Оглавление

Книга посвящена информационной безопасности встраиваемых систем. Считается, что история этого термина начинается в 1966 году, когда в США был разработан бортовой управляющий компьютер для лунной программы «Аполлон». В этом 16-битном компьютере впервые применялись интегральные схемы вместо отдельных транзисторов и электронных ламп, что снизило массу, уменьшило габариты и позволило назвать его первой встраиваемой системой, или встраиваемым устройством. С тех пор такой подход начал применяться в военной технике, авиации и автомобилях.

Следующий рывок связан с распространением интернета. Наличие высокоскоростного доступа к сети привело к взрывному росту встраиваемых устройств, так как теперь они объединялись в сложные системы на базе «интернета вещей» (Internet of things, IoT). Снижение стоимости производства привело к тому, что уже сложно обозначить границу между тем, что является, а что не является встраиваемыми устройствами – такие системы повсюду. Сразу после появления их начали пытаться нелегально копировать, но это не имело массового характера, и производители особо не беспокоились о защите. Однако доступность интернета изменила все – теперь хакеры могли взламывать их удаленно и получать огромное количество информации, которая хранится или обрабатывается на встраиваемых устройствах. Производители же стали использовать различные способы для усложнения анализа и взлома – зашифровывать прошивки устройств (firmware), внедрять электронные подписи и регулярно обновлять устройства через интернет, устраняя найденные уязвимости. Но количество успешных атак показывает, что достаточный уровень защиты не обеспечивается. Часто производители считают, что безопасность через неясность (Security through obscurity) отпугнет хакеров и никто не начнет исследовать устройство, так как это требует специфических навыков и оборудования. Книга покажет, насколько такое предположение ошибочно.

Литературы, рассказывающей о безопасности встраиваемых систем, не так много. «Реверс-инжиниринг встраиваемых систем» – одна из тех книг, которые позволяют вам понять, как происходят атаки на такие системы, как от них можно защититься, и даже поможет исследовать «подозрительное» устройство у вас дома. Профессионалы найдут в данной книге информацию, которая будут полезной в повседневной работе, – современные методы атак на микросхемы и способы защиты, ссылки на современные исследования. Если же вы только начинаете свой путь в исследовании встраиваемых систем, то сэкономите время и бюджет, применяя рекомендации из этой книги. После прочтения вы будете смотреть на свою систему видеонаблюдения или чайник с Bluetooth совсем по-другому.

Я благодарен автору за то, что он предоставил мне возможность ознакомиться с этой книгой в числе первых.

Максим Горячий (@h0t_max),
Device Security Engineer

Intro (Зачем исследовать встраиваемые системы?)

Что движет людьми, которые занимаются реверс-инжинирингом? На мой взгляд, желание разобраться, как устроен мир, и понять, как мыслят другие люди, создающие что-то. С детства я всегда разбирал все, что попадало под руку. Эта тяга осталась до сих пор, только игрушки стали другими. Мне повезло познакомиться и работать с людьми, которые также имеют нездоровую тягу к исследованию внутренностей электронных устройств. Накопленный опыт хочется систематизировать и изложить в максимально доступном виде.

Сразу сделаю уточнение: я не хотел писать академическую книгу сухим языком. Ведь общаемся мы совсем по-другому. Второе уточнение – это слово «реверс-инжиниринг» в названии книги. Лучше бы подошло слово «исследование», ведь в основном люди занимаются исследованием с помощью реверс-инжиниринга. Однако большинству людей понятнее слово реверс-инжиниринг, поэтому пусть в названии будет оно :)

Под словосочетанием встраиваемые системы (англ. embedded) мы будем понимать именно цифровые устройства. Некоторые из них могут иметь аналоговые блоки, выполняющие определенные функции, но темой данной книги будет исследование именно цифровых устройств. Почему именно цифровые устройства? Наверное, потому что, в отличие от реверс-инжиниринга ПО, есть возможность «пощупать» объект исследований, получить тактильные ощущения, совершить с ним различные манипуляции. Рано или поздно увидеть волшебный дым, на котором работает электроника (**https://en.wikipedia.org/wiki/Magic_smoke**). А еще потому, что в современном мире цифровые устройства стали максимально распространены. Они применяются в транспорте, умных домах, игрушках, IoT-системах, заводах и т. д. Какое электронное устройство в руки не возьми – везде встраиваемые системы. Практически в любой современной информационной системе корнем доверия является уровень «железа», т. е. аппаратного обеспечения, лежащего в основе цифровых устройств. Компрометация этого уровня или самих устройств приведет к компрометации всей информационной системы.

Книга была написана после сильного изменения взаимоотношений в мире в 2022 году. Многие технологии для разработчиков в нашей стране стали недоступны. Поэтому реверс-инжиниринг, в том числе встраиваемых систем, становится еще актуальнее. Ведь какие-то из устройств могут перестать работать и надо найти технические пути для их возвращения «в строй». Ну и конечно же, нельзя забывать про аудит безопасности устройств. Иногда проведение иссле-

дований для оценки защищенности – единственный способ удостовериться, что устройство всегда будет работать так, как заявлено. Не менее важен аудит защищенности российских устройств, особенно в контексте возрастающей угрозы атак на цепочку поставок. Все эти проблемы стали сверхактуальными для российского (да и мирового) рынка в последние годы.

Для кого эта книга? Я позиционирую ее как первичное руководство-справочник по погружению в мир цифровых устройств для исследователей, уже попробовавших реверс-инжиниринг ПО. Имеющих представление, что такое дизассемблер и декомпилятор, зачем нужны регистры процессора, из чего состоит бинарный исполняемый файл и что такое виртуальная память. Хотя бы на начальном уровне. В этой книге не будет глав про процесс реверс-инжиниринга ПО, интерфейсы дизассемблеров и т. д., на эту тему написано большое количество книг и статей. Другая, не менее важная целевая аудитория – это разработчики встраиваемых систем. Ведь мало грамотно спроектировать устройство, надо еще его качественно защитить от копирования или исследования. Кто лучше всех знает, как защищаться? Тот, кто умеет нападать. Разработчик, понимающий логику и подходы исследователя, сможет гораздо лучше защитить свое устройство. Также книга будет полезна студентам, обучающимся на кафедрах по специальностям «Информационная безопасность» и «Проектирование электронных устройств».

Я ставил себе задачу показать людям, никогда не имевшим дела с исследованием «железа», что это не сложно, но нужно получить знания во многих областях, которых практически не касаешься в большинстве случаев исследования или разработки ПО. Если у читателя был опыт программирования каких-то микроконтроллеров (хотя бы проекта Arduino), то многие вещи уже будут знакомы и понятны. Фактически данная книга – это методология, показывающая один из вариантов пути исследования электронных устройств, а также объясняющая, почему путь именно такой, какие на этом пути есть распространенные ошибки и как их избежать. Этот путь наверняка не единственный, но он показал свою эффективность более чем за 10 лет практических исследований устройств, так почему бы не начать свои шаги с него? Когда я начинал заниматься исследованием устройств, вся информация собиралась от старших коллег, разрозненных источников, из множества экспериментов, зачастую уводящих в сторону от результата, из «убитых» устройств и размышлений, можно ли было этого избежать. Все это позволило сформировать подход, описываемый в данной книге. На стажировках я рассказывал, почему именно такой порядок действий будет эффективным, но единого материала, который можно было бы прочитать и понять путь и методы проведения исследования цифровых устройств, не было.

При проведении любых исследований мы можем иметь разный уровень знаний об объекте исследования. В зависимости от количества имеющихся знаний о встраиваемой системе исследования могут проводиться с помощью методов белого, черного и серого ящика. Если у нас есть вся информация о системе, то это исследование методом «белого ящика». Если информации минимум, то исследование проводится методом «черного ящика» – анализируется реакция на воздействия, и постепенно получаем все больше информации об объекте исследования. Метод «серого ящика» является промежуточным, когда мы смог-

ли получить какую-то информацию о системе. Большинство описываемых в книге подходов применимы для любого метода исследований, но часть из них будет весьма полезна для исследований методами черного и серого ящика. Исследование методом «черного ящика» является наиболее сложным, особенно для начинающих исследователей цифровых устройств. Поэтому мы сконцентрируемся на том, чтобы максимально быстро получить прошивку устройства и перейти к исследованию методом «серого ящика», основываясь на ее анализе. На схеме представлен путь, которым я предлагаю пройти читателю при первых попытках исследования устройств.

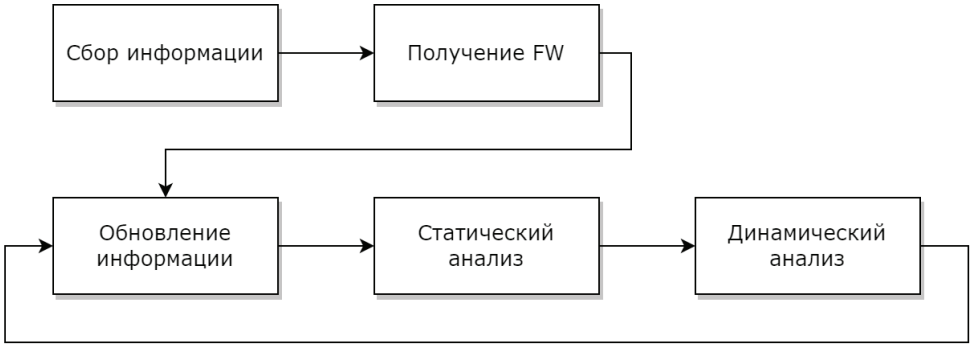

Рис. I.1. Вариант пути проведения исследований встраиваемых систем

Структура книги соответствует предложенной методологии и состоит из пяти основных глав-уровней и нескольких вспомогательных разделов.

В самом начале, в разделе «*Background*», мы договоримся о том, что такое цифровой электрический сигнал и какие существуют особенности его передачи.

Level 0. Первоначальный анализ. С него начинается исследование цифрового устройства. Мы пройдем по пути анализа основных компонентов устройства и восстановим структурную, возможно функциональную, и даже некоторые части принципиальной схемы. (Конечно же не полностью, но на достаточном для понимания базовой структуры устройства уровне. Подробно про разные типы схем можно прочитать в статье по адресу: **https://habr.com/ru/post/451158/**.) В этой главе я расскажу, из чего состоит большинство устройств, как делать этот анализ и почему эти знания помогут в дальнейшем исследовании.

Level 1. Добываем прошивку. Логика работы цифрового устройства заложена в программном обеспечении, оно же прошивка (firmware). Пока у нас ее нет, исследования проводить намного сложнее. В этой главе мы рассмотрим, как прошивку можно получить. От очевидных до весьма нестандартных путей.

Level 2. Начинаем статический анализ. К этой главе мы уже получили образ прошивки, но пока не знаем, что с ней делать. (Ок, «грузить» в дизассем-

блер, но что мы там увидим и что должны увидеть?) Узнаем, какие бывают типы прошивок, как прошивка устроена и как это связано с «железом» устройства.

Level 3. Настраиваем связь с внешним миром (динамический анализ). К этой главе мы уже загрузили прошивку в дизассемблер и даже что-то поискали. Можно и дальше смотреть в дизассемблер, но гораздо эффективнее (и приятнее) заставить устройство что-то нам рассказать о себе в процессе работы, в том числе за счет написания собственных патчей для прошивки.

Level 4. Механизмы защиты встраиваемых систем. Какое хорошее исследование заканчивается без запуска своего кода? Производители устройств очень не хотят, чтобы у нас это получилось. Рассмотрим, какие существуют способы защиты прошивок и устройств от модификации и что с ними можно сделать.

В разделе *«Well Done»* подводим итоги пройденного пути и смотрим, как дальше можно развиваться в исследовании встраиваемых систем.

В *приложении* приведен базовый набор оборудования и ПО, с помощью которого можно проводить исследования большинства встраиваемых систем.

Как читать книгу?

Я старался сделать главы максимально независимыми, чтобы опытный читатель мог использовать книгу как справочник и сам решить, какие блоки информации он знает и может пропустить, а каким стоит уделить внимание. В то же время для начинающих исследователей структура книги позволяет получить знания, объединенные единым подходом-методологией, отвечающей на вопрос «почему последовательность получения информации выстроена в таком порядке?».

Благодарности

Спасибо Юрию Васину, Максиму Горячему, Павлу Иванникову, Алексею Ковриных и Кириллу Малышеву за техническую редактуру книги. Ваши замечания и рекомендации сильно повлияли на итоговое содержание книги и сделали ее лучше.

Почта для связи с автором: **book.re.au@yandex.ru**.

Background: особенности цифровых электрических сигналов

Перед началом исследований цифровых устройств необходимо разобраться, как выглядит цифровой электрический сигнал и какие особенности существуют при его передаче. Я постарался максимально простым языком рассказать самые важные концепции, для более глубокого изучения темы цифровой схемотехники рекомендую начать с книги «Цифровая схемотехника и архитектура компьютера» Дэвида Харриса и Сары Харрис. Ну а мы начнем с определения и различий терминов шина, интерфейс и протокол обмена.

Под понятием «шина» (bus в англоязычной литературе) обычно подразумевают коммуникационную систему для передачи данных между несколькими (>= 2) функциональными блоками. В случае устройства в качестве функциональных блоков могут выступать как компоненты на плате устройства (внутренняя шина), так и само устройство, подключаемое к компьютеру, датчикам или другим устройствам (внешняя шина). В устройстве шины можно различить механический (разъемы и проводники на плате), электрический (физический) и логический (управляющий, или протокольный) уровни. Понятие шина может использоваться как для описания топологии соединения, так и для набора электрических сигналов.

В отличие от прямого соединения точка-точка, позволяющего соединить два функциональных блока, к шине обычно можно подключить несколько устройств по одному набору проводников. Каждая шина предполагает определенные протоколы взаимодействия между подсоединенными к ней функциональными блоками. Протокол – это система правил и соглашений о кодировании, синхронизации и логической организации передаваемой информации.

Интерфейс в общем случае также имеет несколько уровней: механический, электрический, протокольный и управления (программы или функционального блока, непосредственно выполняющего обмен). Несмотря на немного отличающиеся понятия, четко разграничить области применения интерфейса и шины достаточно сложно. Особенно сложно разобраться в переведенной на русский язык иностранной литературе, в которой часто слово interface переводится как шина. Поэтому в реальной жизни между понятиями «шина» и «интерфейс» в большинстве случаев не делается различия и они означают одно и то же: набор проводников, электрических сигналов и протокола, обеспечивающих информационный обмен. При этом также часто шиной называется только набор проводников… В общем, существует большая путаница понятий, мы будем стараться придерживаться следующих общепринятых определений:

- шиной мы будем называть набор электрических проводников и сигналов, передающихся по ним;
- протокол – набор правил, соглашений, сигналов, сообщений и процедур, определяющий взаимодействие между соединяемыми функциональными блоками;
- интерфейс – совокупность программных и аппаратных средств, необходимых для осуществления информационного обмена между функциональными блоками.

Кодирование цифровых сигналов

Для передачи одного бита информации (т. е. 0 и 1) по проводнику необходимо как-то закодировать эти значения. Например, с помощью разных уровней потенциалов относительно «земли», т. е. в общем случае 0 В. Для внешних интерфейсов, применяющихся для связи разных устройств, могут использоваться и отрицательные значения, однако они все равно считаются относительно земли. Для интерфейсов внутри одной платы в основном уровень логического нуля будет от 0 В до некоторого порогового значения $V_{пор}$, а уровень логической единицы – от $V_{пор}$ до значения напряжения питания компонента V_{CC}, как показано на рисунке.

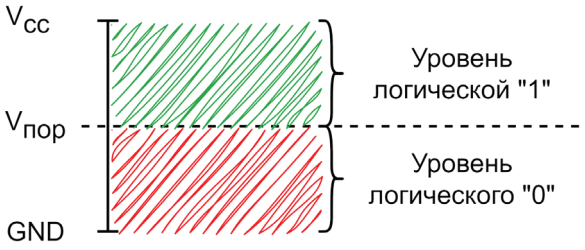

Рис. В.1. Кодирование уровней логического 0 и 1

Чтобы исключить ошибку определения уровня около порогового значения $V_{пор}$, определяют некоторую «буферную зону» напряжений, в которых значение не интерпретируется (зона гистерезиса). Тогда разрешенный диапазон напряжений уровней логического нуля (от GND до V_L) и единицы (от V_H до V_{CC}) будет меньше:

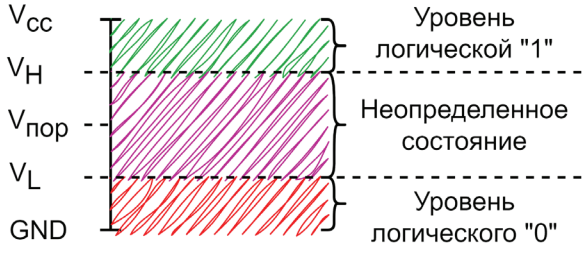

Рис. В.2. Кодирование уровней логического 0 и 1 с зоной неопределенного состояния

Для разных интерфейсов значения уровней логического нуля и единицы могут отличаться. Также уровни напряжений зависят от типов структур транзисторов, из которых состоят микросхемы. Наиболее распространенными на данный момент являются типы КМОП – комплементарная структура металл–оксид–полупроводник (CMOS, complementary metal-oxide-semiconductor) и ТТЛ – транзисторно-транзисторная логика (Transistor-Transistor Logic, TTL). Большинство современных цифровых компонентов построены на базе КМОП-логики. Диапазоны напряжений для ТТЛ- и КМОП-логик представлены на рисунках.

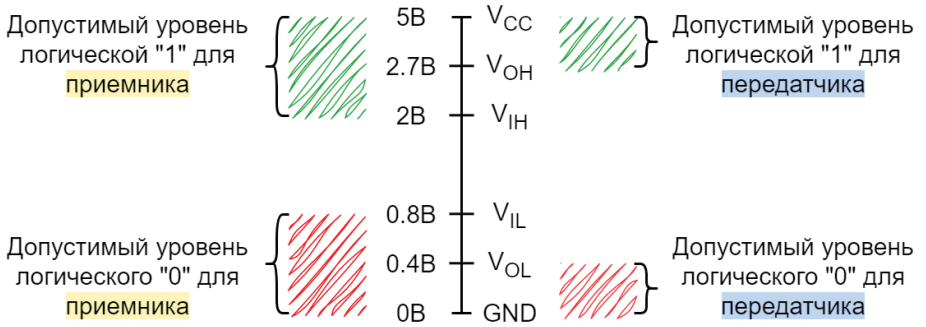

Рис. В.3. Диапазоны уровней напряжений для ТТЛ

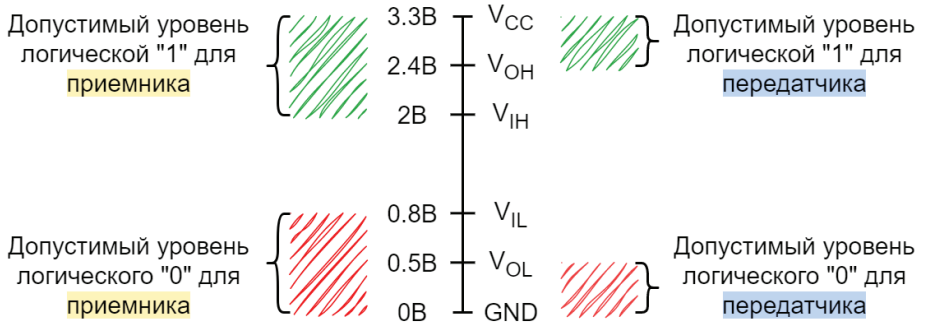

Рис. В.4. Диапазоны уровней напряжений для КМОП

Так как в любых проводниках могут быть потери на сопротивление и рассеивание сигнала, уровни минимального выходного и минимального входного сигналов отличаются. Например, для КПОМ-логики минимальный уровень выходного сигнала логической единицы (V_{OH}) равен 2.4 В, а минимальный уровень входного сигнала логической единицы (V_{IH}) – 2 В. То есть сигнал может «потерять» до 0,4 В при передаче, но будет корректно обрабатываться приемником.

Как отличить электрические сигналы, соответствующие логическому 0 и 1, мы разобрались. Но как отличить один передаваемый бит (со значением 0 или 1) от другого? На выручку приходят временное разделение и синхронизация сигналов.

Синхронизация сигналов

Представим ситуацию, когда нам надо передать один байт (8 бит) информации по одному проводнику. Самый очевидный вариант – использовать для передачи каждого бита информации какой-то временной промежуток, например 1 мс. Получается, что для передачи 8 бит будет достаточно 8 равных временных отрезков по 1 мс, т. е. 8 мс. Принимающая сторона должна проверять уровень напряжения (соответствующий логическому 0 или 1) с аналогичной периодичностью. На рисунке показана временная диаграмма подобной передачи для значения 0x9C (10011100b).

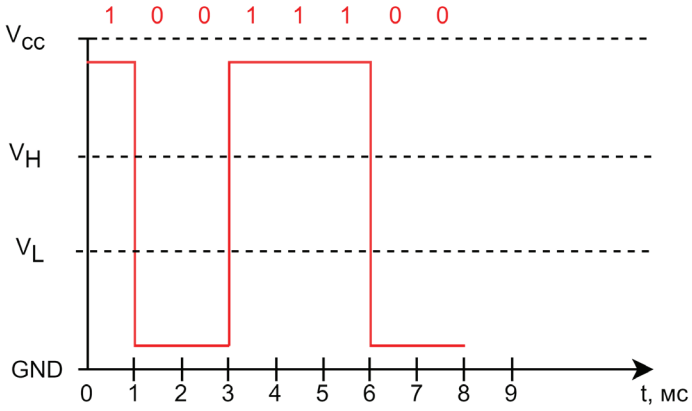

Рис. В.5. Пример временной диаграммы передачи значения 0x9C

Но электрический сигнал не может мгновенно изменить свое состояние с 0 на 1 и наоборот. Всегда существует время, в течение которого происходит переходный процесс, т. е. изменение реального значения напряжения в проводнике.

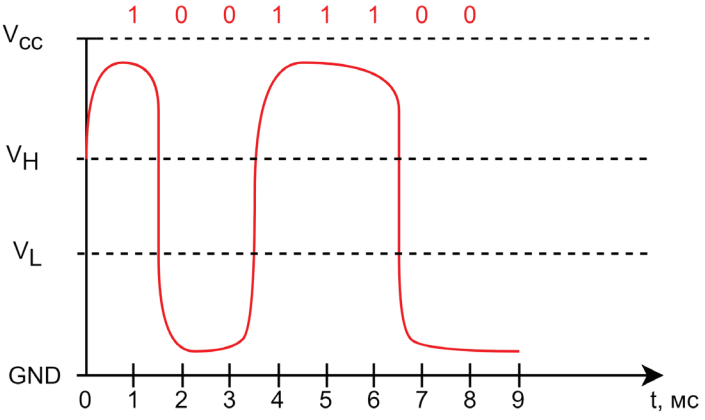

Рис. В.6. Пример переходных процессов изменения сигнала при передаче значения 0x9C

Следовательно, проверять значение уровня напряжения приемнику лучше всего в середине временного отрезка передачи бита. В нашем случае время передачи составляет 1 мс, значит, смотреть значение лучше всего со смещением 0,5 мс от начала передачи бита. Так как любая система не идеальна и всегда могут встречаться какие-то задержки, допустимый временной интервал расширяют до некоторого значения, например ±0,2мс. Получается, что, начиная с 0,3 мс от начала передачи бита и заканчивая 0,7 мс, приемник ожидает корректное значение напряжения, попадающего в границы напряжения для логического 0 и 1 выбранной логики (мы для примера возьмем КМОП). Внешний вид сигнала, передающего первые 4 бита информации (не будем перегружать схему отрисовкой всех 8 бит) байта 0x9C со скоростью 1 Кбит/с (т. е. 1 бит за 1 мс), показан на рис. В.7.

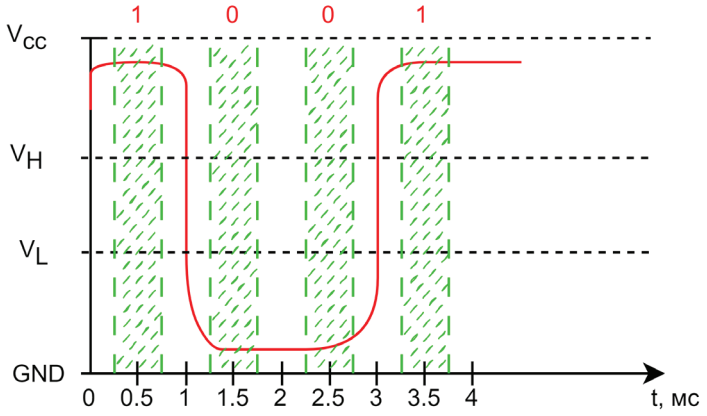

Рис. В.7. Зоны корректных значений сигнала

Этот сигнал соответствует исходной цифровой интерпретации передачи байта 0x9C. Поздравляю, мы с вами придумали простейший интерфейс с временной синхронизацией. Основным минусом подобного типа передачи является то, что приемник и передатчик обязательно должны быть заранее сконфигурированы на одну частоту, которую они должны довольно точно выдерживать. Если приемник или передатчик не могут гарантировать допустимые временные интервалы корректного сигнала, передача будет идти с ошибками. На рис. В.8 показана ситуация, когда время установки сигнала «плавает» и приемник не может корректно интерпретировать подобный сигнал.

Именно из-за отсутствия единой точки отсчета для синхронизации такие интерфейсы не могут похвастаться частотами передачи больше нескольких мегагерц. Для решения этой проблемы существуют различные механизмы, например посылка специальных данных с известным значением при старте передачи и настройка параметров приемника с учетом особенностей сигнала принятых от передатчика данных. Этот подход требует довольно сложной аппаратной поддержки и поэтому используется только в высокоскоростных интерфейсах. Мы рассмотрим другое решение – добавление специального синхронизирующего сигнала отдельным проводником. В таком случае электрический сигнал будет иметь примерный вид, как показано на рис. В.9.

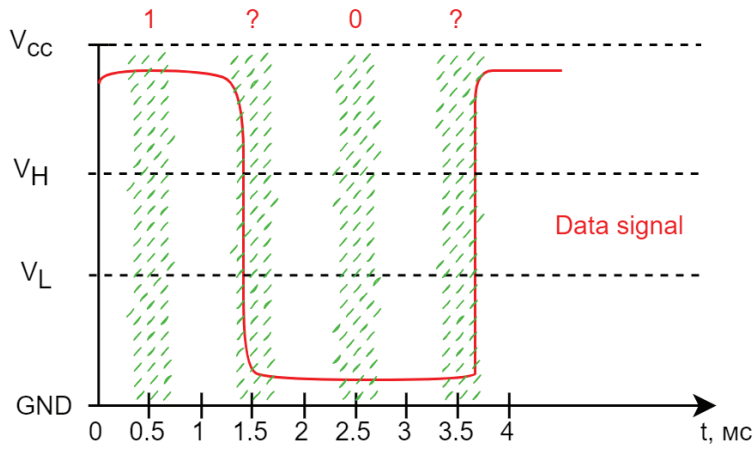

Рис. В.8. Зоны с неопределенными значениями сигнала

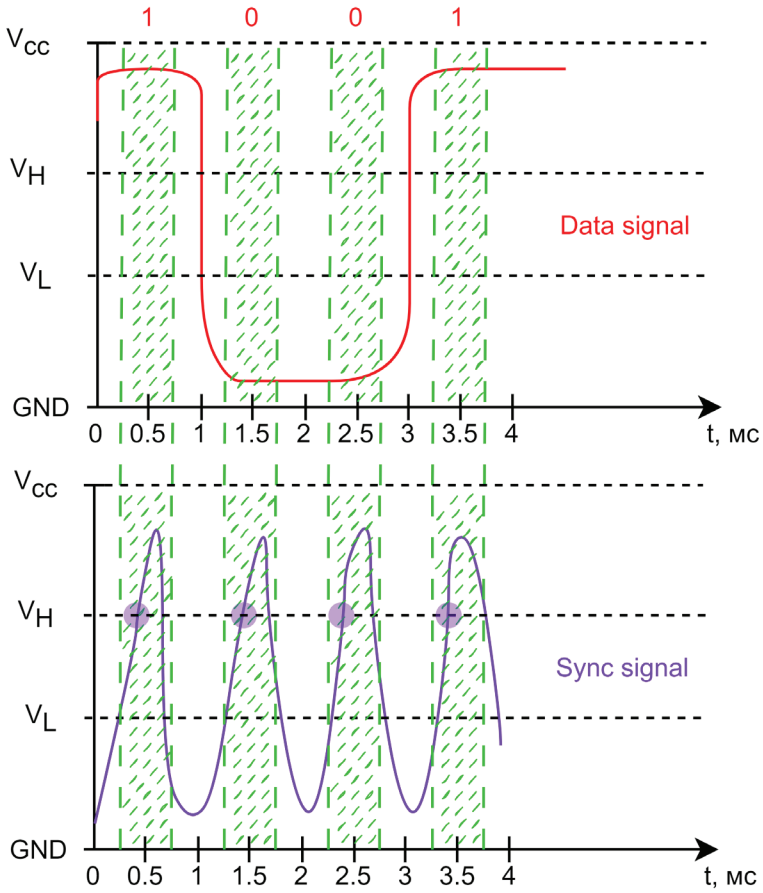

Рис. В.9. Добавление синхронизирующего сигнала

Цифровой сигнал нашего интерфейса примет следующий вид:

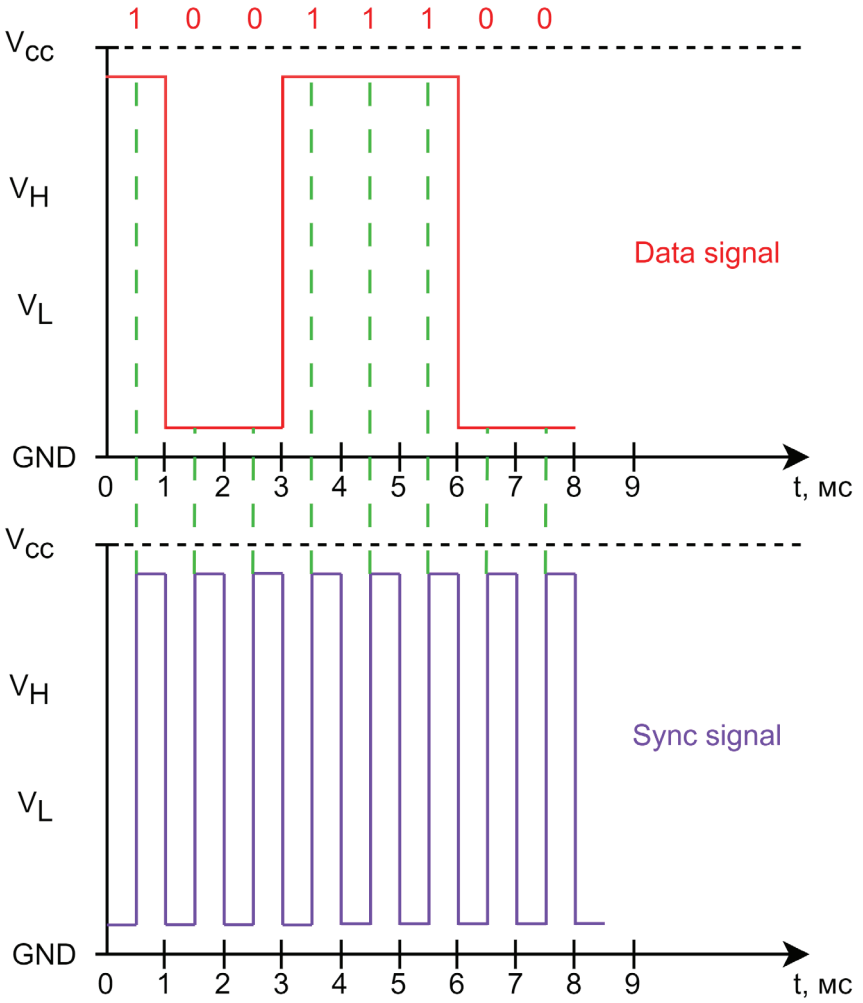

Рис. B.10. Добавление синхронизирующего сигнала

Как видите, теперь для приемника признаком наличия корректного значения на линии данных является изменение сигнала линии синхронизации с уровня логического 0 на 1. Такой сигнал называют синхронизацией по фронту (т. е. по возрастанию сигнала), обратная ситуация носит название по спаду (т. е. по убыванию сигнала с уровня 1 до 0). Существует механизм увеличения пропускной способности передачи, когда сигнал на линии данных меняется и по фронту, и по спаду, т. е. за 1 период изменения сигнала синхронизации передается 2 бита информации.

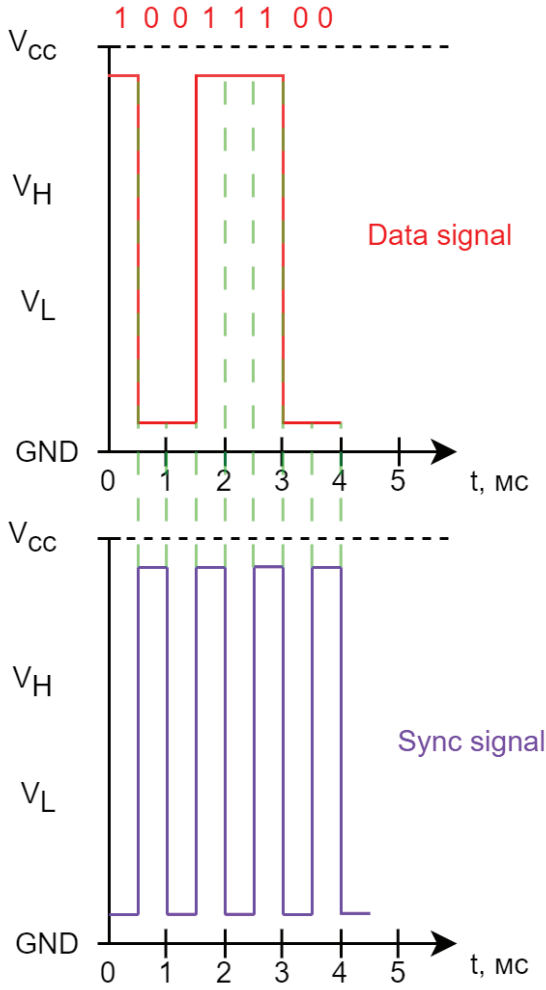

Рис. В.11. Режим передачи сигнала DDR

Фактически, без изменения частоты сигнала синхронизации, происходит удвоение пропускной способности интерфейса, называемое на английском языке хорошо вам известным сокращением DDR (Double Data Rate). Именно этот механизм используется в оперативной памяти современных компьютеров. Обычный вариант передачи данных по фронту или по спаду носит название SDR (Single Data Rate).

Синхронизация с помощью дополнительного проводника широко используется во многих интерфейсах цифровых устройств, их мы подробно рассмотрим чуть позже. А пока давайте разберем, чем последовательные интерфейсы отличаются от параллельных.

Параллельные и последовательные интерфейсы

Снова представим, что нам надо передавать определенный объем информации, чем быстрее, тем лучше. За один период тактового сигнала по двум линиям передачи информации (одна – для передачи данных и одна – для передачи сигнала синхронизации) в режиме SDR передается только один бит информации. Но ведь можно использовать несколько линий для передачи данных? Тогда пропускная способность увеличится, и на той же частоте с двумя линиями данных мы сможем передать уже 2 бита информации, с четырьмя линиями данных (и одним сигналом синхронизации) – 4 бита и т. д.! Вот мы с вами и изобрели параллельный интерфейс передачи данных. На рисунке показана диаграмма – пример передачи значения 0x9C75A16B по интерфейсу с четырьмя линиями данных и одной линией синхронизации:

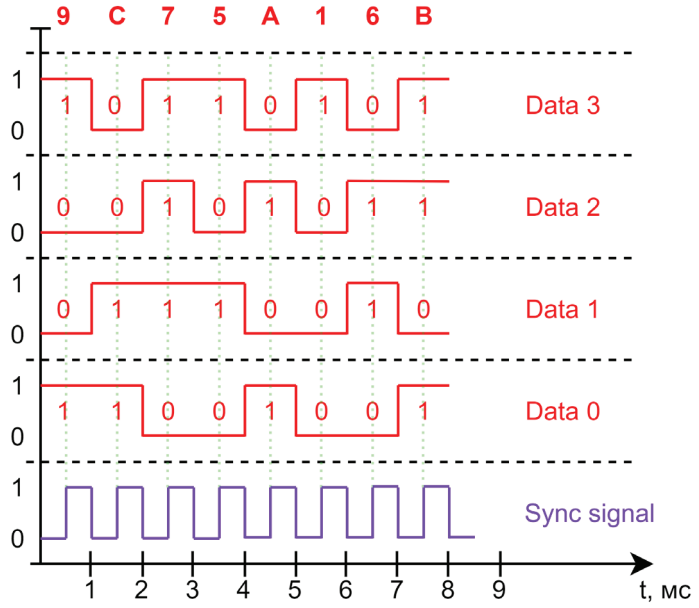

Рис. В.12. Параллельная передача сигнала

Масштабируя количество линий передачи данных, мы легко увеличиваем пропускную способность интерфейса, не меняя частоту передачи. Однако такая логика работает только до определенного порога частот, после которого требования к плате устройства существенно вырастают, т. к. влияние электрических помех все сложнее компенсировать. Это влечет за собой увеличение стоимости разработки устройства. Поэтому параллельные шины на высокоскоростных интерфейсах оправданы только в том случае, если нужна высокая пропускная способность. Например, от оперативной памяти до процессора, где используется несколько десятков параллельных проводников, сигнал по которым идет одновременно.

Поэтому чаще всего используют современные последовательные шины с набором нескольких дифференциальных пар, работающих на гигагерцовых частотах (например, как сделано в знакомом вам USB 3.2). Что такое дифференциальные пары и за счет чего достигаются настолько высокие частоты работы шин на их основе? Одна дифференциальная пара – это набор двух проводников, сигнал в одном из которых инвертирован по отношению к сигналу в другом проводнике. Пример показан на рисунке ниже (верхний график – хорошо знакомый нам обычный сигнал, нижний – дифференциальный сигнал).

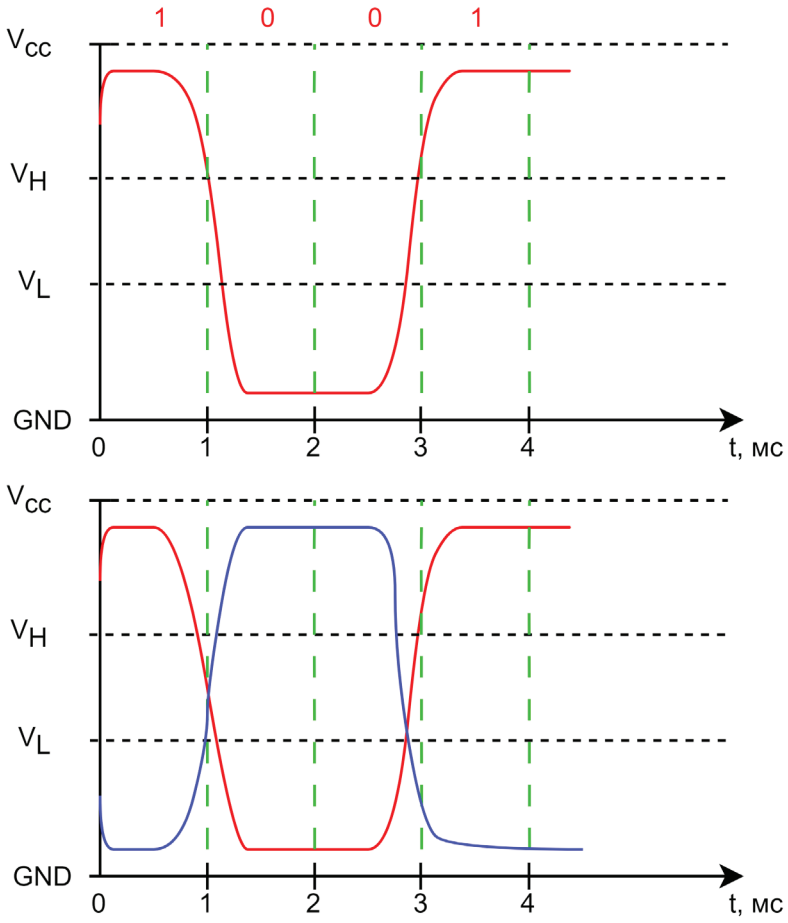

Рис. В.13. Дифференциальный сигнал

При таком способе передачи сигнала существенно снижается вероятность появления ошибок при передаче (т. к. уровень разности потенциалов всегда максимальный и вычитается помеха, влияющая одновременно на оба проводника), следовательно, можно значительно увеличивать частоту передачи. Данные по дифференциальной паре идут последовательно, но в современных высокоскоростных интерфейсах дифференциальных пар может быть несколько и работают они параллельно, например в интерфейсе PCI-Express.

Механизмы детектирования, снижения и исправления ошибок передачи

Фактически многие уже описанные решения были придуманы для минимизации ошибок, неизбежно возникающих при увеличении скорости передачи информации. Использование дифференциальных пар или режима DDR – отличные примеры подобных инженерных решений. Дополнительно во многих интерфейсах используются механизмы контроля целостности и исправления ошибок в передаваемых данных, основанные на помехоустойчивом кодировании:

- коды обнаружения ошибок (контрольные суммы) передаваемых данных. На практике используются циклические коды, известные как CRC8/16/32 и код контроля четности (над всеми битами передаваемых данных проводится операция исключающего ИЛИ (XOR), результат передается вместе с данными и проверяется приемником);
- коды коррекции ошибок (Error correction code, ECC). Позволяют не только обнаружить, но и исправить определенные битовые ошибки при передаче.

Подробнее про помехоустойчивое кодирование можно узнать, прочитав книгу «Коды, исправляющие ошибки» (Питерсон У., Уэлдон Э.). В зависимости от используемого интерфейса в цифровых устройствах могут применять различные механизмы и реализации (аппаратные и программные) обработки ошибок и даже их комбинации. Рассмотрев основы передачи цифровых сигналов, приступаем непосредственно к исследованию цифровых устройств.

Level 0

Первичный анализ

Исследование электронных устройств отличается от исследования высокоуровневого ПО как минимум тем, что нужно принимать во внимание аппаратные особенности устройства, зачастую недокументированные. Именно поэтому нужно получить как можно больше информации обо всем объекте исследований. Устройство далеко не всегда бывает автономно. Оно может подключаться к компьютеру и настраиваться через ПО управления. А может иметь модули связи и подключаться к облаку производителя. Чем больше информации вы соберете вначале, тем проще будет находить закономерности в ходе исследований. Мысль очевидная, но почему-то в пылу «пощупать» что-то руками многие часто забывают об этом шаге. А зря.

Сбор информации

Что мы должны попытаться найти в первую очередь? Правильно, документы. Чем больше – тем лучше. И под документами мы понимаем не всегда самые очевидные вещи, это:

- руководства (пользователя, администратора и т. п.);
- техническая документация (схемы для сервисных центров, информация на форумах ремонтников и т. п.);
- патенты и сертификационные отчеты, например FCC;
- результаты исследований (кто сказал, что мы первые?);
- любая информация о технологиях, применяемых у вендора (используемые ОС, стеки, фреймворки и т. д.). Полезно посмотреть список скиллов у сотрудников вендора на LinkedIn или их персональные блоги.

Вам пригодится умение правильно «гуглить». И пользоваться переводчиком с китайского (китайские форумы – кладезь полезной информации, которой больше нет нигде). Не забывайте про GitHub. Как корпоративный, так и личные репозитории работников вендора. История знает немало случаев, когда там хранились критические данные, например ключи шифрования.

Фактически вы проводите расследование и ищете любую информацию, которая может быть полезна. Полезно или нет, поймете потом, а сейчас просто сохраняйте все, что имеет отношение к исследованию. И еще один тезис: старайтесь мыслить широко. Первичная задача любого производителя устройства – получить прибыль. А как ее получить? Унифицировать все максималь-

но и использовать имеющиеся наработки. Поэтому даже если вы не можете найти информацию по конкретно вашему объекту исследований, посмотрите соседние модели и даже серии устройств. Например, Canon использует DryOS во множестве своих продуктов: от фотоаппаратов до принтеров, а сопроцессор безопасности внутри iPhone (на процессоре Apple A10) и MacBook (в виде чипа Apple T2) используется один и тот же. Ведь если работает, зачем что-то менять?

Ну и конечно же, не забывайте про любой софт. ПО управления, администрирования, *обновления*, *восстановления*. Все это пригодится. Помните, что версии под разные ОС могут значительно отличаться функциональными возможностями, а где-то даже могут быть исходники и отладочные символы.

Инженерный анализ устройства

Наконец-то начинается что-то, имеющее отношение к исследованию встраиваемых систем. Устройство лежит перед нами на столе, и не терпится побыстрее его потрогать. На этом этапе важно помнить о двух вещах.

Статическое электричество. Наш большой враг, ведь «убить» статикой железку очень легко. Поэтому ОБЯЗАТЕЛЬНО соблюдаем технику безопасности:

- «разряжаемся», правильно выбираем одежду и используем антистатические браслеты;
- лишний раз не трогаем выводы антенн и интерфейсные разъемы.

Неизвестное состояние устройства. Даже новое устройство из упаковки производителя может быть бракованным. Поэтому нельзя однозначно сказать, что устройство пришло в негодность после выполнения нашего инженерного анализа или оно уже было неработоспособно. Напрашивается очевидная мысль: перед инженерным анализом взять и проверить работоспособность устройства. Мысль абсолютно правильная, но есть нюанс. Устройство может иметь разные состояния до первого включения и после. И иногда состояние после первого включения содержит в себе меньше информации, чем до. Поясню: устройство может переписывать конфигурацию при каждом включении, перетирать логи (а логи заводских тестов бывают интересными) или иным образом менять свое внутреннее состояние. Возникает дилемма: не включишь – не узнаешь, а включишь – можешь потерять часть данных, полезных для анализа. Выход есть, и он очевидный. Нужно два и более устройств. Одно проверяем, другое препарируем. Но не всегда есть возможность добыть больше одного устройства, или его стоимость превышает стоимость квартиры. Как тут поступить – выбор каждый делает сам. Но из практики лучше все же сначала проверить работоспособность.

Отлично! Про статику помним, железок лежит штуки три, с чего же начинать инженерный анализ? И что мы должны получить на выходе? На выходе мы должны получить перечень основных компонентов устройства, схемы структурную (можно сразу функциональную) и даже принципиальную (в каком-то приближении), понимание организации основных информационных потоков между компонентами устройства. Это программа минимум.

Начнем мы с инвентаризации компонентов на печатной плате. Печатная плата (ПП, по-английски Printed Circuit Board, PCB) в общем случае – это кусок текстолита с проводящими электрический ток дорожками из меди, на котором смонтированы электронные компоненты. Устройства могут быть простые и сложные. В простых плата, скорее всего, одна, компонентов мало, монтаж компонентов неплотный.

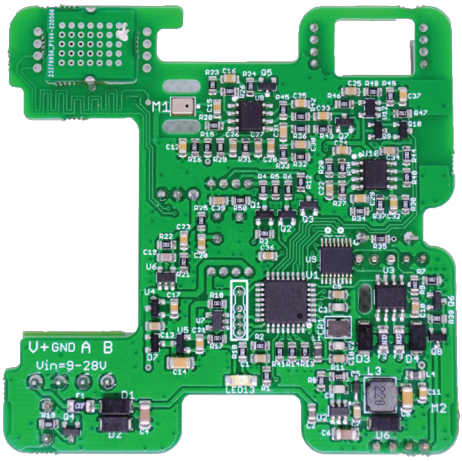

Рис. 0.1. Внешний вид платы простого устройства

В сложных устройствах могут быть различные ухищрения для миниатюризации устройства и эффективной компоновки внутри его корпуса. Размеры компонентов едва различимы на глаз, расстояния между ними практически нет. Например, современные телефоны используют «бутерброд» из двух печатных плат, спаянных между собой. Для примера приведем фото платы iPhone X (видите «точки» по периметру плат? Это линии передачи данных).

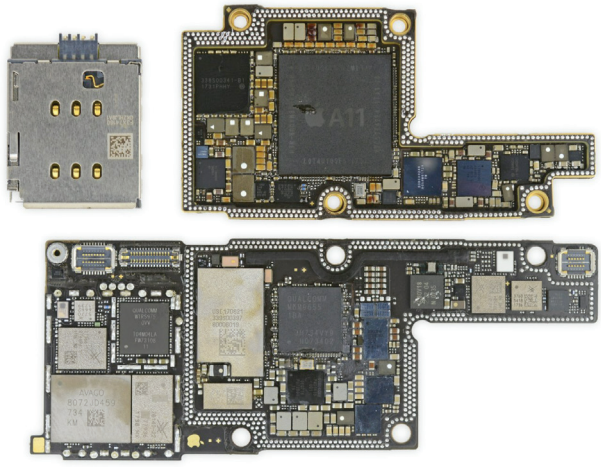

Рис. 0.2. Плата iPhone X как пример сложного устройства[1]

[1] https://ifixit.com/Teardown/iPhone+X+Teardown/98975.

Исследование подобных устройств на порядок сложнее, хотя нет ничего невозможного для человека с опытом и нужным оборудованием. Чтобы добраться до печатной платы, устройство надо сначала разобрать. И тут есть несколько типовых ошибок, которые могут привести к фатальному результату.

Вскрываем корпус и разбираем устройство

Первым делом мы должны обесточить устройство. То есть убрать питающее напряжение («питание»). Если питание внешнее – отключаем и даем пару минут полежать (чтобы конденсаторы разрядились). А если питание автономное (аккумуляторное) и внутри корпуса, то переходим к разбору, но помним, что первым делом после вскрытия надо будет отсоединить аккумулятор. Обратите внимание, что не стоит включать устройство с беспроводными модулями связи без антенн (или стоит использовать аттенюаторы). Это может привести к выходу радиочасти устройства из строя.

В большинстве случаев корпус устройства состоит из нескольких частей, скрепленных между собой как минимум защелками. Существуют специальные инструменты для вскрытия корпусов, продаются в наборах и легко находятся в интернете.

Рис. 0.3. Набор инструментов для вскрытия корпусов электронных устройств[1]

Корпус только на защелках – скорее редкость. Почти всегда к ним добавлены винты, и их надо открутить в первую очередь. Винты бывают разные, но маловероятно, что вы не сможете найти подходящую отвертку в большом наборе. А вот расположение винтов иногда бывает неочевидным. Например, под наклеенными резиновыми ножками на днище устройства. Или под этикеткой. Главное – помнить две вещи: в интернете есть инструкции по разбору многих устройств (гуглим по словам teardown и disassemble), возможно, вы даже нашли что-то на этапе сбора информации. И второе: «сила есть – ума не надо», это не про нас. Сломать можно все, что угодно, мы, собственно, за этим тут и собрались. Но не в смысле «пополам». Поэтому если устройство не хочет разби-

[1] https://www.ifixit.com/Store/Tools/Pro-Tech-Toolkit/IF145-307.

раться, усилие надо прикладывать постепенно. Может, где-то все-таки остался винт, скрепляющий половинки корпуса.

Кстати, еще один подвох: разная длина винтов. При сборке обратно можно легко и непринужденно убить устройство, вкрутив другой винт, имеющий длину даже на 1 мм больше. Оригинальный винт не доходит до печатной платы, а более длинный замыкает на ней что-нибудь. Видел последствия подобных ошибок, максимально обидно. Поэтому фотографируйте процесс разбора, подписывайте винты. Можно раскладывать по коробочкам, можно клеить их на малярный скотч. Каждый винт должен вернуться на свое место.

Случаи склеенных корпусов тоже бывают. Тут уже приходится действовать деструктивно (например, с помощью гравера с микрофрезой), использовать растворители клея или прогревать корпус паяльным феном. Хорошо, что таким приходится заниматься нечасто.

В некоторых встраиваемых системах применяются специальные механизмы защиты от вскрытия (tamper protection), стирающие прошивку, ключи шифрования или просто устанавливающие флаг о вскрытии устройства. Например, в ряде дорогих ноутбуков бизнес-класса есть специальный контакт-тампер, срабатывающий на вскрытие задней крышки ноутбука. Подробнее мы рассмотрим механизмы защиты от вскрытия в главе 4.

На этом этапе считаю, что корпус мы вскрыли без потерь, перед нами открылся вид на печатную плату устройства. На всякий случай напоминаю про статическое электричество и необходимость отсоединения аккумулятора. Держите подальше металлический инструмент: отвертки, пинцеты, скальпели. Замкнуть что-то на печатной плате устройства, уронив на нее металлический инструмент, очень легко. У вас точно получится когда-нибудь, не сомневайтесь.

Если печатных плат несколько, то между собой они могут быть соединены разъемами, например именно таким образом соединяются платы промышленного стандарта PCI/104 (он даже взят за основу форм-фактора для сверхмалых спутников CubeSat размером 10×10×10 см, **https://www.mdpi.com/2076-3417/9/15/3110**). Внешний вид современного компьютера (включающего в себя Intel Core i7 8665UE и 64GB RAM) в форм-факторе PCIe/104 показан на фото:

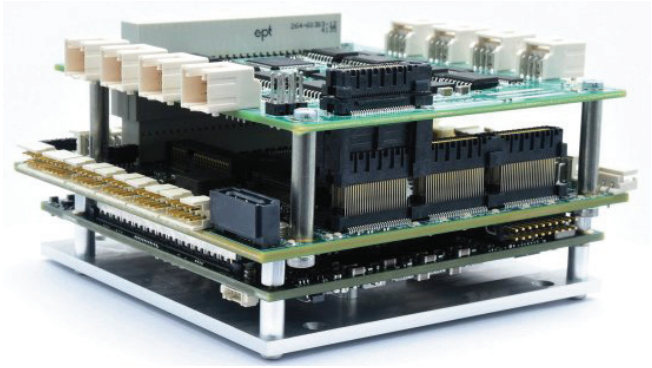

Рис. 0.4. Компьютер в форм-факторе PCIe/104[1]

[1] https://diamondsystems.com/products/gemini.

В большинстве устройств платы соединяются между собой шлейфами и проводами, подключенными к разъемам (коннекторам), пример фотоаппарата Canon – на фото ниже.

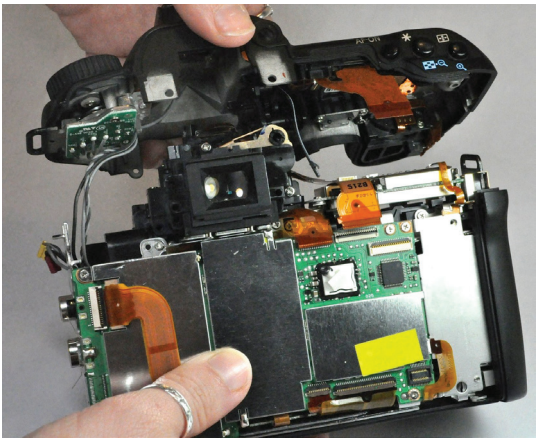

Рис. 0.5. Гибкие шлейфы в устройстве фотоаппарата Cannon[1]

В ряде устройств встречаются гибко-жесткие (rigid-flex) печатные платы. В этом случае шлейф (гибкая часть) является частью печатной платы и не может быть отсоединен. Существуют и чисто гибкие (flex) печатные платы, но в большинстве случаев они используются в качестве шлейфа для соединения жестких плат. Как правило, гибко-жесткие или гибкие платы применяются при плотной компоновке нескольких плат внутри корпуса небольшого устройства, т. к. позволяют располагать платы максимально близко друг к другу. Пример гибко-жесткой печатной платы показан на рисунке.

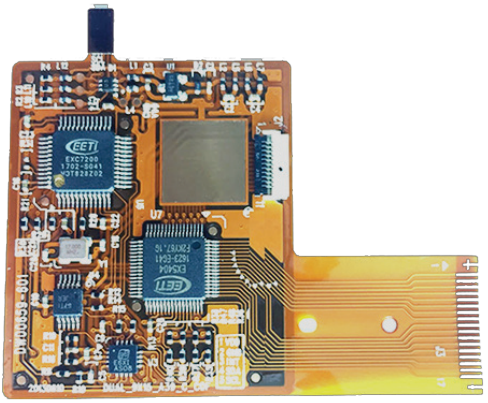

Рис. 0.6. Гибко-жесткая плата[2]

Если разъемы есть, нам нужно их разъединить. Тут все просто, существует всего несколько правил. Первое: найти фиксаторы (например, черная пласти-

[1] https://ifixit.com/Guide/Canon+EOS+40D+Shutter+Button+Replacement/50726.

[2] https://www.fanypcb.com/2-layer-rigid-flex-pcb.html.

ковая деталь на картинке ниже) и разблокировать их. Они могут выглядеть по-разному, но основной смысл – фиксировать соединение. Конструктив бывает разный, информация по разъемам легко находится. Отличным материалом для обучения будут видеоинструкции по разборке устройств на YouTube или обзоры на **https://ifixit.com/**.

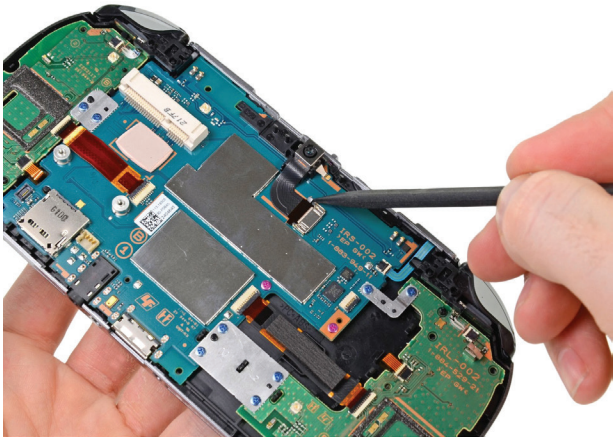

Рис. 0.7. Разъемы на плате устройства[1]

Второе: надо быть аккуратными. Пленочные шлейфы (так называются потому, что проводящие дорожки нанесены на гибкую пленку) легко рвутся или портятся при «переламывании». И если не повезет, можно надломить его и потом долго искать причину, почему устройство не работает, ведь внешне все будет выглядеть целым. Особенно обидно сломать шлейф гибко-жесткой печатной платы, ведь его нельзя заменить.

Не забываем фотографировать все шаги по разборке, времени займет немного, но потом сильно пригодится при обратной сборке устройства.

Из чего состоит плата устройства?

Зачем вообще делать инженерный анализ платы устройства? Часто иначе не достать прошивку, а без прошивки нам дальше делать нечего. Но даже если прошивку уже удалось получить, без информации об аппаратной составляющей устройства исследование может идти медленнее или вообще не получится. Поэтому начинаем анализ платы с изучения общих принципов компоновки.

Плата устройства состоит из непосредственно печатной платы с проводниками и напаянных на нее электронных компонентов. Притом часто говорят «печатная плата», подразумевая плату устройства в сборе с компонентами (еще точнее было бы ввести определения ячейки или электронного модуля, но мы будем оперировать распространенными названиями, пусть и в ущерб точным определениям).

Печатная плата представляет собой «пирог» из проводящих слоев металла (меди) с вытравленными дорожками проводников (4) и слоями диэлектри-

[1] https://ifixit.com/Teardown/PlayStation+Vita+Teardown/7872.

ков (5). Сверху и снизу плату покрывают защитной маской (она еще определяет цвет печатной платы) и наносят надписи с помощью метода шелкографии. На данном этапе нам важно знать следующее: проводящих электричество слоев может быть от одного (тогда это однослойная печатная плата) до нескольких десятков (в очень сложных устройствах, например в материнской плате компьютера). Наиболее распространенным количеством слоев проводников в большинстве печатных плат устройств являются 4, 6 и 8 (так и говорим: «четырёхслойная печатная плата»). Конфигурация «пирога» печатной платы называется стек. Внутренние слои с проводниками позволяют добавить третье измерение в печатную плату и существенно упростить разводку устройства, уменьшить его габариты.

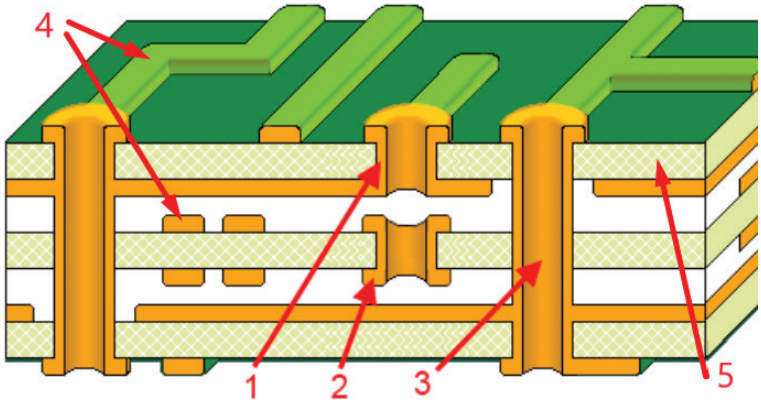

Рис. 0.8. Структура печатной платы[1]

Как правило, часть внутренних слоев используется для передачи питания: слои для «земли» (ground, GND) и для питающих напряжений (V_{CC}). Очевидно, что слои проводников должны как-то соединяться между собой, иначе весь смысл теряется. Делается это с помощью переходных отверстий (VIA) (1, 2, 3). Притом на сложных печатных платах могут быть глухие (1) или скрытые (2) переходные отверстия только между внутренними слоями (но такие ПП намного дороже в производстве). Подобный подход позволяет использовать пространство на внешних слоях печатной платы для размещения компонентов.

Переходные отверстия могут совмещаться вместе с контактной площадкой (VIA in PAD). В таком случае визуально определить наличие переходного отверстия сложно (потому что VIA прячется под дополнительным слоем меди). Кажется, что ножка микросхемы никуда не подключена. Подобное соединение можно обнаружить только прозвонкой или рентгеном (конечно, если нет добытой, например с форума ремонтников, схемы).

Процесс проектирования печатной платы называется трассировкой и делается в специальном ПО (Altium Designer, Cadence Allegro, KiCad и пр.). Часто на печатной плате можно встретить дорожки странного вида, как будто кто-то сделал причудливый узор, на первый взгляд не имеющий практического смысла. Пример распространенных вариантов показан на рисунке ниже.

[1] https://www.altium.com/documentation/altium-nexus/blind-buried-micro-vias.

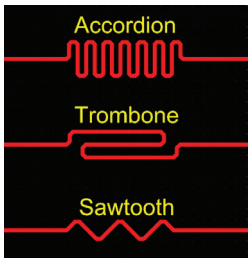

Рис. 0.9. Шаблоны для выравнивая длин проводников на печатной плате устройства[1]

Это выравнивание длин проводников, сигналы по которым должны идти одновременно. Несмотря на то что электрические сигналы распространяются практически со скоростью света (скорость распространения зависит от типа проводника, но в большинстве случаев примерно равна ⅔ от скорости света), при высоких частотах интерфейсов передачи данных дорожки, отличающиеся даже на несколько миллиметров, уже могут быть причиной некорректной работы устройства, т. к. сигналы по ним будут приходить в разное время. Наиболее часто выравнивание встречается для сигналов, передающихся по дифференциальным парам и высокоскоростным параллельным интерфейсам, например DDR. Поэтому когда мы видим на печатной плате эти странные узоры, то сразу понимаем, что это линии какого-то высокоскоростного интерфейса. На фото представлен фрагмент платы Microsoft XBOX, содержащий выровненные по длине проводники.

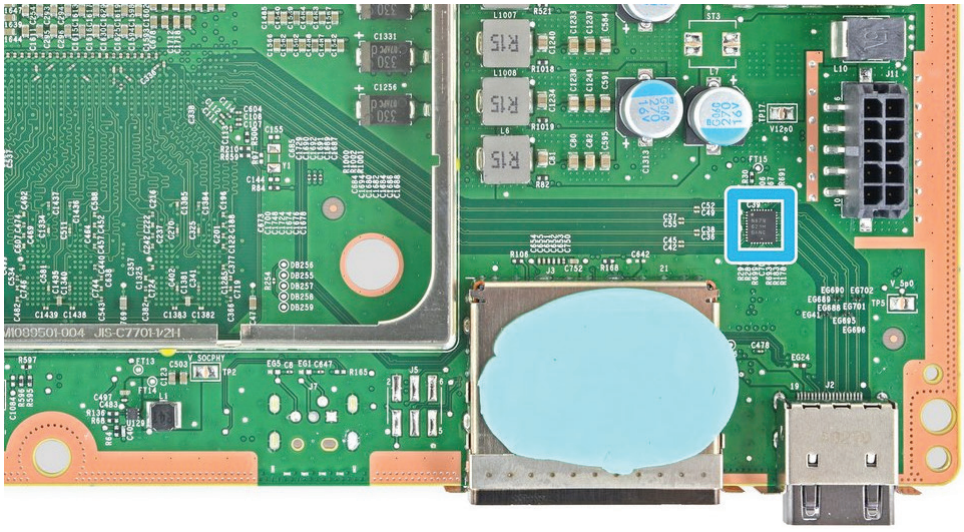

Рис. 0.10. Выровненные проводники на плате устройства[2]

Теперь перейдем к электронным компонентам. Их огромное количество, как и типов корпусов, в которых они существуют. Почти все компоненты мо-

[1] https://www.altium.com/ru/documentation/altium-designer/length-tuning-pcb.
[2] https://ifixit.com/Teardown/Xbox+Series+X+Teardown/138451.

гут быть «упакованы» в разные типы корпусов. Сейчас нас интересует следующее деление: компоненты *выводного* и *поверхностного* монтажа. Отличие – в выводах («ножках») этих компонентов. Наверное, вы видели платы старых устройств, там выводы проходили сквозь печатную плату и запаивались к проводящим слоям с обратной стороны.

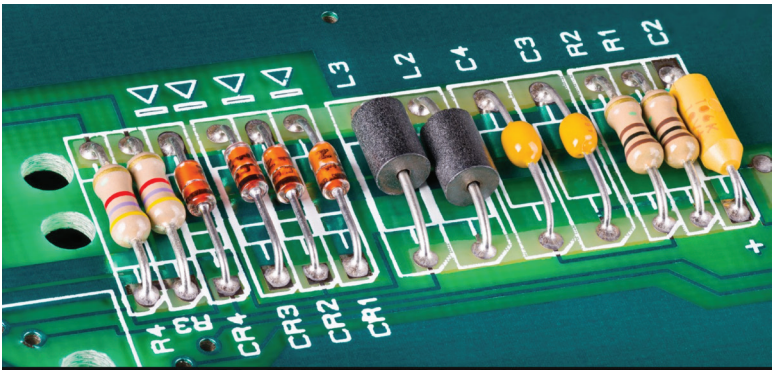

Рис. 0.11. Выводной монтаж компонентов[1]

Сейчас таких компонентов почти не осталось (но, например, их можно встретить в источниках питания, где компоненты могут рассеивать большую мощность). Если компоненты запаиваются с той же стороны, где и находятся, то это *поверхностный* монтаж (такой тип компонентов называется SMD – surface mount devices). Есть разные типы SMD-корпусов: выводные с ножками SOIC, TSOP или безвыводные BGA (Ball Grid Array), WLCSP с шариками контактов под компонентами. Отличие – в плотности и удобстве монтажа. Компонент с BGA-корпусом паяльником уже не запаяешь (да и не отпаяешь). Контакты на печатной плате, к которым подключаются выводы компонентов, называются футпринт (англ. footprint).

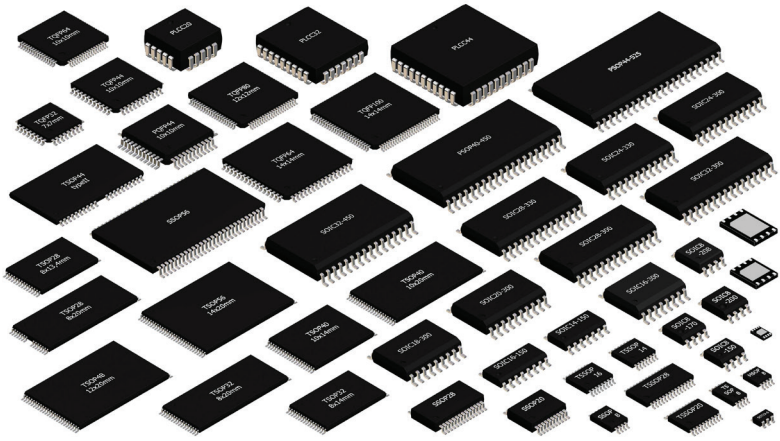

Рис. 0.12. Распространенные корпуса поверхностного монтажа[2]

[1] https://resources.altium.com/p/role-decoupling-inductor-and-resistor-pdn.
[2] https://tened.ru/blog-circuits.

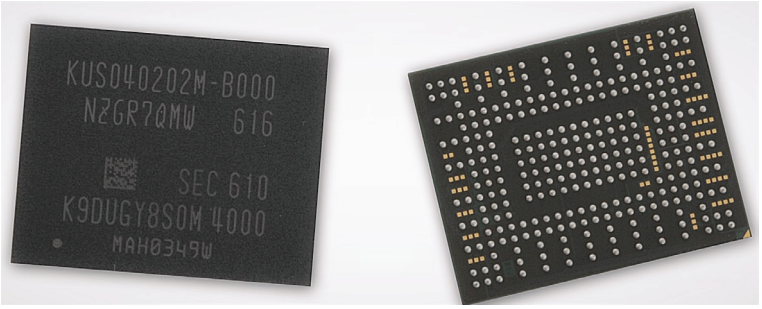

Рис. 0.13. Микросхема с корпусом BGA[1]

Описание процесса производства цифрового устройства

Мы рассмотрели, как выглядит типовое устройство, если вскрыть его корпус. Далее разберем, из каких типовых электронных компонентов состоит устройство, их назначение и какими интерфейсами они могут быть связаны между собой. Но перед этим познакомимся с тем, как выглядит процесс производства устройства на фабриках (не забывая, что разные части устройства могут производиться на разных фабриках).

Типовое цифровое устройство состоит из одной или нескольких электронных плат, установленных в корпус. В первую очередь инженеры разрабатывают электрическую принципиальную схему устройства. После выполняется трассировка печатной платы, результатом которой является полное описание конструкции печатной платы. Далее формируется список компонентов (Bill of Materials, BOM), включающий в себя все компоненты, используемые в плате устройства. Для унификации любое ПО автоматизации проектирования электронных устройств позволяет выгрузить файлы стандарта Gerber, хранящие информацию о проводящем рисунке каждого слоя, отверстия и т. д. Именно эти файлы вместе со списком BOM отправляются на производство. Пример отображения файла стандарта Gerber в ПО просмотра показан на скриншоте ниже.

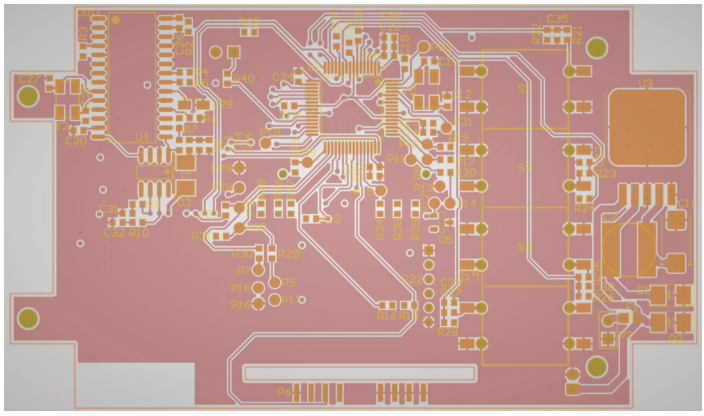

Рис. 0.14. Просмотр файла формата Gerber

[1] https://news.samsung.com/us/512-gigabyte-bga-nvme-ssd-pm971-nvm/.

На выходе первого этапа производства получается печатная плата устройства, полностью покрытая специальной защитной маской, за исключением контактных площадок.

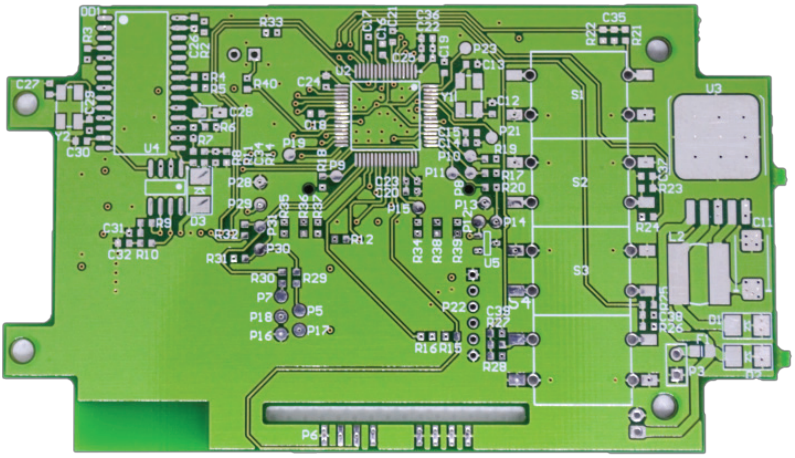

Рис. 0.15. Печатная плата устройства, произведенная на основе файла с форматом Gerber

Далее на контактные площадки наносится паяльная паста (смесь микроскопических шариков припоя с паяльным флюсом) и устанавливаются электронные компоненты (с помощью специальных линий автоматического монтажа) из списка BOM. Печатные платы с установленными компонентами отправляются в специальную печь, где припой, расплавляясь, припаивает выводы компонентов к контактным дорожкам печатной платы. В случае если на печатной плате должны быть установлены элементы, которые не могут подвергаться нагреву в печи, они могут быть запаяны вручную или селективным способом после этапа «запекания» в печи. Далее платы отправляются на процедуры отмывки от остатков флюса и сушки.

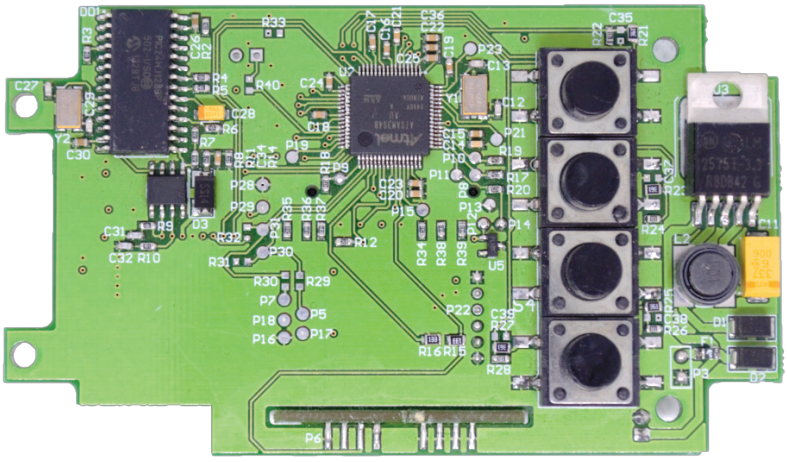

Рис. 0.16. Собранная плата устройства с компонентами

На каждом этапе производства платы устройства существует множество вариантов контроля – оптический, электрический или рентген. Поэтому процент брака при серийном производстве устройства крайне мал. Собранные платы устройства отправляются заказчику или тестируются (и даже программируются) на фабрике. После чего происходят установка плат в корпус и финальное тестирование устройства.

Ознакомившись с общим описанием типового процесса производства серийного цифрового устройства, можно переходить к детальному рассмотрению используемых электронных компонентов. Все компоненты делятся на активные и пассивные. Активные компоненты (микроконтроллер, память, стабилизаторы и т. п.) могут усиливать или преобразовывать электрические сигналы и требуют внешнего источника питания для своей работы. Пассивные элементы (резисторы, конденсаторы, разъемы и т. п.) не нуждаются во внешнем питании. Нас больше всего интересуют активные компоненты. Сейчас наша задача – идентифицировать маркировку основных компонентов и понять их назначение, вбивая маркировку в поисковый запрос и изучая документацию на микросхемы, называемую даташит (datasheet).

Маркировка компонентов на плате устройства

Довольно часто на печатной плате можно встретить надписи (состоящие из одной или нескольких букв и порядкового номера), по которым можно восстановить тип электронного компонента. Ранее мы рассматривали, что процесс нанесения краски на печатную плату устройства называется шелкографией. Рассмотрим основные обозначения на примере отладочной платы фирмы STM Nucleo (стоит учитывать, что иногда производители отходят от распространенных обозначений и маркируют компоненты по-своему):

- U – микросхема (реже встречается обозначение DD для цифровых микросхем и DA для аналоговых);
- J, CN – разъем, JP – перемычка;
- C – конденсатор;
- R – резистор;
- L – индуктивность;
- Q – транзистор;
- VD, D – диод;
- LD – светодиод;
- F – предохранитель;
- X – кварцевый резонатор.

Более подробный список представлен в Википедии (**https://en.wikipedia.org/wiki/Reference_designator**), а полный перечень обозначений элементов, принятых у нас в стране, можно посмотреть в ГОСТ (**http://www.robot.bmstu.ru/files/GOST/gost_2.710-81.pdf**).