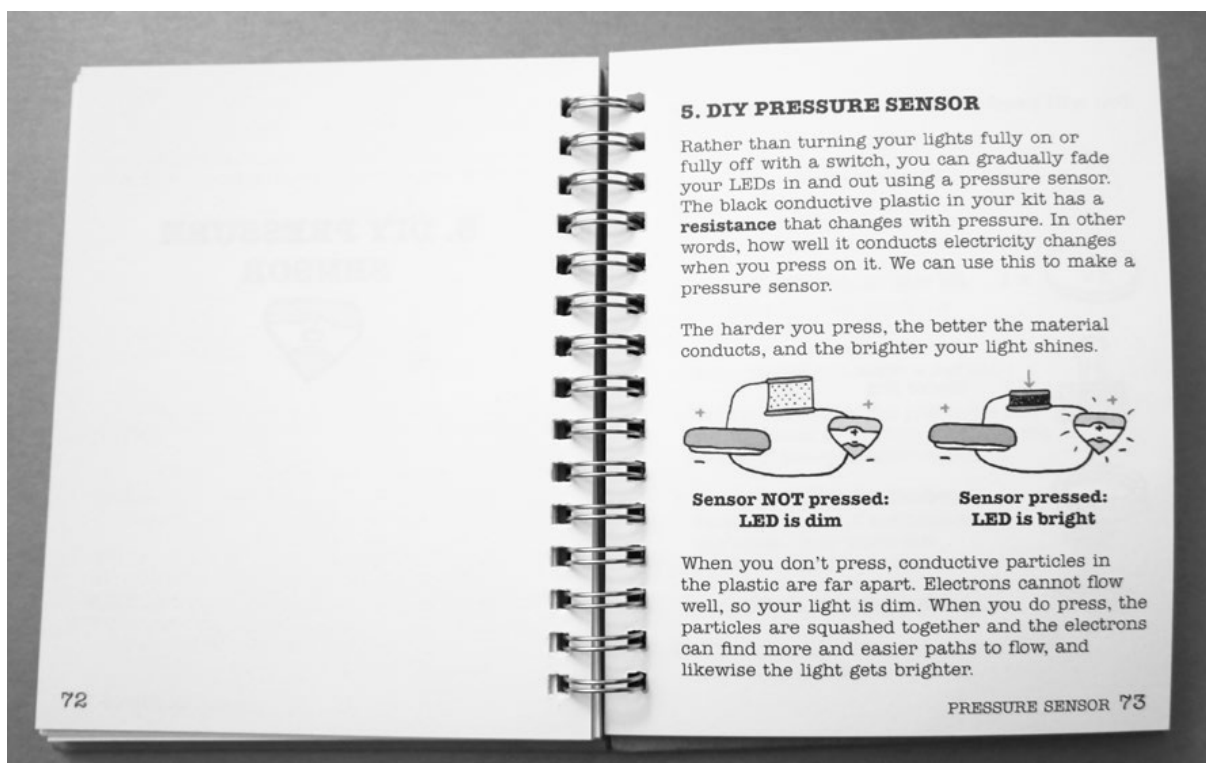
Используя лишь медную ленту, бумагу и капли припоя или скотч, чтобы удерживать компоненты на месте, Jie смогла создавать возвышенные произведения искусства, которые светились и взаимодействовали со зрителями. Эти чарующие работы показали, как электронику можно использовать не только как функциональную среду, но и как выразительную среду, вызывающую удивление и восторг. Фото здесь показывает внутренности одной из ее известных ранних работ, Pu Gong Ying Tu (Dandelion Painting), где сама схема является таким же произведением искусства, как и наложенная поверх картина.

Jie также очень увлечена образованием, и она видела большой потенциал бумажной электроники в том, чтобы сделать технологию более актуальной и доступной для аудитории без инженерной подготовки. Во время нашей поездки в Шэньчжэнь мы обсуждали возможность строить схемы на гибкой подложке, а затем припаивать гибкую схему к бумаге. В итоге она почувствовала, что это было бы в лучшем случае незначительным улучшением. Хотя пайка не является трудным навыком, высокие температуры, химикаты и специализированное оборудование сильно отпугивают начинающих. По-настоящему волшебным было бы, если бы схемы можно было собирать как наклейки на странице. Разве не было бы здорово, если бы мы могли использовать технологию гибких схем с традиционными процессами SMT-пайки оплавлением, чтобы создать модули, которые пользователи затем могли бы наклеивать на дорожки из медной ленты?

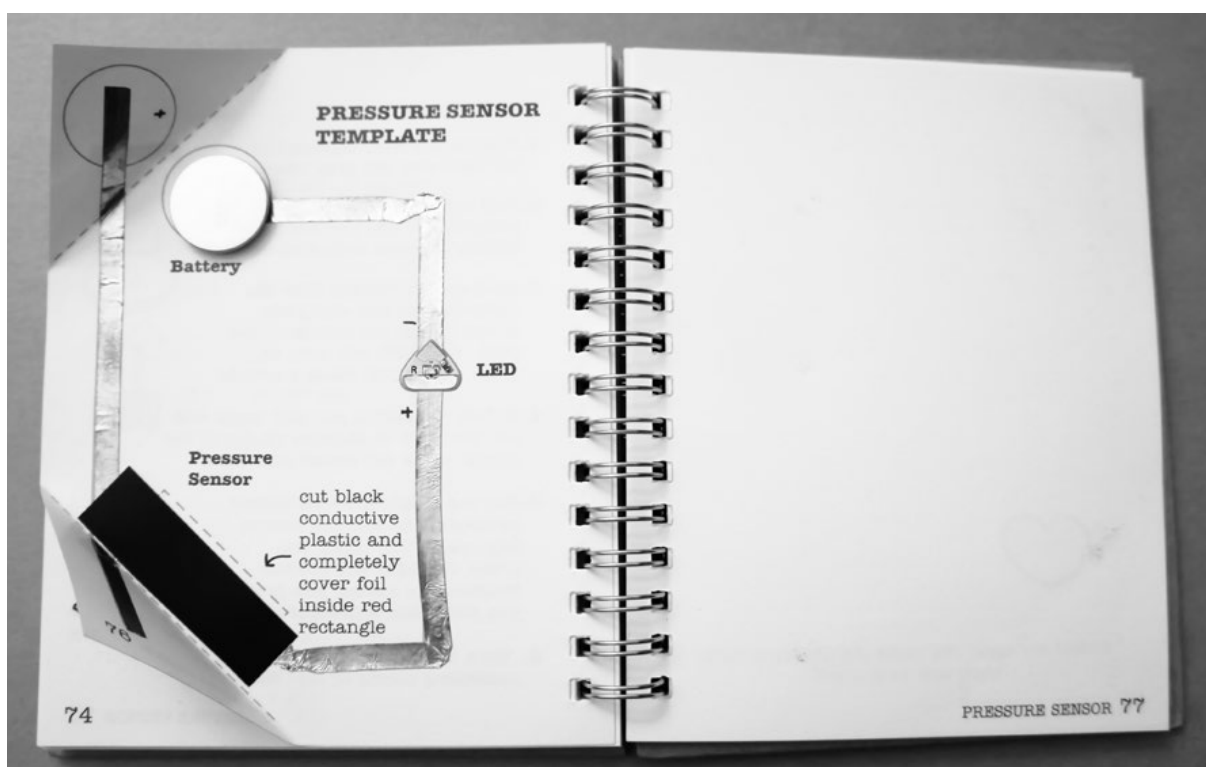
Так мы и пришли к сотрудничеству над Chibitronics, проектом, в котором разработали набор самоклеящихся электронных схем для рукоделия и образования. Chibitronics с самого начала был проектом открытого железа, и вы все еще можете найти все задания из Circuit Sticker Sketchbook, исходный код для всех использованных микроконтроллеров и другие технические подробности в вики проекта по адресу <http://chibitronics.com/wiki/>.



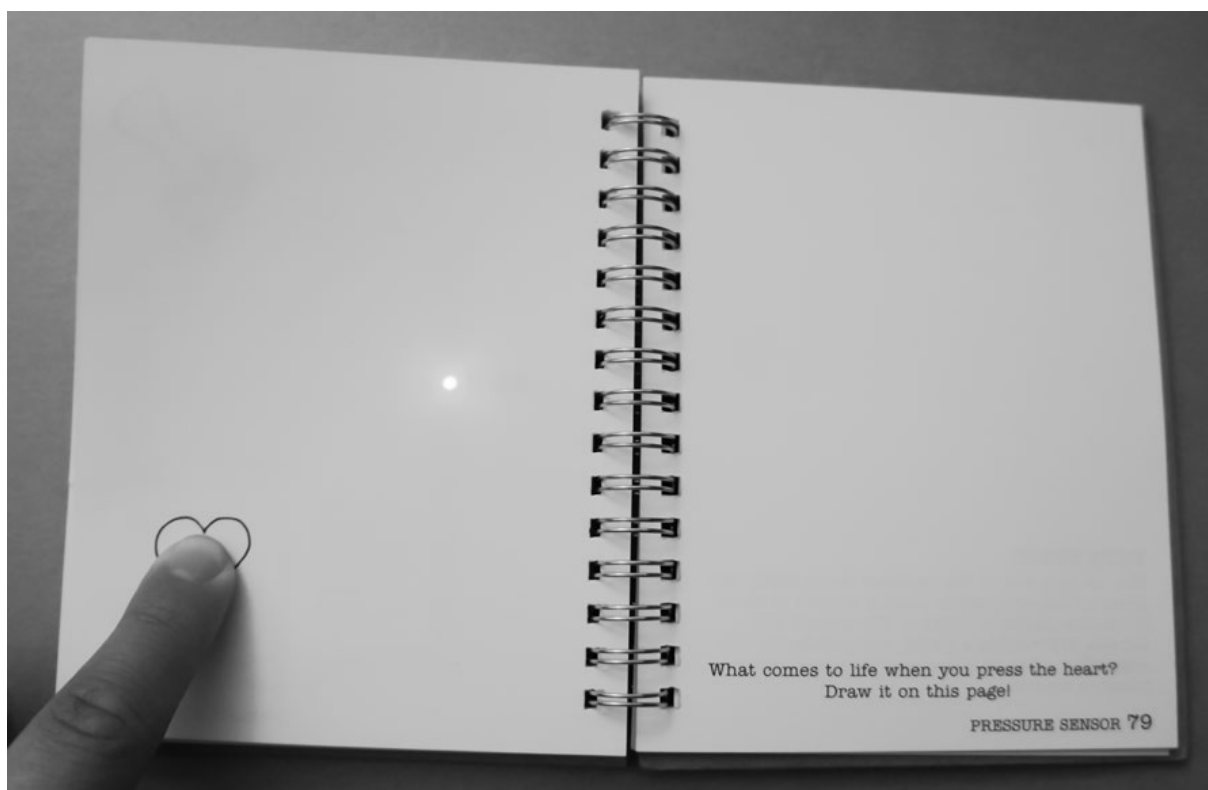
Chibitronics STEM Starter Kit включает Circuit Sticker Sketchbook, LED-наклейки, медную ленту, батарейки и канцелярские зажимы для батареек.



Объяснение, как создать DIY-датчик давления.



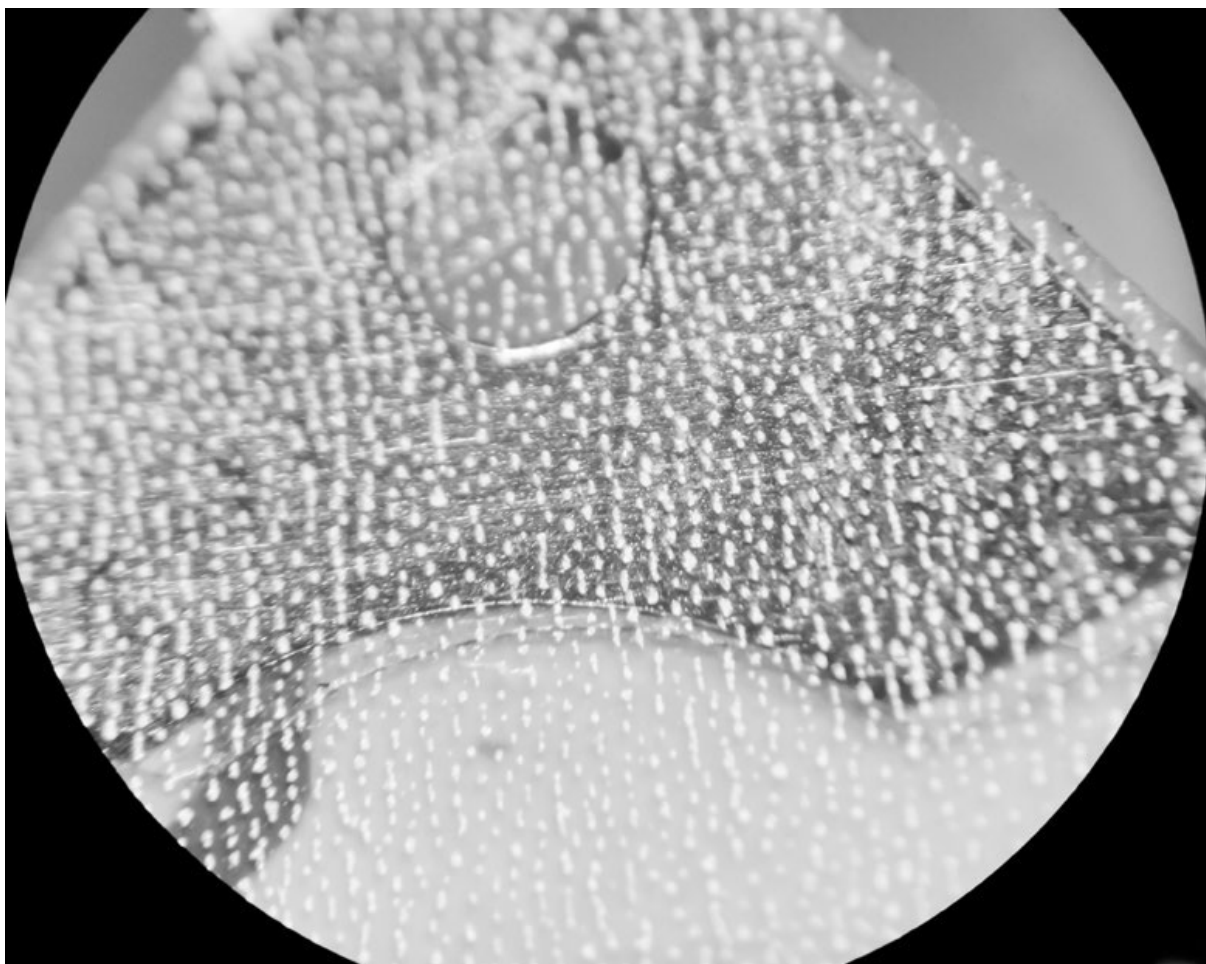
Собранный DIY-датчик давления.



DIY-датчик давления с бумажной накладкой.

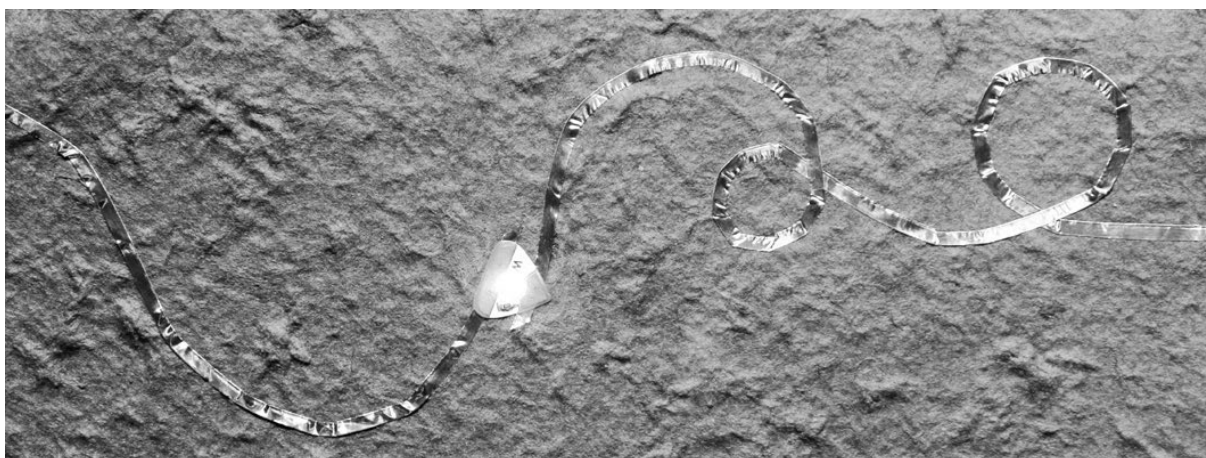
Рукоделие со схемами

Решение, к которому мы пришли в начале 2012 года, опиралось на наработки исследовательской группы High-Low Tech профессора Leah Buechley в MIT. Мы решили строить схемы на гибкой полиимидной подложке с анизотропной лентой (также называемой Z-tape, потому что электричество течет через ленту только вертикально, а не вбок), ламинированной с обратной стороны.

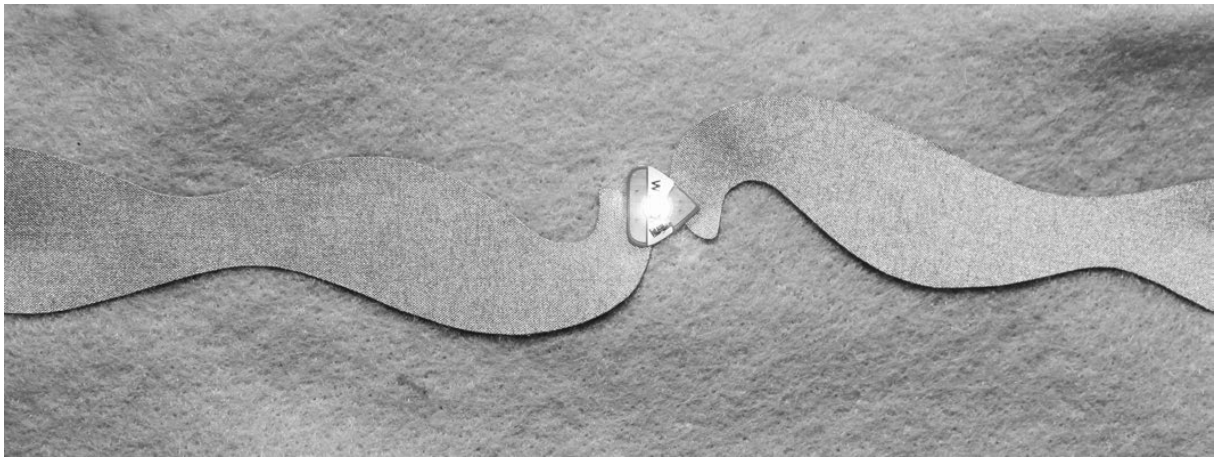


Фрагмент Z-tape под микроскопом.

Использование Z-tape позволяет конечным пользователям собирать схемы без высокотемпературных процессов вроде пайки или оплавления. Возможность просто приклеивать компоненты на место невероятно полезна для художественных проектов, где часто используются чувствительные к теплу и/или податливые материалы вроде бумаги, ткани и пластика. Наклейки-схемы и медная лента тоже гибкие, что дополнительно позволяет любому встраивать электронику в проекты из нетрадиционных материалов. Такие дружелюбные и выразительные материалы побуждают создателей превращать сами схемы в красивые произведения искусства.



Наклейки-схемы на бумаге.



Наклейки-схемы на ткани.

Создание этих наклеек-схем вращалось вокруг ограничений Z-tape. В увеличенном фрагменте Z-tape, ламинированной на полиимидную подложку, серебристо-белые точки - это крошечные металлические частицы, проходящие от одной стороны клеевого слоя к другой согласно статистическому распределению. Учитывая природу распределения металла, для надежного электрического контакта каждая площадка на наклейке-схеме должна была быть довольно большой. Кроме того, дорожки, расположенные очень близко друг к другу, могли быть замкнуты встроенными металлическими частицами, поэтому при проектировании схем я должен был внимательно оставлять достаточно места между открытыми площадками. Datasheet для материала Z-tape содержит правила минимального размера и расстояния между площадками, поэтому я использовал их как руководство.

Разработка нового процесса

Одно дело - спроектировать наклейки, содержащие работающие электронные схемы, и совсем другое - фактически их изготовить. Не существовало стандартных производственных процессов, которые могли бы производить наклейки-схемы так, как мы их представляли. Наконец у меня появилась реальная возможность проверить мою теорию: разработку нового процесса можно выполнить дешево, если вы готовы делать это сами. Поэтому я начал собственную маленькую исследовательскую программу, чтобы изучить среду гибких схем и трудности изготовления из них наклеек-схем, причем на крошечном R&D-бюджете.

Визит на фабрику

В качестве первого шага я посетил предприятие, где производили гибкие PCB. Визит оказался откровением.



Рабочий вручную выравнивает коверлей на материале гибкой схемы.

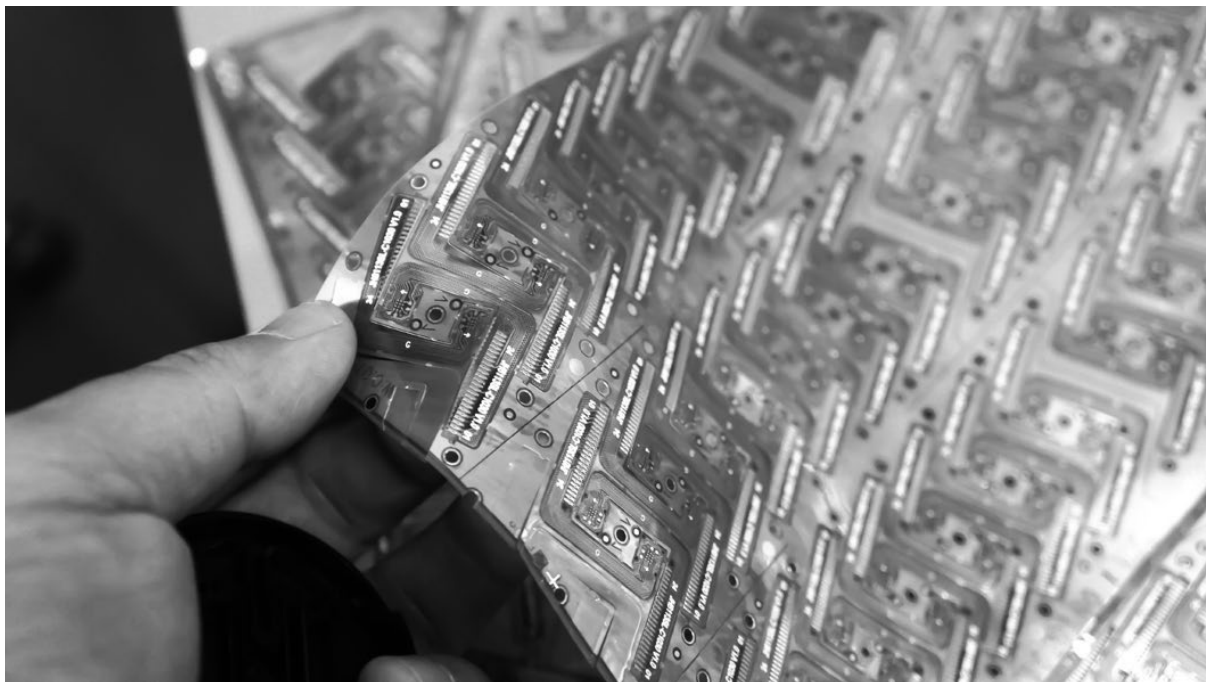
Вместо паяльной маски дорожки гибкой схемы защищены полиимидным листом, который называется коверлей. Паяльная маска слишком хрупкая и треснет при сгибании, но коверлей надежно остается целым на протяжении тысяч циклов изгиба. Иногда, однако, нужно сделать части гибкой схемы жесткими; например, часть схемы может должна оставаться жесткой для механического крепления, а жесткая схема также полезна для SMT-обработки.



Стальные пластины ламинируются на обратную сторону материала гибкой схемы.

Я знал, что полиимидные ребра жесткости можно ламинировать на гибкую схему, но, как оказалось, стальная ламинация тоже возможна. Я бы не узнал этого, если бы сам не посетил фабрику. Личный визит на фабрику также дал мне бесценную возможность увидеть широкий диапазон сложных форм, которые можно производить благодаря высечке. Разнообразие возможных форм было ключевым, потому что мы хотели, чтобы наклейки-схемы тоже выглядели круто. На вопросы вроде

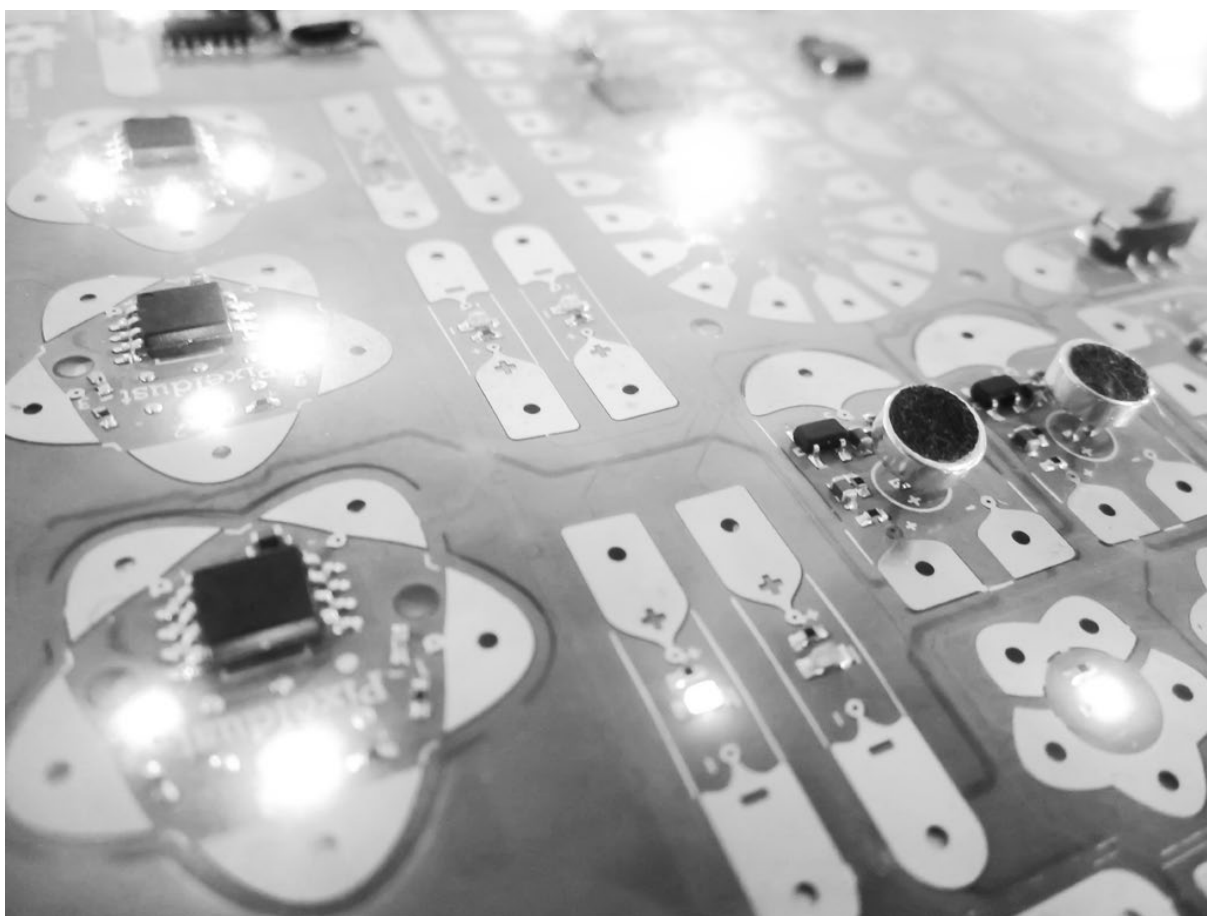
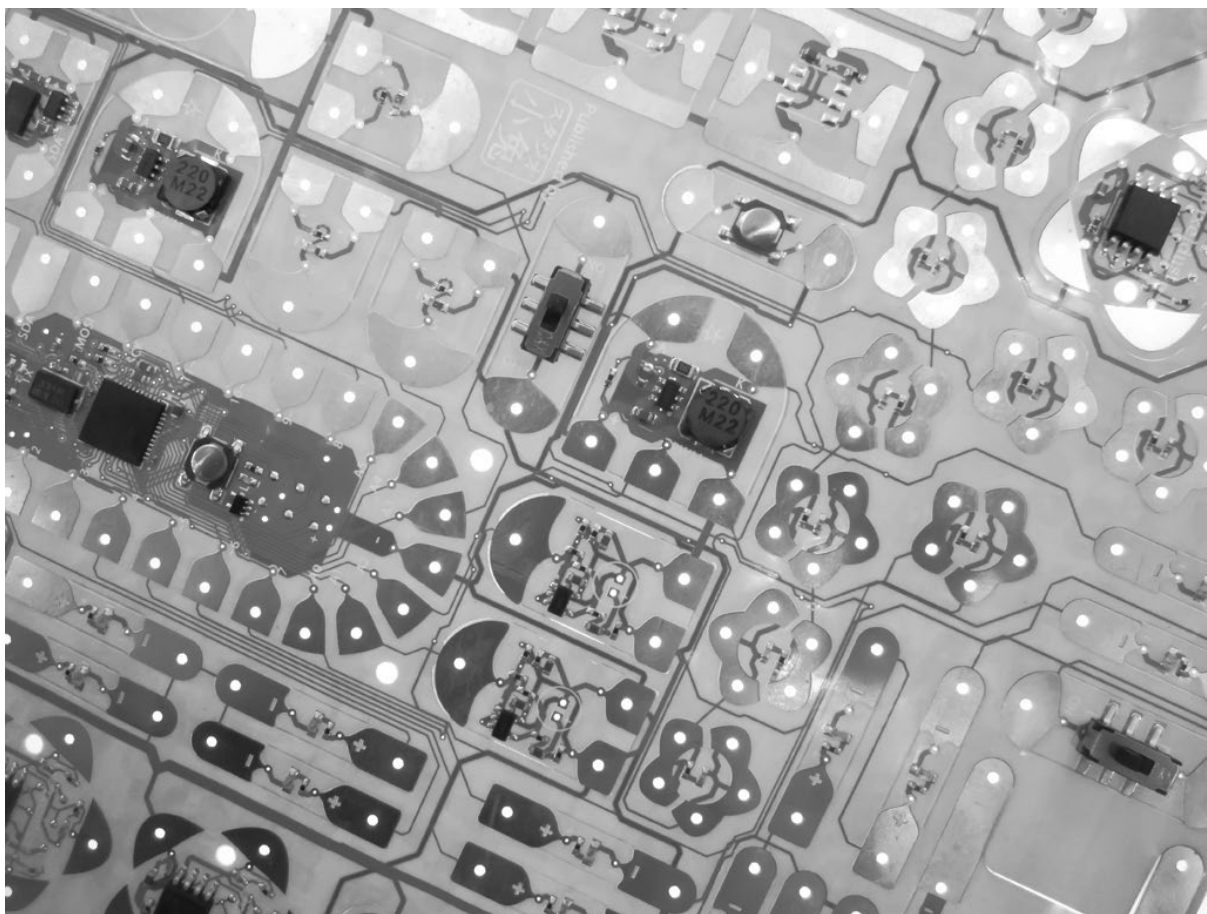
того, насколько узко можно резать материал или насколько малый радиус допустим при высечке, трудно ответить по email, но ответы стали интуитивно очевидны после того, как я увидел процесс лично.



Сложные формы гибких схем, достижимые с помощью высечки.

Проведение теста возможностей процесса

После визита на фабрику следующим шагом был тест возможностей процесса, чтобы раздвинуть пределы производственного процесса. Мы спроектировали неоднородный лист вариантов наклеек, который задействовал самые разные возможности: длинные цепочки переходных отверстий, дорожки шириной 3 mil, компоненты 0201 (малый размер SMT-корпуса), QFN-детали с шагом 0,5 мм (компоненты поверхностного монтажа, у которых все контакты находятся снизу), громоздкие компоненты, использование паяльной маски вместо коверлея, мелкие детали шелкографии, удерживающие язычки, криволинейные вырезы, гибридные техники SMT и пайки в отверстия, ламинирование Z-tape и многое другое. Наш тест возможностей процесса намеренно ломал части производственного процесса, чтобы обнаружить слабые места, которые могли помешать нашей конструкции сработать.



Дизайн наклейки-схемы, который мы изготовили для теста возможностей процесса.

Когда я впервые представил дизайн, фабрика сразу отвергла его, сказав, что изготовить это невозможно. Однако после того как я объяснил свои цели, фабрика согласилась произвести его при условии, что я приму и оплачу все изготовленные изделия, естественно включая дефектные. Анализируя режимы отказа дефектных изделий, я разработал набор правил проектирования для поддержания высокого выхода годных изделий (и, следовательно, снижения стоимости) наклеек-схем.

На основе этих правил проектирования Jie и я создали наш первый набор наклеек - "предсерийный кандидат". В него входили LED четырех разных цветов (белый, красный, синий и желтый), а также два набора умных наклеек. Первый набор умных наклеек содержал предварительно запрограммированный микроконтроллер, который мог генерировать световые паттерны, такие как плавное затухание, сердцебиение, мерцание и мигание. Мы называли их наклейками "эффектов"; это форма физического программирования, которая позволяет людям без навыков кода настраивать поведение своих проектов. Второй набор содержал программируемый пользователем микроконтроллер с забавной возможностью записи и воспроизведения, загруженной в него как демонстрация, вместе с тремя датчиками. Мы называли их наклейками "сенсор и микроконтроллер".

Мы запускали небольшие партии наших предсерийных кандидатов, чтобы найти проблемы, с которыми могли столкнуться при масштабировании, и тщательно расследовали любые вопросы, влияющие на надежность, выход годных изделий или удобство использования. В частности, нам пришлось разработать новый метод ламинирования Z-tape на обратную сторону наклеек, совместимый с типом высеки, необходимым для создания наклеек.

После двух итераций предсерийных кандидатов мы почувствовали, что готовы увидеть, что другие люди смогут сделать с наклейками-схемами. Поскольку это было частью докторского исследования Jie, у нас было два варианта пользовательского тестирования. Традиционный академический подход состоял бы в том, чтобы попросить бюджет у ее научного руководителя, произвести ограниченное количество наклеек и провести серию закрытых воркшопов, чтобы изучить, как молодые и творческие умы взаимодействуют с этой новой средой. Но это происходило в 2013 году, поэтому уже работавшие краудфандинговые платформы открыли возможность предложить наше исследование напрямую заинтересованным пользователям и провести его в большом масштабе. MIT Media Lab, где работала Jie, также очень внимательно относилась к возможностям масштабного исследования, воплощенным в их инициативе "deploy". В 2011 году, когда Joi Ito стал новым директором Media Lab, он начал превращать культуру Media Lab из "demo or die" в "deploy or die" - от "демонстрируй или умри" к "внедряй или умри"; позже это сократилось до менее угрожающей директивы "deploy". При старом режиме "demo or die" исследовательские группы поощрялись создавать эффектные демонстрации технологии, которые могли помочь привлечь финансирование. Согласно директиве Ito идея состояла в том, чтобы вывести технологию из лаборатории в реальный мир, проводя масштабные исследования через инструменты вроде краудфандинга и бережливого аппаратного производства.

В ноябре 2013 года мы запустили краудфандинговую кампанию на Crowd Supply. Для нас было очень важно остаться верными академической миссии, стоявшей за наклейками-схемами, поэтому мы поставили цель финансирования всего \$1. Если хотя бы один человек решил бы, что наклейки-схемы могут быть интересны, мы произвели бы наклейки и работали бы с этим человеком, чтобы собрать обратную связь. И, конечно, мы сделали бы исследование доступным миру, на случай если кто-то захотел бы форкнуть проект или как-то иначе взломать свои наклейки-схемы.

Мы превысили нашу скромную цель на несколько порядков, закрыв кампанию чуть ниже \$60 000 после чуть больше месяца финансирования и очень сдержанной кампании.

Выполнение обещания

В рамках кампании мы заявили, что передадим заказы на отгрузку к маю 2014 года. К счастью, мы смогли выполнить цель точно в срок.



Шестьдесят две коробки, содержащие больше тысячи стартовых наборов Chibitronics, ждут отправки.

Однако доставка вовремя - непростая задача для любого краудфандингового проекта. Я сделал спорный выбор использовать Crowd Supply отчасти потому, что они проявляют больше понимания при проверке аппаратных продуктов, а услуги, которые они предлагают кампаниям (обработка и отправка заказов, клиентская поддержка первого уровня, поддержка предзаказов после кампании и гибкие даты доставки в зависимости от спроса и производственных возможностей), очень полезны начинающим аппаратным командам. Получить обработку заказов, клиентскую поддержку и постоянный сайт электронной коммерции как часть пакета означало, что нам не нужно нанимать кого-то для всего этого. Если ваша "компания" состоит всего из двух человек, пробующих академический проект, или из пары людей, работающих в гараже, это очень важно.

Crowd Supply не имеет такого же медийного охвата или силы бренда, как Kickstarter, что может затруднить сбор такой же суммы. Но в конце концов я чувствовал, что очень важно показать пример устойчивых практик краудфандинга, лучших и для предпринимателя, и для потребителя. Это не просто сегодняшний сбор денег; это построение бренда и репутации, которым можно доверять долгие годы.

Почему своевременная доставка важна

Я поставил перед собой личную задачу для Chibitronics: очень серьезно относиться к нашему обещанию доставить продукт спонсорам. Я видел слишком много провальных краудфандинговых кампаний и сильно беспокоился, что краудфандинг аппаратуры становится синонимом мошенничества и спама.

Kickstarter и Indiegogo страдали от недоставки и мошенничества, а их беспечное отношение к кампаниям в духе *caveat emptor* ("покупатель, будь осторожен") подчеркивает конфликт интересов между потребителями и краудфандинговыми сайтами. Краудфандинговые сайты по сути говорят спонсорам: "Спасибо за пятак, но что случилось с вашим долларом - ваша проблема". Я честно беспокоился, что краудфандинг получит такую плохую репутацию, что в итоге больше не будет жизнеспособной платформой для добросовестных предпринимателей и новаторов.

Суть такова: если я не могу доказать нынешним и будущим спонсорам, что могу доставить проект вовремя, я рискую потерять ценную платформу для запуска будущих продуктов. К счастью, с Chibitronics мы определенно доказали свою состоятельность, и с тех пор я продолжил использовать Crowd Supply для других краудфандинговых проектов.

Извлеченные уроки

Однако мы доставили Chibitronics вовремя не потому, что нам было легко. Когда я составлял исходный график кампании, мои минимальные и максимальные границы срока доставки тянулись от сразу после Китайского Нового года 2014 года (февраль) примерно до апреля. Я добавил к этому графику один месяц сверх максимума, просто для надежности, и мы использовали весь этот запас до последнего дня.

Я сделал много ошибок по пути, но благодаря сочетанию тяжелой работы, удачи, планирования и крепких отношений с фабрикой мы успешно преодолели множество трудностей. Вот несколько уроков, которые я усвоил в процессе.

Не все простые просьбы просты для всех

Каждый стартовый набор Chibitronics включал физический экземпляр замечательной книги, которую Jie написала как пошаговое руководство для самостоятельного обучения проектированию с наклейками-схемами, *Circuit Sticker Sketchbook* (показана на страницах 256-257). Книга необычна тем, что в нее предполагается вклеивать электронные схемы, поэтому нам пришлось настроить несколько аспектов печати. Бумага должна была иметь правильную толщину, чтобы получить хорошее рассеивание света, когда светодиоды размещены под листом. Переплет требовал особого внимания для лучшего опыта схемного рукоделия, а сзади даже есть маленький карман для образцов материалов для рукоделия, используемых в проектах книги.

Типография сочла большинство этих требований относительно легкими, но одно из них особенно сбilo ее с толку. Металлическая спираль переплета книги должна была быть непроводящей, чтобы размещение медной ленты на переплете случайно не вызвало короткое замыкание.

Проверить проволоку на проводимость кажется достаточно простой просьбой для человека, который проектирует схемы профессионально, но для книжной типографии это странно. Никакая часть традиционной печати или переплета книг не требует таких знаний. Типография сначала сказала, что не может ничего гарантировать насчет проводимости проволоки переплета. Так и вышло: первый образец проволоки был непроводящим, второй - проводящим, и типография не могла объяснить почему.

Личные встречи здесь были бесценны. Вместо того чтобы ругаться с типографией по email, мы договорились о встрече во время одной из моих ежемесячных поездок в Шэньчжэнь. У нас состоялось продуктивное обсуждение их опасений, и по итогам встречи мы заказали им мультиметр за \$5 в обмен на гарантию непроводящего корешка книги. В конце концов типография просто не хотела гарантировать то, для чего у нее не было процедуры контроля качества, что совершенно разумно. Нам просто пришлось научить их пользоваться мультиметром.

Это необычное требование непроводимости действительно увеличило наш срок поставки на несколько дней и добавило несколько центов к стоимости книги, но в целом я был готов принять этот компромисс.

Никогда не пропускайте проверочный чертеж

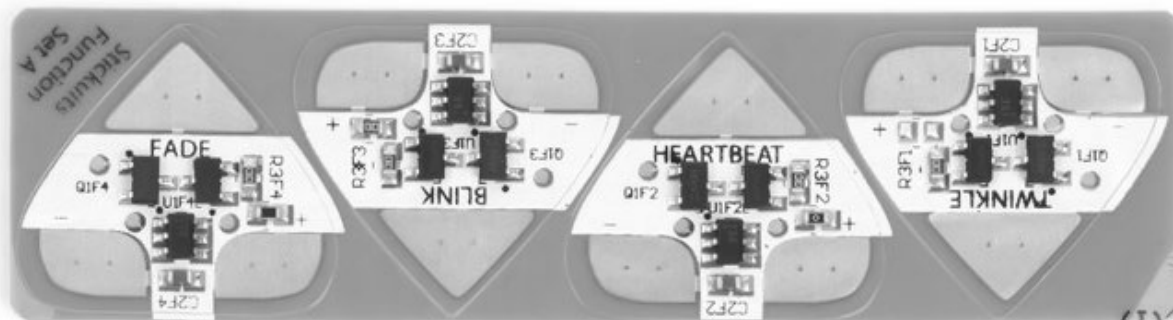
Формы площадок для наклеек-схем - это сложные полилинейные геометрии, с которыми Altium, программа проектирования РСВ, которую я использовал, работала не очень изящно. Я на собственном опыте обнаружил, что в Altium слой паяльной маски иногда исчезает у площадок со сложной геометрией. Старые версии моего дизайна содержали слой паяльной маски, но затем при сохранении файла дизайна слой молча исчез. Такой баг редок, но он случается. Обычно перед оформлением заказа я импортировал бы gerber-файл в сторонний инструмент как проверочный чертеж, но я спешил и перезаказывал существующий дизайн, который уже работал раньше, поэтому пропустил процедуру проверочного чертежа.

Результат? РСВ стоимостью в тысячи долларов пришлось списать, и мы потеряли четыре недели графика. Больно.

Хорошо, что я заложил запас в сроки доставки, и что я держу под рукой бутылку хорошего скотча, чтобы горькие напоминания о том, что случается, когда я расслабляюсь, проходили немного легче.

Если компонент можно установить неправильно, его установят неправильно

Я параноидально отношусь к неправильной установке деталей, потому что эта проблема много раз меня обжигала. Лист наклеек Chibitronics Effects был отличным примером проблемы, которая только и ждала случая проявиться.



Наклейки эффектов Chibitronics.

Лист - это массив из четырех наклеек, которые мигают разными паттернами на LED, но в остальном идентичны. Паттерн мигания управляется программно. Пытаться вести четыре отдельных файла прошивки и загрузить каждый в нужное место в тестере - готовый кошмар. Чтобы решить эту проблему, я спроектировал наклейки так, чтобы они использовали одну и ту же прошивку. Их

поведение вместо этого задавалось значением одного внешнего резистора, которое измерялось встроенным ADC микроконтроллера при загрузке.

Моя логика была примерно такой: если у всех наклеек одинаковая прошивка, нет "неправильного способа" программировать наклейки. Верно?

К сожалению, я также спроектировал мастер-панели PCB идеально симметричными. Панели можно было загрузить в сборочный робот, повернув на π радиан, и сборочная программа выполнялась бы безупречно - за исключением того, что резисторы, задающие поведение прошивки, были бы установлены в обратном порядке относительно подписей шелкографии. Несмотря на реперные отверстия, дававшие систему отсчета, и текст на PCB на китайском и английском, однозначно задававший ориентацию, эта проблема действительно случилась. На первых образцах наклеек эффект наклейка «пульс» (heartbeat) работала как «мигание» (blinking), наклейка «мерцание» (twinkle) работала как «затухание» (fading), и наоборот.

К счастью, фабрика очень последовательно загружала платы задом наперед, что является лучшим случаем для такой проблемы. Я срочно сделал патч прошивки (тоже рискованная вещь), который переворачивал интерпретацию значений резисторов, и мне отправили новый набор образцов FedEx в Сингапур для быстрой проверки здравого смысла. Мы также собрали второе тестовое приспособление, чтобы добавить ручную перепроверку правильного поведения мигания на линии в Китае. Проблема листа Effects была решена, но при создании этого дополнительного теста мы обнаружили другую распространенную проблему.

Некоторые понятия плохо переводятся на китайский

Я написал инструкции на китайском для фабрики, чтобы описать разницу между затуханием (fading) и мерцанием (twinkling), но оказалось, что китайские переводы blink и twinkle похожи. Twinkle переводится как 闪烁 («мерцание») или 闪亮 («сверкать, искриться»), а blink - как 闪烁 («мерцать, сверкать») или 闪亮 («яркий, блестящий»).

Я всегда боюсь писать субъективные описания для тестовых операторов на китайском, и это одна из причин, по которой я стараюсь автоматизировать как можно больше тестов. Как однажды заметил один мой китайский друг, Mandarin - прекрасный язык для поэзии и искусства, но трудный для точной технической коммуникации.

Задача состояла в том, чтобы придумать надежное, межкультурное объяснение разницы между затуханием и мерцанием, используя только простые термины, которые мог бы понять кто угодно; то есть мне пришлось избегать технических терминов вроде «случайный», «частота», «герц» и «периодический».

Я отправил фабрике видео разных LED-паттернов, и наша фабрика рекомендовала использовать 渐变 («плавное изменение») для затухания и 闪烁 («мерцание») для мерцания. Я все еще не уверен, что это было безупречное описание, но оно было лучше любого перевода, который я мог придумать. И по сей день нас преследуют проблемы, когда мы пытаемся объяснить персоналу контроля качества разницу между этими эффектами. Оказывается, неисправная наклейка тоже какое-то время производит довольно хороший эффект мерцания.

Довольно забавно, но для Jie и меня тоже было трудно договориться, как должен выглядеть эффект «мерцания» (twinkle). Она описала нашу первую итерацию эффекта как «ближе к грозе, чем к мерцанию». У нас было несколько долгих разговоров на эту тему, за которыми последовали демо-видео, чтобы прояснить желаемый эффект. Мы фактически крутили код, пока это не стало выглядеть примерно правильно для нас обоих. Учитывая трудность, с которой мы описывали эффект друг другу, неудивительно, что у меня были проблемы с точным описанием эффекта на китайском.

Устраняйте единые точки отказа

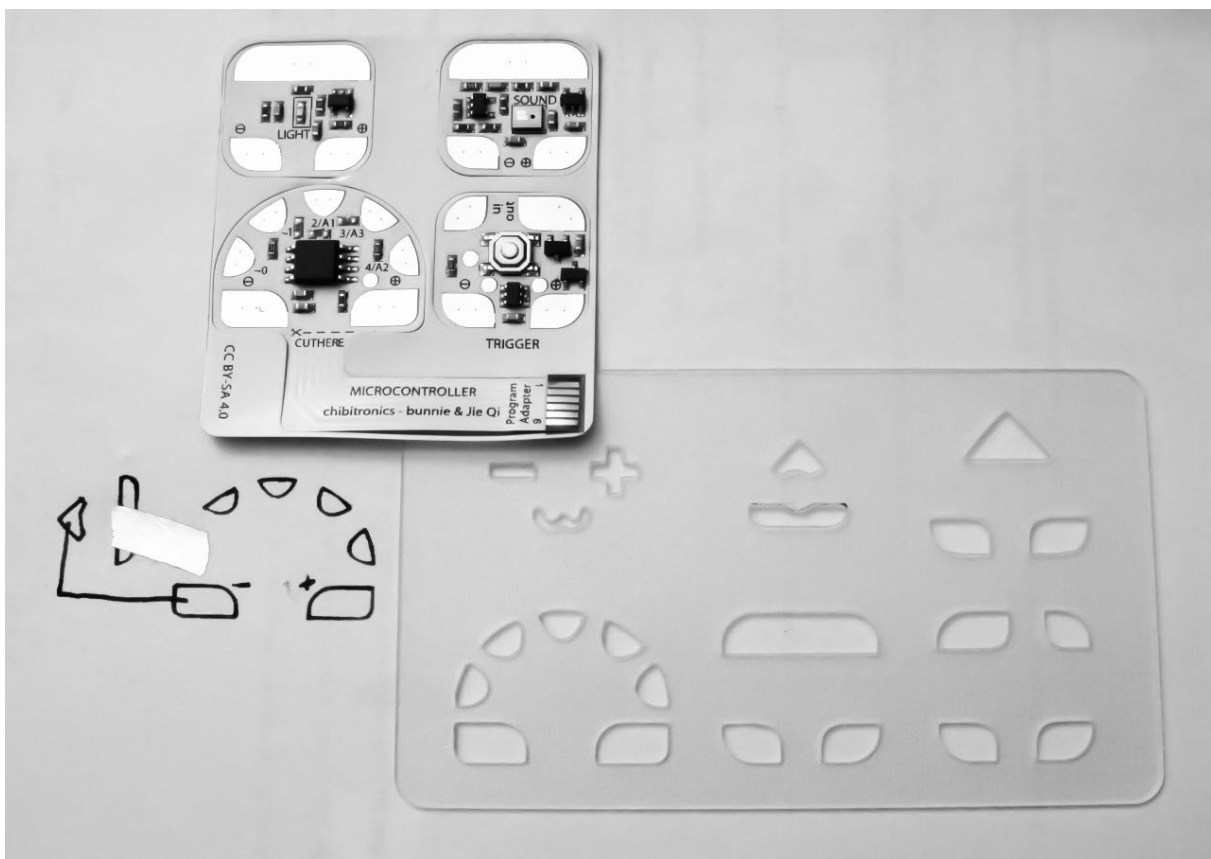
Когда мы собирали тестовые приспособления, мы сделали по две копии каждого, хотя требования по пропускной способности требовали только одну. Почему? Потому что одно могло отказать.

И угадайте что: одно тестовое приспособление действительно отказало. Я до сих пор не знаю почему. Слава богу, что мы сделали две копии; иначе мне пришлось бы срочно лететь в Китай, чтобы диагностировать, почему наше единственное тестовое приспособление не работает.

Некоторые изменения в последний момент того стоят

Примерно за шесть недель до того, как мы окончательно оформили заказ наборов Chibitronics у фабрики, Jie предложила включить трафарет паттернов наклеек в наборы с датчиками и микроконтроллерами. Она рассудила, что пользователям может быть трудно раскладывать паттерны медной ленты для сложных наклеек вроде микроконтроллера с семью площадками без рисунка контактных паттернов. Сначала я сопротивлялся идее; я не хотел задерживать отгрузку из-за того, чего мы изначально не обещали. Как обнаружила Jie, я могу быть очень вспыльчивым, особенно когда речь идет о срывах графика. (Прости, Jie! Спасибо, что терпела меня.)

Но ее аргументы были здравыми, поэтому я поручил нашей фабрике найти поставщика трафаретов. Через две недели мы не нашли никого, готового взяться за работу, но отдел снабжения фабрики не сдался. В конце концов они нашли одного поставщика, у которого было достаточно материала на складе, чтобы оснастить высечной пресс и изготовить пару тысяч трафаретов за две недели - едва-едва вовремя, чтобы уложиться в график.



Лист с датчиком и микроконтроллером, а также трафарет.

Когда я получил образцы набора с датчиком и микроконтроллером вместе с трафаретами, я их попробовал. Jie была совершенно права насчет их полезности. Я обнаружил, что мой опыт резко улучшился, когда у меня был шаблон для работы, особенно для наклейки микроконтроллера с семью близко расположенными площадками, и почувствовал, что пользователи согласятся. Поэтому, хотя трафарет не был обещан как часть исходной кампании, все спонсоры, заказавшие набор с датчиком и микроконтроллером, получили бесплатный трафарет, чтобы помочь им раскладывать дизайны.

Китайский Новый год влияет на цепочку поставок

Хотя Китайский Новый год - двухнедельный праздник, наш исходный график фактически списывал весь февраль. Реальность совпала с этим ожиданием, но я хочу точно рассказать, как Китайский Новый год повлиял на этот проект, на случай если вы рассматриваете производство продукта в Китае.

У нас была черновая рукопись книги, готовая в январе, но я не мог получить полный образец до марта. Не потому, что типография была закрыта целый месяц подряд; как и у всех остальных, их праздник длился около двух недель. Однако поставщик бумаги начал свой отпуск примерно за 10 дней до типографии, а поставщик переплета закончил отпуск примерно через 10 дней после типографии. Хотя каждый поставщик отдыхал только две недели, итоговая цепочка поставок для печати нестандартной книги выпала примерно на 24 дня, или фактически на весь февраль. Разнесенное празднование Китайского Нового года необходимо из-за огромного масштаба человеческой миграции, сопровождающей праздник.

Доставка дорогая и сложная

Когда я сделал первоначальные расчеты доставки, я понял, что мы продаем не совсем наклейки-схемы: если учитывать книгу, по объему и весу нашим основным продуктом была печатная бумага. Чтобы оптимизировать логистическую стоимость, я настаивал на отправке стартовых наборов (которые содержали книгу) и дополнительных отдельных заказов книг морем, а не авиа.

На самом деле стартовые наборы и книги были готовы почти за четыре недели до отправки первых наборов, но мы просто не могли получить разумную оценку стоимости морской перевозки. Мы потратили почти три недели на торг и запросы цен у морских грузовых компаний. В итоге их цена была практически такой же, как при авиадоставке, но доставка заняла бы на три недели больше и несла бы больше риска. Стоимость самого фрахта, по-видимому, является небольшой частью морской доставки; вас убивает множество надбавок, от оплаты портовых грузчиков до оплаты всех промежуточных брокеров и складов, которые обрабатывают ваши товары в порту. Эти фиксированные расходы сложились так, что, хотя мы отправляли больше 60 коробок товаров, авиадоставка все равно была экономичнее.

Примечание. Для справки, 40-футовый морской контейнер Maersk вместил бы больше 1 250 коробок, каждая из которых содержала 40 стартовых наборов. Нам не хватало примерно порядка объема, чтобы эффективно использовать морскую перевозку.

Вы не выбрались из леса, пока не отгрузили продукт

На каждом этапе этого проекта мне приходилось напоминать себе не считать цыплят до того, как они вылупились. Проблемы от обычной ошибки UPS до трагической авиационной аварии, логистической проблемы на складе обработки заказов Crowd Supply или таможенной проблемы могли сорвать своевременную доставку. Но, по крайней мере, мы сделали все, что было в наших силах, чтобы доставить вовремя.

К счастью, когда все было сказано и сделано, наши спонсоры получили свои заказы точно в срок. С тех пор Chibitronics продолжила превосходить мои самые смелые ожидания. Хотя мы начали этот проект как академический эксперимент, интерес пользователей снизу подтолкнул нас превратить эксперимент в полноценную компанию. Поскольку наклейки-схемы - проект открытого железа, спецификации доступны опытным хакерам для экспериментов, но большинство пользователей - люди без технической подготовки, которым напрямую полезнее поддержка по базовому использованию. Для этого компания стремится предоставлять пользователям помощь, задания и новые наклейки, чтобы они могли продолжать учиться и создавать красивые электронные поделки.

Заключительные мысли

Chibitronics стал для меня непрерывным учебным опытом, поскольку у меня никогда не было компании, которая успешно росла бы таким образом. Мне интересно увидеть, как будет развиваться компания, но как инженер я также знаю свои ограничения: я не создан для роли бизнесмена. Когда компания станет достаточно большой, чтобы устойчиво содержать собственный штат, я с нетерпением жду возможности передать руководство, вернуться к своему рабочему столу и придумывать новые изобретения в области открытого железа.

Часть 4. Взгляд хакера

Глава 9. Аппаратный хакинг

Самый большой барьер для хакинга часто заключается в страхе, что вы что-нибудь сломаете, пока будете ковыряться. Но нельзя приготовить омлет, не разбив яйца; точно так же нужно быть готовым пожертвовать устройствами, чтобы взломать систему. К счастью, получить несколько экземпляров массово выпускаемой аппаратной вещи легко. Я часто немного роюсь в мусоре или просматриваю объявления, чтобы получить образцы устройств для исследования. Обычно я стараюсь начать с трех экземпляров: один - чтобы разобрать и никогда не собрать обратно, один - чтобы щупать зондами, и один - чтобы держать относительно нетронутым. Нетронутый экземпляр я использую для контрольной проверки: вызвано ли определенное поведение моим зондированием или же это просто то, как ведет себя железо.

Мой типичный подход к любому аппаратному хаку состоит в том, чтобы сначала открыть устройство, а затем поместить зонд ровно в нужную точку, не влияя на функциональность устройства. Когда вы смотрите внутрь компьютерных чипов, это практически вся задача. Первый хак в этой главе - пример кремниевого хакинга, и вы увидите, что как только корпус снят и вы смотрите на голый кремний, атакующий получает глубокое преимущество.

Некоторые аппаратные хаки требуют большего системного инжиниринга, особенно когда вы хотите выполнить реверс-инжиниринг устройства и перепрофилировать его. В таких ситуациях я склонен разрабатывать дополнительные специализированные инструменты, которые позволяют мне настраивать систему и наблюдать за ней почти в реальном времени или по крайней мере так быстро, как я могу вводить команды, чтобы минимизировать время, затрачиваемое на проверку гипотез. Цель - сделать главным ограничением скорость, с которой вы можете придумывать идеи для проверки, а не то, сколько времени занимает загрузка изменения для проверки этих идей. Второй хак в этой главе рассказывает о реверс-инжиниринге относительно простого устройства System-on-Chip (SoC), находящегося внутри обычных карт памяти SD, и о некоторых инструментах, которые я разработал, чтобы помочь этому процессу.

Наконец, некоторые хаки неизбежно раздвигают границы закона. Третий хак в этой главе рассказывает о NeTV, системе, которую я разработал и которая по-новому смотрит на стандарт шифрования High-Definition Content Protection (HDCP), защищающий большинство HDMI-видеолинков. NeTV - это хак и юридической проблемы, и аппаратной системы. Он обходит колючие проблемы, представленные DMCA, переосмысливая стандарт HDCP так, чтобы сделать возможной атаку man-in-the-middle (MITM), меняющую видеоданные без обхода шифрования. Нет обхода - нет проблемы DMCA. Хаки часто раздвигают границу того, что законно и что было проверено в судах. Как и любую другую систему, правовую систему тоже можно хакать, и один из ключевых выводов этой главы - как думать о законах как о еще одном ограничении, с которым нужно работать на пути к достижению конкретной цели.

Последний хак в этой главе объединяет аппаратное проникновение, создание инструментов и юридические соображения, чтобы выполнить реверс-инжиниринг сложного SoC мобильного телефона. Это еще один проект, над которым я работал с hobs, и снова построение специализированных инструментов для хакинга было бесценным, потому что позволило нам экспериментировать с системой во время ее работы.

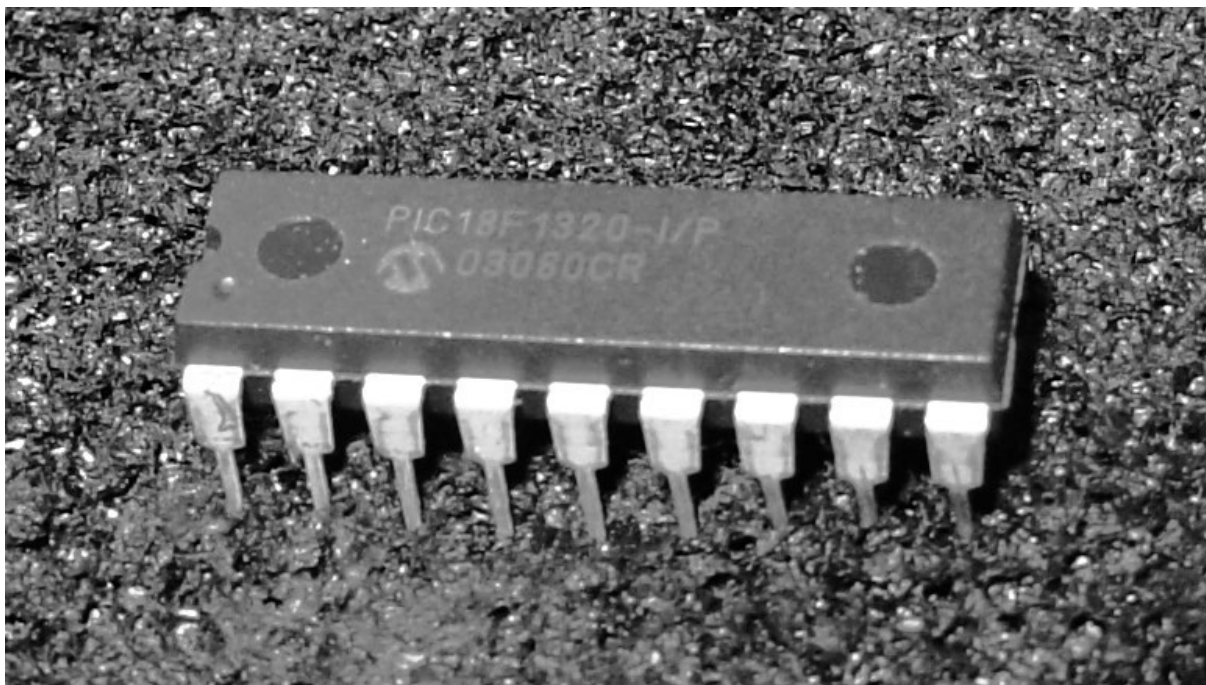
Взлом PIC18F1320

Сохранение секрета - обычная задача для любой системы безопасности. Чтобы решить эту задачу, проектировщики систем безопасности часто прячут секреты внутри кремниевых чипов, потому что прочные эпоксидные корпуса чипов и крошечные геометрические размеры трудно пробить и исследовать.

В теории это звучит хорошо, но на практике проблематично. Проектировщики чипов ошибаются, а когда у чипа есть проблема, проектировщикам нужен способ открыть его и исследовать. Такая ситуация настолько обычна, что существуют коммерческие сервисы, специализирующиеся на открытии чипов специально для этой цели. Эти сервисы, называемые сервисами анализа отказов, освоили несколько техник удаления жесткой эпоксидной смолы с чипов.

За пару лет до моего ускоренного курса по настройке китайской цепочки поставок с Chumby я решил, что будет забавно продемонстрировать, насколько простым может быть взлом чипа, если вы знаете о сервисах анализа отказов. В то время серия микроконтроллеров PIC от Microchip была весьма распространена, поэтому я решил попробовать популярную модель PIC. PIC обычно имеют конфигурационные fuse-ы, которые можно активировать, чтобы запретить чтение или запись определенных областей памяти. Но часто существует законная необходимость прочитать содержимое защищенного, запрограммированного PIC. Например, компания, потерявшая либо документацию на продукт, либо сотрудников, которые изначально создали код для защищенного PIC, оказалась бы без способа прочитать чип. Это проблема, когда компании нужно пересмотреть или обновить устаревшую линейку продуктов.

Я хотел понять, как снять дамп памяти с защищенного PIC. Зная, что мне придется разбить несколько яиц для этого омлета, я раздобыл у друга четыре PIC18F1320 и начал их раздевать. Вот что я обнаружил.

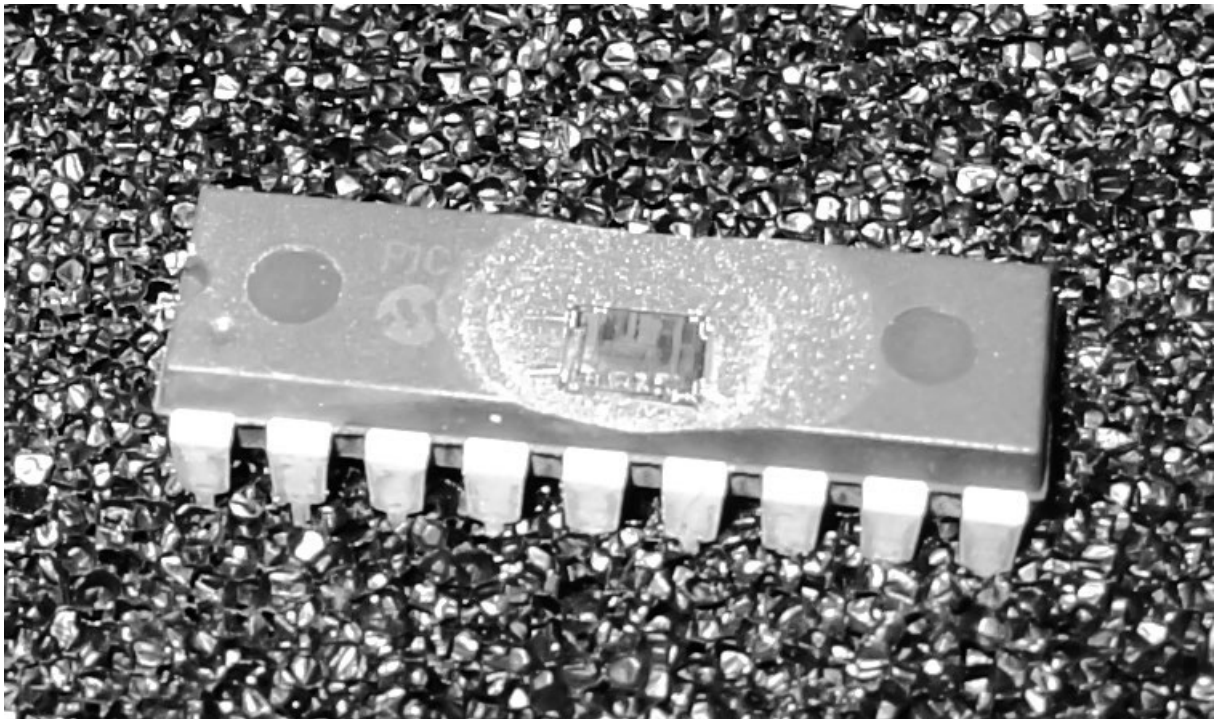


PIC18F1320 в исходном состоянии.

Декапсуляция IC

Сначала мне нужно было снять верх, чтобы увидеть кремний под капотом. Многие самодельные техники декапсуляции чипа обычно включают применение дымящей азотной или серной кислоты, но это не те соединения, которые вы захотели бы держать дома, и достать их тоже нелегко. Азотная кислота, в частности, является важным соединением для изготовления взрывчатых веществ. Поэтому я обнаружил, что самый простой и надежный способ декапсулировать чип - просто отправить его в лабораторию анализа отказов. Примерно за \$50 вы можете получить декапсулированную деталь за два дня.

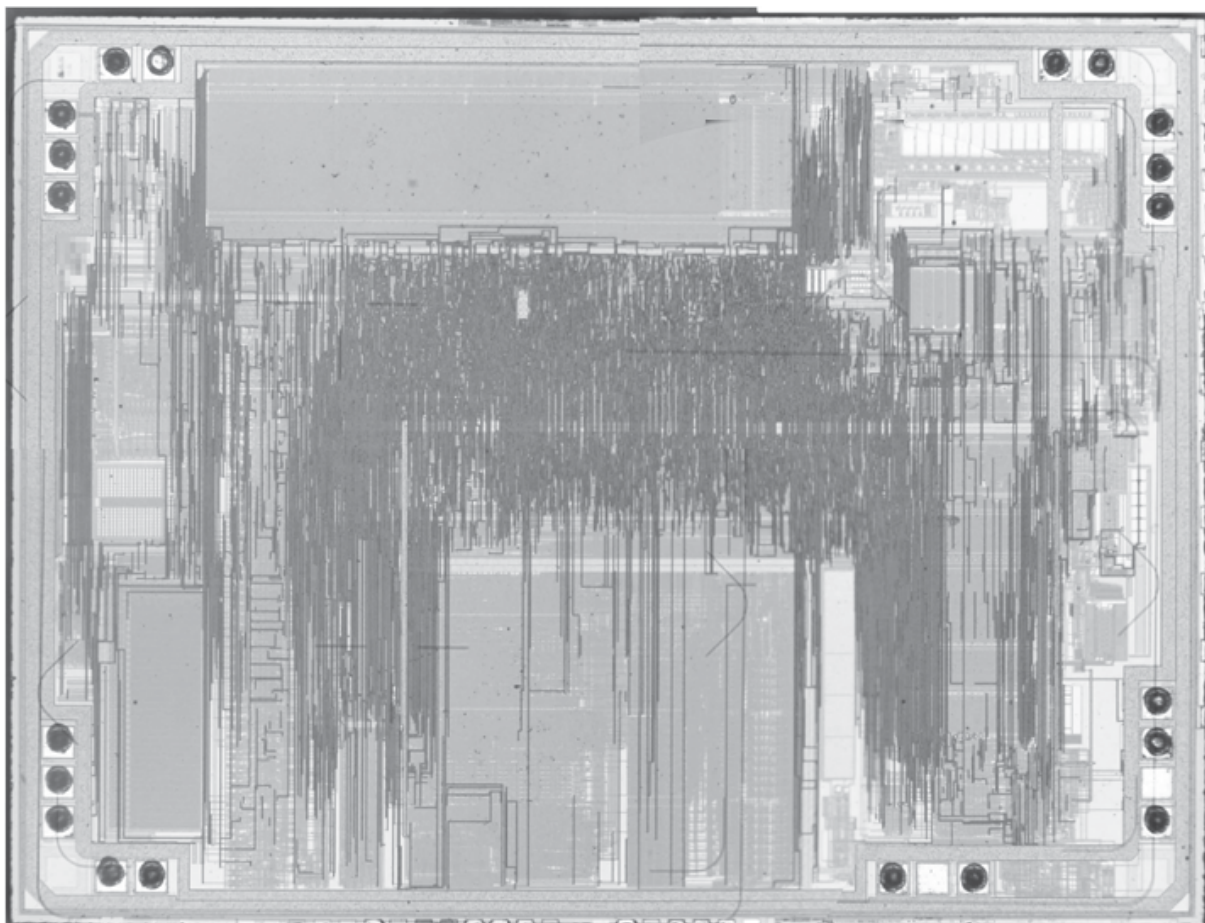
Для этого проекта я декапсулировал три детали. Две были функционально декапсулированы (кремний открыт, устройство все еще в своей выводной рамке и полностью работоспособно), а последняя была полностью декапсулирована (просто голый кремниевый кристалл без корпуса). Один кристалл я полностью декапсулировал потому, что у моего инспекционного микроскопа было очень короткое рабочее расстояние на максимальных увеличениях, и оставшаяся от корпуса эпоксидная смола мешала бы объективу.



Функционально декапсулированный PIC18F1320. Маленький приподнятый квадрат посередине (в реальности он золотистый) - это кремниевый чип.

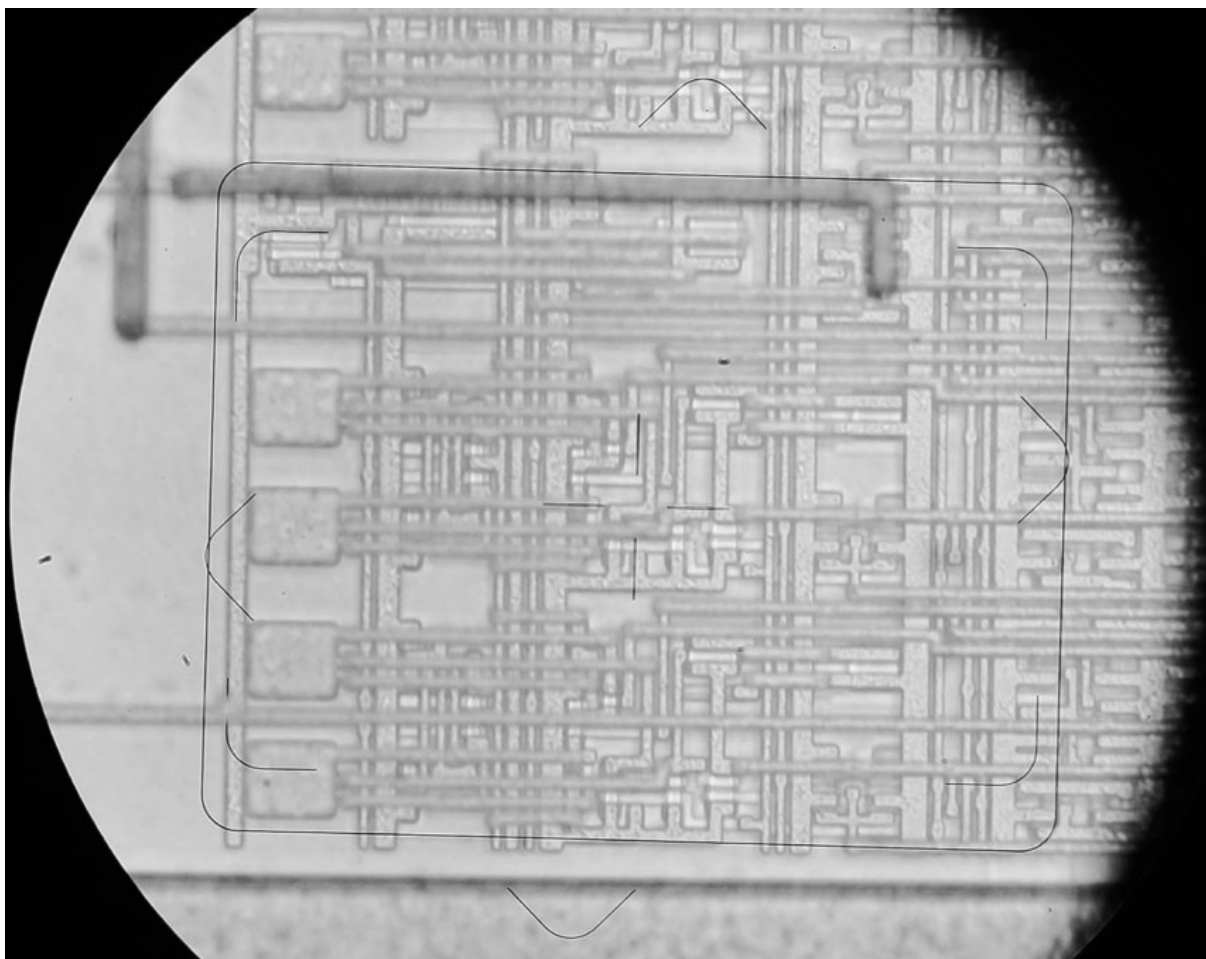
Более пристальный взгляд

Имея на руках декапсулированные IC, я прошелся микроскопом по одному из кристаллов и заметил несколько заметных особенностей. Поскольку физика везде одинакова, большая часть мелкозернистой структуры кремниевого чипа выглядит почти одинаково независимо от того, кто изготовил чип. Эти ограничения распространяются вверх, до системного уровня, и после небольшой тренировки вы можете читать кремниевый чип как книгу.



Моя лучшая догадка о том, что делают различные структуры в этом чипе. Я могу ошибаться.

Один набор структур сразу привлек мое внимание: над некоторыми транзисторами были металлические экраны, и они следовали регулярному шаблону, в котором было примерно правильное количество устройств, чтобы соответствовать всем биты защиты. Полностью металлические экраны, закрывающие устройство, очень редки в кремнии, поэтому они похожи на большой крест, отмечающий место, где хранится что-то очень важное.



Металлические экраны крупным планом.

Стирание флеш-памяти

Экраны были значимы из-за некоторых интересных фактов о технологии flash-памяти, которую это PIC-устройство использовало для хранения состояния защитных предохранителей, а также внутреннего программного кода. Flash-технология использует транзисторную структуру с плавающим затвором, очень похожую на старые технологии ультрафиолетово-стираемой программируемой постоянной памяти (UV-EPROM), например керамические чипы 2716 из 1970-х, у которых были кварцевые окна, чтобы их можно было стирать.

И в flash, и в UV-EPROM-устройствах данные записываются, когда электроны туннелируют в плавающий затвор, где они остаются десятилетиями. Дополнительные электроны в плавающем затворе создают измеримое смещение в характеристиках запоминающего транзистора. Разница в том, что flash-память может вывести сохраненные электроны (стереть устройство), используя только электрические импульсы, тогда как UV-EPROM требует энергичных фотонов, чтобы выбить электроны из плавающего затвора. Ультрафиолетовый свет, необходимый для этого, обычно имеет длину волны около 250 нм. Нужна дорогая кварцевая оптика, чтобы управлять этой длиной волны UV без чрезмерных потерь, что делает ее несколько трудной в использовании.

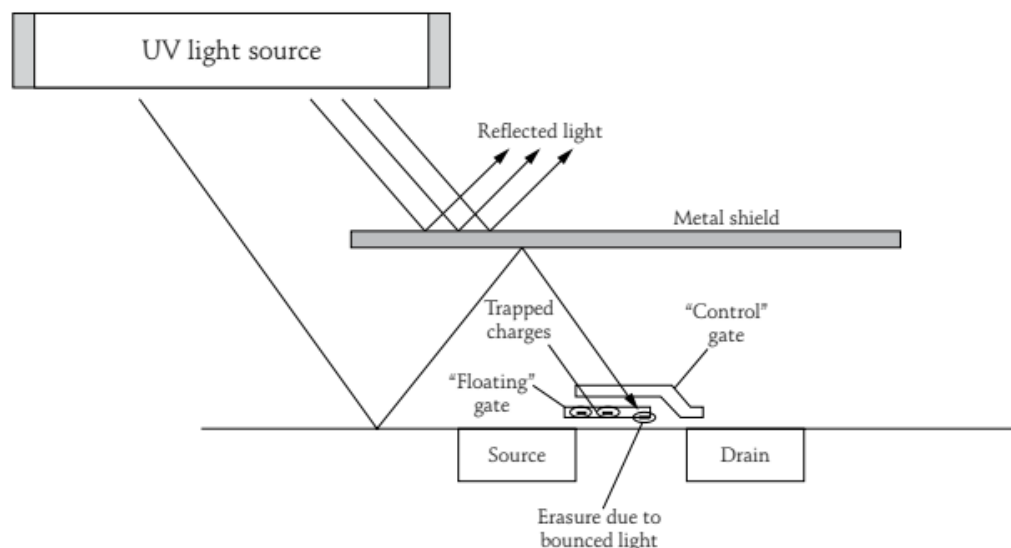
Вот важный вывод, который я сделал из этих фактов: flash-устройства обычно тоже можно стирать с помощью UV-света, поскольку они имеют транзисторную структуру, похожую на UV-EPROM-устройства. Капсулирование вокруг flash-устройства обычно мешает любому UV-свету эффективно добраться до кристалла, но поскольку с PIC-устройств был снят пластик, я мог попробовать применить UV-свет и посмотреть, что произойдет.

Я провел простой эксперимент, запрограммировав PIC-устройство возрастающим шаблоном, где снова и снова сохранял шестнадцатеричные числа от 0x00 до 0xFF. Затем я бросил PIC в свой UV-EEPROM-стиратель, чтобы запечь его... ну, примерно на время хорошего долгого душа и проверки электронной почты. Когда я вынул устройство из стирателя, flash-память действительно была очищена до своего нормального состояния "все 1", а предохранители защиты не пострадали. После запекания еще нескольких PIC-устройств в стирателе я обнаружил, что если не запекать PIC достаточно долго, то получаю странные чтения из массива, в который записывал данные, например все 0, - явление, которое я до сих пор не понимаю.

Стирание биты защиты

Очевидно, металлические экраны над предохранители защиты были там, чтобы пресечь попытки выборочно стереть предохранители защиты, оставив массив flash-памяти нетронутым.

286 CHAPTER 9



A diagram showing how the shields got in the way of the fuse bits, and how to work around them

My problem was that for the flash memory transistor to be erased, high-intensity UV light needed to strike the floating gate. The metal shield effectively reflected all incident light, so the

Figure 9-11. The Problem of Erasing the Flash Memory Transistor

Диаграмма, показывающая, как экраны мешали битам защиты и как их обойти.

Моя проблема заключалась в том, что для стирания транзистора flash-памяти высокоинтенсивный UV-свет должен был попасть на плавающий затвор. Металлический экран фактически отражал весь падающий свет, поэтому свет никогда не достигал затвора. Но я знал, что между оптически прозрачным защитным диэлектрическим слоем диоксида кремния, покрывающим чип, и собственно кремнием есть несоответствие показателей преломления, а это означает, что свет под определенными углами будет отражаться от гладкой кремниевой поверхности. Чтобы увидеть пример этого отражательного эффекта, прыгните в бассейн, погрузитесь под воду и посмотрите вверх, туда, где вода встречается с воздухом. Под косым углом вода должна выглядеть сильно

отражающей, потому что несоответствие показателей преломления между водой и воздухом вызывает полное внутреннее отражение света.

Я планировал использовать это отражение, чтобы направить UV-свет от оксида в металлический экран и отразить его обратно на плавающий затвор. Наклонив PIC внутри ROM-стирателя, я предположил, что смогу получить достаточно света, отраженного в область транзисторов flash-памяти, и стереть биты защиты. После пары попыток с использованием всяких кусочков материала, чтобы зафиксировать угол чипа, я разработал простую технику, которая сработала удивительно хорошо: засунуть чип под углом в антистатическую пенную подкладку UV-стирателя.

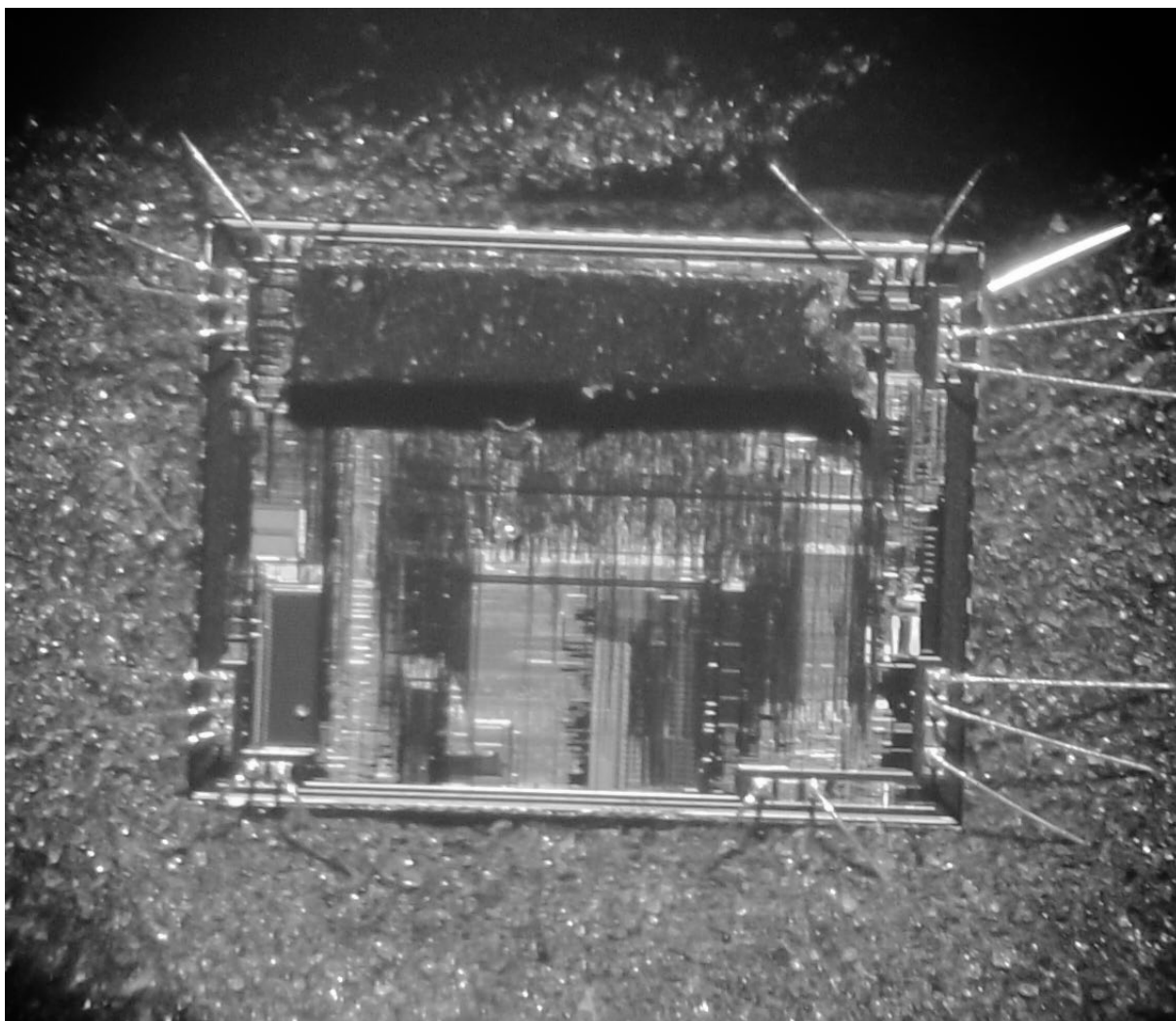


Чип в антистатической пене UV-стирателя.

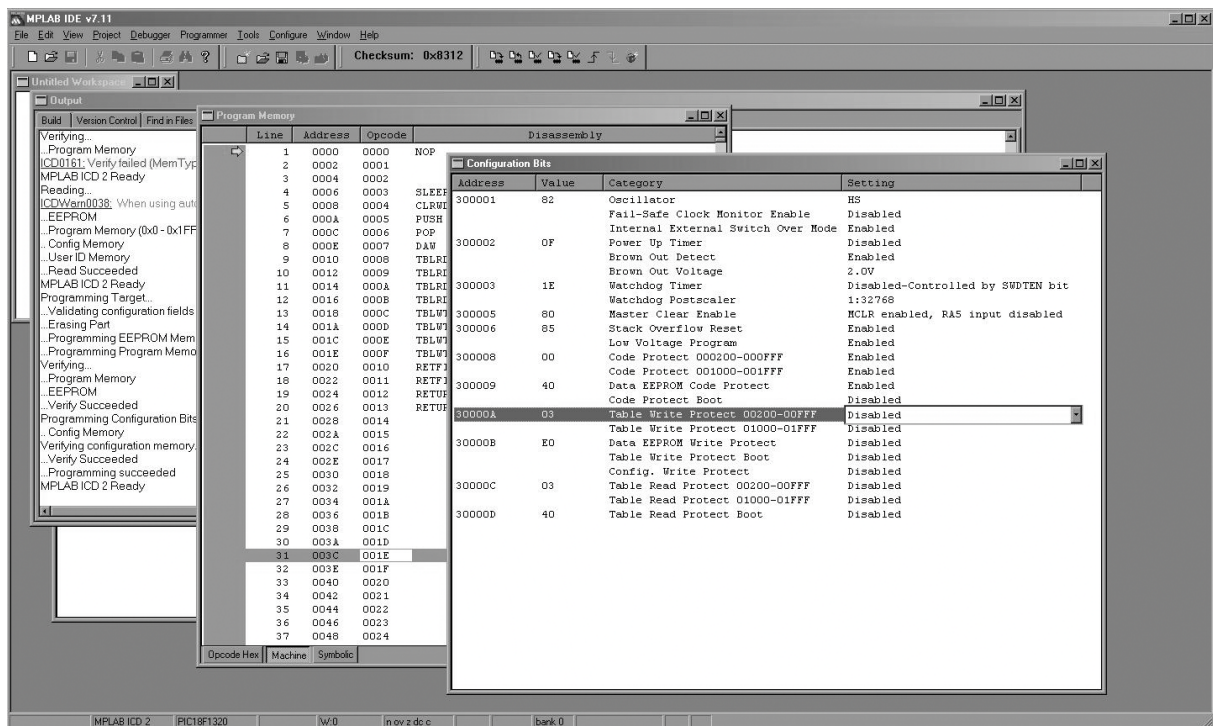
Защита остальных данных

Однако эта техника не защищала flash-данные, которые я хотел сохранить. Чтобы избежать стирания этих данных, я сделал жесткую маску из очень аккуратно вырезанного куска изоленды и приклеил эту маску к поверхности кристалла, используя твердую руку, два пинцета и микроскоп. Изоленда блокировала прямое попадание UV-света на области flash-памяти кода и частично поглощала свет, отраженный обратно от кремниевой подложки.

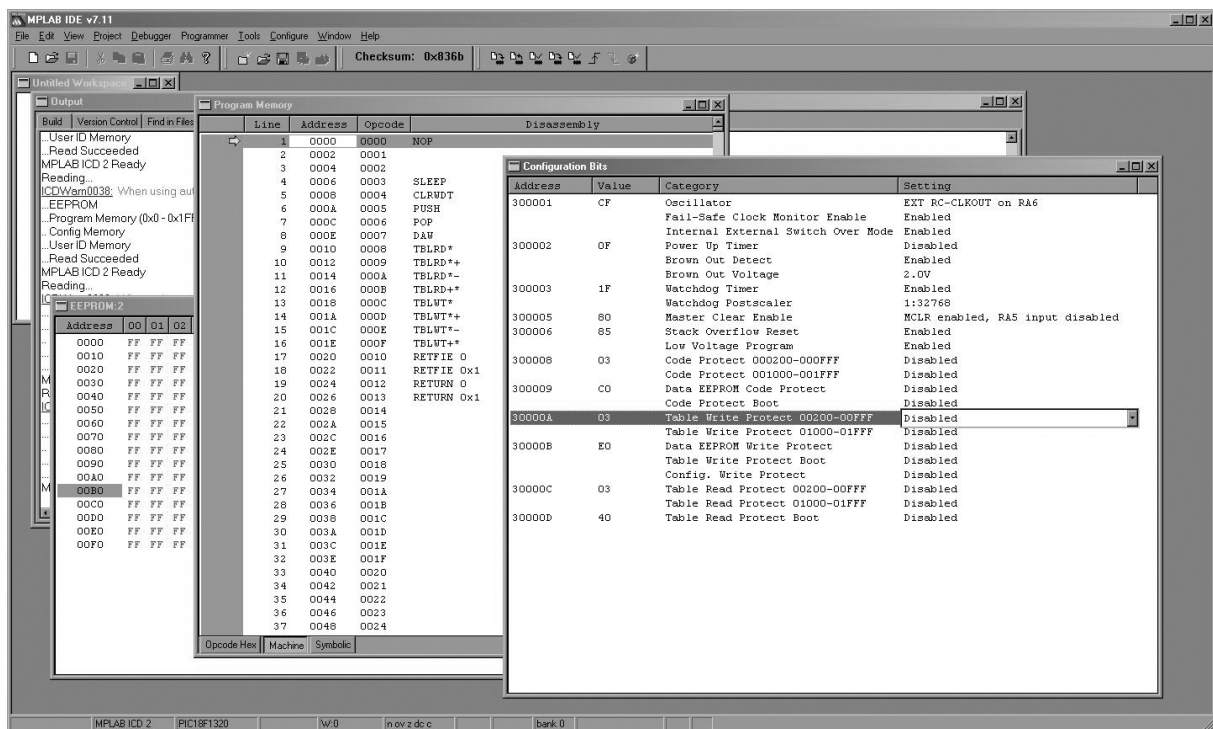
Кристалл в своем корпусе, с изолентой над массивом flash ROM.



Эта маска позволила мне сбросить только предохранители защиты, не слишком сильно затронув массив flash-кода. Следующие снимки экрана показывают состояние памяти массива согласно инструменту программирования и обратного чтения, которым я пользовался.



Мое рабочее пространство PIC-программатора, показывающее настройки устройства до стирания.



Настройки устройства после стирания.

На снимке "до" обратите внимание на настройки предохранителей защиты в окне Configuration Bits и на значения, запрограммированные во flash ROM, показанные в окне Program Memory. На снимке "после" предохранители защиты переключаются в отключенное состояние, тогда как содержимое flash ROM в окне Program Memory читается идентично тому, что было запрограммировано ранее. Другая часть массива кода на самом деле все же была стерта, но я, вероятно, мог бы исправить это, вырезав кусок изоляции побольше.

Я слышал сообщения, что после публикации этого хака Microchip начала ставить металлические экраны не только над fuse-битами, но и над массивом памяти кода, что сделало выполнение этого трюка несколько более трудным. И все же этот хак подчеркивает факт, что довольно часто самая трудная часть кремниевого хакинга - удаление внешнего корпуса, и, к счастью, существуют дешевые, пусть и малоизвестные, сервисы, которые помогают с этой задачей.

Взлом SD-карт

Годы спустя я обнаружил, что взламываю еще одно интересное устройство с flash-памятью: SD-карту. Я уже разбирал SD-карты, когда исследовал партию потенциально поддельных карт, попавших в производственные устройства Chumby; это я обсуждаю в разделе "Поддельные microSD-карты" на странице 156. На этот раз мое намерение было понять, как заставить SD-карту делать то, для чего она не была создана. Этот конкретный хак был еще одной командной работой с моим другом hobs, и он финансировался инициативой DARPA Cyber Fast Track (CFT). Детище uberhacker-а .mudge (одного из первоначальной команды L0pht), CFT было хаком правительства США, чтобы сделать его умнее в вопросах инноваций, особенно в делах, связанных с интернет-безопасностью. Мы провернули это примерно в то же время, когда работали над Novena, а я сотрудничал с Jie Qi над Chibitronics.

Мы с hobs обнаружили, что некоторые SD-карты содержат уязвимости, позволяющие выполнять произвольный код на самих картах памяти. Мы также обнаружили, что похожие классы уязвимостей существуют в родственных устройствах вроде USB-флеш-накопителей и твердотельных дисков. С темной стороны, выполнение кода на карте памяти делает возможными MITM-атаки, при которых карта кажется ведущей себя одним образом, но на самом деле делает что-то другое, пока атакующий перехватывает и манипулирует обменом между картой и устройством, которое ее использует. Со светлой стороны, однако, эта уязвимость также дает энтузиастам железа доступ к очень дешевому и повсеместному источнику микроконтроллеров.



Some of the eggs—or rather, SD cards—we cracked open to find the vulnerability

How SD Cards Work

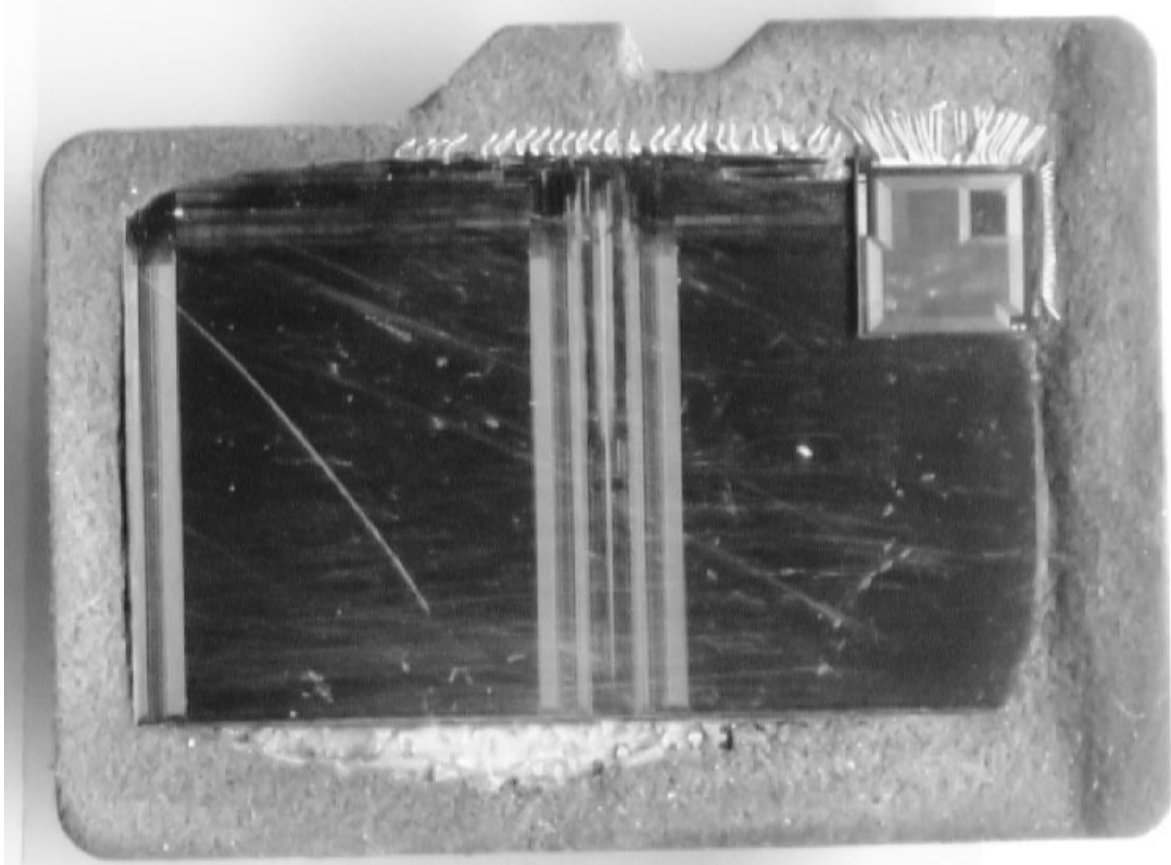
Некоторые из яиц - или, точнее, SD-карт, - которые мы раскололи, чтобы найти уязвимость.

Как работают SD-карты

Чтобы понять хак, нужно знать, как устроены SD-карты. Информация, которую я собираюсь объяснить, относится ко всем устройствам с управляемой flash-памятью, включая microSD, SD и MMC, а также устройства eMMC и iNAND, обычно припаянные к материнским платам смартфонов для хранения операционной системы и других приватных пользовательских данных.

Flash-память рекламируют как сплошной, надежный носитель данных, и она действительно дешевая - настолько дешевая, что первоначальное обещание буквально слишком хорошо, чтобы быть правдой. В реальности вся flash-память без исключения испещрена дефектами. Она создает впечатление надежности с помощью сложных функций коррекции ошибок и управления плохими блоками. Эта система - результат постоянной гонки вооружений между инженерами и матерью-природой: каждый раз, когда производственный процесс уменьшает транзисторы, память становится дешевле, но менее надежной. Точно так же с каждым поколением чипов инженеры создают более сложные и запутанные алгоритмы, чтобы компенсировать склонность природы к энтропии и случайности на атомном уровне.

Эти алгоритмы слишком сложны и слишком специфичны для конкретного устройства, чтобы выполняться на уровне приложения или операционной системы, поэтому каждый flash-диск поставляется с достаточно мощным микроконтроллером, который выполняет специализированный набор алгоритмов дисковой абстракции. Даже крошечные microSD-карты содержат не один, а как минимум два чипа: контроллер и по крайней мере один flash-чип. (В картах высокой плотности несколько flash-кристаллов укладываются стопкой.)



Внутри microSD-карты. Маленький квадрат в правом верхнем углу - это микроконтроллерный SoC, установленный поверх более крупного чипа flash-памяти, которым он управляет.

По моему опыту, качество flash-чипа или чипов, встроенных в карты памяти, широко варьируется. Чип может быть чем угодно: от высокосортного, нового с фабрики кремния до материала, у

которого более 80 процентов плохих секторов. Если вас беспокоят электронные отходы, вам, возможно, будет приятно (а возможно, и нет) узнать, что поставщики карт памяти обычно используют переработанные flash-чипы, извлеченные из выброшенных деталей. Более крупные поставщики обычно предлагают более стабильное качество, но даже крупнейшие игроки твердо оставляют за собой право смешивать и сочетать чипы flash-памяти с разными контроллерами, при этом продавая сборку под тем же номером детали. Это кошмар, если вы имеете дело с ошибками, специфичными для реализации.

Встроенный микроконтроллер карты памяти часто представляет собой сильно модифицированный Intel 8051 или ARM CPU с производительностью около 100 MHz и несколькими аппаратными ускорителями на кристалле. Удивительно, но добавление этих контроллеров к карте памяти стоит всего около \$0,15-\$0,30, особенно для компаний, которые могут производить и flash-память, и контроллеры внутри одного подразделения. Еще интереснее то, что из-за высокой стоимости тестирования чипов на уровне пластины, вероятно, в итоге дешевле добавить микроконтроллер, управляющий плохими блоками, чем тщательно тестировать и характеризовать каждый необработанный чип flash-памяти. Фактически устройства с управляемой flash-памятью обычно дешевле на бит, чем необработанные flash-чипы, несмотря на дополнительную функциональность.

Каждая реализация flash-памяти имеет уникальные алгоритмические требования, умножающие число слоев аппаратной абстракции, с которыми должен справляться микроконтроллер. Эта сложность неизбежно приводит к ошибкам, а значит, просто навсегда зашить статический код во встроенную ROM невозможно, особенно для сторонних контроллеров.

Таким образом, механизм загрузки и обновления прошивки практически обязателен. Конечные пользователи редко сталкиваются с этим процессом, поскольку все это происходит на фабрике, но механизм существует. Исследуя рынки электроники в Китае, я видел, как продавцы прошивают карту прошивкой, которая "расширяет" емкость карты. Иными словами, они загружают прошивку, которая сообщает емкость карты намного большей, чем фактически доступное хранилище. Тот факт, что это возможно в точке продажи, указывает, что механизм обновления, вероятно, защищен плохо.

Реверс-инжиниринг микроконтроллера карты

Мы с hobs обнаружили пример этой уязвимости, исследуя карты памяти, использующие микроконтроллеры ArproTech AX211 и AX215. Мы обнаружили простую последовательность "knock", передаваемую через зарезервированные производителем команды (команда с именем CMD63, за которой следуют байты A, P, P, O), которая переводила контроллер в режим загрузки прошивки. После получения последовательности knock карта принимала следующие 512 байт и выполняла эти данные как код.

Примечание. Чипы ArproTech, которые я описываю здесь, технически интегрируют достаточно функциональности, чтобы в академическом смысле быть не просто микроконтроллерами; это полноценные SoC. Но мне просто странно называть ArproTech SoC, поэтому я не буду. Для меня это всегда будет микроконтроллер!

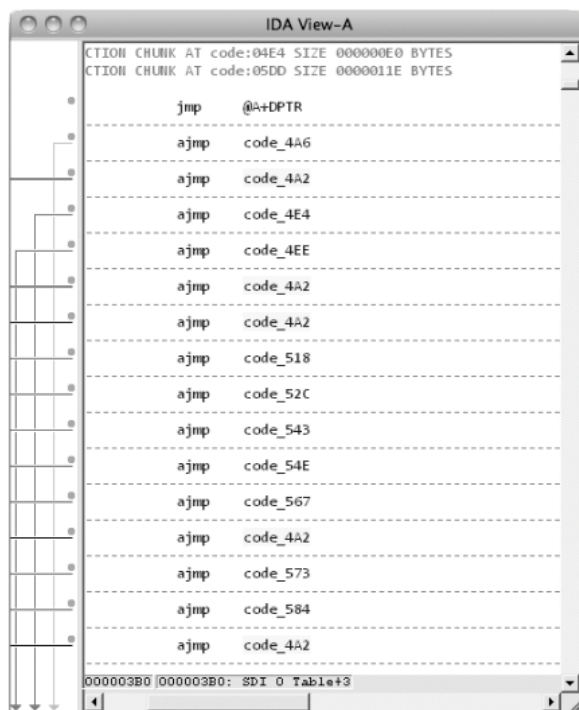
Используя последовательность knock как точку входа в систему ArproTech на этой конкретной карте памяти, мы применили сочетание анализа кода в IDA, интерактивном дизассемблере, и fuzzing (то есть подачи микроконтроллеру недопустимого или случайного ввода, чтобы увидеть, как он реагирует), чтобы выполнить реверс-инжиниринг большинства регистров 8051, отвечающих за конкретные функции. Это позволило нам разрабатывать новые приложения для контроллера без проприетарной документации производителя. Большую часть этой работы мы проделали с аппаратурой ноутбука Novena, которую я описывал в главе 7.

Как я намекал в начале этой главы, мы разработали несколько специализированных инструментов, чтобы помочь себе в реверс-инжиниринге SD-карты. Один из более интересных инструментов, которые мы (и под "мы" я имею в виду прежде всего хобс) сделали, - это интерактивная REPL (read-evaluate-print-loop) shell-оболочка для выполнения произвольного кода на SD-карте. Следующий листинг показывает, как выглядит эта среда.

```
root@bunnie-novena:~/ax211-code# ./ax211 -d debug.bin
FPGA hardware v1.26
Debug mode APP0 response [6]: {0x3f 0x00 0xc1 0x04 0x17 0xab}
Result of factory mode: 0
00000000 0f 41 1f 0f 0f 0f ff ff          |.A.....|
Expected 0x00 0x00, got 0x0f 0x41
Loaded debugger
Locating fixup hooks... Done
AX211> help
List of available commands:
  hello  Make sure the card is there
  peek   Read an area of memory
  poke   Write to an area of memory
  jump   Jump to an area of memory
  dumprom Dump all of ROM to a file
  memset Set a range of memory to a single value
  null   Do nothing and return all zeroes
  disasm Disassemble an area of memory
  ram    Manipulate internal RAM
  sfr    Manipulate special function registers
  nand   Operate on the NAND in some fashion
  extop  Execute an extended код операции on the chip
  reset  Reset the AX211 card
  help   Print this help
For more information on a specific command, type 'help [command]'
AX211> help disasm
Help for disasm:
Disassemble a number of bytes at the given offset.
Usage: disasm [address] [bytes]
AX211> disasm 0x200 16
.org 0x0200
  nop
  nop
  reti
  nop
  mov R7, A
  reti
  mov R7, A
  nop
  mov R7, A
  nop
  mov R7, A
  nop
```

Изнутри этой среды мы могли запускать программы в отладчике, получать список доступных команд и того, что они делают, вводя `help`, и дизассемблировать участки кода, вводя `disasm`. Хотя разработка интерактивного инструмента с таким богатым набором функций заняла много времени, усилия быстро окупались, потому что мы могли проверять сложные гипотезы с помощью автоматизированных fuzzing-фреймворков.

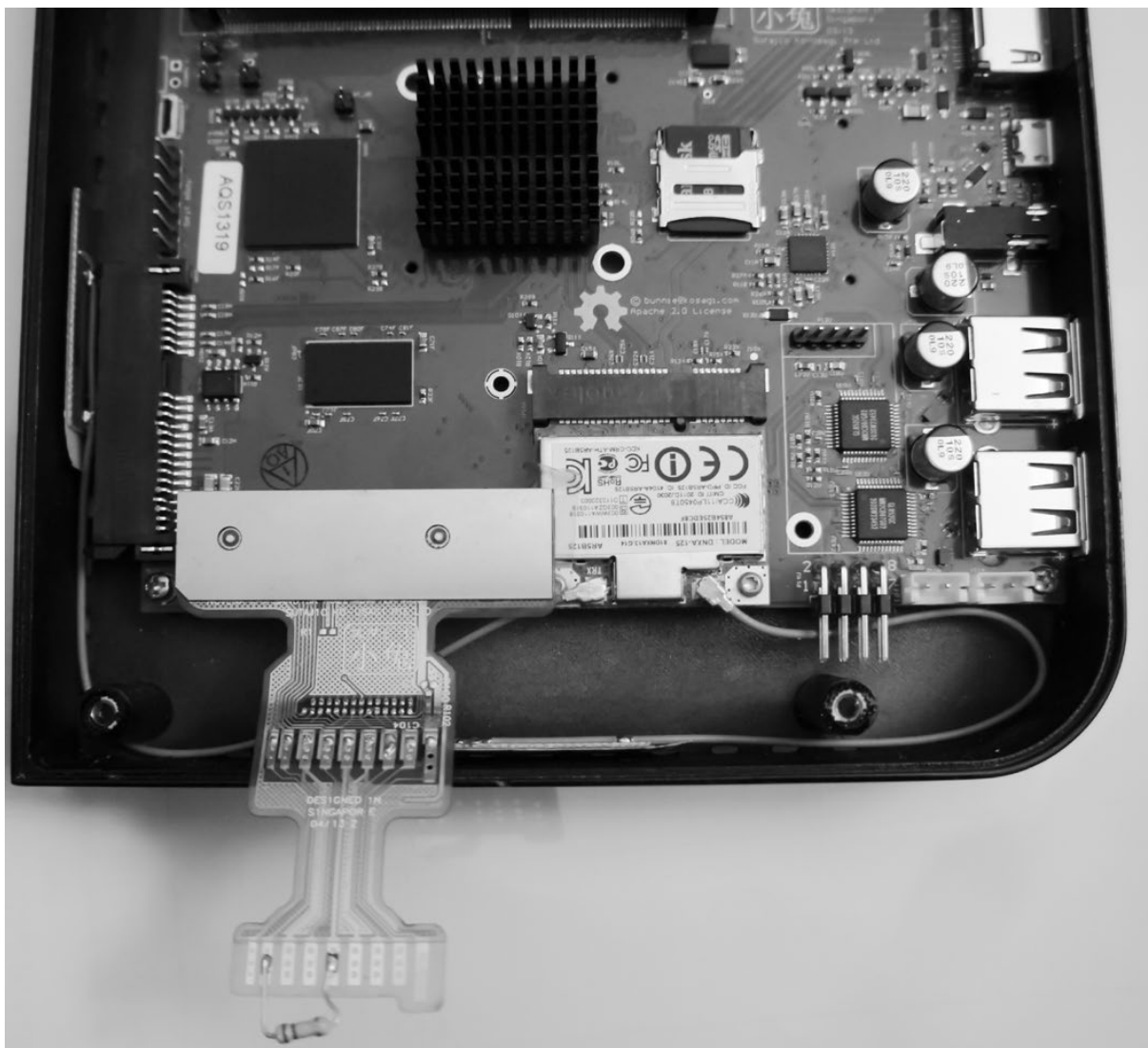
Размер загрузки кода был ограничен 512 байтами, что означало, что нам пришлось разделить REPL-среду между хост-компьютером Novena и целевым устройством.^[1] Например, дизассемблирование конкретной области памяти распадается на сценарий, выполняемый на стороне хоста, который отправляет запросы к AX211 для дампа нужной части памяти, после чего алгоритм дизассемблирования выполняется на ARM CPU хоста.



CMD INDEX	type	argument	resp	abbreviation	command description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all cards to idle state
CMD1	reserved				
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks any card to send the CID numbers on the CMD line (any card that is connected to the host will respond)
CMD3	bcr	[31:0] stuff bits	R6	SEND_RELATIVE_ADDR	Ask the card to publish a new relative address (RCA)
CMD4	bc	[31:16] DSR [15:0] stuff bits	-	SET_DSR	Programs the DSR of all cards
CMD5	reserved for I/O cards (refer to the "SDIO Card Specification")				
CMD7	ac	[31:16] RCA [15:0] stuff bits	R1b (only from the selected card)	SELECT/DESELECT_CARD	Command toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases, the card is selected by its own relative address and gets deselected by any other address; address 0 deselects all. In the case that the RCA equals 0, then the host may do one of the following: - Use other RCA number to perform card de-selection. - Re-send CMD3 to change its RCA number to other than 0 and then use CMD7 with RCA=0 for card de-selection.
CMD8	bcr	[31:12]reserved bits [11:8]supply voltage(VHS) [7:0]check pattern	R7	SEND_IF_COND	Sends SD Memory Card interface condition, which includes host supply voltage information and asks the card whether card supports voltage. Reserved bits shall be set to '0'.
CMD9	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card identification (CID) on CMD the line.
CMD11	reserved				
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission
CMD13	ac	[31:16] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14	reserved				

Разделение функций SD-отладчика между хостом и целью.

Инструмент, который мы построили, начинался с физического слоя эмуляции SD, который я буду называть PHY. Мы использовали FPGA, встроенную в Novena, чтобы представить GPIO-подобный API регистров для SD host PHY. Был один регистр для вывода данных, один регистр для ввода данных и один регистр для побитовой установки направления данных. Карта AX211 была подключена к FPGA через специальный гибкий печатный-адаптер.[2]



Адаптер на гибкой печатной схеме, подключенный к Novena.

SD-команды принимались на AX211 и обрабатывались аппаратной машиной состояний, подключенной к встроенному CPU 8051. Машина состояний обрабатывала прием данных, а также вычисляла и проверяла циклический избыточный код для обнаружения ошибок. Как только машина состояний получала полный пакет, прерывание уведомляло 8051 о прибытии пакета.

Мы захватили механизм обработки прерываний и переназначили обработчик по умолчанию на наш собственный 512-байтный кодовый заглушечный обработчик. Это позволило нам определить новый набор SD-команд, которые мы использовали для реализации callback-функций, нужных нашей REPL-среде, таких как `peek`, `poke`, `jump`, манипуляции с регистрами NAND и так далее. Эти callback-и также были идеальной зацепкой для реализации MITM-атаки.

Callback-функции для REPL, показанные в IDA.

Я не знаю, сколько еще производителей оставляют свои последовательности обновления прошивки незащищенными. ArproTech - относительно небольшой игрок в мире SD-контроллеров; горстка компаний, о которых вы, вероятно, никогда не слышали, тоже производит SD-контроллеры, включая Alcor Micro, Skymedi, Phison и SMI. Конечно, есть также SanDisk и Samsung. У каждой свои механизмы и методы загрузки и обновления прошивки. Но я знаю по крайней мере одну реализацию Samsung eMMC, использующую набор инструкций ARM, в которой была ошибка, потребовавшая отправить обновлятор прошивки на Android-устройства. Это указывает еще на одно потенциально перспективное направление для дальнейших исследований.

Потенциальные проблемы безопасности

С точки зрения безопасности наши находки указали, что хотя карты памяти выглядят инертными, они выполняют код, который можно изменить для проведения MITM-атак, трудных для обнаружения. Не существует стандартного протокола или метода, чтобы проверить и аттестовать содержимое кода, выполняющегося на микроконтроллере карты памяти. Если вы используете SD-карту в ситуации высокого риска и высокой чувствительности, не предполагайте, что выполнение команды безопасного стирания (или какого-то другого инструмента безопасного стирания) на карте гарантирует полное стирание чувствительных данных. Если вам действительно нужно, чтобы данные исчезли, я рекомендую избавиться от карты памяти путем полного физического уничтожения. Измельчите ее ступкой и пестиком, если придется.

Ресурс для любителей

С точки зрения DIY и хакеров наши находки указали на потенциально интересный источник дешевых и мощных микроконтроллеров для использования в простых проектах. Клон Arduino - с 8-битным микроконтроллером 16 MHz - обойдется примерно в \$20. Карта microSD с несколькими гигабайтами памяти и микроконтроллером в несколько раз большей производительности стоит лишь малую долю этой цены. Хотя SD-карты, надо признать, ограничены по I/O, немного умного хакинга микроконтроллера в SD-карте могло бы дать очень экономичное и компактное решение для регистрации данных с I2C- или SPI-датчиков.

Взлом HDCP-защищенных линков для пользовательских оверлей

"Это здорово, но законно ли это?" - частый вопрос, который я получаю, занимаясь хакингом. Так же как у инженерных систем есть хаки, у правовых систем есть лазейки. Некоторые юридические лазейки существуют по замыслу; другие непреднамеренны. В любом случае они могут дать жизненно важное пространство для инноваций. Обдумывая хак, я рассматриваю юридические вопросы так же, как инженерные ограничения, подобно необходимости уместить что-то в корпус определенной высоты или обеспечить работу в течение определенного времени от данной батареи.

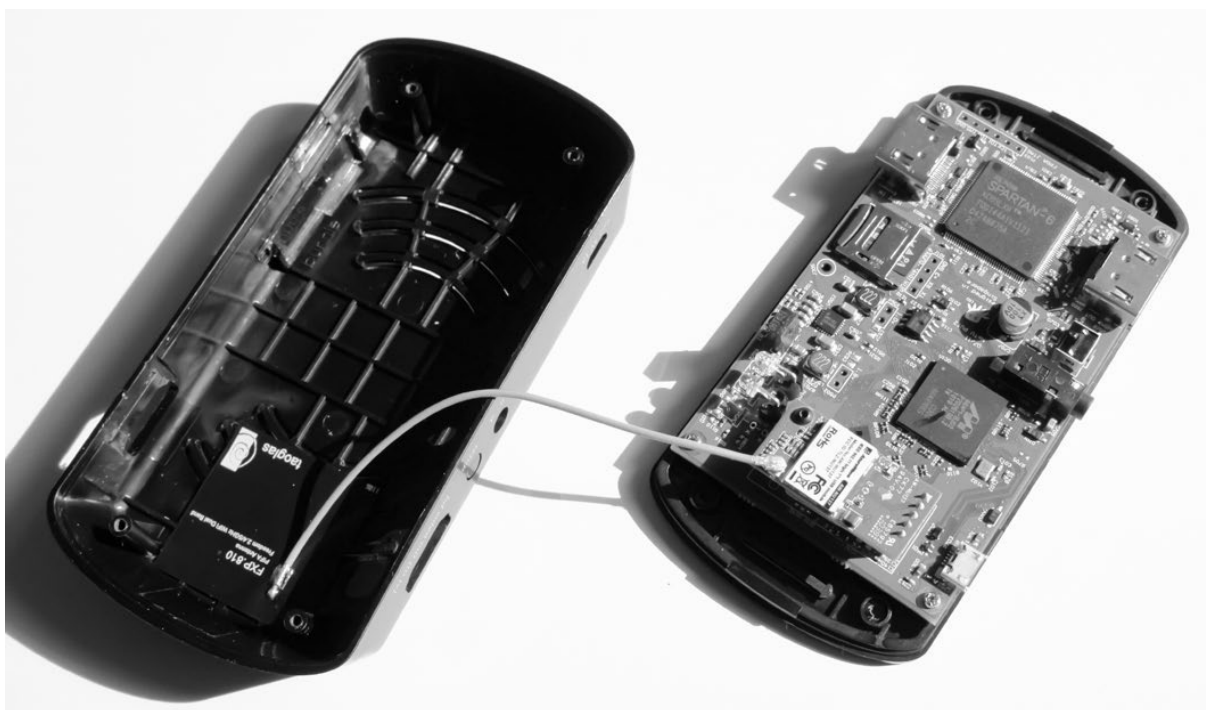
Примерно в 2011 году, когда я все еще был в Chumby, мы ломали голову над тем, как стимулировать распространение продукта, пока iPhone и Android-телефоны поглощали рыночную нишу, которую мы надеялись занять. Стоимость была вечным барьером для пользователей, а встроенный LCD в chumby был самым дорогим элементом. Наш тогдашний гендиректор, Стив Томлин, заметил, что самый большой экран в доме еще не стал подключенным к интернету каким-либо осмысленным образом. И тогда мне был поставлен вопрос: можем ли мы найти способ убить двух зайцев одним выстрелом, убрав экран из нашей ведомости материалов и одновременно приведя телевизоры в интернет-эпоху? Это было до того, как на рынке появились продукты вроде Google Chromecast или Logitech Revue.

Нам пришло в голову, что можно упаковать дешевый компьютер в "стик", подключающийся к HDMI-порту. Это решает проблему вывода chumby на экран телевизора, но тогда, когда выбран chumby, вы не смотрите любимые фильмы или телепередачи. Мы решили, что людям на самом деле нужен какой-то способ смотреть TV и чтобы, скажем, уведомления Twitter или Facebook тоже всплывали на экране.

Концепция достаточно проста. Возьмите существующий выход кабельной приставки, Blu-ray-плеера или AV-ресивера; подайте его в коробку, которая подмешивает chumby-контент; и передайте получившееся видео дальше на TV. Но из-за повсеместного применения HDCP-шифрования в цифровых видеопотоках ремикс контента юридически опасен, если делать его неправильным способом. Поиск правильного способа сделать это и стал рождением NeTV.



NeTV с логотипом Chumby.



Внутри NeTV.

Предыстория и контекст

NeTV был моим ответом на задачу ремикса существующего видео с интернет-контентом при сохранении в юридических границах; этому помогла публичная публикация главного ключа HDCP в сентябре 2010 года. Чтобы помочь вам понять этот хак, начнем с небольшой предыстории HDCP.

High-bandwidth Digital Content Protection - это система шифрования на уровне пикселей, используемая для шифрования видеопередач по HDMI. HDCP дает вещателям и студиям контроль над экранами, на которых воспроизводится их контент, поскольку эти компании используют шифрование как механизм контроля авторского права. HDCP ограничивает законные манипуляции с контентом, такие как отображение картинки в картинке, наложение контента, сторонняя фильтрация и изменение изображения. Объедините HDCP с DMCA, который криминализирует обход контроля авторского права, и вы поймете, что при просмотре некоторых видео незаконно изменять контент на собственном экране. Вот почему существует мало решений для HDMI-видеомикширования, которые действительно работают с вещательным или кинематографическим контентом.

Подытоживая, у меня было четыре цели для NeTV: включить ремикс контента на стороне потребителя, позволить пользователям устранять рекламу или заменять ее рекламой, релевантной им самим, создать интерактивный ТВ-опыт и сделать что-то совместимое с любым телевизором. Чтобы достичь этих целей, я спроектировал NeTV как посредника, который берет данные, скажем, от Blu-ray-плеера и применяет главный ключ, чтобы дать пользователям пользовательский оверлей. У видеооверлея есть много применений, но базовый сценарий таков: пока вы наслаждаетесь контентом X, вы также хотели бы быть в курсе контента Y. Объединение двух источников контента требует механизма видеооверлея.

С моей MITM-атакой NeTV накладывал браузер WebKit (движок, который используют Safari и Chrome) поверх любого видеопотока. Конкретный пример использования этой технологии - накладывать ленты Twitter как бегущие новостные строки поверх телешоу, чтобы смотреть комментарии сообщества в реальном времени на том же экране, на котором вы смотрите само шоу. Некоторые телепрограммы уже пытаются включать ленты Twitter, но они делают это только на стороне источника; пользователи могут смотреть только те хештеги, которые показывает шоу. Однако с этим хаком одна и та же вещательная программа (скажем, политические дебаты) могла бы давать очень разный опыт просмотра в зависимости от того, какой хештег введен в Twitter-краулер зрителя.

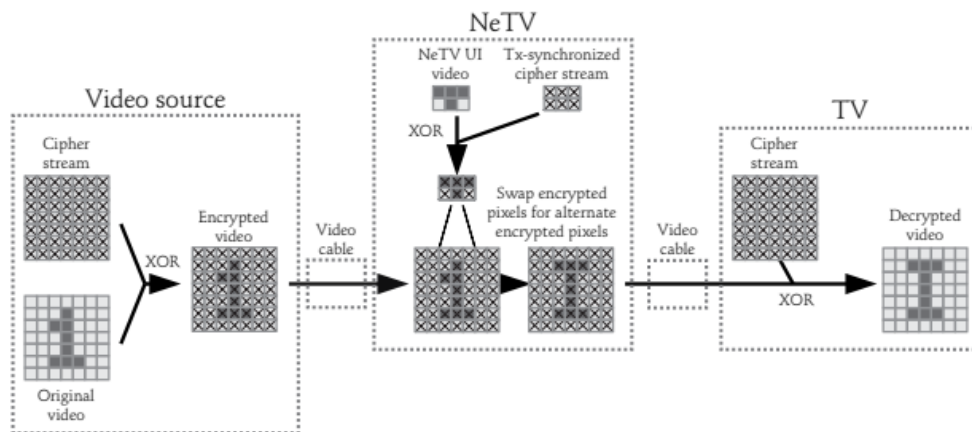
Сам факт того, что тривиальный видеооверлей является интересной темой, иллюстрирует искажение традиционных прав и свобод, принесенное DMCA. Однако, в отличие от HDCP-stripper-ов, которые, как люди предполагали, появятся после публикации главного ключа, мой хак никогда не расшифровывал исходные видеоданные, с которыми он работал. Следовательно, он не обходил авторское право, и DMCA не мог к нему применяться. Лазейка найдена!

Как работал NeTV

Конечно, я выпустил эксплойт как полностью открытый исходный код-проект,[3] включая аппаратную часть и Verilog-реализацию Spartan-6 FPGA, которую я использовал для создания TMD5-совместимых источника и приемника. TMD5 - это стандарт сигнализации, используемый HDMI и DVI. Базовый конвейер внутри FPGA десериализует входящее видео и заново сериализует его на выход. В этом тривиальном режиме NeTV является просто усилителем сигнала для видео: зашифрованные пиксели входят, зашифрованные пиксели выходят - без расшифрования и без манипуляции видео.

NeTV мог смешивать поток пользовательского контента поверх зашифрованного видеопотока, потому что HDCP шифрует без проверки. Иными словами, если посредник вмешивается в зашифрованный поток, приемник просто принимает измененные пиксели как действительные данные, расшифровывает их и показывает пользователю. Отсутствие проверки линка намеренно и необходимо. Естественная частота битовых ошибок HD-видеолинков ужасна, но человеческий глаз не обнаружит битовые ошибки даже на уровне 1 на каждые 10 000 бит. При высоких частотах ошибок пользователи видят "искры" или "снег" на экране, но изображение в основном остается целым. Допущение некоторой пиксельной порчи удерживает потребительские издержки низкими. Иначе потребовались бы кабели гораздо более высокого качества вместе с техниками прямой коррекции ошибок (FEC), чтобы достичь частоты битовых ошибок, совместимой со строгими криптографическими техниками проверки вроде хеширования полного кадра.

Таким образом, главная задача NeTV - вывести поток ключа, идентичный потоку ключа передатчика и синхронизированный с ним, зашифровать пользовательский контент этим потоком ключа и выборочно на лету заменять пиксели передатчика пользовательскими зашифрованными пикселями. Если все выровнено, приемник расшифрует изображение, которое выглядит как идеальный оверлей пользовательского контента поверх исходного видеопотока.



A high-level conceptual diagram of how NeTV worked

CREATING THE OVERRLAY

Высокоуровневая концептуальная диаграмма того, как работал NeTV.

Создание оверлей

Чтобы сгенерировать пользовательский оверлей-контент, мы подключили к FPGA крошечный встраиваемый Linux-компьютер. С точки зрения Linux-компьютера FPGA эмулирует параллельный RGB LCD, к которому можно обращаться через фреймбуфер `/dev/fb0` (путь к первому фреймбуферу в Linux). Linux-компьютер автоматически запускал браузер WebKit на весь экран при загрузке, тем самым заполняя `/dev/fb0` пользовательским контентом.

Система выбирала, какой пиксель заменить, наблюдая цвет оверлей-видео из WebKit; этот прием известен как хромакей. Оверлей-видео не было зашифровано и генерировалось пользователем, поэтому смотреть на цвет оверлей-видео было полностью законно. Однако другие, более выразительные и эстетически привлекательные методы комбинирования пикселей, например альфа-смешивание, потребовали бы расшифровки исходного видео, что было бы незаконно.

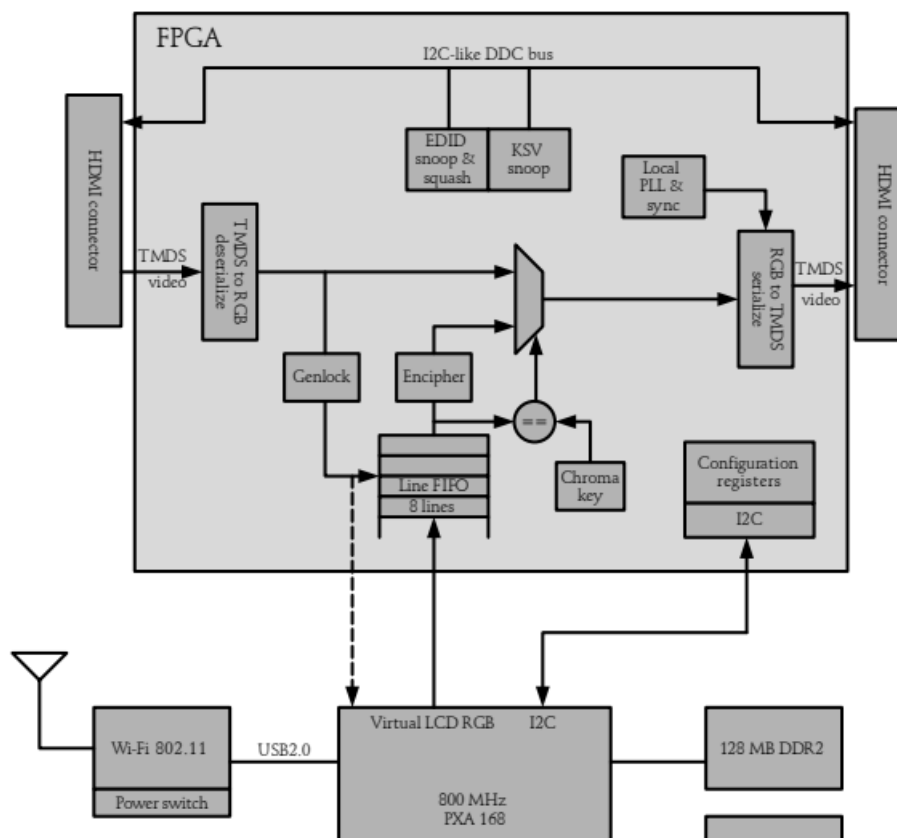
Если оверлей-видео совпадало с определенным хромакей-цветом (в данном случае конкретным оттенком ярко-розового), отображалось входящее видео; иначе отображалось оверлей-видео. Следуя этой системе, пользователи могли создавать прозрачные «дыры» в пользовательском интерфейсе, чтобы показывать исходное видео под ним. Поскольку интерфейс рендерился браузером WebKit, пользователи могли реализовать хромакей, просто установив цвет фона в CSS для страниц интерфейса в этот магический оттенок розового. С такими настройками состояние веб-страницы по умолчанию было бы прозрачным, а все элементы, отрендеренные поверх нее, были бы непрозрачными, пока элементы интерфейса избегали хромакей-цвета и отключали улучшения вроде сглаживания.

Создание ключевого потока

Конечно, хромакей происходил в зашифрованной области. Поэтому второй задачей FPGA было прослушивать HDMI-линк и создавать поток ключа, идентичный потоку ключа передатчика. Сначала FPGA наблюдала I2C-линк, имеющийся в HDMI и известный как канал данных дисплея (display data channel, DDC). DDC позволяет мониторам сообщать свои записи возможностей (расширенные идентификационные данные дисплея, или EDID), и там же происходит обмен ключами шифрования.

Наблюдая рукопожатием обмена ключами между передатчиком и приемником, NeTV мог математически извлечь приватные ключи передатчика и приемника с помощью главного ключа HDCP. После вывода векторов приватных ключей их можно было перемножить ровно так же, как они перемножались бы в источнике или приемнике, чтобы вывести общий секрет, называемый K_m . Когда этот общий секрет записывался в HDCP-движок FPGA, состояние шифра было готово, позволяя NeTV шифровать оверлей поверх видео, передаваемого между источником видео и устройством отображения видео.

Рассматривая юридические ограничения как еще одно инженерное ограничение, я смог создать совершенно новое устройство, доказывающее одну мысль: неверно автоматически приравнивать хаки, которые обходят DRM-систему, к попыткам обойти авторское право. NeTV никогда не расшифровывает ранее зашифрованное видео и не может работать без уже существующего, действительного HDCP-линка, что делает его добросовестным, не нарушающим права, коммерчески полезным применением главного ключа HDCP.



Более подробная блок-схема, показывающая, как работала FPGA NeTV.

Пока в этой главе мы видели примеры разных подходов и техник аппаратного хакинга: от физического проникновения до построения и анализа инструментов системного уровня, а также до рассмотрения юридических ограничений как инженерных задач. В "Кто такие шаньчжай?" на странице 122 я обсуждал юридический подход проекта под кодовым названием Fernvale к реверс-инжинирингу чипсета мобильного телефона. Помимо того что мы с hobs думали о законе как инженеры, нам пришлось выложиться полностью и применить все технические навыки, имевшиеся в нашем распоряжении, чтобы выполнить реверс-инжиниринг такой сложной системы. Остальная часть этой главы погружается в некоторые из этих техник.

Взлом шаньчжайского телефона

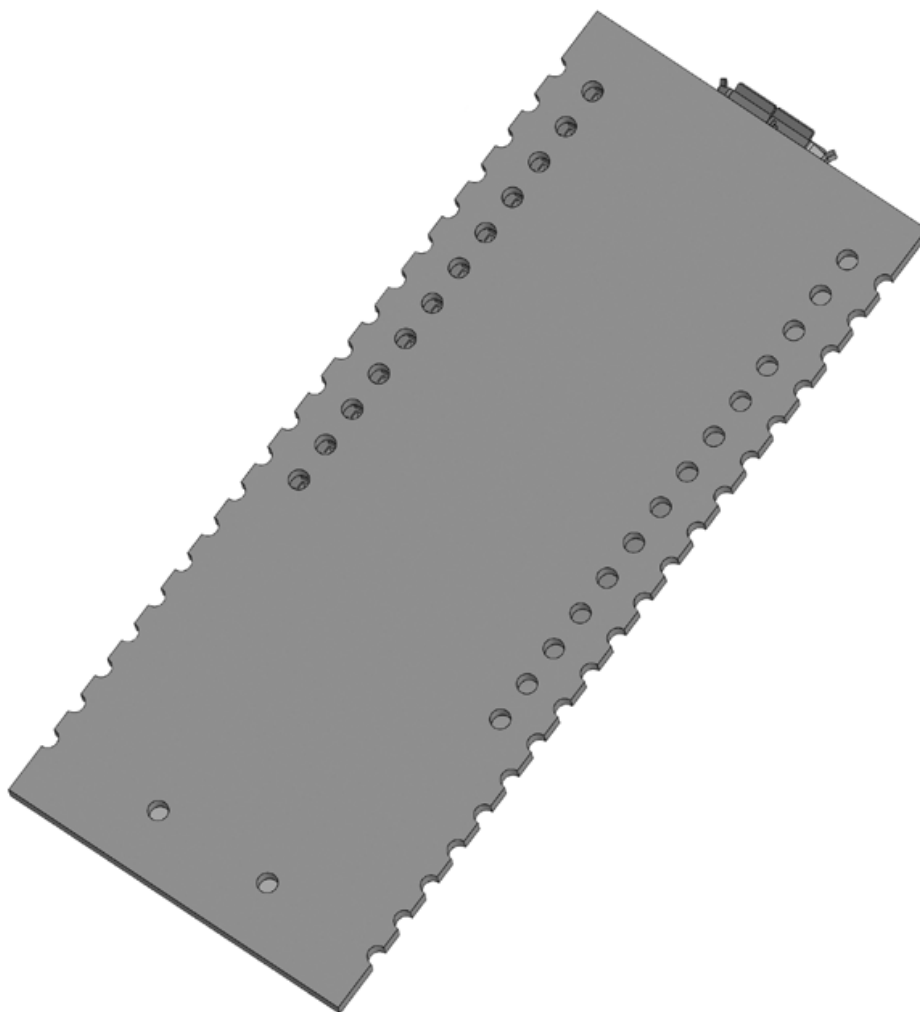
Когда мы с hobs работали над Fernvale, нашей целью было создать новую платформу, производную от аппаратуры моего гонкай-телефона за \$12, и вернуть техническую информацию в открытую систему IP с открытым исходным кодом. По некоторым частям чипа, который мы хотели исследовать методом реверс-инжиниринга, у нас вообще не было документации, но это нас не остановило. Мы разобрались с непростыми юридическими вопросами и создали собственный специализированный язык сценариев для программирования прошивки чипа, чтобы избежать неосознанного плагиата.

Однако по сравнению с прошивкой реверс-инжиниринг аппаратуры был довольно прямолинейным. Документы, которые мы наскребли, дали нам представление о разводке выводов чипа, а схема именования выводов была достаточно описательной, чтобы я мог применить здравый смысл и опыт и догадаться, как подключить чип. Для неоднозначных областей я прозванивал некоторые урезанные телефоны мультиметром или рассматривал их под микроскопом, чтобы определить соединения. В худших случаях я щупом осциллографа проверял работающий телефон, чтобы

убедиться, что понимаю соединения правильно. Более трудным вопросом было то, как спроектировать архитектуру аппаратуры.

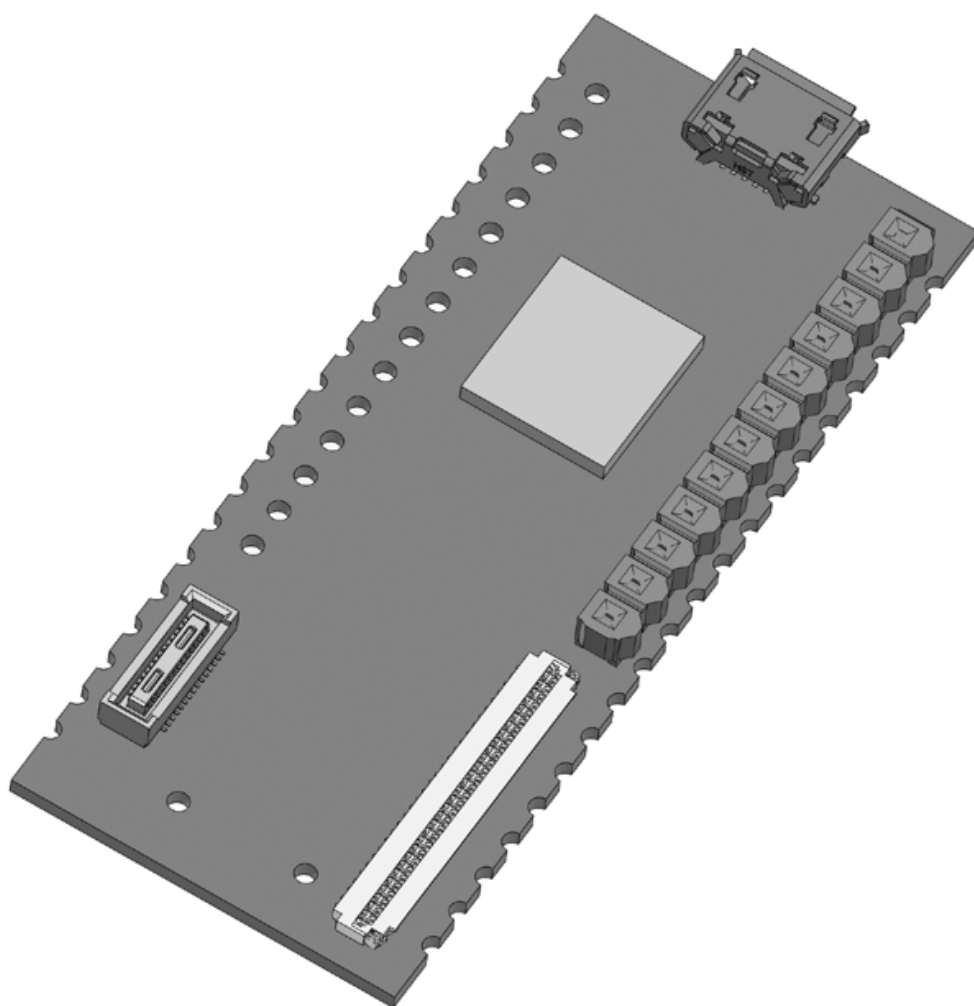
Архитектура системы

Мы не стремились создать телефон; скорее, нечто ближе к Particle Spark Core (позже переименованному в Photon), универсальному одноплатному компьютеру типа System-on-Module, рассчитанному на приложения интернета вещей. Фактически наши первоначальные рендеры и распиновки были спроектированы совместимыми с экосистемой аппаратных расширений Spark, пока мы не поняли, что микроконтроллер MT6260 из гонкай-телефона просто имеет слишком много интересной периферии, чтобы уместиться в такой малый форм-фактор.

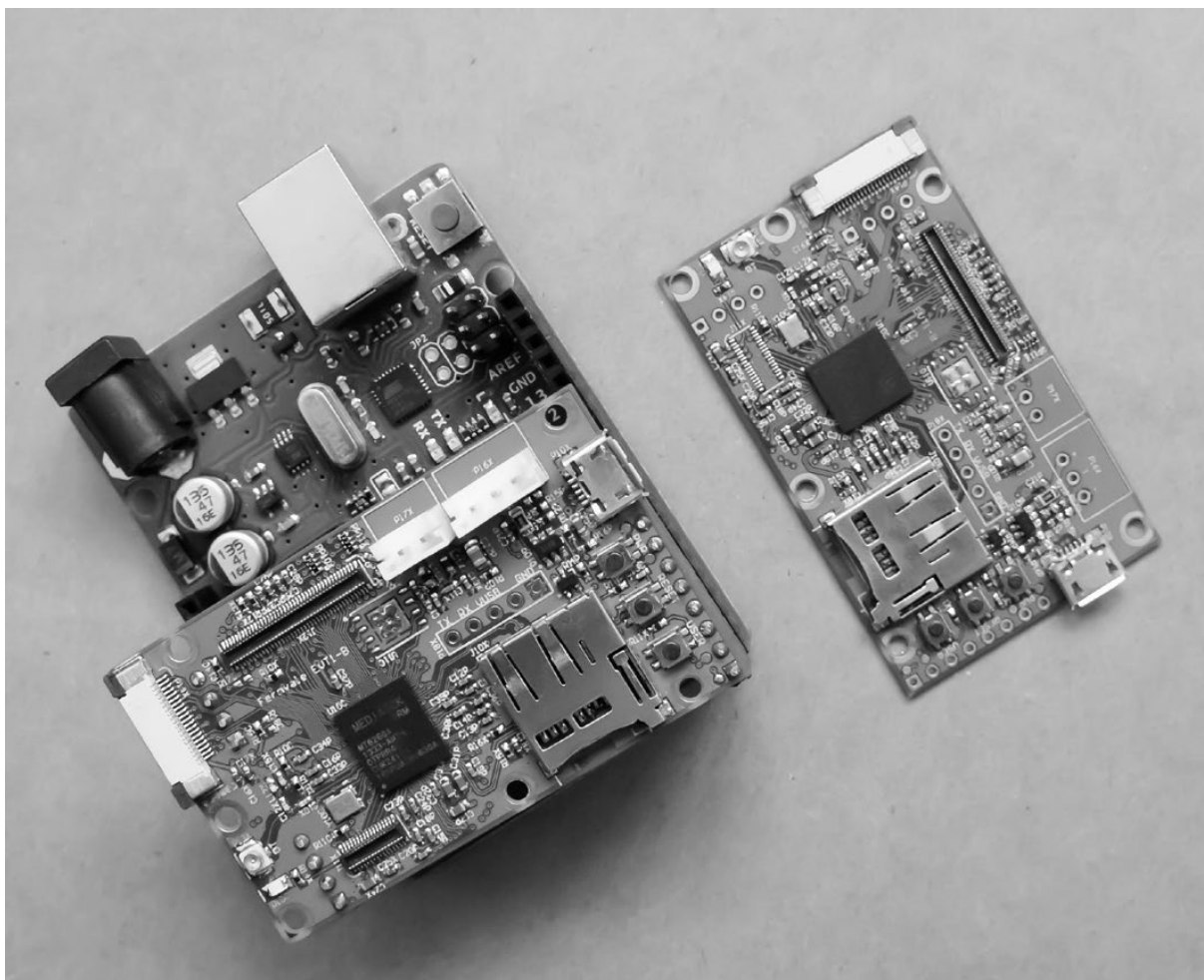


Ранние наброски PCB Fernvale.

В итоге мы остановились на односторонней основной PCB, которую назвали Fernvale Frond. Она встраивала функции microUSB, microSD, батареи, камеры, динамика и Bluetooth (а также обязательные кнопки и LED) на одной плате. Frond получилась тонкой и маленькой: толщиной 3,5 мм, длиной 57 мм и шириной 35 мм. Мы включили отверстия для установки частичного набора штыревых разъемов, расположенных с совместимым с Arduino шагом, хотя плату можно было подключать только к Arduino-устройствам, совместимым с 3,3 В.

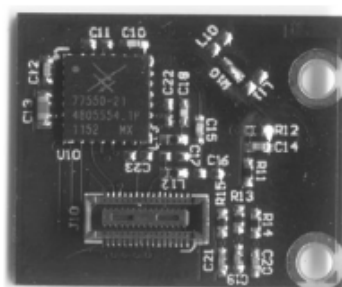


Разметка Fernvale Frond.



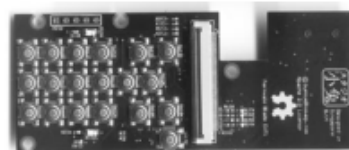
Физическая реализация Fernvale Frond, сфотографированная рядом с Arduino Uno для сравнения размеров.

Оставшуюся периферию мы вывели на пару разъемов: один был выделен для сигналов, связанных с GSM (GSM - это протокол сотовых сетей 2G), а другой - для периферии, связанной с пользовательским интерфейсом. GSM-плату мы назвали Fernvale Spore, а плату пользовательского интерфейса - Fernvale Blade. Мы вынесли GSM в модуль с множеством вариантов RF-фронтенда, чтобы сделать GSM по-настоящему устанавливаемой пользователем функцией и тем самым перенести вопрос регулирования и излучений на уровень пользователя. Выделение функций, связанных с интерфейсом, на другую плату также снизило стоимость основного модуля и позволило пользователям пробовать Frond в многочисленных сценариях, не будучи привязанными к конкретной компоновке LCD или кнопок.



GSM antenna

GSM RF:
PA + TxRx + Filters



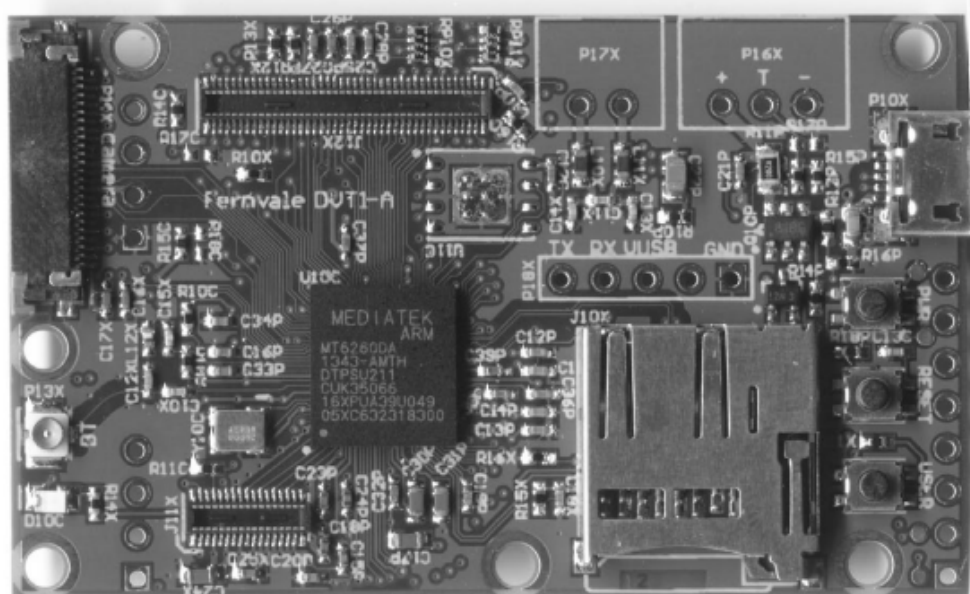
Keypad

SIM

Headphone

TS	LCD
----	-----

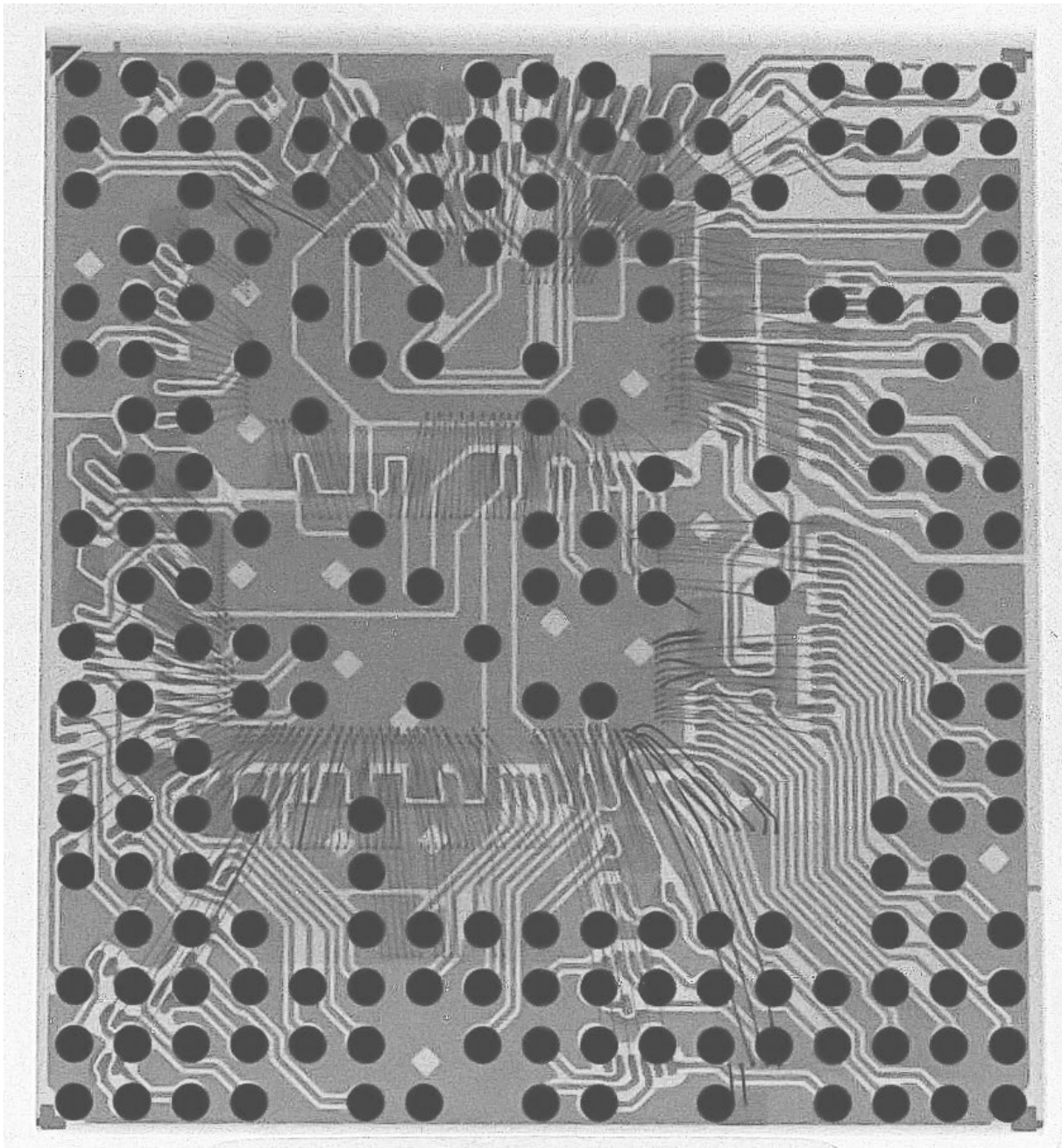
Expansion/breakout board



Системная диаграмма Fernvale, показывающая функции каждой из трех плат.

Внутри MT6260

Я заказал рентгеновские снимки МТ6260, чтобы помочь нам выявлять поддельные компоненты. МТ6260 приходилось закупать на сером рынке, и мы хотели защититься от продажи пустых эпоксидных блоков или перемаркированных версий других чипов. У МТ6260 есть варианты -DA и -A, где различие состоит в объеме встроенной флеш-памяти на кристалле.



Рентгеновский снимок чипа MT6260. Присмотритесь внимательно, чтобы заметить контуры нескольких IC среди проволочных соединений.

К нашему удивлению, этот чип за \$3 содержал не одну IC, а набор как минимум из четырех (возможно, пяти) чипов, интегрированных в один многокристальный модуль (MCM) с сотнями проволочных соединений. Я помню, как в конце 1990-х появился двухкристальный корпус Pentium Pro. Он вызвал споры о стоимости выхода годных MCM по сравнению с использованием одного большого кристалла; в целом MCM считались экзотическими и дорогими.

Я также помню, как примерно в то же время Крсте Асанович, тогда профессор в лаборатории искусственного интеллекта MIT, а позднее в UC Berkeley, сказал мне, что будущее электроники - не устройства система-на-кристалле, а скорее устройства "почти-система-на-кристалле". Корень его утверждения состоял в том, что экономика добавления масочных слоев для объединения DRAM, flash, аналоговой схемотехники, RF и цифровой логики в одном процессе была неблагоприятной; соединять несколько кристаллов в одном корпусе было дешевле и проще.

Это по-прежнему гонка между ценовым влиянием (и с точки зрения себестоимости единицы, и с точки зрения разовых инженерных затрат) добавления дополнительных технологических шагов на полупроводниковой фабрике и влиянием выхода годных, относительной ремонтпригодностью и более низкими разовыми инженерными затратами при сборке модулей. Однокристалльные устройства System-on-Chip были духом времени, когда Крсте сделал это наблюдение, и в некоторой степени остаются им до сих пор, поэтому было интересно увидеть существенную точку данных, подтверждающую его прозрение.

Понимание внутренней структуры чипа также помогало в реверс-инжиниринге системы. Знание того, что MediaTek просто объединяла несколько чипов в одном корпусе, пролило столь необходимый свет на назначение и организацию их API. Оно также подсказало нам, что некоторые элементы системы будут повторно использоваться в нескольких категориях продуктов и поколениях, так что мы знали: из документации по более старым или родственным чипам можно делать полезные выводы. Когда вы собираете головоломку такой сложности, важна каждая улика, включая те, что получены просто при рассмотрении физической структуры чипа.

Реверс-инжиниринг структуры загрузки

Похоже, шаньчжайские инженеры в Китае имеют доступ ровно к такому объему документации, которого хватает, чтобы собрать телефон и настроить его интерфейс, но недостаточно для полного переноса ОС. Посмотрев на достаточное количество телефонов, я в конце концов понял, что все телефоны на основе конкретного чипсета будут иметь одни и те же backdoor-коды, а их пользовательские интерфейсы часто не согласованы с реализованным железом. Например, телефон за \$12, который я разобрал в главе 4, предлагал мне подключить наушники к разъему для наушников, чтобы заработало FM-радио, хотя разъема для наушников у него нет.

Чтобы сделать Fernvale доступным инженерам на Западе через лицензирование с открытым исходным кодом, нам пришлось реконструировать все с нуля, включая цепочку инструментов, инструмент прошивки, ОС и приложения. Но все китайские реализации телефонов просто полагались на проприетарную цепочку инструментов MediaTek, а значит, нам пришлось заняться реверс-инжинирингом, чтобы разобраться в процессе загрузки и протоколе загрузки прошивки.

Мой первый шаг при реверс-инжиниринге чипа - всегда, если возможно, снять дампы ROM. Мы нашли ровно одну модель телефона с внешней ROM, которую смогли выпаять (в ней использовался вариант чипа -D без ROM), и прочитали ее данные обычным ROM-ридером. Мы увидели очень мало шифртекста в ROM, но там было много сжатых данных. Вот страница из наших заметок после статического анализа образа ROM:

0x0000_0000	media signature "SF_BOOT"
0x0000_0200	bootloader signature "BRLYT", "BBBB"
0x0000_0800	sector header 1 ("MMM.8")
0x0000_09BC	reset vector table
0x0000_0A10	start of ARM32 instructions
	- stage 1 bootloader?
0x0000_3400	sector header 2 ("MMM.8")
	- stage 2 bootloader?
0x0000_A518	thunk table of some type
0x0000_B704	end of code (padding until next sector)
0x0001_0000	sector header 3("MMM.8") - kernel?
0x0001_0368	jump table + runtime setup (stack, etc.)
0x0001_0828	ARM thumb code start - possibly also
	baseband code
0x0007_2F04	code end
0x0007_2F05	begin padding "DFFF"
0x0009_F005	end padding "DFFF"

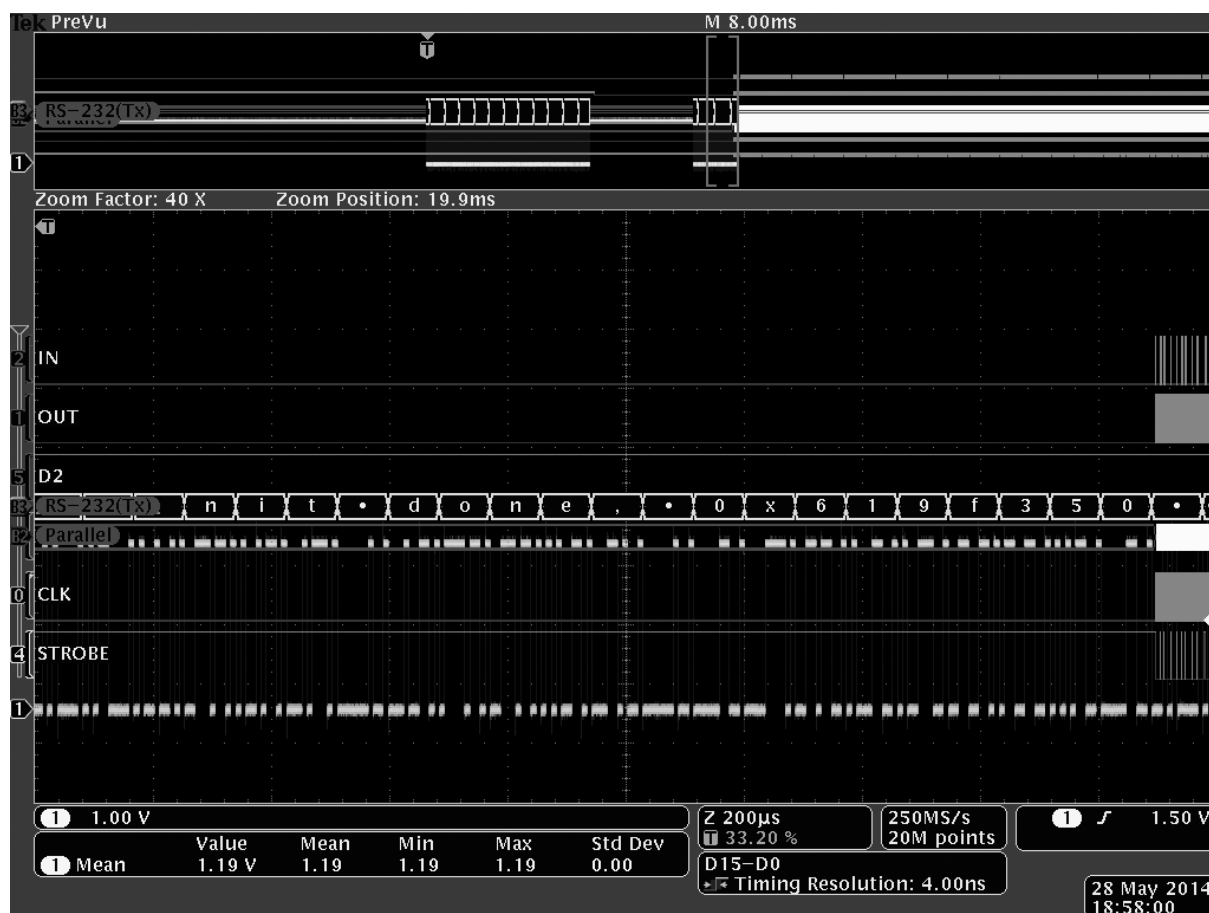
```

0x0009_F006      code section begin "Accelerated
                  Technology / ATI / Nucleus PLUS"
0x000A_2C1A      code section end; pad with zeros
0x000A_328C      region of compressed/unknown data begin
0x007E_E200      modified FAT partition #1
0x007E_F400      modified FAT partition #2

```

Шестнадцатеричные числа слева - это адреса памяти, а текст справа описывает, что, по мнению хобс и меня, хранилось по каждому адресу. Одна из проблем при реверс-инжиниринге SoC состоит в том, что у него есть внутренняя загрузочная ROM, которая всегда выполняется до того, как код загружается с внешнего устройства. Эта внутренняя ROM также может иметь проверки подписи и безопасности, предотвращающие вмешательство во внешний код.

Чтобы определить, насколько трудным будет реверс-инжиниринг этой системы, мы хотели быстро выяснить, какой объем кода выполнялся внутри CPU до перехода к внешнему загрузочному коду. Осциллограф Tek MDO4104B-6 позволил нам выполнить эту задачу всего за пару часов.

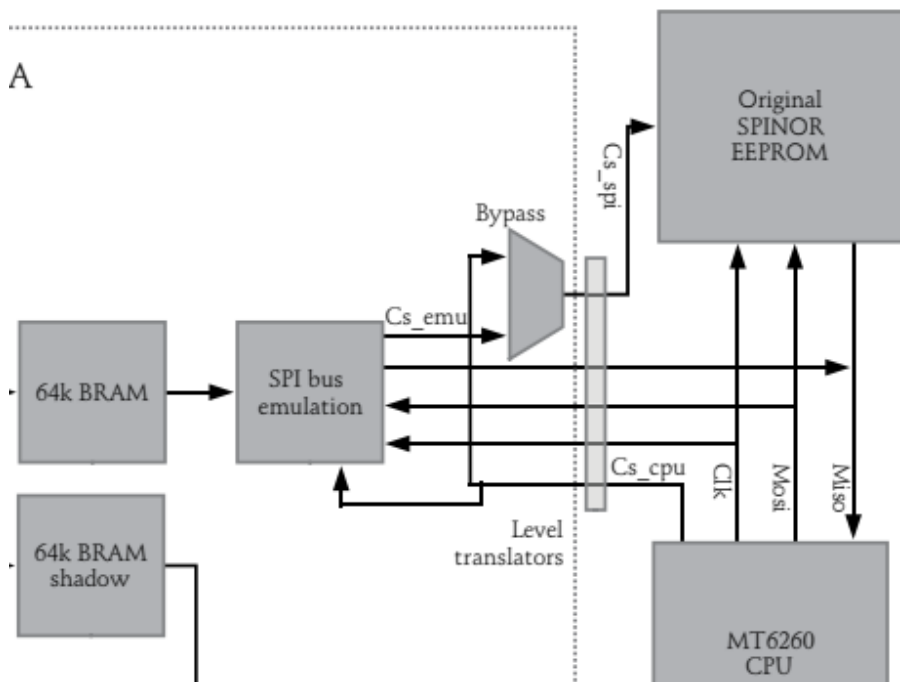


Снимок экрана Tek MDO4104B-6. Верхняя четверть показывает уменьшенный обзор всего захвата. Обратите внимание, как обращения к SPI ROM перемежаются консольным выводом.

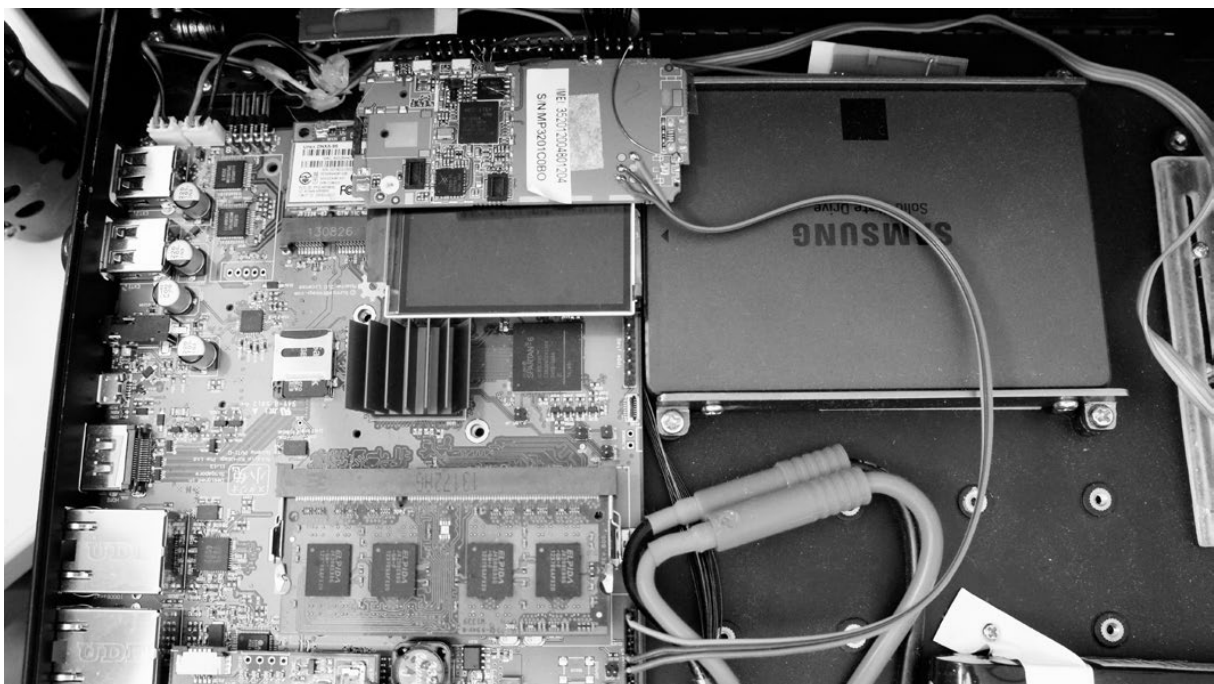
Этот конкретный осциллограф обладает необычной способностью выполнять постобработку глубоких аналоговых трасс с высоким разрешением и выдавать результат как цифровые данные. Например, мы могли просто щупом мультиметра обследовать чип, циклически включая питание, пока не видели что-то похожее на сигнал, закодированный как RS-232, а затем запускали постобработку, чтобы извлечь любой ASCII-текст, закодированный в аналоговых трассах. Аналогично, если мы захватывали SPI-трассы, осциллограф мог извлекать шаблоны обращений к ROM похожим методом. Рассматривая синхронизацию текстового вывода относительно шаблонов обращений к SPI ROM, мы быстро определили: если внутренняя загрузочная ROM вообще выполняла какую-либо проверку, она была минимальной и никак не приближалась к вычислительной сложности RSA-шифрования.

Отсюда нам нужно было ускорить цикл измерить-изменить-проверить. Выпаивать ROM, вставлять ее в программатор и снова припаивать к плате очень быстро стало бы утомительным. К счастью, мы реализовали эмулятор NAND-флеш-ROM (мы любовно сократили это до ROMulator) на Novena, который ранее использовали для реверс-инжиниринга AX211, содержащегося в некоторых SD-картах. Мы просто повторно использовали эту кодовую базу и сделали SPI ROMulator. Мы переделали GPBB и соответствующий FPGA-код, чтобы добавить возможность переключаться между исходной загрузочной SPI ROM и двухпортовой областью эмулятора на 64 KiB, которая также была отображена в адресное пространство Linux-хоста Novena. Затем мы подключили телефон к ноутбуку и пустили ROMulator в работу.

lator to work.



Блок-схема FPGA SPI ROMulator.



В моей Novena есть телефон! Что он там делает?

Имея поток адресов, определенный осциллографом Тек, быструю правку ROM с помощью ROMulator и подсказки статического анализа кода в IDA о существовании функции SHA-1 в ROM, мы определили, что начальный загрузчик (который мы называли 1bl) проверялся хешем с использованием SHA-1-приложения.

Примечание. Ассемблерный код хеш-функции обычно имеет очень характерную форму, или набор инструкций, а данный хеш также имеет некоторое количество уникальных для него магических чисел. Учитывая эти факты, при попытке выполнить реверс-инжиниринг метода аутентификации одно из первых действий хакера - использовать IDA для поиска таких констант рядом с функцией, имеющей форму рассматриваемой хеш-функции.

Создание точки опоры

Следующим шагом было создание небольшой интерактивной shell-оболочки, которую мы могли использовать как точку опоры для экспериментов на целевом железе. Как и в проекте реверс-инжиниринга SD-карты, xobs создал компактную REPL-среду под названием Fernly, поддерживавшую команды вроде просмотра памяти, записи данных и дампа регистров CPU.

Мы спроектировали ROMulator так, чтобы эмулируемая ROM выглядела как отображенное в память окно на 64 KiB на Linux-хосте. Это позволило использовать полезные POSIX-абстракции вроде функции mmap(), функции open() (через /dev/mem), функции read() и функции write() для доступа к эмулируемой ROM. xobs использовал эти абстракции, чтобы создать I/O-цель для radare2, переносимого фреймворка реверс-инжиниринга. I/O-цель автоматически обновляла SHA-1-хеш каждый раз, когда мы вносили изменения в кодовое пространство 1bl. С такой системой мы могли делать полезные вещи вроде интерактивного патчинга и дизассемблирования кода внутри пространства эмулируемой ROM.

```

0x00000c6b 0x0019046b 0x22f9b1f0 0x09200100 0xd0f000a1 k.....".. .....
0x00000c7b 0xfd0020fe 0x70eae4f7 0x0008f4bd 0x00100300 . ....p.....
0x00000c8b 0x0086f800 0x00668070 0x0086b470 0x0085d070 ....p.f.p...p...
0x00000c9b 0x6f6f6670 0x6f742064 0x616d6f79 0x202c616d pfood toyomama,

0x00000c6b 6b04      lsls r3, r5, 17
0x00000c6d 1900      movs r1, r3
0x00000c6f f0b1      cbz r0, 0x00000caf
0x00000c71 f922      movs r2, 249
0x00000c73 0001      lsls r0, r0, 4
0x00000c75 2009      lsrs r0, r4, 4
0x00000c77 a100      lsls r1, r4, 2
==< 0x00000c79 f0d0      beq.n 0x00000c5d ;[1]
0x00000c7b fe20      movs r0, 254
0x00000c7d 00fdf7e4  stc2 4, cr14, [r0, -988] ; 0xfffffc24
0x00000c81 ea70      strb r2, [r5, 3]
0x00000c83 bdf40800 ; <UNDEFINED> 0xf4bd0008 ;[2]
0xffffffff84bdc97() ; hit1_0
0x00000c87 0003      lsls r0, r0, 12
0x00000c89 1000      movs r0, r2
0x00000c8b 00f88600 strb.w r0, [r0, r6]
0x00000c8f 7080      strh r0, [r6, 2]
0x00000c91 6600      lsls r6, r4, 1
0x00000c93 70b4      push {r4, r5, r6}
0x00000c95 8600      lsls r6, r0, 2
==< 0x00000c97 70d0      beq.n 0x00000d7b ;[3]
0x00000c99 8500      lsls r5, r0, 2
0x00000c9b 7066      str r0, [r6, 100]
0x00000c9d 6f6f      ldr r7, [r5, 116]
0x00000c9f 6420      movs r0, 100
0x00000ca1 746f      ldr r4, [r6, 116]
||| ;-- hit1_0:
0x00000ca3 796f      ldr r1, [r7, 116]
0x00000ca5 6d61      str r5, [r5, 20]
0x00000ca7 6d61      str r5, [r5, 20]
0x00000ca9 2c20      movs r0, 44
0x00000cab 2578      ldrb r5, [r4, 0]
0x00000cad 0a0d      lsrs r2, r1, 20
0x00000caf 00f8b500 strb.w r0, [r0, r5, lsl 3]
0x00000cb3 26f64335 ; <UNDEFINED> 0xf6263543 ;[4]
||| 0xffffffff862773d() ; hit1_0
0x00000cb7 0000      movs r0, r0
0x00000cb9 f0a2      add r2, pc, 960 ; (adr r2, 0x0000107c)
||| 0x00000cbb f925      movs r5, 249
||| 0x00000cbd 4841      adcs r0, r1

```

Патчинг некоторого кода в ROM.

Мы также подключили выключатель питания телефона к FPGA I/O. Это позволило нам писать автоматизированные сценарии, которые переключали питание телефона при обновлении содержимого ROM, так что мы могли автоматически fuzz-ить неизвестные аппаратные блоки.

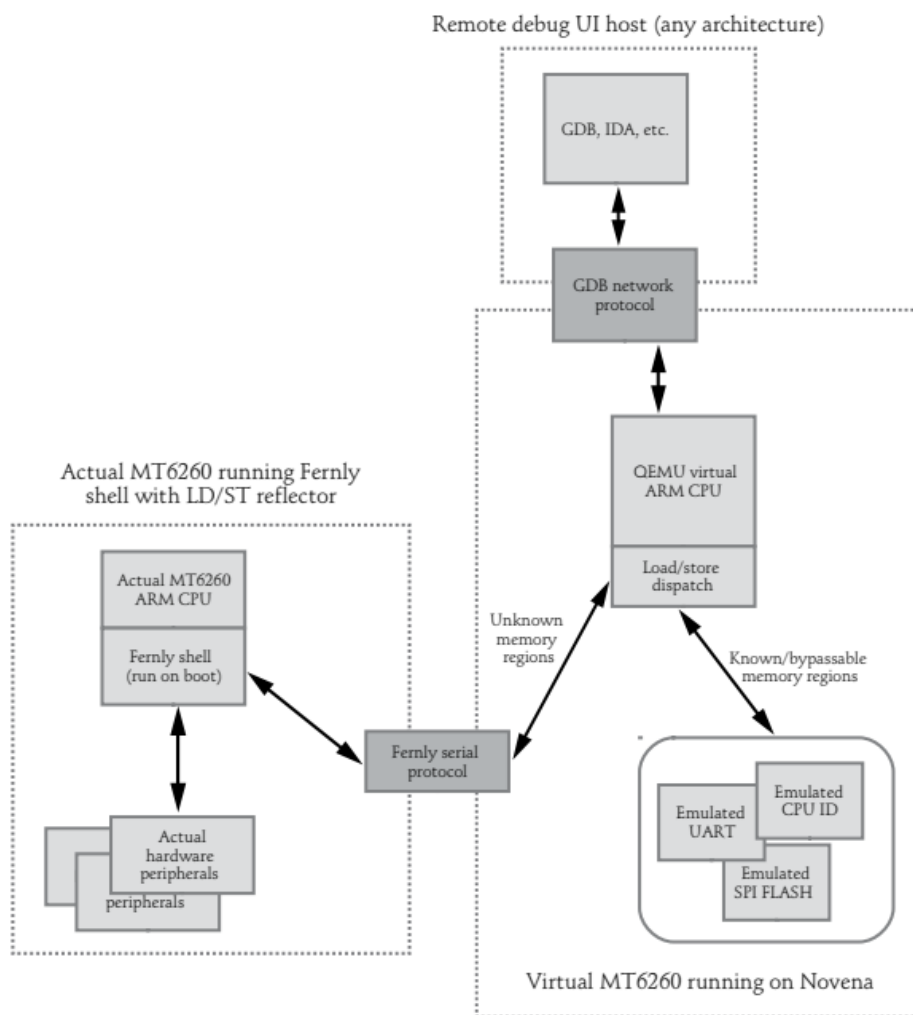
Подключение отладчика

Нам пришлось применить нетрадиционный подход для подключения отладчика к коду в ROM, потому что локализовать критические блоки было трудно, а JTAG был мультиплексирован с критическими функциями на целевом устройстве. xobs эмулировал ARM-ядро и использовал свою shell-оболочку Fernly, чтобы отражать виртуальные операции загрузки и записи на живую цель. Таким образом мы смогли подключить удаленный отладчик к эмулируемому ядру, полностью обойдя необходимость в JTAG. Это также позволило нам использовать кросс-платформенные инструменты вроде IDA на x86 для реверс-инжиниринга.

В сердце этой техники отладки находился QEMU, мультиплатформенный системный эмулятор. QEMU поддерживает эмуляцию ARM-целей, конкретно чипа ARMv5, использовавшегося в нашем целевом устройстве. Мы сделали новый тип виртуальной машины под названием Fernvle, который реализовывал часть наблюдавшегося железа на цели и просто передавал неизвестные обращения к памяти напрямую устройству.

Shell-оболочка Fernly была урезана до поддержки всего трех команд: write, read и zero-memory. Команда write записывает байт, слово или двойное слово данных в RAM настоящего устройства. Команда read читает байт, слово или двойное слово с настоящего устройства. Команда zero-memory является оптимизацией, поскольку операционная система записывает большие количества нулей по большой области памяти.

Мы также перехватили и эмулировали регистры последовательного порта, позволяя хост-системе отображать последовательные данные так, будто они печатаются на целевом устройстве. Наконец, мы эмулировали SPI, IRAM и PSRAM такими, какими они выглядели бы на реальном устройстве. Другие области памяти либо перехватывались и направлялись на настоящее устройство, либо оставались неотображенными и сообщались QEMU как ошибки.



The architecture of the debugger

Архитектура отладчика.

Подключение отладчика было многоступенчатым процессом. Сначала мы подготавливали настоящую цель MT6260 средой shell-оболочки Fernly. Затем мы загружали виртуальный ARM CPU QEMU с версией исходного образа поставщика, подготовленной с известным состоянием регистров в удобной точке процесса загрузки. В этой точке выполнение кода продолжалось на виртуальной машине, пока не выполнялась загрузка или запись по неизвестному адресу. На этой загрузке или записи выполнение виртуальной машины приостанавливалось, а запрос отправлялся настоящему MT6260 через интерфейс shell-оболочки Fernly. Затем загрузка или запись выполнялась на настоящей машине, которая передавала результаты загрузки или записи виртуальной машине, чтобы выполнение могло продолжиться.

Мы не могли запускать Fernly напрямую из SPI ROM, потому что процедура инициализации бинарника поставщика изменяла тайминги SPI ROM. Но, конечно, Fernly упала бы, если бы запись случайно попала куда-нибудь в ее область памяти. Чтобы избежать возможности перезаписи кода shell-оболочки Fernly при загрузке или записи, мы спрятали код в области IRAM, которая перехватывалась и эмулировалась. Эмуляция целевого CPU позволила нам подключить удаленный отладчик вроде IDA через GDB по TCP. Отладчик имел полный контроль над эмулируемым CPU и мог обращаться к его эмулируемой RAM. Вот пример вывода гибридной отладочной связи QEMU/live-target.

```

bunnie@bunnie-novena-ноутбук:~/code/fernvale-qemu$ ./run.sh
~~~ Welcome to MTK Bootloader V005 (since 2005) ~~~
=====
READ WORD Fernvale Live 0xa0010328 = 0x0000... ok
WRITE WORD Fernvale Live 0xa0010328 = 0x0800... ok
READ WORD Fernvale Live 0xa0010230 = 0x0001... ok
WRITE WORD Fernvale Live 0xa0010230 = 0x0001... ok
READ DWORD Fernvale Live 0xa0020c80 = 0x11111011... ok
WRITE DWORD Fernvale Live 0xa0020c80 = 0x11111011... ok
READ DWORD Fernvale Live 0xa0020c90 = 0x11111111... ok
WRITE DWORD Fernvale Live 0xa0020c90 = 0x11111111... ok
READ WORD Fernvale Live 0xa0020b10 = 0x3f34... ok
WRITE WORD Fernvale Live 0xa0020b10 = 0x3f34... ok

```

Этот вывод показывает, как перехваченные последовательные записи появляются на консоли, плюс журнал записей и чтений, выполнявшихся эмулируемым ARM CPU, когда они перенаправлялись настоящему устройству, выполняющему сокращенную shell-оболочку Fernly. Это и была наша точка опоры.

Отсюда мы с hobs обнаружили смещения нескольких IP-блоков, повторно использованных из предыдущих известных чипов MediaTek, путем поиска их "сигнатуры" в памяти. Сигнатура могла быть такой простой, как значения регистров по умолчанию после включения питания, или чем-то более сложным, например изменениями в битовых шаблонах из-за побочных эффектов регистров установки или сброса битов, расположенных по смещениям внутри адресного пространства IP-блока. Следование за сигнатурой помогло нам найти смещения регистров нескольких периферийных устройств и сгенерировать карту памяти.

Начальный адрес	Конечный адрес	Размер области	Описание
0x00000000	0xffffffff	0xffffffff	Карта PSRAM, повторенная и зеркалированная со смещениями 0x00800000
0x10000000	0x1fffffffff	0xffffffff	SPI-чип, отображенный в память
неизвестно	неизвестно	неизвестно	неизвестная область
0x70000000	0x7000cfff	0xcfff	Встроенная SRAM (возможно, кэш?)
неизвестно	неизвестно	неизвестно	неизвестная область
0x80000000	0x80000008	0x08	Блок конфигурации (версия чипа и т. п.)
0x82200000	неизвестно	неизвестно	
0x83000000	неизвестно	неизвестно	
0xa0000000	0xa0000008	0x08	Блок конфигурации (зеркало?)
0x10010000	неизвестно	неизвестно	
(режим SPI?)			неизвестная область
0x10020000	0xa0020e10	0x0e10	Блок управления GPIO
0xa0030000	0xa0030040	0x40	Блок WDT
+ 0x08 -> регистр WDT (?)			
+ 0x18 -> источник загрузки (?)			
0xa0030800	неизвестно	неизвестно	неизвестная область
0xa0040000	неизвестно	неизвестно	неизвестная область
0xa0050000	неизвестно	неизвестно	неизвестная область
0xa0060000	неизвестно	неизвестно	возможные IRQ по 0xa0060200
0xa0070000	=====	=====	== Пусто (все нули) =====
0xa0080000	0xa008005c	0x5c	Блок UART1
0xa0090000	0xa009005c	0x5c	Блок UART2
0xa00a0000	неизвестно	неизвестно	
неизвестная область			

Эта карта памяти показывает, какое содержимое хранится в разных диапазонах адресов на чипе. Например, второй диапазон адресов в карте (от 0x10000000 до 0x1FFFFFFF) состоял из 0x0FFFFFFF байт, соответствующих отображенному в память SPI-чипу.

Загрузка ОС

После обнаружения смещений регистров мы быстро продвинулись по многим направлениям, но наша цель (перенести NuttX, основанную на BSD операционную систему реального времени, на это устройство) оставалась неуловимой. В каноне шаньчайских технических описаний не было документации на контроллер прерываний. Через статический анализ бинарников мы нашли процедуры, устанавливавшие обработчики прерываний, но не могли определить адресные смещения самого контроллера прерываний.

Все, что мы могли сделать, - открыть кодовую базу MediaTek и обратиться к заголовочному файлу, который содержал смещения регистров и определения битов контроллера прерываний. Это укладывалось в наши самоограничения не нарушать авторское право, потому что факты не охраняются авторским правом. Я описываю юридическое обоснование этой идеи в главе 4, в разделе "Работа с авторскими правами" на странице 138. После выяснения этих фактов мы создали собственный специализированный язык сценариев под названием Scriptic, чтобы избежать неосознанного плагиата чего-либо из существующей кодовой базы.

Построение новой цепочки инструментов

Однако требовать от пользователей владеть Novena ROMulator, чтобы хакать Fernvale, не было масштабируемым решением. Чтобы завершить историю, мы создали полную цепочку инструментов разработчика. Компилятор был довольно простым делом; многие стандартные компиляторы поддерживают ARM как цель, включая clang и GCC. Но сделать открытые инструменты для прошивки MT6260 было гораздо сложнее. Все существующие инструменты, о которых мы знали и которые поддерживали версию протокола, требуемую MT6260, были проприетарными Windows-программами. Это означало, что нам пришлось выполнить реверс-инжиниринг протокола прошивки MediaTek и написать собственный инструмент с открытым исходным кодом.

К счастью, MT6260 с не пережженными fuse-битами появляется как /dev/ttyUSB0, когда вы подключаете его к Linux-хосту. Иными словами, он появляется как эмулированное последовательное устройство поверх USB. Это сняло с нас низкоуровневую работу по отправке и приему байтов, оставив реверс-инжиниринг протокольного уровня.

xobs нашел внутреннюю загрузочную ROM MT6260 и выполнил статический анализ кода, чтобы узнать больше о протоколе. Он также сделал статический анализ инструмента прошивки MediaTek и захватил живые трассы с помощью анализатора USB-протокола, чтобы прояснить оставшиеся детали. Вот сводка команд, которые он извлек, в том виде, в каком мы использовали их в нашей открытой версии инструмента USB-прошивки.

```
enum mtk_commands {
    mtk_cmd_old_write16 = 0xa1,
    mtk_cmd_old_read16 = 0xa2,
    mtk_checksum16 = 0xa4,
    mtk_remap_before_jump_to_da = 0xa7,
    mtk_jump_to_da = 0xa8,
    mtk_send_da = 0xad,
    mtk_jump_to_mau_i = 0xb7,
    mtk_get_verсия = 0xb8,
    mtk_close_usb_and_reset = 0xb9,
    mtk_cmd_new_read16 = 0xd0,
    mtk_cmd_new_read32 = 0xd1,
    mtk_cmd_new_write16 = 0xd2,
    mtk_cmd_new_write32 = 0xd4,
    // mtk_jump_to_da = 0xd5,
    mtk_jump_to_b1 = 0xd6,
    mtk_get_sec_conf = 0xd8,
    mtk_send_cert = 0xe0,
```

```

mtk_get_me = 0xe1, /* Responds with 22 bytes */
mtk_send_auth = 0xe2,
mtk_sla_flow = 0xe3,
mtk_send_root_cert = 0xe5,
mtk_do_security = 0xfe,
mtk_прошивка_версия = 0xff,
};

```

Это всего лишь структура C enum, что делает ее очень гиковским способом задать отображение чисел на значения команд. Например, `mtk_cmd_old_write16` - это команда 0xA1, `mtk_command_old_read16` - команда 0xA2 и так далее.

Результаты Fernvale

После примерно года периодических усилий между работой над кампаниями Novena и Chibitronics мы смогли загрузить порт NuttX на MT6260 с поддержкой минимального набора аппаратной периферии. Этого хватило, чтобы грубо воспроизвести функциональность AVR, используемого в Arduino-подобном контексте, но не намного больше.

Мы с hobs представили наши результаты на 31-м Chaos Communication Congress (CCC), и события фактически приняли неожиданный оборот, когда мы писали наше предложение. За неделю до подачи мы узнали, что MediaTek выпустила платформу разработки LinkIT ONE на основе MT2502A совместно с Seeed Studios. LinkIT ONE - это платформа интернета вещей, созданная для предпринимателей и любителей. Она интегрирована в фреймворк Arduino и имеет открытый API, обеспечивающий полную функциональность чипа, включая функции GSM. Но базовая ОС, загружающаяся на MT2502A в LinkIT ONE, по-прежнему проприетарная, и вы не можете обращаться к железу иначе как через API-вызовы, предоставляемые Arduino-shim.

На практике пройдет еще немало времени, прежде чем мы сможем перенести заметную часть функций MT6260 в область открытого исходного кода. Вполне возможно, что мы никогда не сможем сделать реализацию функций GSM-вызовов без blob-ов, поскольку они контролируются DSP-блоком, который еще более непонятен и недокументирован, чем MT6260. Учитывая мощную функциональность LinkIT ONE по сравнению с Fernvale, мы решили оставить вопрос о том, есть ли смысл продолжать реверс-инжиниринг MT6260, сообществу открытого исходного кода. В итоге энтузиазма вокруг проекта было много, но действий - не очень. Появление LinkIT ONE сильно лишило Fernvale импульса, и с тех пор проект фактически заморожен.

Такова, по сути, судьба большинства открытый исходный код-проектов. Есть десятки, если не сотни, открытый исходный код-операционных систем, но Linux только один. Правда в том, что интересных идей гораздо больше, чем способных разработчиков, которые могут их реализовать. Чтобы открытый исходный код-проект загорелся и стал самоподдерживающимся, он должен не только пройти стадию минимально жизнеспособного продукта (MVP), но и встретить восприимчивую аудиторию с реальной потребностью в проекте. Иногда ваш проект попадает в резонанс, и огромное сообщество двигает его вперед. В других случаях вы получаете множество милых, полезных наблюдателей, которые одобрительно кивают, но не хотят или слишком заняты дневной работой, чтобы включиться. А еще иногда вы кричите в пустоту или, что хуже, вас разрывают на части на каком-нибудь интернет-форуме за то, насколько ваш проект ошибочен и бессмыслен.

Заключительные мысли

Учитывая природу проектов с открытым исходным кодом, я склонен вспоминать свои стартап-дни и следовать философии «быстро ошибайся и двигайся дальше». Пробуйте много разных вещей, смотрите, что приживется, учитесь на ошибках и пробуйте снова. Важно не слишком привязываться к какой-то одной идее, особенно если идея не работает. Наконец, вы обнаружите, что полезно больше ценить путь, чем пункт назначения. Fernvale, безусловно, был эпическим путешествием; мы с xobs многому научились, отточили набор инструментов и навыков, которые продолжаем использовать до сих пор в других проектах, и, что важнее всего, отлично повеселились.

В следующей главе мы посмотрим на другой вид хакинга, который в ближайшие десятилетия будет становиться все более актуальным для всех нас: хакинг биологических систем.

Примечания

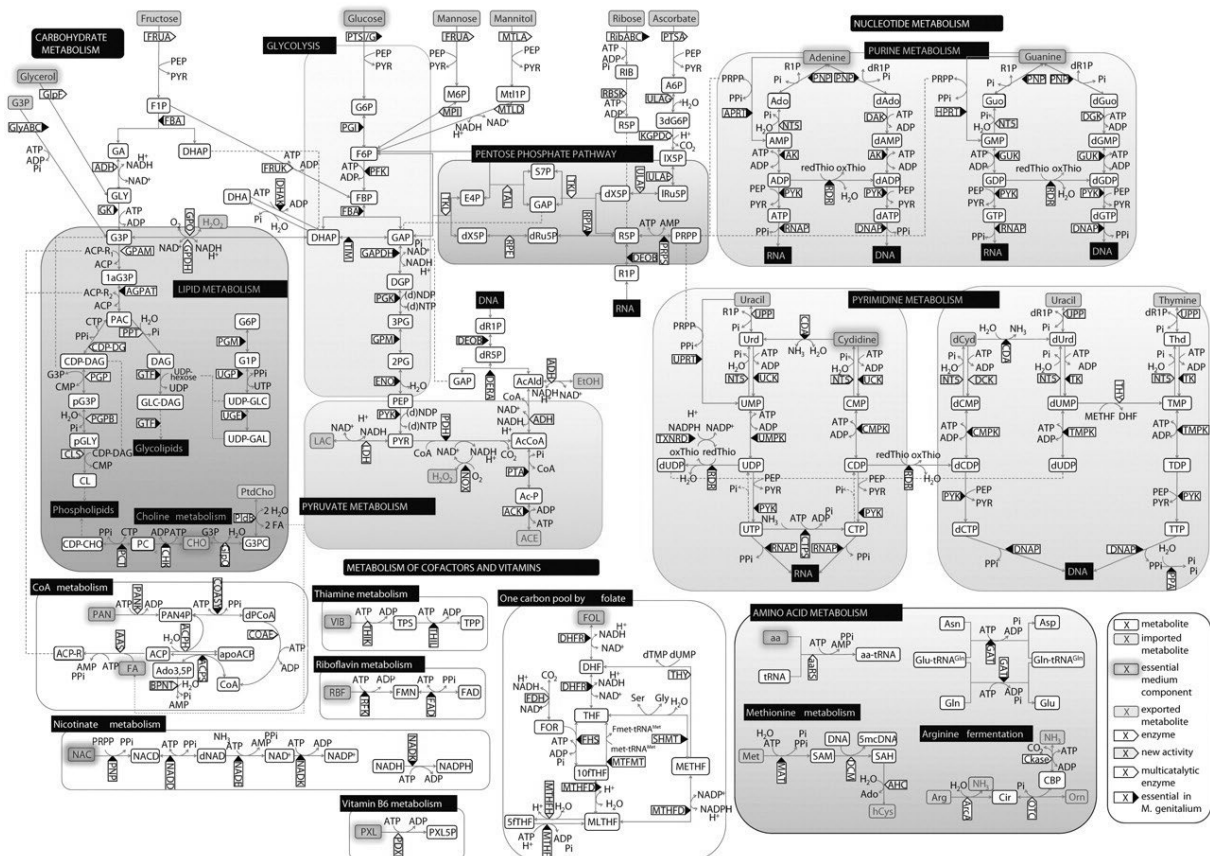
[1] You can find a copy of the code по адресу <https://github.com/xobs/ax211-code/>.

[2] К слову, мы использовали тот же гибкий адаптер, о котором я упоминал в главе 8; отчасти это и привело к разработке Chibitronics.

[3] Документацию можно прочитать в wiki Sutajio Ko-Usagi, хотя ко времени публикации оригинальный продукт NeTV, продававшийся на Adafruit, вероятно, уже будет заменен более новой и лучшей реализацией.

Глава 10. Биология и биоинформатика

Однажды я наткнулся в Science[1] на красивую диаграмму, показывающую метаболические пути одной из самых маленьких бактерий, *Mycoplasma pneumoniae*. Она напомнила мне, как я смотрел на схему Apple II, когда мне было меньше десяти лет. Тогда я знал, что завораживающе сложная масса линий на схеме Apple II была картой компьютера передо мной, хотя я еще не знал достаточно, чтобы что-нибудь сделать с этой картой. Но суть была в том, что карта существовала, и потому, несмотря на ее внушительный вид, она давала мне надежду, что я смогу распутать такие сложности. Биологические "схемы", подобные той, что на следующей странице, дают мне такую же надежду.



Метаболический путь *Mycoplasma pneumoniae*.

Развивая компьютерную аналогию, авторы статьи в Science также включили в дополнительные материалы список, который читался как BOM для *M. pneumoniae*. Пятиугольные блоки на диаграмме - это ферменты, белки, катализирующие конкретные химические реакции. Каждый фермент приведен с функциональным описанием вместе с его геной последовательностью, что эквивалентно исходному коду.

В самом конце этого списка я увидел таблицу неохарактеризованных генов. Если вы немного занимались реверс-инжинирингом, вы, вероятно, составляли похожие таблицы для деталей или вызовов функций в электронной системе. Это первое место, куда я иду за свежими подсказками, когда застреваю. Мне отраднее видеть, что биологи и хакеры применяют похожие техники к реверс-инжинирингу сложных систем.

Сравнение H1N1 с компьютерным вирусом

Сравнение биологических систем с компьютерными системами не останавливается на метаболическом уровне. Однажды я прочитал увлекательную статью в Nature[2], где патогенные компоненты нового вируса H1N1 (более известного как свиной грипп) сравнивались с компонентами других штаммов гриппа, и эта статья заставила меня задуматься о том, как соотносятся цифровые и органические вирусы. Например, насколько велик органический вирус относительно цифрового? Если поставить вопрос иначе, сколько бит нужно, чтобы убить человека или по крайней мере сделать его довольно больным? Исследуя эту идею, я нашел полезным провести несколько аналогий между цифровым и органическим мирами.

ДНК и РНК как биты

Когда в 2009 году разразилась пандемия H1N1, вирус был всесторонне секвенирован и зарегистрирован в базе данных Influenza Virus Resource Национального центра биотехнологической информации (NCBI), и собранные там данные удивительны. Мне нравится конкретность записей. Например, полная последовательность экземпляра гриппа, известного как A/Italy/49/2009(H1N1), выделенного из носа 26-летней самки *Homo sapiens*, возвращавшейся из США в Италию, находится на сайте NCBI. Вот первые 120 бит последовательности ДНК:

```
atgaaggcaa tactagtagt tctgctatat acatttgcaa cgcgaatgc agacacatta
```

При 120 битах в сумме каждый символ (А, Т, Г или С) представляет 2 бита информации. В генах это может быть альтернативно представлено как аминокислотная последовательность, где каждые три символа ДНК являются кодоном, соответствующим одной аминокислоте. Длинные цепочки аминокислот сворачиваются в сложные структуры, называемые белками, которые дают клетке структуру и функцию, а цепочки аминокислот, слишком короткие, чтобы быть полным белком, часто называются пептидами. Используя таблицу перевода, которую биологи называют стандартным генетическим кодом, я преобразовал предыдущую последовательность в следующий пептид:

```
MKAILVLLYTFATANADTL.
```

В этой последовательности каждый символ представляет аминокислоту, что эквивалентно шести битам, или трем основаниям ДНК на аминокислоту. В канонической таблице кодонов есть 20 аминокислот, и каждая буква соответствует другой аминокислоте. М - это метионин, К - лизин, А - аланин и так далее.

Теперь рассмотрим РНК, которая передает остальной части клетки информацию от ДНК о том, как синтезировать белки. Как и в ДНК, каждое основание в РНК задает один из четырех возможных символов (в этом случае А, U, G или С), так что одно основание соответствует двум битам

информации. ДНК и РНК информационно эквивалентны в отображении один к одному. Думайте о ДНК как о программе, сохраненной на диске, а о РНК - как о той же программе, загруженной в RAM. Когда ДНК загружается, инструкции синтеза белков транскрибируются в РНК, но все основания Т заменяются основаниями U.

Белки, таким образом, являются выходом выполнения РНК-программы. Белки синтезируются согласно инструкциям в РНК в отображении три к одному. Вы можете думать о белках как о пикселях во фреймбукфере следующим образом:

- Полный белок похож на изображение на экране.
- Каждая аминокислота в белке похожа на пиксель.
- Каждый пиксель имеет глубину шесть бит из-за отображения три к одному среды, которая хранит два бита на основание.
- Наконец, каждый пиксель проходит через цветовую палитру (таблицу перевода кодонов), чтобы преобразовать сырые данные в окончательный отрендеренный цвет. Однако, в отличие от компьютерного фреймбуфера, разные биологические белки различаются числом аминокислот (аналогично числу пикселей).

Чтобы привязать это к конкретному примеру, представьте, что шесть бит, сохраненные как ATG на вашем жестком диске (ДНК), загружаются в RAM (РНК) как AUG, потому что Т транскрибируется как U при переходе от ДНК к РНК. Когда РНК-программа в RAM выполняется, AUG переводится в пиксель (аминокислоту) цвета М, или метионин, который является биологическим старт-кодоном, то есть первой инструкцией в каждой действительной РНК-программе.

В качестве сокращения, поскольку ДНК и РНК эквивалентны один к одному, биоинформатики представляют генные последовательности в формате ДНК, даже если биологический механизм находится в формате РНК. Вирус гриппа имеет РНК-архитектуру, а не ДНК, и 120 бит ДНК, которые я показал ранее, соответствуют РНК-подпрограмме в вирусе гриппа. Эта подпрограмма кодирует ген НА, который производит разновидность Н1 белка гемагглютинаина. Это Н1 в обозначении свиного гриппа Н1N1.

У организмов есть уникальные порты доступа

С учетом этой фоновой информации, если вы думаете об организмах как о компьютерах с IP-адресами, каждая функциональная группа клеток в организме слушает окружающую среду через собственный активный порт. Как порт 25 конкретно отображается на SMTP-сервисы на компьютере, порт Н1 конкретно отображается на область дыхательного горла у человека. Интересно, что тот же порт Н1 отображается на кишечный тракт у птицы. Таким образом, один и тот же вирус Н1N1 будет атаковать дыхательную систему человека и кишечник птицы. В отличие от этого, Н5 - разновидность белка hemagglutinin, найденная в Н5N1, смертельном птичьем гриппе, - задает порт ваших внутренних легких. В результате Н5N1 намного смертоноснее Н1N1, потому что атакует ткань внутренних легких, вызывая тяжелую пневмонию. Н1N1 менее смертоносен, потому что атакует более безобидный порт, который просто заставляет вас часто сморкаться и откашливать мокроту.

Примечание. Исследователи все еще узнают больше о порте Н5. В статье Nature, которую я читал, указывалось, что, возможно, у некоторых человеческих мутантов легкие не реагируют на порт Н5. Люди, чьи легкие игнорируют порт Н5, имели бы больше шансов пережить инфекцию птичьего гриппа, тогда как у тех, у кого порт Н5 открыт в легких, нет шансов выжить ("make your time . . . all your base pairs are belong to H5N1").[3]

Зная, что вирус смертоносен, вы можете выяснить, сколько бит нужно, чтобы убить человека (или по крайней мере сделать его довольно больным), посчитав число бит в вирусном геноме. Тогда

вопрос таков: сколько бит в этом экземпляре H1N1? Сырое число бит, по моему подсчету, равно 26 022; число фактических кодирующих бит приблизительно 25 054. Я говорю "приблизительно", потому что в некоторых местах вирус делает эквивалент самомодифицирующегося кода, чтобы создать два белка из одного гена. Трудно сказать, что считается кодом, а что считается побочным, неисполняющимся NOP-sled-участком, требуемым для самомодифицированного кода.

Это означает, что нужно около 25 Kb, или 3,2 KB данных, чтобы закодировать вирус, имеющий нетривиальный шанс убить человека. Это эффективнее, чем компьютерный вирус вроде MyDoom, который имеет размер около 22 KB. Знание, что меня могут убить 3,2 KB генетических данных, смиряет. С другой стороны, при примерно 800 MB данных в моем геноме там наверняка найдется эксплойт-другой.

Взлом свиного гриппа

Одно интересное следствие чтения этой статьи Nature и доступа к последовательности вируса состоит в том, что теоретически теперь я знаю, как изменить последовательность вируса, чтобы сделать его смертоноснее. Например, статья Nature отмечает, что варианты гена PB2 вируса гриппа с глутаминовой кислотой в позиции 627 последовательности имеют низкую патогенность, то есть они не очень смертоносны. Однако варианты PB2 с лизином в той же позиции повышают вероятность смертности.

Посмотрим последовательность PB2 для H1N1. Вернувшись к базе данных NCBI, я нашел следующие аминокислотные последовательности вокруг позиции 627:

```
601 QQMRDVLGTFDTVQIIKLLP
621 FAAAPPEQSRMQFSSLTVNV
641 RGSGLRILVRGNSPVFNYNK
```

Числа слева указывают позицию первого символа в каждой строке последовательности; я буду следовать этому соглашению до конца обсуждения. Проверьте строку с меткой 621 и обратите внимание на E в позиции 627. E - это символ глутаминовой кислоты. К счастью, H1N1, похоже, является менее смертоносной версией гриппа; возможно, поэтому от заражения H1N1 умерло меньше людей, чем вас могли заставить поверить СМИ.

Теперь обратим это назад в ДНК-код:

```
621 F A A A P P E Q S R
1861 ttt gct gct gct cca cca gaa cag agt agg
```

Обратите внимание, что GAA кодирует E. Чтобы изменить этот геном и сделать его смертоноснее, вам просто нужно было бы заменить GAA одним из кодов лизина (K). Лизин может иметь код AAA или AAG. Таким образом, более смертоносный вариант H1N1 имел бы такую кодирующую последовательность:

```
621 F A A A P P K Q S R
1861 ttt gct gct gct cca cca aaa cag agt agg
                        ^ changed
```

Итак, изменение одной пары оснований - по сути переключение двух битов - может быть всем, что нужно, чтобы превратить вирус свиного гриппа H1N1 в более смертоносный вариант. Теоретически я мог бы применить серию хорошо известных биологических процедур, чтобы синтезировать этот штамм и фактически провести такой хак. В качестве первого шага я мог бы зайти на сайт синтеза ДНК и заказать измененную последовательность, чтобы запустить свой маленький смертоносный проект чуть дороже \$1 000. У некоторых из этих компаний есть процедуры скрининга для защиты от ДНК-последовательностей, которые можно использовать для создания биологически опасных продуктов, но даже если бы они случайно проверяли HA-варианты, существуют хорошо известные

протоколы сайт-направленного мутагенеза, которые, возможно, позволили бы изменить одно основание РНК в материале, извлеченном из нормального H1N1.

Адаптируемый грипп

Конечно, я должен отдать вирусу гриппа должное. Он упаковывает смертоносный удар в 3,2 КВ, и несмотря на лучшие усилия ученых мы его не искоренили. Может ли вирус гриппа уже самостоятельно делать хаки вроде того, который я только что описал?

Короткий ответ - да.

Фактически вирус гриппа эволюционировал так, чтобы позволять такие адаптации. Обычно, когда ДНК копируется, белок проверки ошибок проходит по скопированному геному, чтобы убедиться, что не было допущено ошибок. Это удерживает частоту ошибок довольно низкой. Но помните, что вирус гриппа использует РНК-архитектуру. Поэтому для копирования ему нужен механизм, отличный от ДНК.

Внутри своей белковой капсулы вирус гриппа упаковывает код белкового комплекса под названием РНК-зависимая РНК-полимераза, который является крошечной машиной для копирования РНК с РНК-шаблонов. Обычно РНК создается только транскрибированием ДНК, а не копированием существующего фрагмента РНК, поэтому этот механизм необходим для репликации РНК-основанного вируса гриппа. Существенно, что РНК-зависимая РНК-полимераза пропускает белок проверки ошибок, который предотвращал бы мутации. В результате вирус гриппа делает примерно одну ошибку на каждые 10 000 пар оснований, которые копируются. Геном гриппа имеет длину около 13 000 пар оснований, так что в среднем каждая копия вируса гриппа имеет одну случайную мутацию.

Некоторые из этих мутаций ничего не меняют; другие делают вирус безвредным; и вполне возможно, что некоторые делают вирус гораздо более опасным. Поскольку вирусы реплицируются и распространяются в астрономических количествах, шанс, что этот маленький хаки может в итоге произойти естественно, на самом деле довольно высок. Думаю, это одна из причин, по которой чиновники здравоохранения так беспокоились о H1N1: у людей не было к нему устойчивости, и хотя он не был настолько смертоносным, насколько мог быть, штамм, вероятно, был всего в паре мутаций от того, чтобы стать гораздо более крупной проблемой для здоровья.

Есть еще одна важная тонкость РНК-архитектуры вируса гриппа, помимо высокой частоты мутаций: генетическая информация вируса хранится как восемь отдельных, относительно коротких фрагментов РНК. Во многих других вирусах и простых организмах генетическая информация вместо этого хранится как одна непрерывная нить.

Чтобы понять, почему это важно, рассмотрим, что происходит, когда хозяин заражен двумя типами вируса гриппа одновременно. Если бы гены хранились как единый фрагмент ДНК или РНК, у генов двух типов было бы мало возможностей перемешиваться. Но поскольку грипп хранит свои гены как восемь отдельных фрагментов, эти гены свободно смешиваются внутри зараженной клетки и случайно тасуются в вирусные пакеты при выходе. Если вам настолько не повезло, что вы заразились двумя типами гриппа сразу, результатом становится потенциально новый штамм гриппа, поскольку РНК-нити копируются, смешиваются, вытаскиваются из метафорической шляпы, а затем упаковываются в вирусные частицы. Этот процесс изыщен тем, что тот же механизм позволяет смешивать произвольное число штаммов в одном хозяине. Если вы можете заразить клетку тремя или четырьмя типами гриппа сразу, результатом будет еще более дикая вариация частиц гриппа.

Этот механизм - часть причины, по которой новый H1N1 называется тройным реассортантным вирусом. Через серию двойных инфекций или, возможно, одну катастрофическую инфекцию несколькими разновидностями гриппа, новый H1N1 приобрел смесь РНК-фрагментов, которая дала ему высокие скорости передачи и сделала его тем, к чему люди не имели врожденного иммунитета. Это идеальный шторм для пандемии.

Если бы существовала компьютерная аналогия этой модели РНК-перетасовки, это был бы вирус, который распространяет себя в форме несвязанных файлов объектного кода плюс небольшая вспомогательная программа, которая при заражении хоста заново связывает свои файлы в случайном порядке перед копированием и повторным распространением. Он также искал бы похожие вирусы, которые уже могут заражать этот компьютер, и иногда связывал бы объектный код с подходящими шаблонами функций из других вирусов. Такая перестановка и новое связывание самого кода сорвали бы классы антивирусного программного обеспечения, ищущие вирусные сигнатуры на основе фиксированных шаблонов кода. Это также размножило бы в дикой среде разнообразный набор вирусов с менее предсказуемыми свойствами.

Многоуровневый механизм адаптации вируса гриппа примечателен. У вируса есть и медленно эволюционирующий механизм точечных мутаций, и механизм радикального изменения своих свойств за одно поколение через смешивание на уровне генов с другими вирусами. Это работает не совсем как секс, но результат, вероятно, столь же хорош, если не лучше. Также примечательно, что оба этих важных свойства вируса возникают как следствие использования РНК вместо ДНК в качестве носителя генетического хранения.

Светлая сторона

Поскольку существует так много вариантов гриппа, ни одна вакцина не может быть нацелена на все типы вируса, но история H1N1 все же имеет светлую сторону. По-видимому, пациент, заразившийся свиным гриппом во время пандемии, создал новое антитело с замечательной способностью давать иммунитет ко всем 16 подтипам гриппа А. Группа исследователей просеяла белые кровяные клетки пациента и сумела изолировать четыре В-клетки, содержащие код для производства этого антитела. Они клонировали клетки и произвели антитела, что облегчило дальнейшие исследования потенциальной вакцины, способной дать широкую защиту от гриппа.

Меня это очень заинтересовало на интуитивном уровне, потому что дает мне надежду: если вирус-убийца действительно уничтожит большую часть человечества, может быть, небольшая группа людей его переживет.

Реверс-инжиниринг супербактерий

В 2011 году в Европе произошла вспышка штамма-супербактерии *E. coli* (вида бактерий с подтипами, способными вызывать пищевое отравление) под названием EHEC O104:H4. Когда я узнал, что ученые из BGI, расположенного в Шэньчжэне, бесплатно выложили в интернет полную последовательность O104:H4, чтобы любой мог ее изучить, мне стало очень любопытно. Я не мог не задуматься, какие именно инструменты биоинформатики используют для анализа ДНК-последовательностей. Вручную рассматривать относительно простые последовательности вируса гриппа - это одно, но должны же быть вычислительные инструменты, помогающие осмыслить более сложные организмы вроде *E. coli*.

К счастью, моя девушка-программистка ([s/perl/girl/](#)) также является известным биоинформатиком. Она выкроила время в своем плотном расписании, чтобы показать мне некоторые инструменты

ремесла. Оказалось, что большинство инструментов для анализа ДНК свободно доступны онлайн. Поскольку ДНК - это просто последовательности А, Т, Г и С, стандартный формат обмена данными - старый добрый ASCII-текст, а значит, многое в анализе можно делать командными инструментами вроде `grep`, `sed` и `awk`.

ДНК-последовательность O104:H4

Сырые данные последовательности, предоставленные BGI, представляли собой набор избыточно выбранных подпоследовательностей, которые нам нужно было собрать, сопоставляя перекрывающиеся области. Сшивание подпоследовательностей немного похоже на составление большой картины из маленьких фотографий, сделанных случайным образом. При достаточно плотной выборке вы в конце концов создадите в основном полную картину, но изображение все равно будет иметь неоднозначности, особенно в областях с регулярными шаблонами.

Геном O104:H4 был предоставлен как список из более чем 500 000 коротких образцов ДНК. Процесс сборки сшил короткие образцы ДНК в 513 непрерывных фрагментов ДНК (контигов), с общей длиной генома 5,3 миллиона пар оснований. Организм вроде *E. coli* имеет одну большую петлю ДНК, так что существовало 513 мест, где ограничения технологии секвенирования (или просто невезение) пропустили неизвестное число пар оснований, мешая нам узнать всю непрерывную последовательность. Примечательно, что типичный штамм *E. coli*, не являющийся супербактерией, имеет около 4,6 миллиона пар оснований, так что O104:H4, вероятно, как минимум на 15 процентов длиннее. Аналогично, этому штамму потребовалось бы больше времени на репликацию, чем штамму без лекарственной устойчивости. Посмотрите на контиг 34 этой сборки:

```
AAATGGTATTCTGTTCACGATACTATTGCCAGAGTTGTATCCTGTATCAGTCCTGC
AAAATTTTCATGAGTGCTTTATTAACTGGATGCGTGACTGCCATTCTTCAGATGATAA
AGACGTCATTGCAATTGATGGAAAAACGCTCCGGCACTCTTATGACAAGAGTCGCCG
CAGGGGAGCGATTTCATGTCATTAGTGCCTTCTCAACAATGCACAGTCTGGTCATCGG
ACAGATCAAGACGGATGAGAAATCTAATGAGATTACAGCTATCCCAGAACTTCTTAA
CATGCTGGATATTAAGGAAAAATCATCAACTGATGCGATGGGTTGCCAGAAAGA
TATTGCAGAGAAGATACAAAAACAGGGAGGTGATTATTTATTCGCGGTAAGGAAA
CCAGGGGCGGCTAAATAAGCCTTTGAGGAAAAATTTCCGCTGAAAGAATTAATAA
TCCAGAGCATGACAGTTACGCAATTAGTGAAAAGAGTCACGGCAGAGAAGAAA
```

Я мог бы выбрать любой контиг, и он, вероятно, имел бы для вас примерно столько же смысла, сколько этот блок букв. Если не считать случайных поп-культурных отсылок (слово GATTACA встречается 252 раза в геноме O104:H4), сырая ДНК-последовательность не слишком информативна. Это немного похоже на созерцание бинарного машинного кода. Чтобы анализировать данные, нужно "декомпилировать" "методы", содержащиеся внутри кода.

В этом случае мы искали ДНК-последовательности, кодирующие белки. Как я упоминал ранее, белки - это сложные, часто переплетенные цепочки молекул, состоящие из маленьких строительных блоков, известных как аминокислоты. Клетки делают дела с помощью белков: одни белки превращают сахар в энергию, другие используют эту энергию, чтобы двигаться или менять форму клетки, а третьи отвечают за копирование и ремонт клетки.

К счастью, белковые последовательности в ДНК сильно консервативны. Природа склонна повторно использовать белковые структуры, с небольшими модификациями, между организмами. Поэтому функция, определенная биологическим экспериментом даже на другом виде, часто может быть соотнесена с ДНК-последовательностью. Например, один распространенный эксперимент для определения функции последовательности - вырезать кусок ДНК из клетки и наблюдать, что происходит с клеткой; потеря функции, возникающая из-за отсутствующей ДНК, часто указывает на роль белка в клетке.

Биологи накопили десятилетия исследований о том, что делают определенные белки, в огромных базах данных. Поэтому, чтобы понять, что означает фрагмент ДНК, можно выполнить нечеткое сопоставление шаблона между интересующей вас ДНК и базой данных известных белков.

Инструменты реверсинга для биологии

Чтобы выполнить реверс-инжиниринг ДНК, мне нужны были два инструмента: база данных белков и программа под названием BLASTX. Оба можно бесплатно скачать онлайн.

База данных UniProt

Я скачал список известных белков из Universal Protein Resource, или UniProt (<http://www.uniprot.org/>). В 2011 году поиск в базе данных по «лекарственной устойчивости», ограниченный организмами *E. coli*, давал список из 1 378 белков, которые ученые за годы идентифицировали как части механизма лекарственной устойчивости бактерий *E. coli*. Каждый год в базу данных добавляются новые открытия.

Вот фрагмент из базы данных, описывающий белок, который дает O104:H4 устойчивость к лекарству, которое вы можете узнать:

```
>sp|P0AD65|PBP2_ECOLI Penicillin-binding protein 2
OC=Escherichia coli (strain K12) GN=mrdA PE=3 SV=1
MKLQNSFRDYAESALFVRRALVAFLGILLTGVLIANLYNQIVRFTDYQTRSNEENRIK
LVPIAPSRGIYDRNGIPLALNRTIYQIEMMPEKVDNVQQTLDALRSVVDLTDDIAAFR
KERARSHRFTSIPVKTNLTEVQVARFAVNQYRFPGVEVKGYKRRYPYGSALTHVIGYVS
KINDKDVRLNNDGKLANYAATHDIGKLGIERYYEDVLHGQTGYEEVEVNNRGRVIRQLK
EVPPQAGHDIYLTLDLKLQYIETLLAGSRAAVVTDPRGTGGVLALVSTPSYDPNLFVDG
ISSKDYSALLNDPNTPLVNRATQGVYPPASTVKPYVAVSALSAGVITRNTTLFDPGWWQL
PGSEKRYRDWKKWGHGRNLNVTSLSEESADTFYQVAYDMGIDRLSEWMGKFGYGHYTGID
LAEERSGNMPTREWKQKRFKKPWYQGDITPVGIGQGYWTATPIQMSKALMILINDGIVKV
PHLLMSTAEDGKQVPWVQPHEPPVGDIIHSGYWELAKDGMYGVANRPNGTANHYFASAPYK
IAAKSGTAQVFLKANETYNANKIAERLRDHKLMTAFAPYNNPQVAVAMILENGGAGPAV
GTLMRQILDHIMLGDNNTDLPANPAVAAAEDH
```

PBP2_ECOLI[4] связан с устойчивостью к пенициллину и является мутировавшим геном, определяющим форму бактерии. Похоже, этот устойчивый вариант адаптировался к работе несмотря на присутствие пенициллина; бактерии с неустойчивыми формами гена не способны формировать правильные клеточные стенки в присутствии пенициллина и убиваются лекарством. Другие гены могут вызывать более активные контрмеры, например выкачивать антибиотик из клетки или модифицировать антибиотик, чтобы он был менее токсичен для клетки. Просмотр базы данных UniProt дает ощущение огромного разнообразия генов, доступных в природе, которые могут делать бактерии устойчивыми к лекарствам.

Декомпилятор

Далее мне был нужен настоящий декомпилятор. Здесь на сцену вышел BLASTX (в конечном итоге обновленный до BLAST+). BLASTX - это вариант BLAST, что означает Basic Local Alignment Search Tool. Сначала я заставил эту программу анализа вычислить все возможные трансляции ДНК *E. coli* в белковые последовательности. Трансляция ДНК дает шесть возможных белковых последовательностей: ДНК можно читать вперед и назад (известно как 5'→3' и 3'→5'), и каждое направление имеет три возможные рамки считывания. Затем я заставил программу проверить среди получившихся аминокислотных последовательностей шаблоны, совпадающие с базой данных последовательностей, известных тем, что дают лекарственную устойчивость. (Я мог бы также проверить другие типы шаблонов, введя что-то другое в запрос к базе данных.) Результатом был отсортированный список каждого известного белка лекарственной устойчивости вместе с областью генома *E. coli*, которая лучше всего совпадает с этим белком.

Ниже приведен вывод BLASTX для примера с пенициллином.

```
# BLASTX 2.2.24 [Aug-08-2010]
# Query: 43 87880
# Database: uniprot-drug-resistance-AND-organism-coli.fasta
# Fields: Query id, Subject id, % identity, alignment length,
mismatches, gap openings, q. start, q. end, s. start, s. end,
e-value, bit score
43 sp|P0AD65|PBP2_ECOLI 100.00 632 0 0 29076 30971 1 632 0.0 1281
43 sp|P0AD68|FTSI_ECOLI 25.08 650 458 21 29064 30926 6 574 2e-33 142
43 sp|P60752|MSBA_ECOLI 32.80 186 120 6 12144 12686 378 558 6e-17 87.0
43 sp|P60752|MSBA_ECOLI 27.78 216 148 5 77054 77677 361 566 8e-14 76.6
43 sp|P77265|MDLA_ECOLI 27.98 193 133 6 12141 12701 370 555 2e-10 65.5
--snip--
```

Строка Fields описывает, что показывает каждый столбец в таблице. В столбце % identity видно, что ген PBP2_ECOLI имеет 100-процентное совпадение внутри генома O104:H4.

Ответы на биологические вопросы с помощью shell-скриптов Unix

Имея этот список, я мог отвечать на некоторые интересные вопросы, например: "Сколько известных генов лекарственной устойчивости находится внутри O104:H4?" Вот однострочная программа, которую моя подруга написала, чтобы ответить на этот конкретный вопрос:

```
cat uniprot_search_m9 | awk '{if ($3 == 100) { print;}}' | \
  cut -f2 |grep -v ^# | cut -f1 -d"_" | cut -f3 -d"|" | \
  sort | uniq | wc -l
```

Вывод этого сценария сообщил нам, что 1 138 генов в O104:H4 имели 100-процентное совпадение с базой данных из 1 378 генов, которые могут давать лекарственную устойчивость. Когда мы ослабили критерии, чтобы также перечислить 99-процентные совпадения, допуская одну или две мутации на ген, список расширился до 1 224 из 1 378. Супербактерия O104:H4 заслужила свой титул, приобретя примерно 90 процентов известных генов устойчивости!

Я также хотел ответить на обратный вопрос: каких генов лекарственной устойчивости точно нет в O104:H4? Рассматривая гены устойчивости, отсутствующие у супермикроба, мы могли бы собрать подсказки о том, какие методы лечения могут быть эффективны против этого микроба.

Чтобы исключить ген лекарственной устойчивости, мы составили другой поиск, который должен был показать, какие гены устойчивости в базе данных имеют менее чем 70-процентное совпадение с последовательностью O104:H4. Порог 70 процентов был просто произвольным числом, которое я выбрал; вероятно, существует строгий стандарт, которым пользуются ученые и клиницисты.

Вот список, как он выглядел в моем терминале:

```
A0SKI3 A2I604 A3RLX9 A3RLY0 A3RLY1 A5H8A5 B0FMU1 B1A3K9 B1LGD9
B3HN85 B3HN86 B3HP88B5AG18 B6ECG5 B7MM15 B7MIF1 B7NQ58 B7NQ59
B7TR24 BLR CML D2I9F6 D5D1U9 D5D1Z3 D5KLY6 D6JAN9 D7XST0 D7Z7R4
D7Z7W9 D7ZDQ3 D7ZDQ4 D8BAY2 D8BEX8 D8BEX9 D9R21 D9R22 D9R23
E0QC79 E0QC80 E0QE33 E0QF09 E0QF10 E0QYN4 E1J2I1 E1S2P1 E1S2P2
E1S382 E3PYR0 E3интерфейс84 E3XPK9 E3XPQ2 E4P490 E5ZP70 E6A4R5 E6A4R6
E6ASX0 E6AT17 E6B2K3 E6BS59 E7JQV0 E7JQZ4 E7U5T3 E9U1P2 E9UGM7
E9VGQ2 E9VX03 E9Y7L7 085667 Q05172 Q08JA7 Q0PH37 Q0T948 Q0T949
Q0TI28 Q1R2Q2 Q1R2Q3 Q3HNE8 Q4HG53 Q4HG54 Q4HGV8 Q4HGV9 Q4HH67
Q4U1X2 Q4U1X5 Q50JE7 Q51348 Q56QZ5 Q56QZ8 Q5DUC3 Q5UNL3 Q6PMN4
Q6RGG1 Q6RGG2 Q75WM3 Q79CI3 Q79D79 Q79DQ2 Q79DX9 Q79IE6 Q79JG0
Q7BNC7 Q83TT7 Q83ZP7 Q8G9W6 Q8G9W7 Q8GJ08 Q8VNN1 Q93MZ2 Q99399
Q9F0D9 Q9F0S4 Q9F7C0 Q9F8W2 Q9L798
```

Вы можете вставить любой из этих белковых кодов в базу данных UniProt и узнать о нем больше. Например, BLR - это бета-лактамаза, фермент, вызывающий устойчивость к бета-лактамам антибиотикам. UniProt описывает его так:

```
Has an effect on the susceptibility to a number of antibiotics involved
in peptidoglycan biosynthesis. Acts with beta lactams, D-cycloserine
and bacitracin. Has no effect on the susceptibility to tetracycline,
chloramphenicol, gentamicin, fosfomycin, vacomycin or quinolones.
Might enhance drug exit by being part of multisubunit efflux pump.
Might also be involved in cell wall biosynthesis.
```

К сожалению, беглый осмотр показал, что большинство функций, которых O104:H4 не хватало, были просто маленькими, плохо понятыми фрагментами машин, участвующих в лекарственной устойчивости. В результате в его геноме не было очевидного кандидата на убийцу супербактерии.

Больше вопросов, чем ответов

Хорошая новость в том, что любой может получить доступ к инструментам анализа геномов, а некоторые инструменты, такие как `grep`, `awk` и `sed`, уже знакомы компьютерным инженерам. Плохая новость в том, что хотя мы можем задавать вопросы о геноме с помощью этих инструментов, у нас все равно остается больше вопросов, чем ответов. Например, устойчивость к антибиотикам кажется полезной для выживания бактерий, так почему же она есть не у всех бактерий? И как бактерии приобретают (или теряют) такие гены?

Появление устойчивой к антибиотикам супербактерии - следствие нашей любви к антибиотикам. Поскольку ДНК в *E. coli* копируется со скоростью около дюжины пар оснований в секунду, избавление даже от одного неиспользуемого гена может дать значимое преимущество в гонке экспоненциального роста; в конце концов, популяция *E. coli* в оптимальных условиях может удваиваться каждые 20 минут. В результате естественный отбор благоприятствует избавлению от генов, не нужных для выживания. Геном O104:H4 на 15 процентов длиннее, чем у типичного штамма *E. coli*, а это означает, что через семь поколений типичный штамм *E. coli* имел бы вдвое большую популяцию, чем O104:H4. За половину дня в оптимальных условиях роста без антибиотиков штамм *E. coli*, не обремененный генами устойчивости к антибиотикам, имел бы более чем в 20 раз большую популяцию, чем O104:H4. Поэтому бактерия, сохраняющая гены устойчивости к антибиотикам, похожа на спринтера, который выходит на старт в бронежилете. Точно так же одна из величайших природных угроз для супербактерий - компактная, быстро размножающаяся обычная бактерия, которая может вытеснить супербактерию просто за счет численного превосходства.

Однако бактерицидные и бактериостатические антибиотики соответственно убивают или подавляют рост неустойчивых бактерий, оставляя только устойчивые бактерии расти беспрепятственно. Со временем и при воздействии нескольких типов антибиотиков разумно ожидать, что популяция устойчивых бактерий продолжит селективно размножаться ради множества генов устойчивости, создавая супербактерию.

И все же меня поражает, что устойчивые бактерии, похоже, развивают гены устойчивости так быстро. Нас учат, что эволюция - медленный процесс, поэтому кажется удивительным, что бактерии могут удачно накапливать целый набор генов устойчивости к антибиотикам общей длиной в сотни тысяч пар оснований. Новые гены действительно возникают спонтанно очень долго (существует очень мало ясно документированных случаев этого, таких как Long-Term Evolution Experiment Ричарда Ленски). Вместо этого большинство генов устойчивости приобретается из окружающей среды через горизонтальный перенос генов.

Наша окружающая среда кишит фрагментами ДНК. GitHub биологии находится повсюду вокруг нас: в грязи, в море и в воздухе, которым мы дышим. Некоторые фрагменты ДНК кодируют полезные признаки; некоторые являются просто мусором. Когда бактерия находится под стрессом (как при воздействии антибиотиков), она может начать поглощать случайные фрагменты ДНК из окружающей среды и производить белки на основе этого кода. Если ей все равно умирать, почему бы и нет? Большую часть времени включенные фрагменты ДНК не помогают, но если одной удачливой бактерии удастся подобрать из окружающей среды нужный ген устойчивости, она сможет быстро превзойти других в среде, насыщенной антибиотиками.

Таким образом, хотя неустойчивые штаммы бактерий быстро превзойдут числом устойчивые к антибиотикам штаммы, крошечная оставшаяся популяция устойчивых бактерий (или, возможно, даже их безжизненные тела, плавающие в окружающей среде) образует резервуар генетического материала, который можно привлечь во времена стресса. И поскольку генетический код совместим между всеми видами, гены устойчивости можно приобретать даже от неродственных организмов.

Обнаружение того, что функции, которых не хватало O104:H4, были плохо поняты, само по себе стало интересным уроком. Художественная литература популяризирует представление, что знание ДНК-последовательности - это то же самое, что знание того, какие болезни или признаки может иметь организм. Но хотя мы знаем последовательности и общие свойства многих белков, гораздо труднее связать белки с конкретной болезнью или признаком. В какой-то момент кому-то приходится пачкать руки и делать биологические эксперименты с реальными организмами, чтобы присвоить биологическое значение данному семейству белков.

Поп-культурные отсылки к ДНК-анализу беспечно не осознают это недостающее звено процесса, что приводит к завышенным ожиданиям от генетического анализа, особенно в его полезности для диагностики и лечения человеческих болезней и применений в евгенике. Давайте внимательнее посмотрим на некоторые из этих мифов.

Развенчание мифов персонализированной геномики

Мы определенно живем в Будущем во многих отношениях. Например, у нас есть электромобили! Но голливудские ролики 60-х и 70-х также предсказывали, что к настоящему времени я буду перемещаться по городу на летающем автомобиле, а не только на электромобиле по земле. Конечно, автомобильная технология - не единственная жертва голливудского хайпа.

Потенциальное влияние персональной геномики сильно преувеличено в фильмах вроде GATTACA, которые создают миф, что ваш геном похож на хрустальный шар и что каким-то образом ваша судьба предопределена вашей генетической программой. Моя девушка, о которой я упоминал

ранее, была соавтором статьи в Nature[5], исследовавшей предложения персональной геномики напрямую для потребителя (DTC) от 23andMe. Давайте посмотрим на ее статью, и пусть развенчание мифов начнется!

Миф: прочитав ваш геном - это как шестнадцатеричный дамп ROM вашего компьютера

Недорогая техника для просмотра частей генома называется генотипированием. Здесь выполняется выборочное сравнение между вашим геномом и эталонным человеческим геномом; иными словами, ваш геном проверяется только в потенциально интересных местах на одноточечные мутации, называемые однонуклеотидными полиморфизмами (SNP, произносится "снипы"). Концепция генотипирования естественно ведет к двум вопросам. Во-первых, как решить, какие SNP достаточно интересны, чтобы включить в выборку? И во-вторых, откуда вы знаете, что эталонный геном является надежной базой для сравнения? Это подводит к еще двум развенчанным мифам.

Миф: мы знаем, какие мутации предсказывают болезнь

Некоторые мутации в человеческом геноме просто коррелируют с болезнью; не доказано, что они являются предиктивными или причинными. По правде говоря, мы действительно не понимаем, почему возникают многие генетические заболевания. Для плохо понятых болезней все, что мы можем сказать, - это что люди, имеющие конкретную болезнь, обычно имеют определенный шаблон SNP. Важно не путать причинность с корреляцией.

Таким образом, хотя ученые могут делать предсказания о болезнях на основе SNP, большинство этих предсказаний коррелятивны, а не причинны (и к тому же слабо коррелятивны). В результате генотип не следует считать хрустальным шаром для предсказания вашего будущего в отношении болезней. Скорее, он ближе к пятну Роршаха, на которое нужно некоторое время щуриться и смотреть, прежде чем вы сможете сказать, что оно означает. Например, в статье, которую написала моя девушка, она обнаружила, что компании часто не совпадали в своих предсказаниях риска болезней, потому что по-разному интерпретировали значения мутаций.

Миф: референсный геном - это точный эталон

Слово "референсный" в выражении "референсный геном" должно подсказать вам проблему: оно подразумевает существование "референсных людей". В конечном счете для создания сегодняшнего референсного генома были секвенированы всего несколько индивидуумов, и большинство из них имеют европейское происхождение. Со временем, по мере сбора большего количества полнопоследовательных генетических данных, референсный геном будет объединяться и подгоняться, чтобы представлять более точную картину всей человеческой расы, но сейчас важно помнить, что генотипирование - это diff относительно исходного репозитория сомнительной универсальной валидности.

Например, некоторые SNP имеют разную частоту в разных популяциях. Основание А может доминировать в европейской популяции, но в той же позиции в африканской популяции может доминировать основание G. Также важно помнить, что референсный геном имеет совокупную частоту ошибок около 1 ошибки на 10 000 пар оснований, хотя, справедливости ради, процесс обнаружения варианта болезни обычно очищает любые ошибки в референсном геноме для соответствующих областей последовательности.

Пройдут десятилетия, прежде чем у нас будет полное понимание того, что означают все последовательности в человеческом геноме, и даже тогда они могут не быть по-настоящему предиктивными для риска болезни или чего-либо еще в нашем здоровье. Здесь, пожалуй, находится самое важное сообщение и пункт, который я не могу подчеркнуть достаточно: в большинстве ситуаций окружающая среда имеет больше отношения к тому, кто вы, кем вы станете и какие болезни у вас будут, чем ваши гены. Любая польза персональной геномики будет связана не с предсказаниями в стиле хрустального шара, а скорее с тем, что знание о собственных генетических предрасположенностях может побудить больше людей изменить образ жизни так, чтобы оставаться здоровыми. Если я чему-то научился, встречаясь с выдающимся биоинформатиком, так это тому, что независимо от вашего генетического состава большинство распространенных болезней можно предотвратить или задержать правильным питанием и упражнениями.

Патчинг генома

До сих пор в этой главе я приводил примеры секвенирования и анализа геномов. Это более или менее эквивалент возможности снять дампы исполняемой программы и проанализировать ее в IDA. Часто после анализа исполняемого файла вы захотите пропатчить его, чтобы он делал что-то новое. Патчинг программного обеспечения относительно прямолинеен и надежен: просто запустите hex-редактор и измените файл. В худшем случае вам может понадобиться использовать фокусированный ионный луч (FIB), чтобы изменить отдельные провода масочной ROM внутри чипа.

Но исторически возможность патчить геном была сильно ограничена. Информация в клетках хранится на молекулярном уровне, и изменение конкретной части гена может быть кропотливым процессом. Так же как электронные лампы и транзисторы предшествовали интегральным схемам, нуклеазы цинковых пальцев (ZFNs) и TALEN-нуклеазы (TALENs) сделали редактирование генов возможным, но с серьезными оговорками по эффективности, производительности и в конечном счете стоимости. В 2012 году появилась своего рода интегральная схема редактирования генов: система CRISPR/Cas.[6]

CRISPR в бактериях

CRISPR, сокращение от "кластеризованные регулярно перемежающиеся короткие палиндромные повторы", описывает определенную РНК-структуру, тогда как Cas - это белки, ассоциированные с CRISPR-ами. CRISPR-ы, насколько известно биологам, распространены только у бактерий и архей (например, грибов), и они являются частью дьявольски хитрой системы иммунитета простых организмов. Как и у людей, у бактерий есть иммунные системы, которые могут программироваться через воздействие патогенов. Когда бактерии встречают вирусного захватчика, у них есть белки, способные вырезать короткие последовательности вирусной ДНК и архивировать эти последовательности как вставки в CRISPR.

Лаборатории, которые месяцами безуспешно редактировали ген с помощью TALEN, переключились на CRISPR/Cas и преуспели с первой попытки. Они преуспели так быстро потому, что процесс просто включает проектирование короткого фрагмента РНК, вставляемого в CRISPR, - простое упражнение, которое можно выполнить целиком на компьютере или, осмелюсь сказать, вручную. Сам РНК-фрагмент можно изготовить примерно за неделю дешевле чем за \$50 у одного из нескольких поставщиков услуг, заменив значительный объем лабораторной сложности упражнением в информатике.

Каждая CRISPR-область помечена лидерной последовательностью, за которой сразу следует собственно CRISPR. Сам CRISPR состоит из направляющей РНК (гРНК), или "спейсерной" последовательности, ограниченной хорошо определенной ДНК-последовательностью прямого повтора, которая является палиндромной.

Примечание. Термин "спейсер" (spacer) используется при обсуждении иммунной системы, тогда как "направляющая РНК" (guide RNA) используется при обсуждении редактирования генома. Называть интересующую область спейсером сбивает с толку, но при обратной разработке названия нередко оказываются неудачными. Я не могу винить ученых за то, что они сначала заметили шаблон повторяющихся разделителей и назвали то, что находится между разделителями, "спейсерами". В конце концов, физики задали соглашение о направлении тока наоборот и остались с ним. Кто мы такие, чтобы судить?

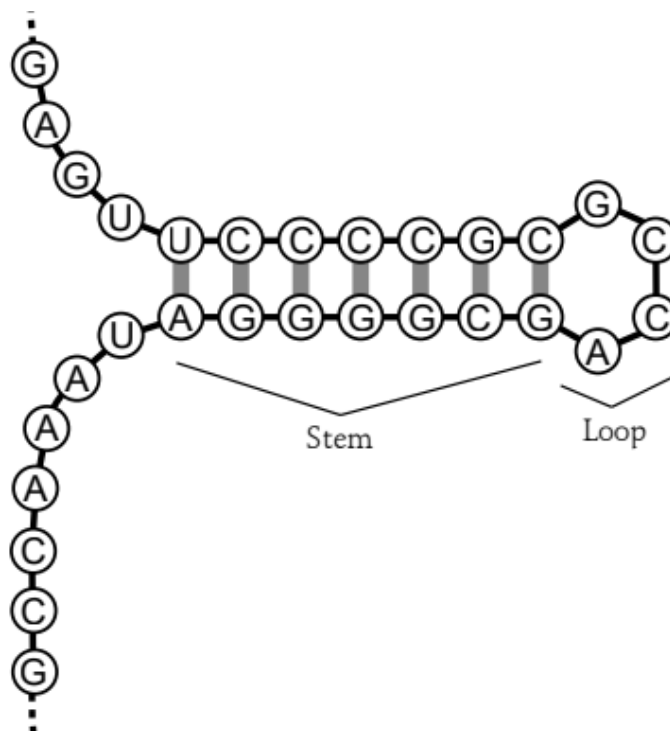
Палиндромность обычно означает, что строка эквивалентна при простом обращении, как английское слово *gasesag* («гоночная машина»). Когда биологи говорят, что последовательность «палиндромна», они имеют в виду, что последовательность эквивалентна, если ее сначала дополнить (A->T, T->A, G->C, C->G), а затем обратить. Например, GAATTC считается биологически палиндромной, хотя лексически она не палиндромна.

Система CRISPR/Cas была описана вскоре после гибели Chumby, и в то время я стажировался в лаборатории инфекционных заболеваний доктора Swaine Chen в Genome Institute of Singapore. Среди прочего я изучал различные штаммы *E. coli*, вызывающие инфекцию мочевыводящих путей, под руководством Lu Ting Liow. Помогая исследованию участков ДНК фагового вируса, которые нашли путь в *E. coli*, я получил просьбу написать сценарий для идентификации палиндромных и повторяющихся последовательностей ДНК в геноме *E. coli*. Мой сценарий показал, что геном усеян такими последовательностями; я решил, что в коде есть ошибка, и не придал результату большого значения. Но, возможно, некоторые из прямых повторов, которые я видел, были участками CRISPR.

Теперь посмотрим на CRISPR из штамма *E. coli*. Это CRISPR последовательность прямого повтора для *E. coli* O104:H4:

```
GAGTTCCCCGCGCCAGCGGGGATAAACCG
```

Выделенные жирным пары оснований являются палиндромными областями. Когда эта ДНК-последовательность транслируется в РНК (так что T->U), палиндромная область может спариваться сама с собой, образуя шпильку, как показано здесь.



A stem loop

Шпилечная петля.

Эта форма намекает на значение повторяющихся палиндромных структур в CRISPR: при трансляции в РНК последовательность может свернуться сама на себя, формируя вторичную структуру. Важно помнить, что гены - это не просто строки кода; это физические молекулы, общая форма которых существенно влияет на их функцию. Биологи используют четырехуровневую систему для описания физической структуры, которую молекулы вроде ДНК, РНК и белков могут принимать на основе своего исходного кода. Первичная структура - это просто последовательность мономеров (оснований или аминокислот). Вторичная структура относится к физическим формам, возникающим из локализованных взаимодействий мономеров из-за физических свойств, таких как расстояние и число водородных связей между молекулами или сродство определенных мономеров к воде. В РНК и ДНК это означает структуры вроде шпилечных петель; в белках это означает структуры вроде спиралей и листов. Третичная структура относится к сложной 3D-форме молекулы, возникающей из дальнедействующих взаимодействий между потенциально удаленными участками первичной последовательности. Третичная структура особенно применима к белкам, поскольку некоторые аминокислоты, такие как цистеин, могут сшиваться друг с другом на больших расстояниях. Четвертичная структура относится к структурам, сформированным взаимодействием нескольких молекул. Комплекс Cas9/РНК является примером четвертичной структуры. Итоговая химически активная и нацеленная молекула возникает только тогда, когда белок Cas9 объединяется с гРНК, и вторичная структура шпилечной петли у гРНК необходима, чтобы Cas9 ее распознал.

Определение того, где резать ген

РНК, полученная из области CRISPR через транскрипции, включается в белковый комплекс с другими Cas-белками. Конкретные Cas-белки (такие как Cas9) используют РНК как шаблон поиска и уничтожения: комплексы Cas9/РНК плавают по клетке, и когда они находят ДНК-последовательность, совпадающую с РНК-шаблоном, они выборочно режут ДНК в месте шаблона, фактически нейтрализуя вторгшийся вирус. Но вы, возможно, заметили проблему рекурсии: комплекс Cas9/РНК должен также разрезать область CRISPR в геноме организма-хозяина, поскольку эта область тоже имеет целевой шаблон. Это фактически уничтожило бы область CRISPR для будущего использования.

Чтобы избежать уничтожения области CRISPR, комплекс Cas9/РНК нацеливается на шаблонную ДНК плюс короткую, определенную последовательность из трех-пяти пар оснований, называемую мотивом, прилегающим к протоспейсеру (PAM). Например, PAM для популярного белка Cas9 из *S. pyogenes* записывается как [AGTC]GG в формате регулярного выражения; биологи используют другое соглашение, NGG, чтобы сказать то же самое. Пока архив CRISPR не включает последовательность PAM, он не будет разрезан комплексом.

Требование PAM означает, что существуют некоторые ограничения на то, где можно резать ген. Это немного похоже на нацеливание только на шестнадцатеричные строки, заканчивающиеся на 0xC3, или на поиск ROP-гаджетов. Так же как хакеры, ищущие ROP-гаджеты, ищут короткие последовательности инструкций, заканчивающиеся кодом операции RET, биоинформатики должны искать короткие последовательности ДНК для редактирования, которые заканчиваются PAM.

Несмотря на эти ограничения, CRISPR/Cas оказался универсальным и надежным инструментом редактирования генов. Он был адаптирован и для разрезания генов, и для вставки новых последовательностей. Сделать точный разрез в произвольном месте ДНК - самый трудный шаг вставки новой ДНК. Но в сочетании с хорошо изученными техниками вроде негомологичного соединения концов (non-homologous end joining, NHEJ) или гомологически направленной репарации (homology-directed repair, HDR), CRISPR/Cas можно использовать для внесения модификаций в ген.

Последствия для инженерии людей

Хотя CRISPR/Cas - естественная система, найденная у бактерий и грибов, универсальный генетический код означает, что система бинарно совместима со всеми видами, включая людей. До открытия этой системы гены были в основном доступны только для чтения, особенно в живых организмах. CRISPR/Cas дает нам гораздо более надежный и эффективный инструмент для патчинга и ремонта генов без обязательного нарушения жизнеспособности организма-хозяина. Биологам удалось упаковать необходимую ДНК для CRISPR/Cas-эксплойта в вирусы, что позволяет им протаскивать эти инструменты редактирования генов через клеточные стенки живых, сложных организмов, таких как мыши, растения и люди. Структура CRISPR также позволяет ученым выполнять несколько правок в одном эксперименте, расширяя экспериментальную и терапевтическую универсальность техники.

Эта технология уже была проверена на человеческих клетках, даже на человеческих эмбрионах, и последствия просто поражают воображение. Независимо от этических стандартов, установленных научным и юридическим сообществами в вашей стране проживания, я думаю, что обещание специально спроектированных детей, свободных от генетических болезней, некогда мучивших родителей, является слишком сильным искушением. Даже если большинство стран запретит такую практику, мне кажется неизбежным, что кто-то, где-то, возможно финансируемый богатым миллиардером, неспособным иметь собственных жизнеспособных детей, начнет возиться с генетически спроектированными людьми. Если результаты будут положительными, это, вероятно,

изменит ход человечества глубже, чем закон Мура. И это если механизм под названием "генный драйв" не доберется туда первым.

Взлом эволюции с помощью генного драйва

Генный драйв переписывает правила полового размножения и, следовательно, эволюции способом, прежде не виденным в природе. Вы, возможно, знаете, что у вас есть две копии каждого гена: одна от матери и одна от отца. Каждая копия - это аллель. Если аллели совпадают, о вас говорят, что вы гомозиготны по этому гену. Если аллели различаются, вы гетерозиготны по нему. Обычно то, какой аллель ребенок получает от каждого родителя, является броском монеты, а приспособленность ребенка в данной среде - главный решающий фактор для передачи набора аллелей новому поколению.

Генный драйв устраняет этот бросок монеты. Естественный отбор коротко замыкается, позволяя генам с потенциально негативными побочными эффектами быстро распространяться в популяции. Этот эксплойт становится возможен благодаря оснащению желаемого аллеля механизмом редактирования генов с помощью CRISPR/Cas, который нацеливается на гетерозиготный аллель и превращает его в гомозиготный. Например, если у матери есть ген, оснащенный механизмом генного драйва с помощью CRISPR/Cas, не имеет значения, какие гены у отца. Внутри ребенка материнская копия выразит механизмы редактирования CRISPR/Cas, найдет отцовскую копию и отредактирует ее так, чтобы она стала такой же, как материнская.

С точки зрения разрушительной силы, если CRISPR/Cas - это команда gm , то генный драйв похож на вызов $gm -r *$.

Это оказывает глубокое влияние на естественный отбор. Забудьте "выживание наиболее приспособленных"; изменениям больше не обязательно строго повышать приспособленность организма, чтобы распространяться по популяции. Более того, изменения, управляемые генным драйвом, могут проходить через природную популяцию с экспоненциальной скоростью (гораздо быстрее типичных мутаций), потому что они не полагаются на броски монеты и естественный отбор для усиления мутации.

С положительной стороны, генный драйв можно использовать, чтобы принудительно внести в мир хорошие изменения, например создать комаров, свободных от малярии. С отрицательной стороны, этот новый механизм, прежде не виденный в природе, может устроить хаос в эволюции и экосистеме. Хотя наши изменения могут быть хорошо спроектированы и иметь добрые намерения, природа любит встряхивать вещи через мутации, спонтанные перестройки и горизонтальный перенос генов. Если организм с генным драйвом подхватит дополнительные гены в полезной области, исход может быть непредсказуемым.

Например, комары, свободные от малярии, принесут пользу людям, но комары также играют большую роль в экосистеме Земли как источник пищи для рыб и птиц. Если модифицированные комары не смогут процветать и занимать свою экологическую нишу, может возникнуть эффект домино, вредящий другим видам. Все это может произойти на таком коротком временном масштабе, что мы, возможно, не сможем обратить это вспять, даже если попытаемся. Более того, организмы вроде комаров не признают геополитических границ. Поэтому запрет генного драйва в большей части мира не делает никого защищенным от его потенциальных последствий. Если всего один хорошо спроектированный организм попадет в дикую природу, всем придется иметь с ним дело.

Возможно, не случайно CRISPR/Cas был найден только у бактерий и архей - организмов, которые, как известно, размножаются бесполым путем. Возможно, способность коротко замыкать

требование приспособленности при половом размножении быстро ухудшает общую приспособленность любой зародышевой линии, несущей мутацию CRISPR/Cas, так что линия вымирает прежде, чем успевает захватить популяцию. В конце концов, любые случайные гены или спонтанная мутация, попавшие в полезную нагрузку CRISPR/Cas, также пройдут по популяции так же быстро, как первоначальный драйв.

Тогда вопрос в том, сколько времени требуется, чтобы это ухудшение и вымирание произошли? Пример искоренения переносчиков малярии имел бы совершенно иной исход, если бы модифицированные комары вымерли в течение нескольких лет, а не нескольких тысячелетий.

Заключительные мысли

Очевидно, на фронтире биологической инженерии много вопросов без ответа, и все это происходит прямо сейчас. Хороший или плохой, результат сегодняшних экспериментов, вероятно, повлияет на человечество так же глубоко, как закон Мура и интернет. Электронная технология изменила то, как мы думаем и общаемся, а биотехнология изменит наши тела и нашу среду. Большая разница в том, что в биотехнологии мы еще не развили способность делать резервные копии, но развиваем технологию с потенциальной силой команды $r_m - r^*$.

Лично я оптимистичен; думаю, эти технологии могут и будут использоваться для улучшения нашей жизни. Но чтобы это произошло, обществу нужно понимать стоящие на кону вопросы и вести энергичные и открытые дебаты. Даже если эти биологические техники имеют пугающие последствия для нашего здоровья и безопасности, неспособность раскрывать и обсуждать уязвимости просто приглашает уязвимости нулевого дня. А кто хочет однажды утром проснуться, зараженным калечащей вредоносной программой, без жизнеспособного патча?

Аппаратные прорывы изменили нашу жизнь в ее привычном виде, но закон Мура замедляется, а секвенирование ДНК его обогнало. Кто знает, какой новый мир будет создан достижениями биотехнологий? И так же как общество выигрывает от ответственного раскрытия и обмена уязвимостями и эксплоитами, участие в научном дискурсе конструктивнее, чем попытка его цензурировать. Возможно, опыт и перспективы, полученные при взрослении аппаратной индустрии за последние 50 лет от карманных калькуляторов до карманных суперкомпьютеров, помогут направить биотехнологии к столь же положительному исходу.

Примечания

- [1] Eva Yus et al., "Impact of Genome Reduction on Bacterial Metabolism and Its Regulation," Science 326, no. 5957 (2009): 1263-1268, <http://science.sciencemag.org/content/326/5957/1263/>.
- [2] Gabriele Neumann, Takeshi Noda, and Yoshihiro Kawaoka, "Emergence and Pandemic Potential of Swine-Origin H1N1 Influenza Virus," Nature 459, no. 7249 (2009): 931-939, <http://www.nature.com/nature/journal/v459/n7249/full/nature08157.html>.
- [3] Если вам незнаком этот оборот, см. https://en.wikipedia.org/wiki/All_your_base_are_belong_to_us.
- [4] Кстати, меня забавляет, что последовательность RBP2 короче, например, моего блока публичного ключа PGP.
- [5] P.C. Ng et al., "An Agenda for Personalized Medicine," Nature 461, no. 7265 (2009): 724-726, <http://www.nature.com/nature/journal/v461/n7265/full/461724a.html>.
- [6] У Addgene есть отличный технический обзор, очень подробно описывающий эту систему. Рекомендую заглянуть туда, если мой беглый обзор разбудил ваш аппетит: <https://www.addgene.org/CRISPR/guide/>.

Глава 11. Избранные интервью

За эти годы я дал несколько интервью, и эта глава собирает пару из них, которые, как я подумал, могут вам понравиться. Первое интервью изначально было опубликовано China Software Developer Network (CSDN), которая описывает себя как "журнал программистов". В конце вы найдете историю из Blueprint, сборника интервью с основателями и новаторами в аппаратуре.

Эндрю "bunnie" Хуан: Hardware Hacker (CSDN)

Это интервью изначально появилось в CSDN на китайском языке в 2013 году, и журнал любезно разрешил мне опубликовать английский перевод в моем блоге. В первой части я обсуждал свои мысли о движении мейкеров, которое в то время было относительно новым, и свой опыт создания аппаратных продуктов. Вторая часть больше посвящалась аппаратному хакингу и тому, что, как я чувствую, означает иметь хакерский дух. Оригинальную китайскоязычную версию можно найти по адресу <http://www.csdn.net/article/2013-07-03/2816095>.

Об открытой аппаратуре и движении мейкеров

Движение мейкеров и движение открытой аппаратуры привлекли много внимания. Крис Андерсон написал книгу под названием Makers, а Пол Грэм назвал это время "аппаратным Ренессансом". Как, по вашему мнению, это движение повлияет на обычных людей, разработчиков и нашу IT-индустрию?

Это движение, если его так можно назвать, на мой взгляд, скорее симптом, чем причина. Сначала давайте вспомним, как мы пришли к этому моменту.

В 1960 году существовала только аппаратура, и вся она была открытой. Когда вы покупали транзисторный радиоприемник, сзади у него была напечатана схема. Если радио ломалось, вам приходилось чинить его самому. Было популярно покупать наборы, чтобы собирать собственные радиоприемники.

Между 1980 и 1990 годами началась революция персональных компьютеров. Компьютеры стали достаточно мощными, чтобы выполнять программное обеспечение, которое было интересным и давало новые возможности.

С 1990 по 2005 год закон Мура заставлял компьютеры становиться вдвое быстрее и иметь вдвое больше памяти каждые 1,5-2 года. Значение имело только программное обеспечение, потому что если вы не могли позволить себе изготовить чип по новейшей технологии, заниматься аппаратурой не стоило. К тому времени, когда вы собирали компоненты вместе, новый чип уже делал вашу разработку медленной. Оптимизация программного обеспечения также значила меньше, чем функции, удобство и креативность. Пользователи могли просто купить более быстрый компьютер и запускать старое программное обеспечение быстрее. Мейкерство вышло из моды, потому что на него не было времени: нужно было "ship code or die" - выпускать код или умирать.

С 2005 по 2010 год компьютеры не стали намного быстрее с точки зрения тактовой частоты, но стали меньше. Родились смартфоны. Все стало приложением, и все по-прежнему становится более подключенным.

Примерно с 2010 года до настоящего времени закон Мура замедляется. Это замедление волнами расходится по цепочке инноваций. PC уже не становятся заметно быстрее, лучше или дешевле. Мы

покупаем новые PC просто чтобы заменить сломанные, а не потому что последняя модель настолько лучше. Еще слишком рано говорить уверенно, но смартфоны тоже, возможно, становятся зрелой платформой: iPhone 5 довольно похож на iPhone 4, и телефоны Samsung тоже выглядят довольно похожими от версии к версии.

Тогда вопрос в том, как внедрять инновации? Как отличаться на рынке? При замедлении закона Мура можно внедрять инновации в аппаратуре, и ваша идея не устареет только потому, что вышел новый чип. У вас есть устойчивые платформы (PC, смартфоны, планшеты), под которые можно подстраивать свои аппаратные идеи. Вам не нужно изготавливать чипы просто чтобы получить преимущество. Теперь все перебирают старые технологии в поисках упущенных ниш. Даже устаревшая материнская плата смартфона выглядит потрясающе, если поставить ее в квадрокоптер, спутник, HVAC-систему, автомобиль, систему мониторинга энергии, систему мониторинга здоровья и так далее.

Более того, как люди мы фундаментально по-разному относимся к физическим вещам и виртуальным вещам. Приложения замечательны, но человеческие дома - это больше, чем смартфон, поднос с едой, кровать и туалет. Люди все еще окружают себя безделушками, фотографиями друзей и физическими подарками по особым случаям. Я не думаю, что когда-либо наступит время, когда приложение с виртуальным плюшевым медведем вытеснит физического плюшевого медведя для обнимания ночью.

В результате всегда будет место для людей, делающих аппаратуру, которая удовлетворяет эту потребность в осязаемых товарах. Эта аппаратура будет объединять больше технологий и выполнять больше программного обеспечения, но в конце концов существует пространство для мейкеров и аппаратных стартапов, и это пространство сейчас только расширяется, поскольку аппаратная технология стабилизируется.

Arduino и Raspberry Pi, похоже, снижают порог проектирования аппаратуры. Как, по вашему мнению, это повлияет на аппаратную индустрию? Думаете ли вы, что эти платформы продвинут индустрию семимильными шагами? Если нет, что нужно, чтобы сделать действительно инновационный аппаратный продукт?

Arduino и Raspberry Pi обслуживают конкретные рыночные ниши.

Ключевой вклад Arduino - сведение вычислений к простой в использовании физической форме. Он был сделан прежде всего дизайнерами и художниками, а не столько технологами. Этот уникальный взгляд на технологию очень силен, потому что люди, которые не являются программистами или разработчиками аппаратуры, тоже хотят получить доступ к аппаратной технологии. С помощью Arduino были сделаны очень трогательные, глубокие интерактивные художественные работы, позволяющие аппаратуре превращать приземленные управляющие приложения в произведения искусства, которые меняют ваше настроение или заставляют думать о жизни иначе. Думаю, Arduino - лишь первый шаг к тому, чтобы убрать "техно" из технологии и позволить обычным людям не просто использовать технологию, но создавать с ее помощью. Другие платформы, конечно, будут.

Raspberry Pi - это очень недорогой встраиваемый аппаратный эталонный модуль, и я думаю, что другие платформы пойдут по его следам. Он достаточно дешев, чтобы для многих приложений можно было использовать Raspberry Pi как есть и не получить никакого чистого ценового преимущества от проектирования и сборки собственной аппаратуры. Для профессионалов аппаратной разработки приятная сторона этой платформы в том, что вместо покупки эталонного дизайна и последующей разводки собственной платы можно просто купить Raspberry Pi и поставить его в свой продукт. Для людей с относительно малосерийными продуктами это имеет смысл.

Я вижу продолжающуюся тенденцию: проектировать продукты небольшими партиями становится все реалистичнее. Все еще существует рынок для миллионных хитов вроде смартфонов и

кофеварок, но в конечном счете появится также рынок для устройств с тиражом всего 1 000-10 000 единиц и намного более высокой маржой. Эти малотиражные продукты будут разрабатываться и продаваться командами всего из одного-двух человек, так что прибыль все равно будет хорошим заработком для их создателей. Ключ к успеху таких продуктов в том, что они точно подстроены под конкретную задачу и решают задачу узкой группы пользователей, готовых платить больше за подходящее решение.

Когда новые концепции или технологии впервые появляются, они всегда вызывают оптимистичные обсуждения, но большинство из них действительно повлияют на нашу жизнь только после долгого периода развития. Обсуждая движения мейкеров и открытой аппаратуры, не слишком ли мы оптимистичны? Есть ли у среднего человека распространенные заблуждения об этой области?

Да, технологии действительно требуется много времени, чтобы по-настоящему изменить нашу жизнь.

Движение мейкеров, как я думаю, меньше связано с разработкой продуктов и больше - с развитием людей. Оно о том, чтобы помочь людям осознать: поскольку технология создана человеком, каждый человек имеет власть управлять ею при наличии небольшого знания. В технологии нет магии. Можно также сказать, что каждый может стать волшебником после небольшой тренировки.

Открытая аппаратура - это скорее философия. Успех или провал продукта в значительной степени отделен от того, открыта аппаратура или закрыта. Закрытие аппаратуры не мешает людям клонировать или копировать, а открытие аппаратуры не означает, что плохие идеи будут скопированы просто потому, что они открыты. В отличие от программного обеспечения, аппаратура требует цепочки поставок, распространения и сети отношений, чтобы строить ее с низкой стоимостью. Эти накладные расходы означают, что открытость или закрытость - лишь малая часть уравнения, а вопрос о том, открывать или закрывать проект, вращается вокруг того, насколько сильно вы хотите вовлечь конечных пользователей или третьи стороны в модификацию или взаимодействие с вашим продуктом.

Если смотреть на будущее открытой аппаратуры, думаете ли вы, что оно будет аналогично отрасли открытого ПО, где многие коммерческие компании также поддерживают открытое ПО? В чем различия между ними?

Я не думаю, что они вполне аналогичны. В программном обеспечении стоимость копирования, модификации и распространения по сути равна нулю. Я могу клонировать копию исходного репозитория Linux, выполнить команду make и получить то же высококачественное ядро на моем настольном компьютере, которое работает на серверах высшего класса и суперкомпьютерах.

Но копирование аппаратуры имеет реальную стоимость: детали, фабрики и квалифицированные рабочие, используемые для их сборки; процедуры контроля качества; и производственный процесс - все это важные факторы итоговой стоимости, внешнего вида, ощущения и производительности продукта. Просто дать кому-то копию моих схем и чертежей не означает, что он сможет сделать мой точный продукт. Даже в литье под давлением есть искусство. Если я дам один и тот же CAD-чертеж двум изготовителям оснастки, результат может быть очень разным в зависимости от того, где изготовитель формы решит разместить литники, выталкиватели, охлаждение для формы, время цикла формы, температуру и так далее.

А затем нужно подумать о канале распространения, обратной логистике, финансировании и прочем. Даже когда мир становится эффективнее в логистике, вы никогда не сможете купить телевизор так же легко, как скачать фильмы, которые вы смотрели бы на этом телевизоре.

Какая бизнес-модель, по вашему мнению, идеальна для компании открытой аппаратуры? Можете привести пример?

Одна из моих ключевых теорий об открытой аппаратуре состоит в том, что независимо от лицензии аппаратура по своей природе открыта, по крайней мере на уровне схем и разводки PCB. За относительно небольшие деньги можно заплатить сервису, чтобы извлечь детали, необходимые для копирования PCB-дизайна. Поэтому можно считать, что как только аппаратное изделие попадает в мир, его можно скопировать. Если принять это предположение, то из него следует, что отказ публиковать схемы и разводки PCB не остановит людей от копирования ваших товаров. Если кто-то хочет скопировать аппаратное изделие, он это сделает, делитесь вы проектными файлами или нет.

Но публикация проектных файлов действительно имеет значение для отдельной и важной группы людей. Есть другие бизнесы и индивидуальные инноваторы, которые могли бы использовать ваши проектные файлы, чтобы проектировать аксессуары, обновления или сторонние улучшения, полагающиеся на ваш продукт. В таком случае публикация ваших проектных файлов улучшает ваши возможности для новых деловых отношений, что делает это (с лицензией открытой аппаратуры для сохранения нескольких базовых прав и защит) практичным предложением.

Очевидно, некоторые аппаратные стратегии несовместимы с открытым исходным кодом. Если ваша единственная ценность для потребителя - ваша способность делать автономную аппаратуру, и у вас нет стратегического преимущества по стоимости, тогда вы захотите держать свои планы в секрете, чтобы задержать дешевые копии как можно дольше.

Но самые инновационные продукты сегодня - не просто аппаратные устройства. Они также включают программное обеспечение и сервисы. Бизнес-модели открытой аппаратуры лучше работают в таких гибридных продуктах. Во многих случаях потребители готовы платить за продукт ежегодно - через подписки, рекламу, допродажи, аксессуары, роялти или обновления. Фактически наиболее прибыльно просто собирать эти платежи и не заниматься частью, связанной с производством аппаратуры. Контролировать доступ к действующему сервису гораздо проще, чем контролировать чертежи аппаратного изделия.

Таким образом, если вы связываете прибыльный онлайн-сервис со своей аппаратурой, открытая аппаратура вполне оправдана. Позволить другим людям копировать аппаратуру, продавать ее и приводить больше пользователей в ваш онлайн-сервис просто означает, что вы увеличиваете выручку без дополнительного риска.

Вы часто приезжаете в Китай и много знаете об этой стране. Китайские программные технологии развиты слабо. Как вы думаете, поможет ли роль мирового фабричного центра Китая повысить общий уровень технологий? Как эта страна может измениться из просто производственного центра в место, сосредоточенное на дизайне, исследованиях и разработке? Чего не хватает Китаю?

Я бы не сказал, что много знаю о Китае. Я знаю немного об одном маленьком уголке Китая в одной конкретной области - производстве аппаратуры. Однако если есть одна вещь, которую я действительно знаю, так это то, что Китай - очень большая страна с множеством разных людей и длинной историей, которую я только начинаю понимать. Тем не менее я прожил почти всю историю высоких технологий, поэтому могу прокомментировать связь между высокими технологиями и людьми, из чего могу вывести некоторую перспективу о Китае.

Во-первых, каждая страна, которая сегодня является технологической державой, начинала с производства. Соединенные Штаты начинали как колонии Британии, добывая руды и меха, выращивая хлопок и табак. Со временем в США появились сталелитейные заводы и производство льна. США на самом деле не начали развивать оригинальную технологию до начала 1900-х, и этот процесс набрал силу только к середине 1900-х.

Япония развивалась похожим образом. Она начинала с производства, копируя многие товары, сделанные в США. Фактически, если верить историческим свидетельствам, первые автомобили и радиоприемники, сделанные в Японии, были не очень хороши. США и Японии потребовались десятилетия, чтобы перейти от производственных экономик к экономикам, основанным на услугах.

Сравните это с Китаем, где индустрия производства электроники началась, пожалуй, всего 20 лет назад, максимум, и Китай только сейчас переходит от производственно ориентированной экономики к экономике, способной делать больше дизайна и программных технологий. Я считаю, что это естественная последовательность событий. Некоторая часть работников начального уровня в конечном счете станет техниками, затем некоторые техники станут дизайнерами, и наконец некоторые дизайнеры станут успешными предпринимателями.

В конкретных числах: если у вас есть 10 миллионов фабричных рабочих, возможно, 1 процент, или 100 000 рабочих, через несколько лет узнают достаточно, чтобы стать техниками. После нескольких лет работы техником, возможно, 1 процент приобретет достаточно навыков, чтобы стать оригинальными дизайнерами, что даст 1 000 дизайнеров. Эти опытные дизайнеры низового уровня станут ядром предпринимательской экономики, и оттуда экономика сможет начать трансформироваться.

В течение десятилетия или двух из тысячи компаний в конечном счете останется всего несколько глобальных брендов. Я считаю, что Китай сейчас проходит эту финальную фазу. У многих людей в Шэньчжэне есть опыт производства, дизайнерское чутье и способность применить свой талант к инновациям и оригинальному проектированию продуктов. Следующее десятилетие будет захватывающим для китайской технологической индустрии, если текущие политики экономического и интеллектуального развития останутся примерно на нынешнем курсе.

Этот шаблон применим прежде всего к аппаратуре или продуктам, где доминирует аппаратура. Программные продукты имеют похожий шаблон, но я считаю, что есть уникальные культурные аспекты, дающие Западу преимущество в проектировании программного обеспечения. В аппаратуре, если процесс неэффективен или дает низкий выход годных изделий, можно легко выявить первопричину и представить прямое физическое доказательство проблемы. Аппаратные проблемы по сути неоспоримы.

В программном обеспечении, если код неэффективен или плохо написан, очень трудно определить точную причину проблемы. Можно видеть доказательства того, что программы падают или работают медленно, но нет сломанного провода или отсутствующего винта, который можно поднять и показать всем, почему программа сломана. Вместо этого разработчики должны пересматривать сложные проекты, учитывать множество мнений и в конечном счете выявлять проблему, которая сводится всего лишь к плохому решению одного человека. Все программные API - просто конструкции человеческих мнений.

В азиатских культурах сильно внимание к guanxi, репутации и уважению к старшим. Запад склонен быть более бунтарским и готовым принимать чужаков как героев, и там меньше уважения к советам старших. В результате, думаю, в азиатском контексте культурно очень трудно обсуждать качество кода и архитектурные решения. Сама область программного обеспечения существует всего 30 лет, и более старшие, более опытные инженеры также наиболее устарели с точки зрения методологии и знаний. Фактически именно молодые инженеры часто имеют лучшие идеи. Но если молодым инженерам культурно трудно оспаривать решения старших инженеров, вы получаете плохо спроектированный код и никакой надежды быть конкурентоспособными.

Преодолеть эти препятствия возможно, но навязывание правильных стимулов и культуры потребовало бы очень сильной управленческой философии. Работники должны получать справедливое вознаграждение за правильные решения, и не должно быть фаворитов на основе дружбы, отношений или старшинства. Старшие инженеры и менеджеры должны видеть реальное финансовое вознаграждение за принятие своих ошибок вместо сохранения лица путем

принуждения младших инженеров писать патчи вокруг плохих решений высокого уровня. Американские компании обычно достигают такого выравнивания, распределяя долю в компании между инженерами, так что большая выплата приходит только если компания в целом выживает, независимо от эго отдельного человека.

Как, по вашему мнению, в будущем будут соотноситься отдельные мейкеры и коммерческие компании? И поскольку отдельные мейкеры в будущем могут конкурировать не только с коммерческими компаниями, но и с другими мейкерами, какие факторы критичны для успеха продукта?

По мере того как минимальные объемы заказа уменьшаются, а инновации приближаются к краю, я думаю, коммерческие компании увидят больше конкуренции со стороны мейкеров, особенно когда логистическая отрасль превратится в API, который можно подключать напрямую к веб-сайтам. В конце концов самым критическим фактором успеха все равно будет то, сколько ценности потребители воспринимают в продукте. Это связано с превосходными функциями и хорошим качеством продукта, но презентация для потребителя и то, насколько ясно объяснены преимущества, тоже важны.

В результате любой продукт должен быть визуально привлекательным, простым в использовании и иметь маркетинговый материал, который ясно объясняет преимущества его использования. Эти элементы часто сложны для индивидуальных мейкеров, которые хорошо делают технически ценные продукты, но обладают меньшим талантом к продажам и маркетингу. Мейкеры, которые смогут освоить обе стороны, получают преимущество.

Об аппаратных хакерах

Вы участвовали в процессе разработки многих продуктов, но какова ваша личная цель?

Я хотел бы делать людей счастливыми, создавая вещи, которые так или иначе улучшают их жизнь. Самое большое удовольствие - видеть, как кто-то наслаждается тем, что я сделал, и знать, что я в какой-то небольшой мере улучшил жизнь этого человека. Иногда продукт решает большую проблему для своих пользователей; в других случаях продукт более причудливый, и счастье пользователя приходит от веселья или красоты. Но в любом случае мне важно знать, что, создавая что-то, я помогаю другому человеку. Я понял, что деньги сверх определенного уровня не делают меня счастливее. Из-за этого со мной трудно работать: людям сложно просто нанять меня, предложив много денег. Вместо этого им нужно убедить меня, что эта работа тоже сделает людей счастливее.

Другая важная цель для меня - просто понять, как работает мир. У меня естественное любопытство, и я хочу учиться и понимать всевозможные вещи. Во вселенной много закономерностей, и иногда вы обнаруживаете, что кажущиеся несвязанными части складываются вместе как магия. Обнаружение этих связей и видение мира как большого пазла глубоко и приятно.

Неудача обычно дает людям больше опыта. Можете рассказать о не очень успешных проектах, в которых вы участвовали, или о том, видели ли вы другие провалившиеся проекты, которые вас вдохновили?

Моя жизнь - это история неудач. Единственное, что я делал неоднократно и надежно, - терпел неудачи. Но у меня есть два правила обращения с неудачами:

1. Не сдавайся.
2. Не совершай одну и ту же ошибку дважды.

Если следовать этим правилам, в конце концов вы найдете успех после многих неудач. Тем не менее у меня есть интервью, сосредоточенное на одной из моих недавних неудач. Его можно прочитать по адресу <http://makezine.com/2012/04/30/makes-exclusive-interview-with-andrew-bunnie-huang-the-end-of-chumby-new-adventures/>.^[1]

Ваша книга Hacking the Xbox была опубликована 10 лет назад. Для людей, которые сегодня хотят изучить реверс-инжиниринг или стать аппаратным хакером, как этот опыт и навыки все еще применимы?

Мне хотелось бы думать, что основные принципы, рассмотренные в книге, все еще актуальны сегодня. Xbox был просто примером, который я использовал, чтобы показать, как подходить к таким задачам. Подход и техники применимы к широкому спектру проблем.

Для китайской аудитории я обнаружил, что руководства по ремонту мобильных телефонов довольно интересно читать, хотя я плохо читаю по-китайски. Их описания теории электроники не всегда полностью точны, но с практической точки зрения они достаточно хороши и дают быстрый способ начать, одновременно изучая сразу полезные навыки ремонта телефонов.

Есть также китайский журнал под названием 无线电 (что-то вроде Radio Electronics по-английски), который, как я обнаружил, довольно хорош. Если вы начнете собирать проекты оттуда, думаю, вы будете учиться очень быстро.

Xbox One имеет более строгие ограничения для пользователей. Что вы думаете об этом? Интересно ли вам исследовать этот черный ящик и обновить свою книгу?

Я давно не работал много с игровыми консолями; есть целое новое поколение консольных хакеров, которым интересно их исследовать, и я рад этому. Что касается безопасности Xbox One, я уверен, что это одна из самых защищенных систем из когда-либо созданных. Microsoft очень хорошо поработала над Xbox 360, и участники команды безопасности Xbox One, которых я лично знаю, очень хорошо понимают принципы, необходимые для создания защищенной аппаратуры. Ее должно быть очень трудно взломать.

С учетом сказанного, я рад, что у меня нет желания покупать или использовать его. Думаю, я очень быстро начал бы раздражаться из-за их правил использования и ограничений.

Есть много споров о том, должны ли электронные устройства иметь блокировку, чтобы предотвратить получение пользователем root-доступа. Что вы думаете об этом? Есть ли противоречие между обеспечением безопасности пользователя и предоставлением пользователям полного контроля над своими устройствами?

Я считаю, что пользователи должны владеть своей аппаратурой, а владение чем-то означает право модифицировать это и иметь права root-доступа. Если компания беспокоится о том, что пользователи будут небезопасны, тогда достаточно легко позволить пользователям отказаться от защиты, подписав электронный отказ от поддержки и гарантийных прав в обмен на полный доступ к собственным машинам. Большинство людей, способных получить root-доступ на своих машинах, уже умнее сотрудников телефонной поддержки, которым они звонили бы внутри компании, так что у них не должно быть проблем.

Законы изменились так, что некоторые действия по получению root-доступа стали незаконными даже на аппаратуре, которую вы купили и которой владеете. Я думаю, это сокращение наших естественных прав собственности опасно и может ставить потребителей в несправедливые ситуации. Это также отбивает у потребителей желание исследовать и узнавать больше о технологиях, от которых они стали настолько зависимы.

По мере того как аппаратные системы становятся более интегрированными, думаете ли вы, что аппаратный хакинг становится все труднее, или беспокоитесь ли вы о вымирании аппаратных хакеров? Если да, как можно изменить эту ситуацию?

Интеграция аппаратных систем давно растет. TX-0 использовал просто транзисторы, Apple II использовал TTL-микросхемы, PC используют контроллерные чипсеты, а мобильные телефоны имеют всего один System-on-Chip. Рост интеграции действительно делает некоторые части труднее для хакинга, но всегда есть возможности на уровне системной интеграции.

Иными словами, я все еще думаю, что в аппаратуре есть искусство, но уровень, на котором аппаратные хакеры должны работать, становится выше каждый день, и это хорошо. Это означает, что хаки со временем также становятся мощнее.

Hacking the Xbox посвящена Аарону Шварцу. Можете рассказать, почему, по вашему мнению, дух хакера важен сегодня?

Хакерский дух - это предельное выражение человеческой способности решать проблемы. Это способность видеть мир таким, какой он есть, а не как конструкции и условности, которые общество устанавливает. Например, кирпич используется не только для строительства зданий; он может быть дверным упором, оружием, пресс-папье, тепловым балластом, или его можно измельчить и использовать как грунт. Хакеры ставят условность под вопрос через призму того, что наиболее практично и правильно для данной ситуации. Иногда их методы не всегда гармоничны, поскольку хакеры часто ставят правильный поступок выше того, чтобы быть милыми или играть по правилам.

Я замечаю, что чем труднее становятся ситуации, тем более распространенным и сильным становится хакерский дух среди обычных людей. Я вижу доказательства этого по всему миру. Этот дух связан с человеческой волей к выживанию и процветанию. Думаю, обществу важно культивировать и терпеть хакерский дух. Он есть не у всех, но немногие, у кого он есть, помогают сделать общество более стойким и живучим в трудные времена.

Есть ли еще слова, которыми вы хотели бы поделиться с китайскими читателями?

Я читал некоторые комментарии на китайском веб-форуме и был удивлен, что многие китайцы считают термин шаньчжай негативным. Как сторонний наблюдатель, я чувствую, что шаньчжай сделали много очень интересных и полезных инноваций.

В английском у нас похожая проблема. Термин hacker в английском начинался как хороший термин, но со временем стал ассоциироваться со множеством видов негативных действий. Термин maker был придуман, чтобы различать положительные и отрицательные аспекты хакеров, но я все еще называю себя hacker, потому что все еще придерживаюсь традиционного определения слова.

Может быть легче объяснять инновации, происходящие в Китае, если похожее лингвистическое раздвоение могло бы произойти в китайском. Недавно я предложил называть инновационные, открытые аспекты того, что делают шаньчжай, например их метод обмена проектными файлами, словом gongkai (公开). Существенно, что, как я чувствую, термин 公开 (kāi fāng, что означает раскрывать или открывать публично), используемый в 公开源代码 (kāi fāng yuán dài mǎ, что означает программное обеспечение с открытым исходным кодом), не вполне применим. Он относится к конкретному, западноцентричному юридическому аспекту открытости, который неприменим к методам, задействованным в китайской экосистеме.

Примечание. Между прочим, kāi fāng также означает "цвести", так что по-китайски это звучит поэтично. Gongkai, с другой стороны, означает просто "публичный" или "открытый" - нравится вам это или нет. Его значение не так поэтично или оптимистично, как kāi fāng.

Тот факт, что Китай нашел собственный способ обмена IP, отличный от западной системы, не означает, что китайская система плоха. На самом деле она довольно интересна, и мне очень любопытно посмотреть, куда она пойдет. Поскольку я вижу положительную ценность в некоторых методах, которые используют шаньчжай, я бы предложил использовать более позитивный, общий термин gongkai для описания стиля обмена IP, широко используемого в Китае, но я бы не стал формально связывать его со строгим определением открытого исходного кода.

Но опять же, кто я такой, чтобы говорить? Я не носитель китайского языка, и, может быть, есть гораздо лучший способ подойти к этой ситуации.

The Blueprint беседует с Эндрю Хуаном

The Blueprint публикует истории об основателях в аппаратной сфере, и это интервью сосредоточено на том, что автор назвал моим "личным путешествием". Я обсуждаю, что привело меня в аппаратуру в детстве, над какими проектами я работал, когда давал интервью, и ловушки, за которыми стоит следить аппаратным стартапам. Оригинальное интервью, включающее несколько фотографий моих проектов и ответы на несколько других интересных вопросов, не вошедших в собственно интервью, находится по адресу <https://theblueprint.com/stories/andrew-huang/>.

Как бы вы описали свои первые встречи с аппаратурой?

Мой отец купил клон Apple II, когда мне было восемь лет, и это пробудило мой интерес к аппаратуре. Клон шел без корпуса, так что вся электроника была открыта. Я видел электронику и хотел с ней повозиться. Отец не хотел, чтобы я трогал компьютер, потому что я мог его сломать, но когда его не было дома, я все равно возился с электроникой. Я ломал его несколько раз, потому что чипы были в панельках. Хотя отец говорил мне этого не делать, я просто хотел посмотреть, что произойдет, если вставить чипы задом наперед. Очень рано я усвоил, что вставлять чипы задом наперед - плохая идея!

Прекрасно было то, что Apple II поставлялся с отличным набором схем и исходного кода. Я был тем странным ребенком в начальной школе, который таскал с собой справочное руководство к Apple II. На игровой площадке я просто доставал схему и смотрел на нее, потому что это было так увлекательно. Я не понимал, на что смотрю, но у меня было некоторое представление о связи между линиями на схеме и проводниками на плате. Со временем я научился постепенно сопоставлять символы на схеме с функциями компьютера, и все начало складываться.

К средней или старшей школе я уже мог строить собственные подключаемые карты для компьютера, и я сделал маленький синтезатор речи. Именно этим занимаешься, когда растешь среди кукурузных полей в Michigan, а дети не хотят играть с тобой, потому что ты странно выглядишь и ты единственный китайский ребенок.

Как ваш ранний опыт повлиял на решение пойти в аппаратную индустрию?

После этого я просто продолжал учиться дальше. Когда я поступил в MIT, я подбросил монетку и вместо биологии пошел в электронику. Я получил диплом, в конце концов ушел в индустрию, возненавидел это, а затем вернулся за PhD, потому что хотел еще немного спрятаться в свою скорлупу. После получения PhD я участвовал в куче стартапов, которые все провалились. У меня никогда не было успешного стартапа, но я многому научился на неудачах.

До производства я занимался проектированием кремниевых чипов и реверс-инжинирингом. Многие годы я хотел делать самый большой, самый крутой и самый трудный проект, какой только мог, а это означало работу в чисто технологическом стартапе. В таком проекте вы работаете на далекое будущее, и фактически к тому времени, когда технология начинает работать и выходит на рынок, патенты уже истекли. Капитал не удастся превратить в отдачу, вы очень тяжело работаете, а продукт остается крайне нишевым. В результате у меня никогда не было ничего, что выпускалось бы большими объемами. Это было самым разочаровывающим: вложить свою жизнь во что-то и так и не увидеть, как оно выходит на рынок.

Какие уроки вы извлекли, работая над chumby?

Я устал работать в чисто технологической компании и решил, что пора присоединиться к компании, которая может быстро монетизировать бизнес-идею. Когда я пришел в Chumby, я хотел заниматься открытой аппаратурой и производством, и начал накапливать опыт в обеих областях. Я работал над первым chumby, а затем над несколькими поколениями после него с 2005 по 2010 год.

Когда я начинал, я никогда не выпускал продукт массово и не занимался механическим проектированием. Я даже не знал, что такое литье под давлением. Но у меня была привилегия сидеть вместе с другими инженерами в РСН, и я просто выходил в фабричный цех, смотрел, что они делают, и учился этому. К тому времени, когда я закончил с Chumby, я уже мог использовать SolidWorks, чтобы проектировать собственные корпуса и с нуля делать литые под давлением корпуса.

Это был очень образовательный опыт. Я научился делать планы тестирования, заниматься производством, поиском поставщиков и другим навыкам, которые просто приходится подхватывать по пути. Когда Chumby прекратила работу, я жил в Сингапуре, где пытался открыть местный офис. Я остался, чтобы свернуть офис, аккуратно его закрыть и убедиться, что все нашли работу в других местах. После того как все было улажено, я решил один год побыть безработным; первым делом я разработал радиационный датчик для Японии после ужасного землетрясения и цунами 11 марта 2011 года.

Затем я начал думать о том, каким будет мой следующий проект. Я сделал серию проектов вроде реверс-инжиниринга SD-карт и познакомился с Jie Qi, которой помог производить наклейки-схемы под брендом Chibitronics.

Одним из людей, работавших со мной в Сингапуре, был Sean Cross, и мы сидели и спрашивали себя, что нам стоит создать. Мы решили сделать что-то, чем могли бы пользоваться сами, потому что в Chumby я строил вещи для других людей, а не для себя. Я каждый день пользуюсь ноутбуком, и нам нужна была платформа разработки, так что мы создали ноутбук, которым действительно стали бы пользоваться. Сейчас мы проводим краудфандинговую кампанию вокруг этого продукта.

Как бы вы описали свой процесс перехода от прототипа к производству?

В проектировании вещей, которые легко производить, на самом деле много искусства. Один отличный подход к этому - быть полностью ответственным за собственную цепочку поставок. Мне не нравится иметь менеджера цепочки поставок и менеджера производства. Я хочу делать вещь сам. Я настаиваю на том, чтобы делать все тестирование самостоятельно. Я настаиваю на том, чтобы самому разбираться с производственными проблемами, потому что с инженерной точки зрения это заставляет думать: "Смогу ли я это изготовить? Если я сглажу эту деталь, позже я могу дорого за это заплатить".

С самого начала, когда вы начинаете проектирование, я думаю о том, как сделать вещь пригодной к производству. Какой производственный процесс мне использовать? Как убедиться, что я смогу закупить все эти компоненты? Когда фактически приходит время производства, я уже принял все решения, потому что именно мне в конце концов придется платить цену.

Что люди чаще всего упускают из виду при проектировании?

Есть много аспектов, о которых можно забыть. Первые два, которые приходят в голову, - это доступность материалов и выход годных изделий. Например, инструкции для классного проекта в Make: magazine часто предлагают найти малоизвестную или устаревшую вещь, вроде мотора от VHS-плеера 1980-х. В теории это было бы отлично, потому что у многих людей такая дешевая деталь лежит в гараже. Но внезапно все идут на eBay искать одну и ту же деталь, и ее невозможно закупить.

Со стороны выхода годных изделий многие люди не учитывают, что на самом деле означает процент выхода. На каждом шаге производственного процесса есть некоторый отсев. Если каждый шаг дает примерно 99-процентный выход, и у вас 10 таких шагов, ваш выход будет около 90

процентов. Люди, по сути, закладывают перекося прямо в свой проект, и в конце концов проблемы накапливаются, мешая отгрузке. Критически важно создать систему, которая надежна и пригодна к доработке, чтобы каждый шаг можно было сочетать с другим шагом и минимизировать потери выхода. Иначе вы потеряете много денег.

Как бы вы описали, как изменилось восприятие аппаратуры с тех пор, как вы занялись производством?

Это странно. Примерно в то время, когда я работал над Xbox в 2001 году, аппаратура, вероятно, была на самом дне. Во время бума доткомов работа над Web 2.0 была невероятно модной, и если вы делали что-то с Amazon или XML, это было круто. Пайка считалась занятием низкого статуса, которое выполнялось где-то в другом месте.

Но я был тем странным парнем, который умел паять в лаборатории, так что люди приходили ко мне со сломанными вещами, и я их чинил. Я просто продолжал этим заниматься, потому что это то, что я делаю, и я люблю это делать. Одна из причин, почему защиту Xbox было относительно легко сломать, состояла в предположении, что аппаратура трудна, а пайка сложна. Но если вы умеете паять, сломать защиту очень легко. Я сделал это на бюджете аспиранта примерно за \$150. После взлома Xbox я выступал на конференциях, по сути говоря людям, что аппаратура не трудна, что за ней нет магии. Я показывал людям, что "магия" на самом деле была довольно простыми производственными техниками.

Потом появился Kickstarter. Деньги начали идти в систему, куда раньше не шли, потому что венчурные инвесторы не трогали аппаратуру. Они думали, что аппаратура - это розничная пропасть, где все эти деньги нужно заплатить заранее, затем стартапы по сути все умирают, а инвесторы не получают доходности.

Внезапно эти крутые компании начали собирать миллионы долларов на Kickstarter как свой посевной раунд и в конце концов достаточно часто выполнять обещания по выпуску продуктов. Ничто так не привлекает интерес ребят из Silicon Valley, как деньги. С тех пор восприятие аппаратуры радикально изменилось. Все больше людей начинают заниматься аппаратурой. Проблема в том, что многие теперь думают, что должны добавлять аппаратуру к продуктам, но не имеют понятия как.

Другая проблема - растущее число мошеннических проектов на Kickstarter, где есть все эти аппаратные кусочки и детали, а спонсоры не могут понять, что реально, а что подделка. Я знаю, что отрасль определенно уже ощущается как пузырь; я чувствую, как пузырь сейчас растет.

Думаю, возможно, мне больше нравилось, когда никто не знал об аппаратуре, потому что тогда мне по крайней мере не приходилось беспокоиться о конкуренции с мошенниками.

Как вы подходили к поиску собственных фабрик?

Если вы стартап и единственная ценность, которую вы можете принести фабрике, - это деньги, тогда вы фактически ничего не стоите. У стартапов нет денег, а если у вас есть деньги, они конечны. Все фабрики это знают.

Многие стартапы хотят пойти куда-то вроде Foxconn, но у Foxconn огромное количество людей и возможностей. Им не нужна ваша помощь. Но им нужны ваши деньги, а их у вас немного. Если вы попытаетесь работать с самыми престижными фабриками, вы очень быстро израсходуете все деньги и не сможете запуститься.

Я ищу фабрики, которым не хватает определенных возможностей, чтобы я мог дать им больше ценности, чем деньги. Когда я прихожу со своим продуктом, я помогаю обучать персонал строить мой продукт. Фабрики видят ценность в этом обучении, и я достигаю той точки, где строю отношения, обмениваясь не только деньгами.

В чем проблема для аппаратных онлайн-стартапов, когда они доходят до розничной фазы?

В мире интернета, где все автоматизировано, кажется, что любую проблему можно решить технологией. Но розница - это продавец, который встречается с покупателями лицом к лицу, проводит демонстрации и идет в штаб-квартиру Walmart или Target, чтобы выстраивать отношения и заключать сделки. Это ощущается как более старая система, и многие люди этого не ожидают, потому что они ведут бизнес через Kickstarter.

Проблема в том, что люди хотят увидеть продукт вживую, потрогать его и понять ощущения, прежде чем потратить на него пару сотен долларов. Best Buy становится витриной для Amazon, но возможность показать продукт в магазине действительно ценна. Вероятно, эту модель можно немного подорвать (например, убедить надежных обозревателей попробовать вашу аппаратуру и описать ее другим людям), но в конце концов розничное присутствие нужно, чтобы эффективно продавать аппаратуру.

Онлайн-маржи гораздо толще, поэтому компании, которые с самого начала начинают бизнес онлайн, склонны занижать цену своих продуктов. Затем, когда они доходят до розницы, они не могут выжить.

Какие самые распространенные вопросы задают вам аппаратные предприниматели?

Вопросы, которые команды обычно задают, как правило, сосредоточены на слабостях в составе их команды. У некоторых команд есть суперзвездные инженеры-электрики, но нет опыта в механическом проектировании. У таких команд куча "механических" вопросов. В некоторых командах нет инженеров-электриков, и тогда большой вопрос - как создать аппаратный стартап, где нет никого, кто умеет проектировать электронику.

Команды аппаратных стартапов в целом обычно технические, поэтому они часто слабы в маркетинге и бизнесе. У некоторых бизнес-люди подключаются рано и могут все разложить по плану и поставить стратегию на место, но у многих команд есть отличные технические идеи без понимания, что в их стратегии отсутствуют критически важные аспекты.

В этот момент я заставляю их рассказать мне, что они делают, и даю обратную связь. Дело почти не в том, что команды спрашивают, а скорее в том, что они забывают спросить; именно там им больше всего нужна помощь.

Чего, по вашему мнению, не хватает стартапам, но что будет необходимо для постоянной поддержки аппаратной экосистемы?

Есть огромный разрыв между тем, как производство делалось раньше, и тем, как его нужно делать, чтобы соответствовать этим более гибким, бережливым и, честно говоря, менее опытным компаниям. Но я не думаю, что это непроходимая пропасть.

Производители оригинальных конструкций (ODM), у которых есть фабрики и ресурсы, должны повысить свой уровень сервиса. Люди ожидают, что ODM смогут ответить на множество вопросов. Между стартапами и ODM есть необоснованные ожидания, потому что ODM не могут дать равным счетом никакого понимания того, как снижать стоимость вашего продукта. Люди расстраиваются, потому что просто не видят этот конфликт интересов.

Многие думают, что создание продукта в Китае означает, что стоимость деталей волшебным образом становится ниже. Они не понимают. Фабрика - это не дизайнер; ее задача - убедиться, что ваш проект работает и изготовлен по спецификации. Если вы указываете дорогую деталь, а фабрика заменяет ее более дешевой версией, кого обвинят, когда продукт не будет работать так же хорошо? Более того, фабрика зарабатывает деньги как процентная маржа поверх ведомости материалов. Поэтому рекомендация использовать более дешевые детали подвергает ее большему риску, одновременно принося ей меньше денег. Многие злятся на фабрики за то, что они не более агрессивны в снижении затрат, но если подумать, вам действительно нужно включаться. Вам нужен инженер, работающий с этими ребятами, чтобы снижать стоимость, потому что в конечном счете

это ваша итоговая прибыль. Это ваша чистая прибыль. Нельзя просто поехать в Китай и ожидать, что они сделают все правильно.

ODM, возможно, может решить эту проблему, наняв персонал, специально занимающийся снижением затрат, но тогда ODM должен будет либо брать с клиента доплату, чтобы сделать услугу устойчивой, либо требовать существенно больший объем заказа, на котором можно амортизировать дополнительные расходы предоставления таких услуг.

Больше совместимости в отрасли тоже было бы хорошо. Один стартап, с которым я работаю, - Spark,[2] который действительно пытается дать людям возможность использовать свою аппаратную платформу через открытость. Я чувствую, что одна недостающая часть для Spark - добиться, чтобы ODM были "Spark certified" и могли делать продукты, которые используют платформу Spark. Часто кто-то хочет встроить один продукт в другой продукт, и советы о том, как это эффективно сделать, разбросаны повсюду. Даже если у вас есть вся необходимая информация, все равно нужно многое расшифровать. Даже более крупные компании боятся этого, потому что у них нет компетенции, чтобы нанять людей, способных разобраться во всех деталях.

На чем сейчас сосредоточено ваше внимание в аппаратной индустрии?

Прямо сейчас я работаю с Jie Qi над наклейками-схемами. Мы доходим до момента отгрузки устройств, и я одержимо настроен убедиться, что уложусь в сроки, которые установил для своей компании. Я действительно хочу отправить все вовремя и доставить вещи людям тогда, когда сказал, потому что в краудфандинговых кампаниях было слишком много опозданий. Это не обязательно так. Нужно просто правильно задать ожидания, собрать все воедино до того, как объявляешь дату, и знать, когда запас уже почти готов к отправке. У нас есть несколько продуктовых линеек, которые продаются; примерно половина уже произведена и просто ждет отгрузки с фабрики. Пара новых линеек отстает, но у нас еще есть время до мая, чтобы решить эти проблемы. Думаю, это не будет проблемой, и я с нетерпением жду, как наши линейки будут расти, развиваться и работать с большим числом людей.

Другая вещь, над которой я работаю, - этот проект ноутбука Novena с Sean Cross, который мы в прошлом году на самом деле не планировали делать. В декабре прошлого года я построил сделанный вручную прототип; это была маленькая, довольно убогая штука из кожи и бумаги. Мы использовали ее, чтобы сделать презентацию на CCC, и отклик был ошеломляющим. Это было здорово, и я переработал дизайн, чтобы сделать его более пригодным к производству и более удобным для закупки компонентов. Кампания пока, кажется, идет хорошо. Думаю, она будет профинансирована, и я с нетерпением жду, когда Novena будет произведена и выйдет в мир.

Чему вы научились из двух своих краудфандинговых кампаний?

Завершение почти двух краудфандинговых кампаний дало мне много понимания. Ранее я упомянул, что люди, продающие онлайн, назначают цену на свой продукт слишком низко, чтобы позже перейти в розницу. Но поддерживать высокую цену, о которой я говорю, что всем остальным следует ее поддерживать, оказалось действительно болезненно. Так соблазнительно опуститься ниже, до неустойчивой точки.

Причина, по которой многие краудфандинговые кампании не выполняют обещания, состоит в том, что они назначают цену слишком низко. Они на самом деле не могут изготовить продукт за цену, которую установили. Даже зная это, мне все равно пришлось стиснуть зубы с ноутбуком, потому что я должен был назначить цену выше, чем мне хотелось бы. Несмотря на высокую цену, если бы мы закрыли кампанию ровно на той сумме, которую я надеялся собрать, я, вероятно, едва не потерял бы деньги на нем, но многие этого не видят. Посмотрите на что-то вроде Ubuntu Edge, который собрал \$12 миллионов, но нуждался в \$25 миллионах для успеха. Это потому, что, чтобы установить цену \$700-800 за телефон, им нужно было изготовить 40 000 телефонов. Так что, хотя

люди считали Ubuntu Edge крутым и он собрал много денег, он не достиг цели финансирования, и это стало печальным итогом для всех.

Я знал, что могу либо назначить цену на свой ноутбук гораздо ниже, и тогда мне понадобятся тысячи покупателей, чтобы достичь цели, либо ориентироваться на узкий рынок из нескольких сотен энтузиастов открытого исходного кода, которые полностью на одной волне со мной. В конце концов, особенно на ранних фазах, вам действительно нужны именно эти энтузиасты. Они будут вашими лучшими пользователями. Вы хотите заботиться о них и дать им лучший возможный сервис. Вы будете брать немного больше, но создадите для них действительно хороший продукт, и они будут довольны. В моем представлении это гораздо более удачный итог, чем гнаться за быстрыми деньгами и провалиться.

Примечания

[1] Отрывок из этого интервью приведен в главе 6.

[2] Позже Spark сменила название на Particle.

Эпилог

Когда я начинаю хакинг или мейкерство, мной движет любопытство. Лишь малая часть моей работы в итоге оказывается релевантной или интересной, но я веду журнал своих успехов и неудач в своем блоге, <http://bunniestudios.com/>, и время от времени публикую наблюдения в Twitter как @bunniestudios. Трудно знать заранее, что станет хитом, а что промахом; но пока я учусь, путь того стоит. И поэтому я продолжу бродить по электронному фронтиру...

Об авторе

Andrew «bunnie» Huang всегда было трудно вставать до полудня. Это, в сочетании с его склонностью ставить авторитет под вопрос, означает, что он никогда не удержится на работе в компании из Fortune 500. Поэтому он благодарен за все кружки пива, которые получил через краудфандинг, потому что это значит, что он может получать немного калорий через гидратацию.