Министерство науки и высшего образования Республики Казахстан

Карагандинский университет имени академика Е.А. Букетова

Т.А. Рахимгалиев

Проектирование встраиваемых систем на примере одноплатного компьютера Orange Pi3 LTS

**ДИПЛОМНАЯ РАБОТА**

Образовательная программа

6B06201 – «Радиотехника, электроника и телекоммуникации»

Министерство науки и высшего образования Республики Казахстан

Карагандинский университет имени академика Е.А. Букетова

«Допущен к защите»

Заведующая кафедрой РФиЭ

доктор PhD, ассоц.проф.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Г.К. Алпысова

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 202\_\_\_ г.

**ДИПЛОМНАЯ РАБОТА**

**на тему:**

**«Проектирование встраиваемых систем на примере одноплатного компьютера Orange Pi3 LTS»**

по образовательной программе

6B06201 – «Радиотехника, электроника и телекоммуникации»

Искомая степень: бакалавр

Выполнил:

студент Т.А. Рахимгалиев

Научный руководитель:

к.т.н., ассоц. проф., Л.В. Чиркова

Карагандинский университет имени академика Е.А. Букетова

Факультет физико-технический

Образовательная программа «6B06201 Радиотехника, электроника и телекоммуникации»

Кафедра радиофизики и электроники

«УТВЕРЖДАЮ»

Заведующий кафедрой

радиофизики и электроники

\_\_\_\_\_\_\_\_\_\_\_\_\_ Г.К. Алпысова

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 202\_\_ г.

**ЗАДАНИЕ**

**на выполнение дипломной работы**

Студенту \_\_\_\_\_\_\_\_\_\_\_\_\_Рахимгалиеву Т.А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ф.И.О.

4 курс, РТ-413р-21, 6B06201 – РЭТ, очная образовательная программа.

1. Тема выпускной работы «Проектирование встраиваемых систем на примере одноплатного компьютера Orange Pi3 LTS» Утверждена приказом Председателя Правления-Ректора НАО «Карагандинский университет им. академика Е.А. Букетова» от «26» ноября 2024 г.

2. Срок сдачи студентом законченной работы «25» мая 2025 г.

3. Исходные данные к работе (законы, литературные источники, лабораторно-производственные данные)

* Техническая документация микросхемы AXP805 (datasheet)
* Стандарты проектирования печатных плат: IPC-7351, ГОСТ Р 55693-2013
* Методические указания кафедры по выполнению выпускных квалификационных работ
* Рекомендации по использованию ECAD-систем (Altium Designer)
* Лабораторные наработки по работе с Orange Pi 3 LTS
* Требования к файловой системе встраиваемых систем (Filesystem Hierarchy Standard, Linux Foundation)

4. Перечень вопросов, подлежащих к разработке в выпускной работе

1. Анализ и выбор инструментов ECAD для проектирования печатных плат
2. Проектирование и моделирование подсистемы управления питанием с использованием AXP805
3. Разработка программной среды для микрокомпьютера Orange Pi 3 LTS на базе Linux

5. Перечень графических материалов (чертежи, таблицы, диаграммы и т.д.)

1. Принципиальная схема подсистемы питания
2. 3D-модель и топология печатной платы
3. Таблица компонентов (BOM)
4. Диаграмма загрузки системы (ROM → Bootloader → Kernel → RootFS)
5. Иллюстрации интерфейса Altium Designer и Docker-среды

6. Перечень рекомендуемых источников:

1. AXP805 Datasheet — SUNXI
2. IPC-7351: Generic Requirements for Surface Mount Design and Land Pattern Standard
3. «The Art of Electronics», Horowitz & Hill
4. Документация по Altium Designer (официальный сайт)
5. Embedded Linux System Development Training — Bootlin
6. Хабр: статьи по сборке прошивки для Orange Pi и проектированию электроники

7. График выполнения выпускной работы

|  |  |  |  |
| --- | --- | --- | --- |
| № | Этапы работы | Сроки выполнения | Примечание |
| 1 | 2 | 3 | 4 |
| 1 | Сбор материалов | Сентябрь-декабрь 2024 г. |  |
| 2 | Разработка структуры выпускной работы | Сентябрь-октябрь 2024 г. |  |
| 3 | Подготовка Введения работы | Декабрь 2024 г. |  |
| 4 | Подготовка разделов работы | Январь-февраль 2025 г. |  |
| 5 | Подготовка Заключения работы | Апрель 2025 г. |  |
| 6 | Консультации научного руководителя | Сентябрь-май 2025 г. |  |
| 7 | Предоставление выпускной работы для проверки на плагиат и для нормоконтроля | Май 2025 г. | За полтора месяца до итоговой аттестации |
| 8 | Предоставление выпускной работы на предзащиту | По графику предзащиты май 2025 г. | За месяц до начала итоговой аттестации |
| 9 | Предоставление выпускной работы на рецензию | 26.05.2025 г. | В течение первой недели после предзащиты |
| 10 | Предоставление окончательного варианта выпускной работы с отзывом научного руководителя и рецензией | 2.06.2025 г. | За три дня до начала итоговой аттестации |
| 11 | Подготовка доклада на защиту | 2.06.2025 г. | За три дня до начала итоговой аттестации |
| 11 | Защита выпускной работы | По графику | По расписанию |

Дата выдачи задания «15 » октября 2024 г.

Научный руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_к.т.н., асс. профессор Чиркова Л.В

(подпись) Ф.И.О., ученое звание, должность

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Т.А. Рахимгалиев

(подпись) Ф.И.О.

Содержание

[Введение 7](#_Toc198844069)

[1 Проектирование печатной платы 10](#_Toc198844070)

[1.1 Выбор системы автоматизации проектирования электроники (ECAD) 11](#_Toc198844071)

[1.2 Создание библиотеки компонентов 11](#_Toc198844072)

[1.3 Проектирование принципиальной схемы 21](#_Toc198844073)

[1.4 Расположение компонентов на плате 25](#_Toc198844074)

[1.5 Топология печатной платы 26](#_Toc198844075)

[2 Сборка программного комплекса 31](#_Toc198844076)

[2.1 Linux для встраиваемых систем 31](#_Toc198844077)

[2.2 Подготовка рабочей среды и компиляция системы 33](#_Toc198844078)

[Заключение 36](#_Toc198844079)

[Список использованных источников 38](#_Toc198844080)

[Приложение A 40](#_Toc198844081)

[Приложение B 46](#_Toc198844082)

# Введение

Развитие встраиваемых систем является одним из ключевых направлений современной электроники и телекоммуникаций. Под встраиваемыми системами понимаются специализированные вычислительные устройства, встроенные в состав различных технических средств и предназначенные для выполнения одной или нескольких строго определённых функций в режиме реального времени. Такие устройства активно используются в самых разных сферах: от бытовых приборов и автомобилей до промышленных контроллеров, медицинского оборудования, телеметрических систем и систем безопасности. Объёмы применения встраиваемых систем продолжают расти с каждым годом, особенно с учётом стремительного развития Интернета вещей (IoT), автоматизации и цифровизации всех сфер жизни.

Отличительной особенностью встраиваемых решений является высокая степень интеграции, автономности и ориентации на конкретную задачу. В отличие от универсальных компьютеров, встраиваемая система проектируется с прицелом на минимизацию затрат, энергопотребления и объёма программного обеспечения. Поэтому надёжность и стабильность их работы во многом зависят от правильного проектирования как аппаратной, так и программной части. Ошибки, допущенные на раннем этапе, могут привести к необходимости перепроектирования всего устройства, что требует значительных временных и финансовых ресурсов.

В контексте глобальных тенденций развития микроэлектроники, актуальной становится задача комплексного проектирования встроенных систем, начиная от схемотехники и заканчивая созданием оптимизированного программного окружения. Учитывая это, практические навыки проектирования встроенных решений становятся неотъемлемой частью подготовки современного инженера в области радиотехники, электроники и телекоммуникаций.

Данная дипломная работа посвящена практической реализации встроенной системы на примере одноплатного компьютера Orange Pi 3 LTS. Этот микрокомпьютер представляет собой универсальную платформу, сочетающую в себе мощный процессор, контроллеры ввода-вывода, достаточные ресурсы памяти и широкий набор интерфейсов. Он является конкурентом известного Raspberry Pi и активно используется в образовательных, исследовательских и промышленных проектах. Благодаря открытому аппаратному обеспечению и доступной стоимости, Orange Pi 3 LTS представляет интерес как объект исследования и разработки.

Несмотря на высокую функциональность, для полноценного использования Orange Pi 3 LTS требуется настроенная система электропитания и адаптированная операционная система. Без этого невозможно обеспечить надёжную и стабильную работу устройства в условиях ограниченных ресурсов. Именно эти задачи — аппаратная интеграция и программная адаптация — и были поставлены в рамках дипломной работы. Актуальность выбранной темы обусловлена необходимостью подготовки специалистов, способных выполнять полный цикл проектирования встраиваемых систем — от схемотехники до программной реализации.

Цель работы — создание полноценной цепи питания и сборка собственной минималистичной операционной системы Linux для платформы Orange Pi 3 LTS. Реализация поставленной цели требует междисциплинарного подхода: с одной стороны — глубоких знаний в области схемотехники, с другой — понимания архитектуры операционных систем, компиляции, работы с низкоуровневыми инструментами и кросс-компиляцией.

Для достижения данной цели необходимо было пройти весь путь: от анализа требований и выбора компонентов — до разработки схемы, трассировки печатной платы, компиляции ядра и формирования загрузочного образа. Такой подход позволяет не только получить практически работоспособную систему, но и сформировать системное понимание всех этапов её создания, что особенно важно для будущей инженерной деятельности.

На первом этапе был выполнен подробный анализ требований к питанию микрокомпьютера. Платформа Orange Pi 3 LTS требует нескольких стабилизированных напряжений, необходимых для работы процессора, периферии и логических узлов. Для этих целей была выбрана специализированная микросхема AXP805, представляющая собой PMIC (Power Management Integrated Circuit), предназначенную для систем на кристалле. Эта микросхема управляет пятью независимыми каналами питания, имеет встроенные регуляторы напряжения и интерфейс конфигурации. Благодаря широким возможностям настройки и высокой интеграции, она идеально подошла для построения гибкой системы питания, что позволило сократить количество внешних компонентов и улучшить надёжность всей схемы.

Процесс аппаратного проектирования начался с создания библиотеки компонентов в ECAD-среде Altium Designer — одного из наиболее распространённых инструментов для проектирования печатных плат. Были разработаны условно-графические обозначения (УГО), посадочные места (footprints) и трёхмерные модели всех компонентов, включая AXP805, конденсаторы, резисторы, индуктивности и разъёмы. Далее была построена принципиальная схема, в которой учтены все требования по подключению, фильтрации помех, развязке питания и защите от перегрузок. Особое внимание уделялось правильному размещению конденсаторов, трассировке земляных полигонов и обеспечению стабильной подачи напряжений — критически важным аспектам при работе с чувствительной цифровой логикой.

После завершения проектирования схемы была выполнена трассировка печатной платы. Сформирована топология, учитывающая минимальные расстояния, допустимую ширину проводников и рекомендации по размещению компонентов. Проект был проверен с помощью встроенного инструмента Design Rule Check, что позволило выявить и устранить возможные ошибки до этапа производства. Правильно выполненная трассировка не только улучшает электрические характеристики устройства, но и упрощает его сборку и последующую отладку.

Вторая часть работы — разработка программного обеспечения — имела своей целью получение полностью работоспособной и минимизированной операционной системы, адаптированной под платформу Orange Pi 3 LTS. В отличие от использования готовых дистрибутивов, была выбрана стратегия ручной сборки системы, что позволило контролировать все этапы, уменьшить размер итоговой прошивки и точно соответствовать аппаратным требованиям. Были вручную собраны все необходимые программные компоненты: загрузчик U-Boot, ядро Linux, системная библиотека glibc и набор базовых утилит BusyBox. Использование Docker-контейнера позволило создать чистую и управляемую среду для сборки, избежать конфликтов зависимостей и гарантировать воспроизводимость результата.

Компиляция выполнялась с применением кросс-компилятора для архитектуры ARM64. Все действия были детально прописаны в Dockerfile, что обеспечивает автоматизацию сборки и облегчает повторение всех шагов другими разработчиками или в рамках будущих проектов. На выходе был получен загрузочный образ, который можно записать на SD-карту и использовать для загрузки Orange Pi 3 LTS. Система протестирована на устойчивость, стабильность загрузки и работоспособность основных компонентов.

Таким образом, работа охватывает полный цикл создания встроенной системы — от проектирования схемы питания до реализации собственной Linux-платформы. Это даёт не только практические навыки, но и глубокое понимание процессов, лежащих в основе современной электроники. Выполненная дипломная работа демонстрирует, что при грамотной организации труда и наличии фундаментальных знаний студент может самостоятельно реализовать проект уровня, приближённого к промышленному, что делает его ценным вкладом в профессиональное становление.

# Проектирование печатной платы

Проектирование печатной платы является одним из ключевых этапов разработки любого устройства. Для реализации данного процесса применяются системы автоматизированного проектирования (CAD — Computer-aided design). Выбор соответствующей системы CAD определяется особенностями отрасли: например, для специалистов в области электроники используется ECAD (Electrical computer-aided design) [1], в то время как инженеры конструкторы выбирают из программ типа MCAD (Mechanical computer-aided design).

Кроме того, у специалистов в области разработки аппаратного обеспечения используется система управления жизненным циклом продукта (PLM — Product Lifecycle Management) [2], которая обеспечивает интеграцию и взаимодействие между различными CAD системами. PLM-системы позволяют эффективно управлять данными, связанными с проектированием, производством и эксплуатацией продукции, обеспечивая координацию между различными этапами разработки и участниками процесса. Например, инженер электронщик, после расположения компонентов на печатной плате, может передать трехмерную модель платы инженеру конструктору, который в свою очередь сделает корпус устройства. В ситуациях, когда PLM нет — используется нейтральный формат для передачи трехмерных моделей, такой как STEP.

На рисунке 1 перечислены существующие ECAD / PLM / MCAD системы, отдельно выделены стоящие рассмотрения. В рамках данной работы будет рассмотрена исключительно ECAD система.

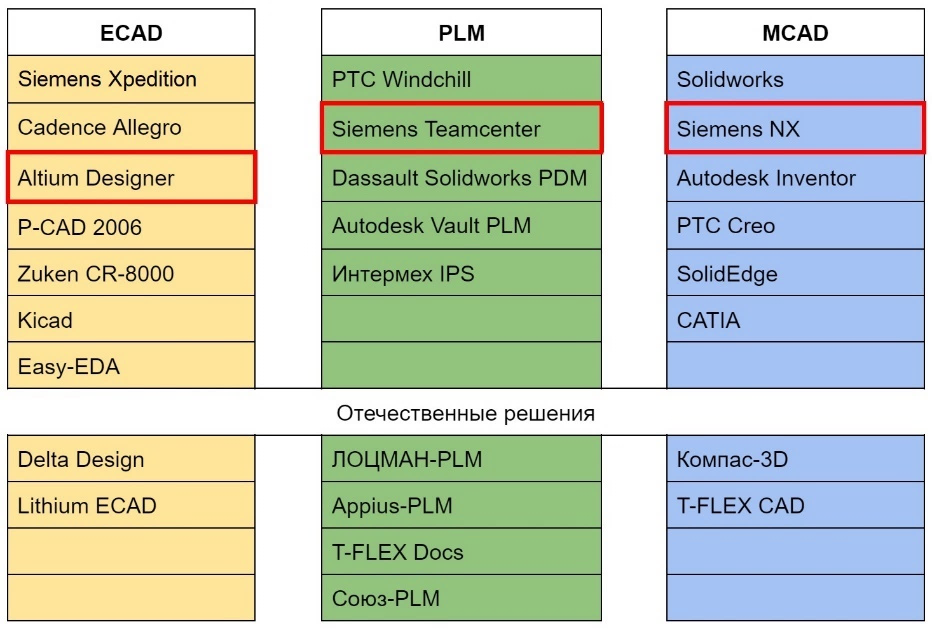


Рисунок 1. Существующие CAD системы. Выделены наиболее актуальные

Примечание — отдельного упоминания стоит система контроля версий (например git), которая может быть использована для сохранения и быстрого возвращения к предыдущей версии проекта, что сильно упрощается разработку, если по каким-то обстоятельствам на более поздних этапах обнаруживается критическая ошибка.

## Выбор системы автоматизации проектирования электроники (ECAD)

В настоящее время существует множество программных продуктов для ECAD, каждый из которых обладает своими особенностями и предназначен для различных категорий пользователей. Например, KiCad — это бесплатная и открытая система автоматизированного проектирования, которая является хорошим выбором для пользователей, не располагающих бюджетом на коммерческие решения.

Среди профессиональных решений особое место занимает Cadence Allegro — один из самых мощных и сложных инструментов ECAD, который применяется в таких высокотехнологичных компаниях, как Intel и Apple. Эта система ориентирована на корпоративных пользователей, работающих с многослойными платами высокой плотности и требующих строгого соблюдения требований к сигналам, частотам и электромагнитной совместимости. Cadence Allegro поддерживает широкие возможности для симуляции, совместной работы и интеграции в корпоративные процессы разработки. Однако сложность интерфейса и высокая стоимость делают его менее доступным для студентов и начинающих инженеров.

С другой стороны, Altium Designer — является коммерческим продуктом, ориентированным на средние и крупные компании, которые требуют более мощных и функционально насыщенных инструментов для разработки сложных электронных систем. Altium предоставляет широкий спектр возможностей для проектирования, включая интеграцию с другими системами и поддержку сложных проектов. Имеется студенческая лицензия.

В рамках выполнения дипломной работы будет использоваться Altium Designer, поскольку данный инструмент ориентирован на применение в малых и средних предприятиях, что делает его актуальным как с образовательной, так и с практической точки зрения.

## Создание библиотеки компонентов

Поскольку в рамках данной работы в качестве примера используется микрокомпьютер Orange Pi 3 LTS, а полное рассмотрение процесса его проектирования выходит за пределы объема выпускной работы, внимание сосредоточено на одной из его подсистем. В качестве иллюстрации выбран процесс проектирования системы управления питанием. На её примере последовательно рассматриваются этапы выбора компонентов, разработки принципиальной схемы и проектирования печатной платы.

Объекта исследования — интегральная схема управления питанием AXP805, которая является PMIC (Power Management Integrated Circuit) и отвечает за управление цепями питания на данной платформе.

### Создание проекта

Для создания проекта в главном меню выбирается пункт *File → New → Project*. Далее задаётся имя проекта и его расположение на диске. Например PowerManagement.PrjPcb.

После создания проекта требуется добавить в его структуру основные файлы: схему (schematic), топологию печатной платы (PCB layout), а также библиотеки условно-графических обозначений (symbol libraries) и посадочных мест (footprint libraries). Для этого необходимо щёлкнуть правой кнопкой мыши по имени проекта в панели навигации и выбрать пункт *Add New to Project*. В появившемся контекстном меню следует выбрать соответствующие элементы, задать им имена и сохранить.

В результате дерево проекта должно приобрести структуру, аналогичную представленной на рисунке 2.



Рисунок 2. Структура проекта после добавления всех необходимых для работы файлов

### Получение информации о компоненте AXP 805.

Информацию о компоненте Altium Designer и другие ECAD импортируют с сайтов крупных поставщиков. В число таких входит DigiKey, Mouser Electronics или Octopart. Последний, строго говоря, не является поставщиком, а всего лишь предоставляет функционал для поиска электронных компонентов, их УГО, 3D моделей и посадочных мест.

Для того чтобы импортировать компонент необходимо знать его артикул (part number), который можно выяснить, поискав компонент на сайте поставщика. PMIC AXP 805 имеет артикул AXP805 [3]. Там же находятся некоторые характеристики компонента, продублированные из технической спецификации для удобства поиска, что будет продемонстрировано позже.

Для импорта информации о компоненте нужно перейти в панель Manufacter Part Research. В строке поиска (*Search*) нужно ввести артикул компонента (AXP805). После загрузки компонент отобразится, а третьей строкой, под его артикулом, будет находиться кнопка «N SPNs», где N это количество доступных поставщиков, а SPNs акроним от Supplier Part Number. Получив список поставщиком, нужно кликнуть на желаемого поставщика правой кнопкой мыши и в контекстном меню выбрать *Import Into PowerManagement.SchLib As <Part Number>*, как показано на рисунке 3.

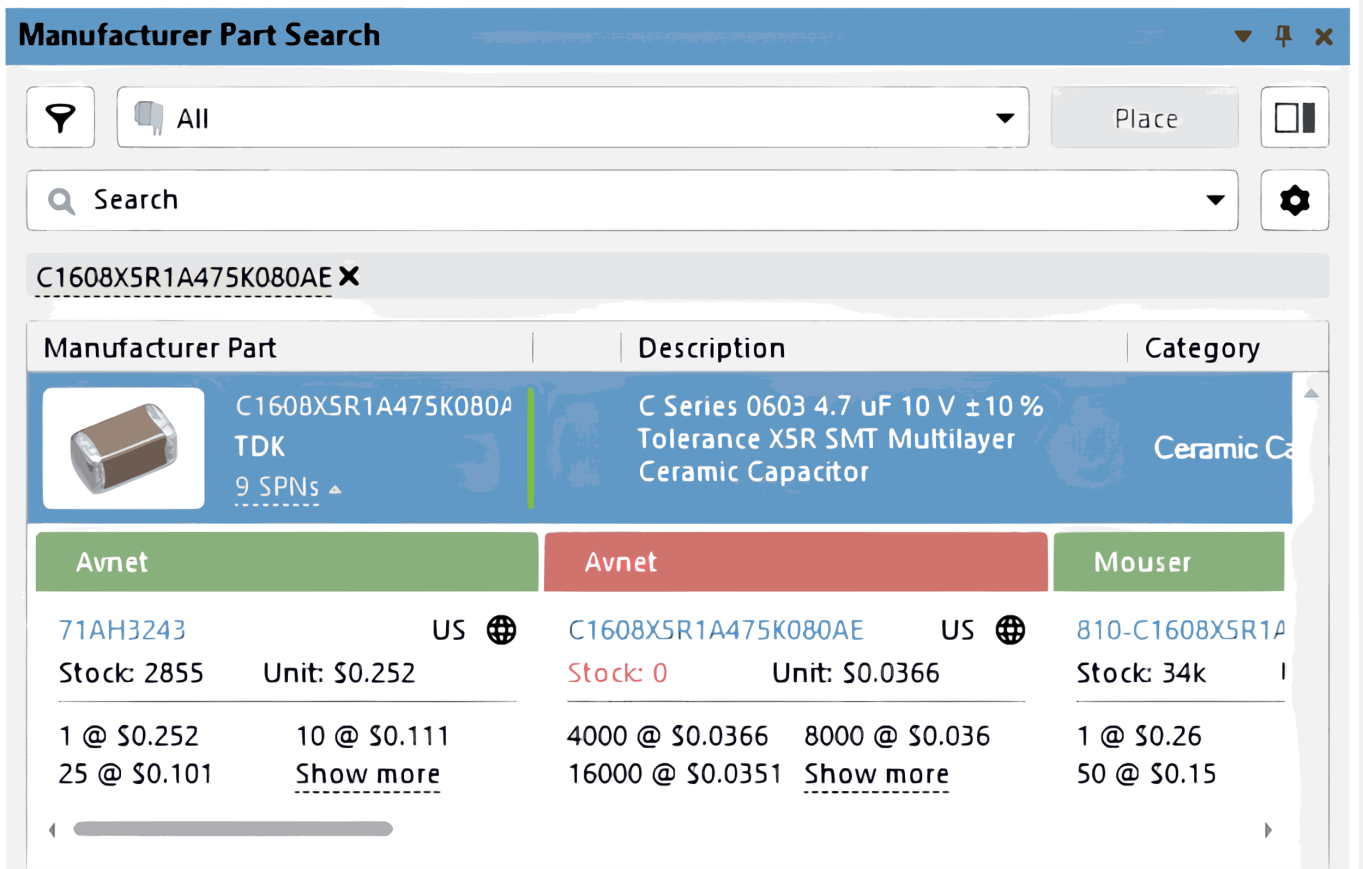


Рисунок 3. Панель Manufacter Part Research. Импортируется конденсатор

Примечание — настоятельно рекомендуется использовать техническое программное обеспечение на английском языке. Такая практика обусловлена рядом объективных причин. Во-первых, подавляющее большинство обучающих материалов, технической документации и справочной информации представлено именно на английском языке. Во-вторых, большинство программных продуктов изначально разрабатываются с ориентацией на англоязычный интерфейс, что гарантирует корректность отображения терминов и стабильность работы. В-третьих, локализация программного обеспечения на другие языки зачастую сопровождается неточностями перевода, что может привести к неправильному пониманию функций и затруднить процесс обучения или разработки.

На рисунке 4 показана импортированная от поставщика информация. Основное на что следует обратить внимание — это производитель, артикул компонента и ссылки на поставщика. На основе этих данных в дальнейшем можно сформировать спецификацию материалов (Bill of Materials, BOM) — документ, служащий основой для заказа компонентов, необходимых при производстве и сборке печатной платы на производстве.

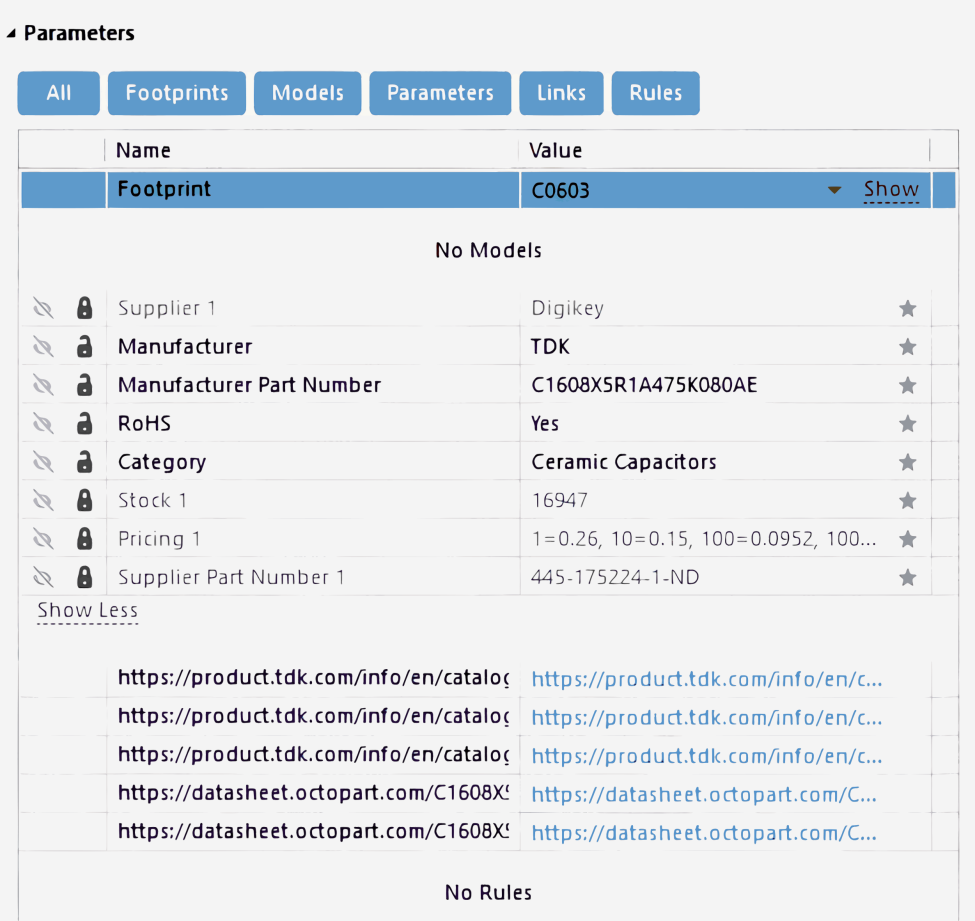


Рисунок 4. Импортированная от поставщика информация

Для того чтобы завершить импорт компонента ему необходимо назначить дезигнатор, который позже будет использоваться при его нумерации на схеме. На рисунке 5 интегральной микросхеме назначается дезигнатор *U?*. U — это префикс для всех интегрированных микросхем. На место вопроса будет подставлен номер компонента на схеме.

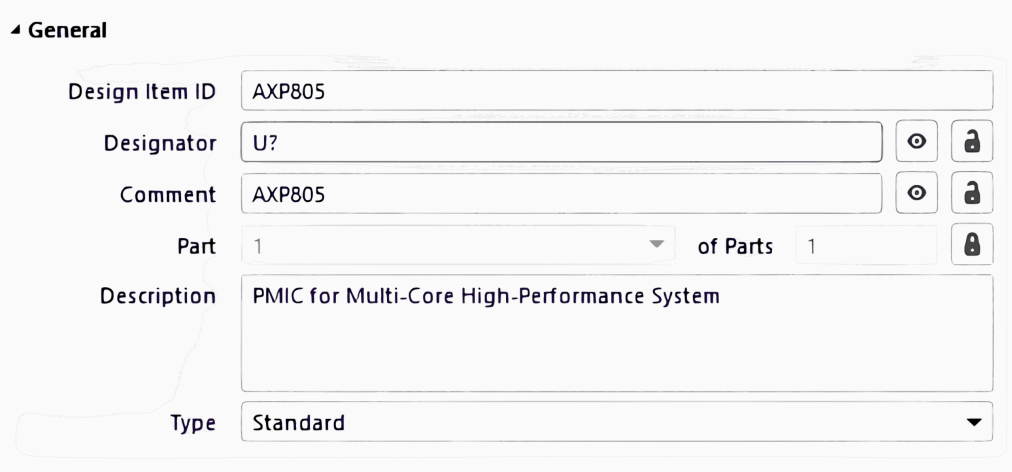


Рисунок 5. Основная информация о компоненте AXP805

### Создание условного графического обозначения AXP 805.

Для создания УГО компонента следует ознакомиться с назначением выводов AXP805 в технической спецификации [5], показанного на рисунке 6, затем скопировать и разместить информацию о них в модели.

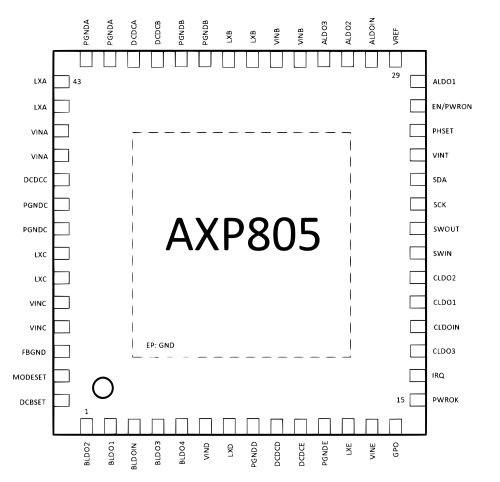


Рисунок 6. Схема выводов AXP805, нумерация выводов против часовой стрелки

Для размещения вывода в редакторе компонента используется инструмент *Place → Pin*. После его активации с помощью клавиши TAB открывается окно *Properties*, в котором указываются дезигнатор и имя вывода (поле Name), соответствующее данным из технической документации. Аналогичным образом осуществляется добавление всех остальных выводов.

Затем поверх размещённых выводов добавляется прямоугольник, который вызывается с помощью команды *Place → Rectangle*. Для корректного отображения прямоугольника под остальными элементами условного графического обозначения необходимо воспользоваться командой *Edit → Move → Send to Back*, после чего указать элемент, который требуется переместить на задний план. Результат данного действия представлен в приложении A, на рисунке A.1.

Примечание — строго регламентированных правил для размещения пинов на условно-графическом обозначении компонента не существует, однако есть общепринятые рекомендации [6], соблюдение которых способствует улучшению читаемости схем и упрощает трассировку.

Входные сигналы (Input) рекомендуется располагать с левой стороны условного графического обозначения, а выходные сигналы (Output) — с правой. Такой подход способствует интуитивному восприятию направления прохождения сигнала через компонент. Линии питания целесообразно размещать в верхней части символа, а соединения с землёй — в нижней. Это улучшает читаемость схемы и способствует упорядочиванию электрической иерархии.

В случаях, когда строгое соблюдение указанных рекомендаций затрудняет восприятие схемы или её компоновку, допускается отступление от этих правил в пользу повышения наглядности и удобства проектирования.

### Создание посадочного места для AXP 805.

Микросхема AXP805 поставляется производителем в стандартном корпусе типа QFN 7×7 мм (Quad Flat No-Lead), что упрощает процесс создания посадочного места (footprint). Благодаря наличию точных размеров в технической документации и стандартизированному типу корпуса [7], посадочное место можно сгенерировать автоматически с использованием встроенных инструментов Altium Designer.

Для этого необходимо перейти в редактор библиотеки посадочных мест и нажав на вкладку *Tools → IPC Compliant Footprint Wizard* выбрать из доступных типов компонентов QFN. Программа предоставит форму, которую необходимо заполнить данными из раздела Package технической спецификации AXP 805 [5]. Назначьте компоненту имя QFN7X7MM и запустите генерацию.

Примечание — Инструмент *IPC Compliant Footprint* генерирует посадочные места в соответствии стандарту IPC-7351 [8], который регламентирует размеры и допуски для компонентов поверхностного монтажа. Altium Designer предоставляет более универсальный инструмент Footprint Wizard, который выходит за рамки стандарта IPC-7351, обеспечивая большую гибкость в проектировании посадочных мест.

После генерации посадочного места необходимо перейти в правую нижнюю часть редактора и открыть вкладку *Panels → PCBLIB Filters*. С помощью запроса фильтрации IsVia производится выделение всех переходных отверстий, которые затем подлежат удалению. После выполнения операции фильтрацию следует отключить. В результате будет получено посадочное место, представленное на рисунке 7.

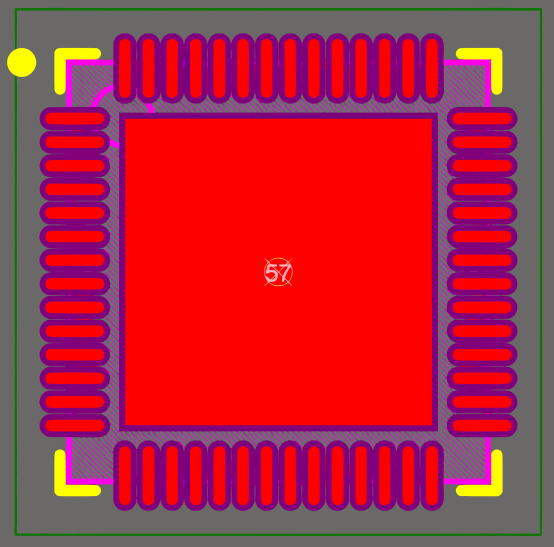


Рисунок 7. Посадочное место QFN7X7MM

После создания посадочного места его необходимо связать с УГО. Для этого в редакторе библиотек УГО в нижней панели моделей выбирается команда *Add Footprint → Browse…* из списка выбирается посадочное место QFN7X7MM и выбор подтверждается. На этом этапе компонент считается полностью подготовленным к использованию в проекте.

### Создание резисторов, конденсаторов и катушки индуктивности.

Для реализации типовой схемы применения, представленной в технической документации AXP 805 [5] и продублированной в приложении A на рисунке A.2, необходимо добавить несколько конденсаторов, резисторов и катушек индуктивностей.

#### Конденсаторы

Для обеспечения эффективной развязки питания интегральной микросхемы малого форм-фактора были выбраны керамические конденсаторы в корпусе 0603 с температурным коэффициентом X5R и номинальным напряжением 10 В. Обозначение X5R относится к системе классификации температурной стабильности керамических диэлектриков, установленной стандартом EIA.

Данные по каждому выбранному компоненту приведены в приложении A, таблице A.1.

При выборе конденсатора для развязки питания важно учитывать номинальное напряжение, которое должно быть не менее 1,5–2 раз выше рабочего напряжения системы, чтобы обеспечить надежность и долговечность компонента. Кроме того, использование керамических конденсаторов класса X5R или X7R предпочтительно для развязки питания из-за их стабильности и низких потерь.

После импорта данных из справочника / поставщика в проект, необходимо заполнить дезигнатор конденсатора и комментарий, в данном случае это *C?* и 10u / 10V соответственно (рисунок 8).

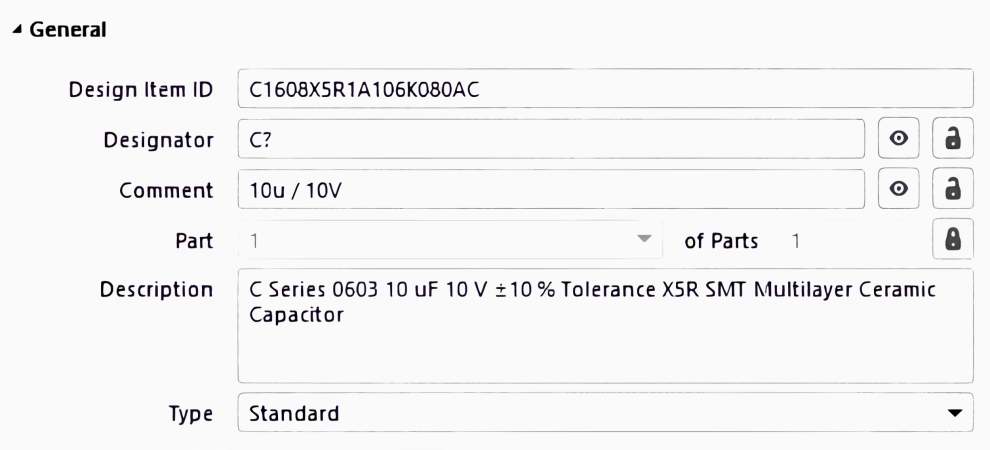


Рисунок 8. Основная информация о конденсаторе

Поскольку во всех случаях используется типовой корпус, достаточно создать посадочное место один раз, после чего оно может быть повторно использовано для аналогичных компонентов. Для этого необходимо открыть редактор посадочных мест, перейти во вкладку *Tools → Footprint Wizard → Capacitors → Surface Mount* и задать соответствующие размеры контактных площадок.

Для многослойного керамического конденсатора (Multilayer Ceramic Capacitor, MLCC), предназначенного для монтажа методом пайки оплавлением (reflow soldering) и выполненного в корпусе 0603, контактные площадки будут представлять квадраты со сторонами 0,8 мм и расстоянием в 1,1 мм от центра каждого [10], как показано на рисунке 9.

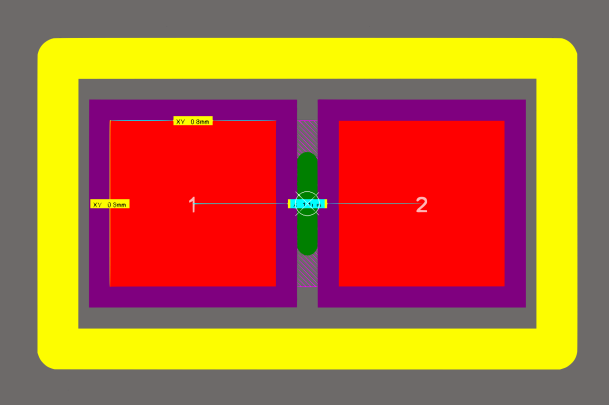


Рисунок 9. Контактные площадки конденсатора в корпусе 0603, со сторонами 0,8 мм и расстоянием 1,1 мм

Примечание — Как можно увидеть из документа [10], существует разница в подходах к созданию посадочных мест для пайки методом оплавления (reflow soldering) и пайки волной (wave soldering). Первый метод используется для компонентов поверхностного монтажа (SMT), в то время как второй — для компонентов сквозного монтажа (THT). В некоторых случаях необходимо комбинировать эти методы. Более того, в случае если заранее известно, что компоненты будут монтировать на плату вручную, можно принять ряд определенных мер уже на этапе создания посадочного места, чтобы упростить данную задачу. Например, следовать стандартам IPC [2] или специальным образом увеличить контактные площадки.

После того как посадочное место будет готово обязательно нужно найти и импортировать трехмерную модель компонента, это значительно упростит процесс создания корпуса и позволит заранее избежать ошибок при расположении компонентов на плате. Для поиска трехмерных моделей можно использовать один из следующий ресурсов: *3dcontentcentral.com*, *grabcad.com*, *snapeda.com*. Обычно в справочниках или у поставщиков есть ссылки на готовые модели. Кроме того, некоторые производители публикуют готовые модели продукции на своих сайтах.

После того как файл модели найден, следует перейти к следующему шагу.

В редакторе необходимо перейти во вкладку *Place → 3D Body*. В открывшемся диалоговом окне выбирается STEP файл с моделью компонента. После импорта 3D-модели её необходимо выровнять относительно уровня контактных площадок. Для этого следует перейти в трёхмерный режим (клавиша 3) и выбрать *Tools → 3D Body Placement* → *Align Face With Board*. Затем следует кликнуть на модель и на плоскость, к которой она должна быть выровнена относительно посадочного места. Программа автоматически выполнит выравнивание.

Иногда 3D-модель компонента не содержит центральной точки привязки (snap point), что затрудняет её точное размещение по центру посадочного места, как показано на рисунке 10. В таких случаях необходимо вручную задать точку привязки для упрощения позиционирования. Для этого следует дважды щёлкнуть на 3D-модели, чтобы открыть окно свойств (*Properties*), и задать координаты S*nap Points*, рассчитав их как половину длины и половину ширины модели. Значения длины и ширины можно измерить непосредственно в редакторе с помощью сочетания клавиш *Ctrl + M*.

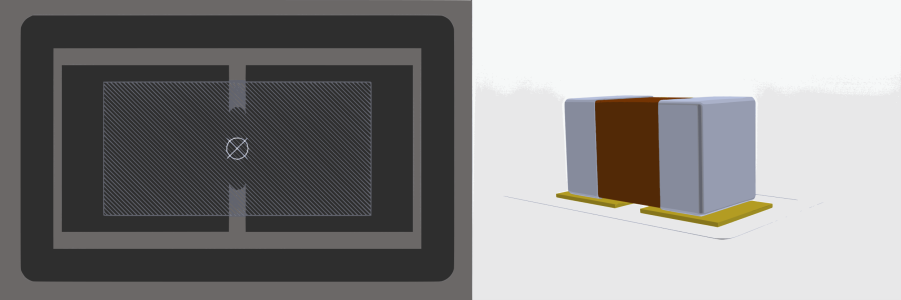


Рисунок 10. Трехмерная модель и ее проекция (вид сверху)

После завершения посадочного места его можно привязать к УГО (поскольку само УГО типовое, его изображение здесь было опущено).

Остальные компоненты оформляются аналогичным образом: осуществляется импорт данных о компоненте с сайта поставщика, создаётся УГО, разрабатывается посадочное место и подбирается соответствующая 3D-модель (при наличии ранее созданного, например для конденсатора 0603, его можно использовать повторно). После этого посадочное место связывается с УГО для дальнейшего использования в проекте.

#### Катушка индуктивности

Микросхема AXP805 содержит три понижающих DC-DC конвертера (step-down), которые осуществляют подачу питания с использованием индуктивных элементов. Согласно разделу 7 (электрические характеристики) технической спецификации [5], через каналы DCDCA, DCDCB и DCDCC протекает ток до 3А, а через DCDCD и DCDCE — до 2А. На основании этих значений необходимо выбрать подходящую катушку индуктивности.

Процесс выбора осуществляется аналогично подбору конденсаторов: в электронном справочнике (или на сайте поставщика) подбирается компонент, соответствующий заданным параметрам, после чего его данные импортируются в Altium Designer. Далее создаётся УГО, посадочное место и трехмерная модель, которая в случае катушки индуктивности особенно критична из-за ее габаритов. Данные по выбранному компоненту приведены в приложении A, таблице A.1.

Посадочное место и трехмерная модель катушки индуктивности показана на рисунке 11.

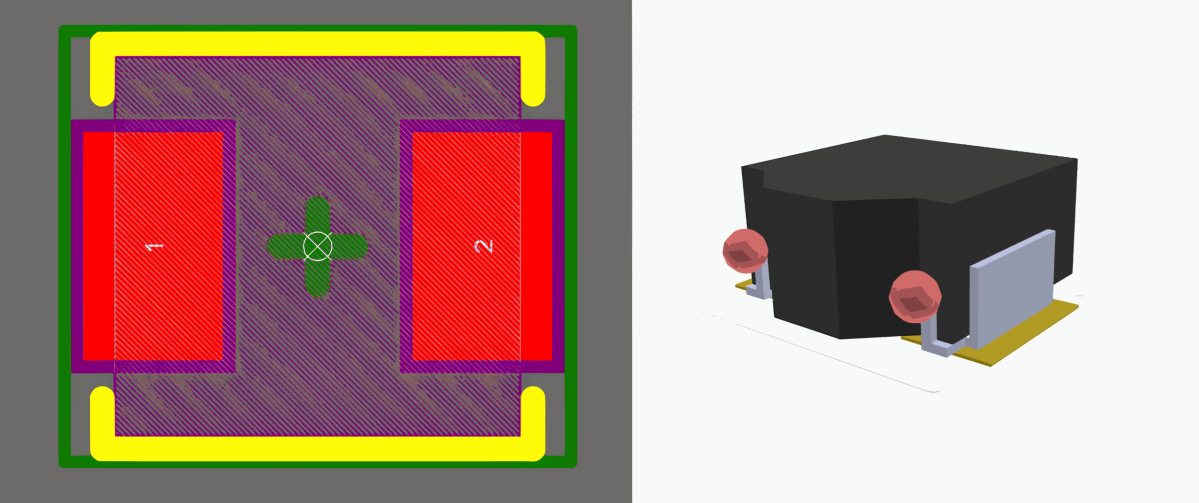


Рисунок 11. Посадочное место и трехмерная модель индуктора

#### Резисторы

Что касается резисторов, процесс их добавления аналогичен предыдущим случаям. Однако следует отметить необходимость включения резистора перемычки [12] (резистор чье сопротивление стремится к нулю), с помощью такого резистора будет осуществляться конфигурация платы.

### Прорезные отверстия

Для реализации ввода-вывода (I/O) на печатной плате были выбраны прорезных отверстий (castellated holes) [13]. Такой тип отверстий обеспечивает удобство пайки платы в качестве модуля на другую плату и гарантирует надежный электрический и механический контакт при соединении.

Важно отметить, что для прорезных отверстий в панели *Properties* должен быть установлен тип No BOM. Это необходимо для того, чтобы компонент не попал в спецификацию материалов, так как он не является закупаемым элементом.

## Проектирование принципиальной схемы

После завершения формирования библиотеки компонентов можно переходить к этапу проектирования принципиальной электрической схемы.

Для выполнения задачи, необходимо извлечь все компоненты из библиотеки и подключить их согласно типовой схеме, представленной в приложении A, на рисунке A.2.

### Детали проектирования

В данном разделе рассматриваются такие элементы, как перемычки, net-метки и порты, особенности обработки неиспользуемых выводов, процесс аннотирования схемы, а также методы её отладки.

#### Перемычки, net-метки и порты

Важно учитывать не только корректность подключения, но и правильное именование и расположение нет-меток (net labels) / портов (ports), чтобы обеспечить читабельность схемы.

На примере перемычки, изображенной на рисунке 12, показано как используются net-метки и порты. Здесь *PMIC\_DCBSET* это соединение, идущее от вывода AXP805 с одноименным названием. Для того чтобы не вести его через всю схему, ухудшая ее читабельность, достаточно просто пометить соединения одним именем.

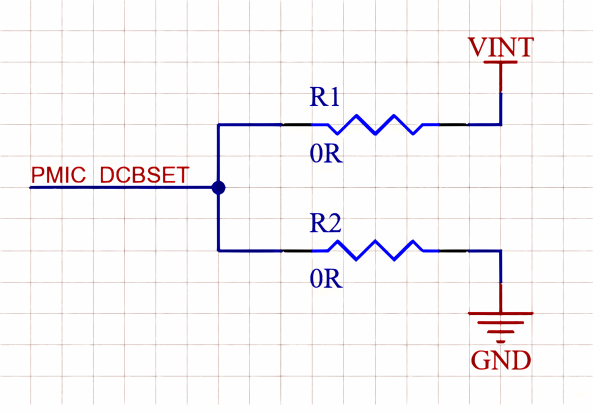


Рисунок 12. Пример перемычки

Также на рисунке 12 изображены использование порта *VINT*. Здесь *VINT* это порт на линии питания логики внутри AXP805, выведенный наружу через соответствующий пин компонента.

Аналогичная топология используется для остальных перемычек в проекте (*PMIC\_MODSET*, *PMIC\_PHSET*).

#### Не используемые выводы

Если определённые пины не используются в проекте, их следует пометить с инструментом *Generic No ERC*, как показано на рисунке 13.

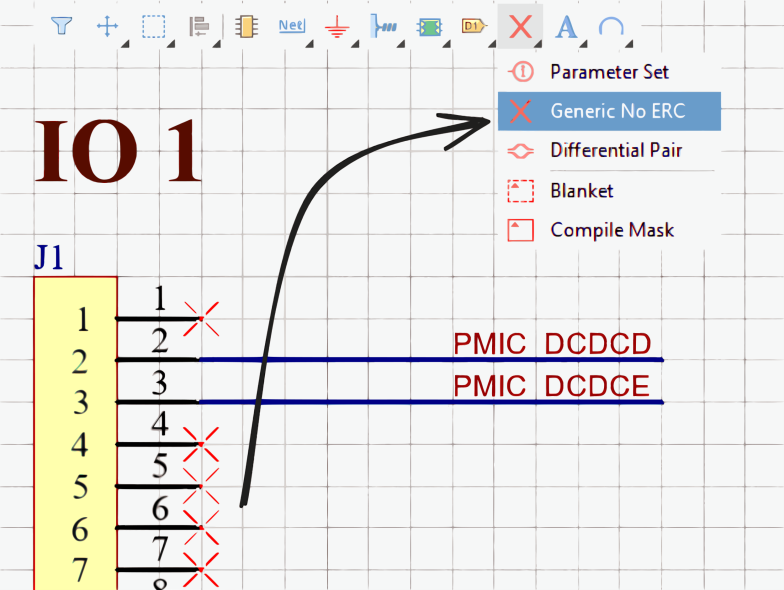


Рисунок 13. Использование *Generic No ERC* на примере портов ввода-вывода

#### Аннотация схемы

После завершения этапа подготовки схемы необходимо выполнить её аннотирование. Для этого следует воспользоваться пунктом меню: *Tools → Annotations → Annotate Schematic*. После чего обновить список изменений и применить его ко всем элементам схемы.

В результате выполнения каждому компоненту схемы будет присвоен дезигнатор, обеспечивающий однозначную идентификацию элементов в рамках проекта.

#### Отладка схемы

В случае необходимости проведения отладки схемы рекомендуется использовать встроенный в Altium Designer инструмент — Навигатор. Для его активации требуется открыть вкладку *Panels → Navigator*, после чего будет отображена панель, содержащая перечень всех компонентов и связанных с ними электрических цепей.

Для визуального анализа соединений необходимо выбрать одну из цепей и, вызвав контекстное меню правой кнопкой мыши, активировать опцию *Show Graph*. После этого Altium Designer отобразит граф электрических соединений в виде точек, что значительно упрощает процесс поиска и устранения ошибок в схеме. Пример такого графа для сети GND изображен на рисунке 14.

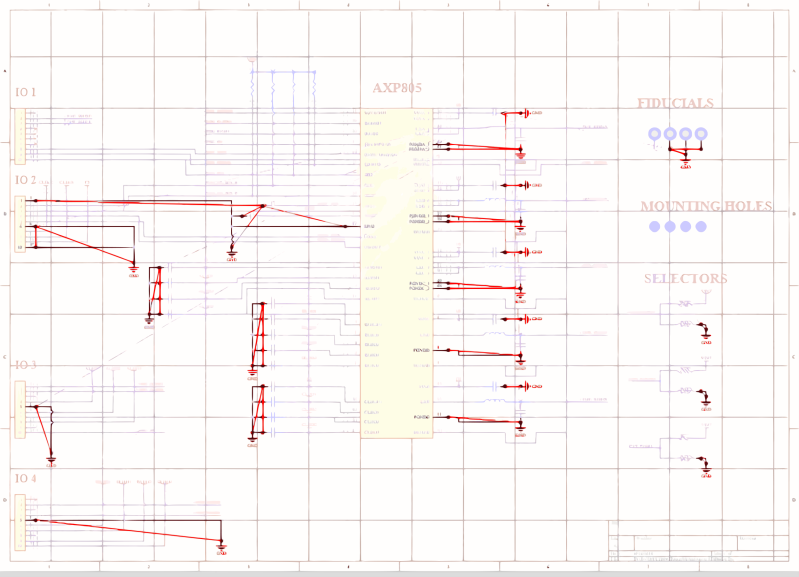


Рисунок 14. Схема в режиме навигации, отображен граф сети GND

### Настройки генерации PCB

Перед генерацией PCB (Printed Circuit Board) из принципиальной схемы важно отключить генерацию регионов (rooms).

Регионы — это логические блоки или области, которые группируют компоненты на схеме с целью упрощения их размещения и организации на печатной плате. Обычно их используют для структурирования и упорядочивания компонентов на основе их функциональных групп или других критериев. Однако в данном случае компонентов достаточно мало, а регион всего один, поэтому он не требуется.

Чтобы отключить генерацию регионов, необходимо перейти в меню *Options → Project Options...*, затем выбрать вкладку *Class Generation* и снять галочку с пунктов *Generate Rooms* и *Components Classes*. Это предотвратит автоматическую генерацию логических областей и классов компонентов при переходе от схемы к плате.

### Генерация PCB

Перед началом генерации PCB необходимо убедиться, что проект компилируется без ошибок. Конечный вариант принципиальной схемы представлен в приложении A, рисунок A.3. Для этого следует нажать правой кнопкой мыши на проект и выбрать пункт *Validate...*. Если после проверки не появятся сообщения об ошибках, это означает, что проект готов к дальнейшему экспорту.

Для генерации PCB из принципиальной схемы необходимо перейти в редактор платы (PCB Editor), выбрать пункт меню *Design → Import Changes from PowerManagement.PrjPcb* и принять все изменения.

После импорта изменений необходимо скорректировать положение компонентов на плате. Для этого следует выделить все объекты, используя комбинацию клавиш Ctrl + A, а затем выбрать пункт меню *Tools → Components Placement → Arrange Outside the Board*, что позволит автоматически разместить компоненты вне границ платы.

### Структура слоев

Ранее, в эпоху менее сложной электронной разработки, печатные платы, как правило, ограничивались двумя слоями. Однако с развитием технологий и увеличением функциональных требований к электронным устройствам произошёл значительный прогресс в области проектирования печатных плат. В настоящее время во многих потребительских устройствах количество слоёв может достигать восьми и более. В связи с этим в системе проектирования Altium Designer предусмотрен специализированный инструмент управления слоями — Layer Stack Manager. Доступ к нему осуществляется через редактор платы, в меню *Design → Layer Stack Manager*.

Существует множество профилей печатных плат, и выбор толщины платы, как правило, определяется спецификой отрасли и целевым назначением устройства. Так, в потребительской электронике, например в смартфонах, предпочтение отдается более тонким платам — порядка 1,2 мм — с целью снижения габаритов устройства. В то же время в изделиях, предназначенных для применения в оборонной промышленности, часто используются более толстые платы, обеспечивающие повышенную механическую прочность и надёжность. Для двухслойных печатных плат стандартом де-факто является толщина 1,6 мм [14] (рисунок 15).



Рисунок 15. Структура слоев печатной платы

### Установка формы платы

При задании формы печатной платы необходимо учитывать её функциональное назначение, а также предполагаемое количество размещаемых компонентов. Рациональное использование доступной площади способствует более плотной компоновке элементов, что, в свою очередь, позволяет сократить длину сигнальных трасс. Уменьшение длины прохождения сигнала положительно сказывается на электрических характеристиках и общей надёжности устройства.

Для определения габаритов платы следует перейти в режим *Board Planning Mode* в редакторе печатной платы, что можно сделать нажатием клавиши «1» на клавиатуре. Далее, в меню *Design* необходимо выбрать команду *Redefine Board Shape* для изменения формы платы.

После определения габаритов печатной платы необходимо перейти на слой Keep-Out Layer и воспроизвести контур платы на нем. Это обеспечивает корректное применение правил размещения элементов, предотвращая возможность их установки за пределами рабочей области. Кроме того, слой Keep-Out Layer используется производителями при изготовлении печатной платы в качестве ориентира для соблюдения границ конструкции.

## Расположение компонентов на плате

Перед началом трассировки рекомендуется предварительно выделить все дезигнаторы компонентов на плате используя инструмент *Panels → PCB Filter* и временно скрыть их отображение. Дополнительно целесообразно отключить визуализацию соединений с помощью *View → Connections → Hide Net.* Это обеспечит более удобную и не загромождённую рабочую область.

После подготовки рабочего пространства следует приступить к размещению наиболее габаритных компонентов [15], таких как разъёмы и микроконтроллеры, поскольку именно они в наибольшей степени определяют топологию будущей платы. После размещения указанных элементов рекомендуется зафиксировать их положения по осям X и Y с помощью панели *Properties*, чтобы исключить случайное смещение при последующей работе.

Наиболее рациональным способом размещения компонентов на печатной плате является ориентация на ранее разработанную принципиальную схему. Для повышения удобства работы и снижения необходимости постоянного переключения между редакторами схемы и платы, рекомендуется воспользоваться функцией вертикального разделения экрана (*Vertical Split*), которая активируется через контекстное меню вкладки схемы нажатием правой кнопки мыши.

После этого следует перейти в меню *Tools* и включить режим *Cross Select Mode*, обеспечивающий перекрёстное выделение компонентов между схемой и платой, что значительно упрощает процесс их размещения.

Отдельное внимание при размещении компонентов на печатной плате следует уделить размещению развязывающих конденсаторов (decoupling capacitors) [16]. Их необходимо располагать как можно ближе к питаемым выводам микросхем, что позволяет минимизировать высокочастотные помехи и обеспечить стабильную работу устройства (рисунок 16). Несоблюдение данного требования может привести к возникновению шумов на линиях питания и ухудшению электромагнитной совместимости всей системы.



Рисунок 16. Компоновка (верхний слой)

Как правило, корректное размещение компонентов с первого раза оказывается затруднительным, и в процессе последующей трассировки платы их положение может неоднократно изменяться. Как видно на рисунке 16, в двухслойных печатных платах общепринятой практикой является размещение всех компонентов на верхнем слое (Top Layer) [17]. Такой подход упрощает процесс сборки и облегчает отладку устройства при использовании ручного монтажа.

## Топология печатной платы

Топология печатной платы представляет собой схему размещения и соединения электронных компонентов на плате. От правильного проектирования топологии зависит не только работоспособность устройства, но и его электромагнитная совместимость, тепловой режим и возможность массового производства [18]. В этом разделе рассматриваются основные принципы разработки топологии, включая размещение элементов, трассировку проводников, а также особенности многослойных плат.

### Настройка правил проектирования

Прежде чем приступить к разработке топологии печатной платы, необходимо задать ряд правил проектирования, которые будут применяться в среде Altium Designer на протяжении всего процесса. Установленные правила (таблица 1) служат основой для автоматизированной проверки и трассировки, обеспечивая соблюдение технологических и электрических требований.

Таблица 1

Настройки проверки правил.

|  |  |  |  |
| --- | --- | --- | --- |
| Директория | Раздел | Правило | Значение |
| Report Options | DRC Report Optoins | Stop When [ \* ] violations found | 1 000 000 |
| Rules To Check | Electrical | Un-Routed Net | Disable |
| Rules To Check | Routing | \* | Enable |
| Rules To Check | SMT | \* | Enable |
| Rules To Check | Testpoint | \* | Disable |
| Rules To Check | Manufacturing | \* | Enable |
| Rules To Check | High Speed | \* | Enable |
| Rules To Check | Placement | \* | Enable |
| Rules To Check | Signal Integrity | \* | Disable |

Для настройки инструмента проверки проектных правил (Design Rule Checker) необходимо перейти в меню *Tools → Design Rule Check…*. Все параметры, рекомендованные для данного проекта приведены в таблице 1.

Примечание — символ астериска означает, что значение применяется ко всем правилам, находящимся в разделе.

При подготовке печатной платы к производству необходимо заранее уточнить у выбранного производителя три ключевых технологических параметра [19]:

1. Минимальное допустимое расстояние между проводниками и отверстиями.
2. Минимальный диаметр переходного (via) отверстия.
3. Минимально допустимая ширина проводника.

Эти параметры также следует учитывать при установке проектных правил, чтобы обеспечить соответствие конструкции возможностям производства.

После завершения настройки инструмента необходимо определить сами правила во вкладке *Design → Rules…*. Все используемые в данном проекте правила систематизированы в таблице 2, а их описания приведены в таблице 3.

Таблица 2

Настройка правил.

|  |  |  |
| --- | --- | --- |
| Категория | Правило | Значение (мм) |
| Clearance | ExceptionCastellatedHolesKeepOut | 0 |
| Clearance | Clearance | Min: 0.2 |
| Width | ExceptionWidthPowerSupply | Min: 0.3 / Pref: 0.5 / Max: 1.0 |
| Width | Width | Min: 0.3 / Pref: 0.3 / Max: 0.5 |
| Routing Via Style | Routing Via Style / RoutingVias / Via Diameter | 0.6 |
| Routing Via Style | Routing Via Style / RoutingVias / Via Hole Size | 0.3 |
| Solder Mask Expansion | SolderMaskExpansion | 0.1 |
| Hole Size | HoleSize | Min: 0.2 / Max: 10 |
| Hole To Hole Clearance | HoleToHoleClearance | 0.3 |
| Minimum Solder Mask Sliver | MinimumSolderMaskSliver | 0.1 |
| Silk To Solder Mask Clearance | SilkToSolderMaskClearance | 0.1 |
| Silk To Silk Clearance | SilkToSilkClearance | 0.1 |
| Component Clearance | ComponentClearance | Horizontal: 0.2 / Vertical: 0.2 |
| Component Clearance | ExceptionJumpResistors | 0 |

Таблица 3

Описание правил.

|  |  |
| --- | --- |
| Правило | Описание |
| ExceptionCastellatedHolesKeepOut | Исключение для расстояния между прорезными отверстиями и краем платы |
| Clearance | Общее правило для расстояния между цепями |
| ExceptionWidthPowerSupply | Исключение для ширины цепей питания |
| Width | Общее правило для ширины проводников |
| Routing Via Style / RoutingVias / Via Diameter | Настройки переходных отверстий |
| Routing Via Style / RoutingVias / Via Hole Size | Настройки переходных отверстий |
| SolderMaskExpansion | Расширение маски паяльной пасты |
| HoleSize | Допустимые размеры отверстий |
| HoleToHoleClearance | Минимальное расстояние между отверстиями |

Продолжение таблицы 3

|  |  |
| --- | --- |
| MinimumSolderMaskSliver | Минимальная ширина между областями маски |
| SilkToSolderMaskClearance | Зазор между шелкографией и паяльной маской |
| SilkToSilkClearance | Зазор между элементами шелкографии |
| ComponentClearance | Минимальное расстояние между компонентами |
| ExceptionJumpResistors | Исключения для расстояний между резисторами-перемычками |

### Трассировка печатной платы

Трассировка печатной платы является задачей класса NP-полных [20], что исключает возможность применения универсального алгоритма, способного эффективно решать её во всех случаях. В результате проектирование топологии требует творческого подхода и инженерной интуиции. Рекомендуется начинать с разводки компонентов, критичных к размещению вблизи контроллера — таких как развязывающие конденсаторы и кварцевые генераторы. После этого следует перейти к прокладке наиболее протяжённых соединений, обеспечивающих ключевые электрические связи. В случае ограниченного пространства допускается использование нижнего слоя (Bottom Layer), однако его предпочтительно оставлять свободным для последующей заливки сплошной землёй, что способствует улучшению экранирования и снижению уровня электромагнитных помех.

В результате проделанной работы топология платы должна приближаться по структуре и организации к изображению, представленному на рисунке 17.

Как можно видеть на рисунке 17, на нижнем слое печатной платы питающая дорожка выполнена с увеличенной шириной. Такое решение обусловлено необходимостью снижения сопротивления проводника и обеспечения надёжной подачи питания к нагрузке, особенно при протекании значительных токов [21].

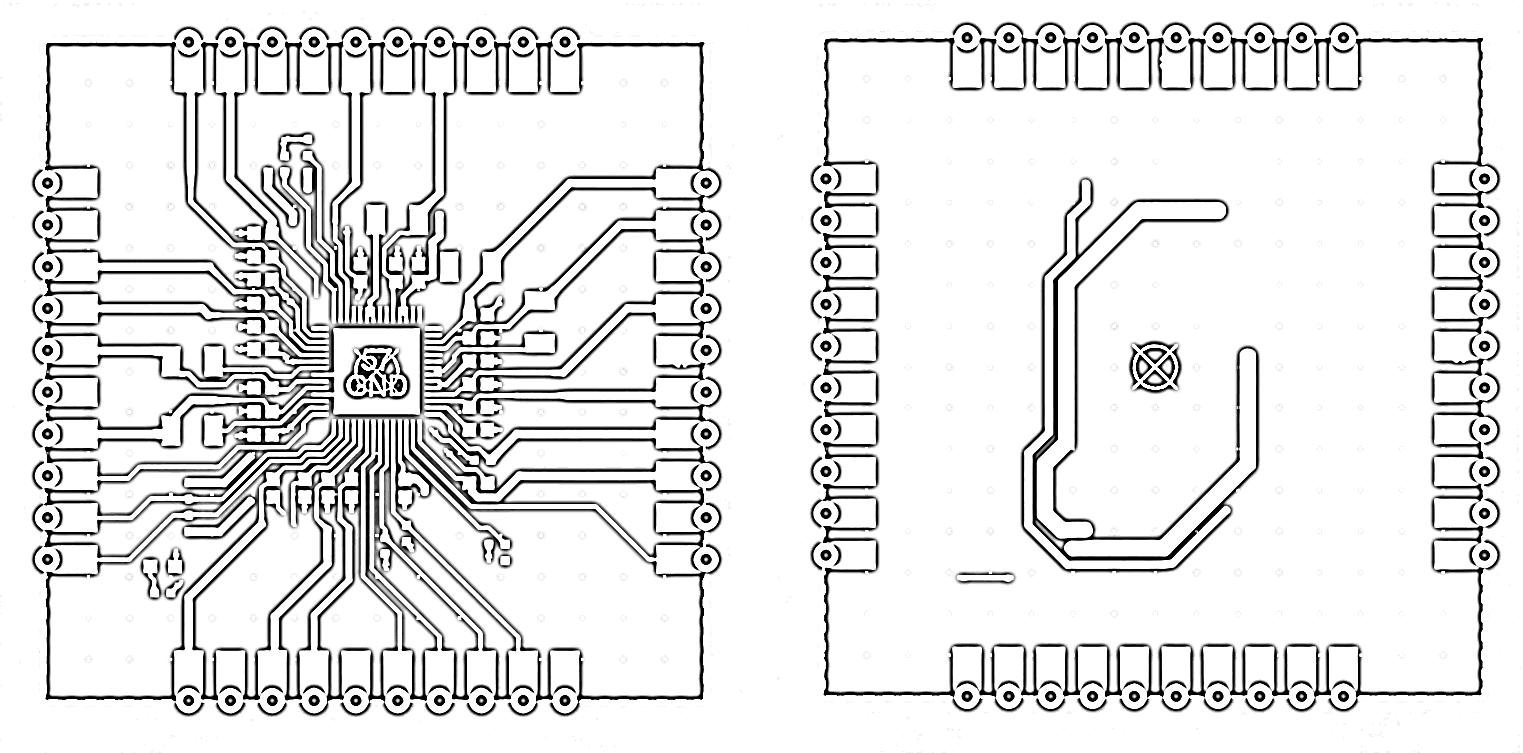


Рисунок 17. Топология печатной платы, верхний и нижний слои

### Земляные полигоны

Заключительным этапом трассировки печатной платы является заливка слоёв земляными полигонами (рисунок 18) и размещение переходных отверстий (vias) для их соединения. Такая мера значительно повышает помехоустойчивость устройства за счёт формирования сплошного экранирующего слоя и обеспечения надёжного заземления всех участков схемы [22].

Следует отметить, что полигоны в данном проекте используются не только для земли, но и для контактов питания в нижней части платы, по тем же причинам, из-за которых питающая дорожка выполнена с увеличенной шириной.

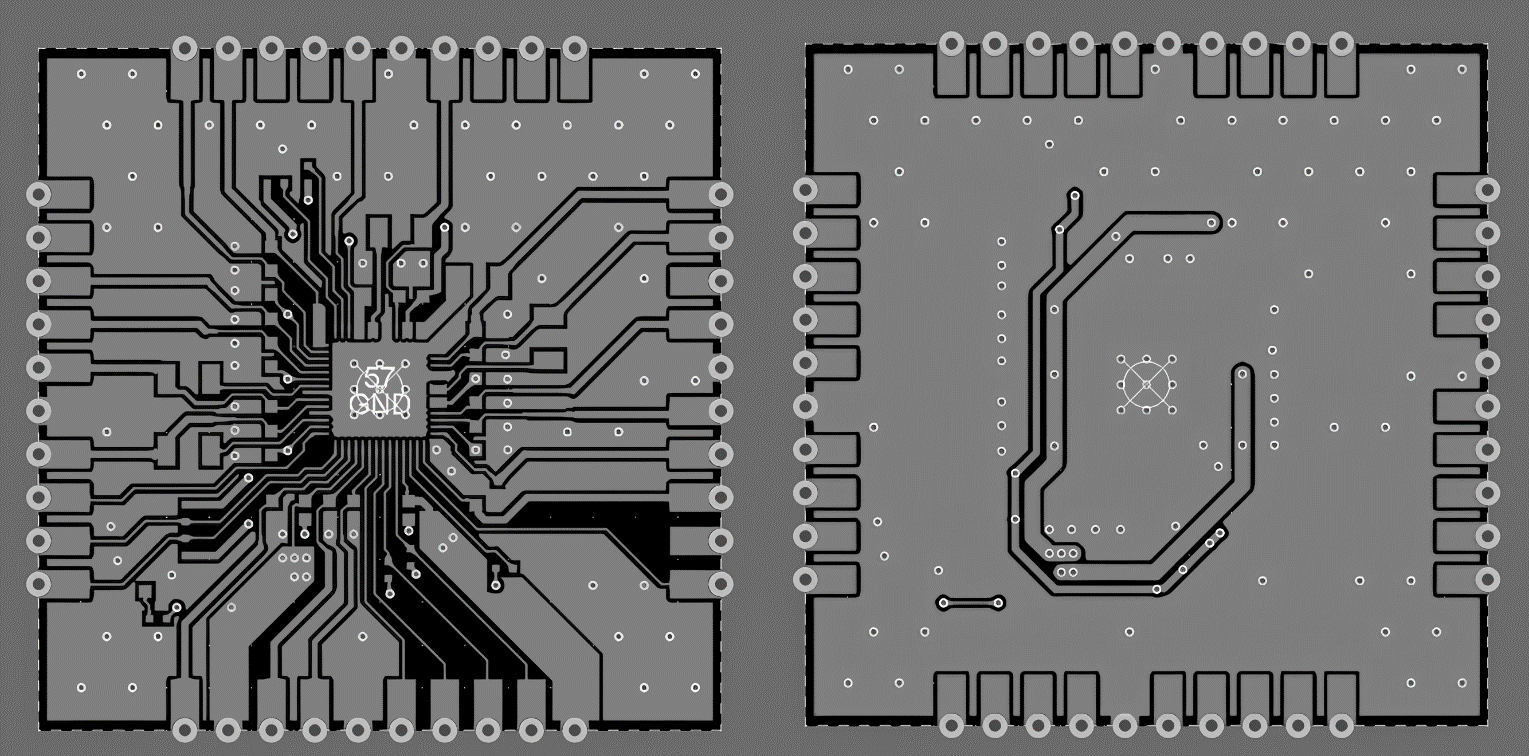


Рисунок 18. Верхний и нижний слои платы, залитые земляными полигонами

# Сборка программного комплекса

Помимо проектирования печатных плат, современная разработка встраиваемых систем также включает написание значительного объема прикладного программного обеспечения. Для достаточно крупных встраиваемых систем, таких как использующаяся в исследовании платформа Orange Pi 3 LTS, целесообразно применять операционные системы, например, GNU / Linux, которая зарекомендовала себя как эффективное решение для встраиваемого оборудования.

В данной главе рассматриваются теоретические основы работы встраиваемых операционных систем, а также представлен один из подходов к их сборке.

## Linux для встраиваемых систем

Операционная система Linux широко применяется в разработке встраиваемых решений, где используется ядро и определённый набор компонентов с открытым исходным кодом. В данной главе рассматриваются ключевые элементы такой системы: загрузчик операционной системы U-Boot, ядро Linux, минимальный набор утилит BusyBox, а также одна из реализаций стандартной библиотеки языка программирования C — glibc, необходимая для функционирования вышеуказанных компонентов.

Несмотря на длительную историю развития, ядро Linux по состоянию на 2025 год сохраняет относительную компактность. Это объясняется тем, что значительная часть его объёма приходится на многочисленные драйверы оборудования. В контексте встраиваемых систем возможна значительная оптимизация путём исключения неиспользуемых модулей и драйверов, что становится возможным при заранее определённой функциональности устройства.

Процесс построения программной среды начинается с подготовки сборочной среды (environment), в которой происходит компиляция всех необходимых компонентов системы. На этапе разработки осуществляется сборка загрузчика и ядра, для чего используется набор инструментов (toolchain), создаваемый на компьютере разработчика. В работе применяется предварительно подготовленный набор инструментов, что упрощает процесс сборки.

По завершении компиляции всех программных компонентов формируется образ файловой системы, предназначенный для последующей записи на физический носитель. В качестве такого носителя, например, для микрокомпьютера Orange Pi 3 LTS, может использоваться карта памяти SD, с которой осуществляется загрузка операционной системы.

### Процесс загрузки операционной системы

Большинство современных встраиваемых систем содержат так называемый ROM-код [23], представляющий собой начальный этап последовательности загрузки. Этот код прошит непосредственно в микроконтроллер или процессор производителем и не подлежит модификации со стороны разработчика. Подробное описание поведения ROM-кода обычно приводится в технической документации (datasheet) соответствующего микропроцессора.

Основная функция данного кода заключается в поиске загрузчика (bootloader) на доступных внешних носителях, таких как SD-карты, eMMC или USB-устройства, с последующей загрузкой его в область внутренней памяти SRAM. Это накладывает ограничение на размер начального загрузчика, так как он не должен превышать доступный объём встроенной памяти.

В связи с этим загрузчик, как правило, реализуется в виде двухэтапной схемы. Первая стадия загрузки (англ. *First Stage Bootloader*) загружается в SRAM и выполняет инициализацию внешней оперативной памяти (DRAM). После этого она передаёт управление второй стадии загрузки (англ. *Second Stage Bootloader*), которая размещается уже в более вместительной DRAM и осуществляет дальнейшую подготовку системы к запуску операционной системы.

Последовательность загрузки и распределение данных в адресном пространстве памяти приведены на Рисунке 19.

A diagram of a booting sequence

Description automatically generated

Рисунок 19. Расположения данных в памяти и этапы загрузки системы [24]

Загрузчик представляет собой программный компонент, предназначенный для первичной инициализации аппаратного обеспечения вычислительной системы, а также для загрузки в оперативную память и запуска исполняемого кода, которым, как правило, является ядро операционной системы.

Помимо базовых функций, многие современные загрузчики оснащаются интерфейсом командной строки (англ. *Command Line Interface*, CLI) или графическим меню, позволяющим пользователю выбирать между различными вариантами загрузки операционных систем или конфигураций системы [24].

Следует отметить, что загрузчик является первой программой в процессе старта системы, которую разработчик имеет возможность создать и модифицировать самостоятельно. В связи с этим в рамках настоящей работы данный компонент будет собран вручную, скомпилирован и записан на загрузочный носитель [25].

### Задача операционной системы

После завершения процесса загрузки операционной системы, загрузчик передаёт ей полный контроль над аппаратными ресурсами. С этого момента управление всеми ключевыми компонентами вычислительной системы — центральным процессором, оперативной памятью, а также устройствами ввода-вывода — осуществляется непосредственно операционной системой [26].

В контексте работы с одноплатным микрокомпьютером Orange Pi 3 LTS следует отметить, что официально поддерживаемой операционной системой для данного устройства является Ubuntu Linux. Этот дистрибутив представляет собой систему с открытым исходным кодом, основное достоинство которой заключается в простоте установки и использовании, что делает её особенно привлекательной для широкой аудитории.

Тем не менее, универсальность Ubuntu Linux, ориентированная на максимальное покрытие пользовательских сценариев, приводит к значительному увеличению объёма дистрибутива и снижению эффективности его применения в сфере встраиваемых систем.

Среди прочих функций, реализуемых операционной системой, следует выделить обеспечение унифицированного, аппаратно-независимого программного интерфейса (API) для запускаемых приложений. Кроме того, операционная система обязана организовать многозадачную среду, позволяющую нескольким приложениям функционировать одновременно, не нарушая стабильность работы друг друга и всей системы в целом [27].

## Подготовка рабочей среды и компиляция системы

В индустрии давно утвердился стандарт, согласно которому для разработки, сборки, отладки и тестирования встраиваемых систем используется операционная система Linux. Этот стандарт стал де-факто, поскольку сборка компонентов по многим причинам становится практически невозможной на системах типа DOS и Windows.

Для изоляции рабочей среды часто применяется система контейнеризации Docker. Она использует конфигурационный файл, называемый Dockerfile (приложение B, листинг B.1), в котором прописаны команды, необходимые для подготовки системы к работе.

Данный Dockerfile описывает процесс создания контейнера для разработки и сборки встраиваемых систем на платформе ARM с использованием различных компонентов, включая ядро Linux, загрузчик U-Boot, библиотеку glibc, систему BusyBox и другие необходимые инструменты. Процесс состоит из нескольких этапов, каждый из которых выполняет конкретные задачи для подготовки окружения, сборки и создания образа для загрузки системы.

### Этап запуска

На этом этапе происходит загрузка исходных архивов с необходимыми компонентами, такими как:

* + Ядро Linux для платформы Orange Pi 3 LTS (через ссылку на репозиторий GitHub) [28].
  + Загрузчик U-Boot, предназначенный для платформы ARM.
  + ARM Trusted Firmware, который управляет низкоуровневыми функциями безопасности [29].
  + glibc — стандартная библиотека языка программирования Си.
  + BusyBox — набор утилит для минимизации размера файловой системы.
  + Инструменты компилятора для архитектуры ARM.

Все архивы скачиваются с использованием хеш-сумм для проверки целостности данных. Далее, архивы распаковываются, а ненужные файлы удаляются. На этом этапе также происходит модификация Makefile для настройки сборки U-Boot [30].

### Этап компиляции

На этом этапе происходит настройка рабочей среды для сборки встраиваемой системы. Основное внимание уделяется установке необходимых зависимостей и инструментов, таких как компилятор GCC для ARM, а также различные библиотеки и утилиты для сборки. Затем выполняется сборка следующих компонентов:

* + ARM Trusted Firmware: производится компиляция firmware для платформы ARM.
  + Загрузчик U-Boot: конфигурация и компиляция загрузчика для конкретной платформы.
  + Ядро Linux: настройка конфигурации и сборка ядра Linux для архитектуры ARM64.
  + glibc: настройка и сборка стандартной библиотеки для целевой архитектуры.
  + BusyBox: сборка минималистичной операционной системы для работы с основными утилитами.

Далее происходит установка собранных компонентов в файловую систему, подготовку модулей ядра и заголовков для использования в будущем.

### Этап сборки

На данном этапе происходит сборка финального образа для встраиваемой системы. В частности:

* + Копируются необходимые файлы из предыдущих этапов, включая компоненты файловой системы и загрузчика.
  + Используется утилита cpio для упаковки файловой системы в initramfs, которая будет использована как начальная загрузочная файловая система.
  + Генерируется образ загрузочного скрипта boot.scr, который будет использован для настройки загрузки.
  + Создаются образы дисков в формате ISO, в которые записываются данные: сначала создается загрузочный сектор с загрузчиком, затем формируется файловая система с разделами.
  + Используется утилита fdisk для создания разделов на виртуальном диске.

### Финальный этап (ISO-образ)

На последнем этапе создается итоговый образ, который будет использоваться для прошивки устройства. Все необходимые файлы и компоненты записываются в ISO-образ, который затем может быть использован для загрузки и установки встраиваемой системы на платформе ARM.

Таким образом, данный Dockerfile описывает полный процесс создания среды для разработки встраиваемых систем, начиная от загрузки исходных архивов с необходимыми компонентами, их сборки, до создания образа для загрузки и работы системы.

# Заключение

В ходе выполнения дипломной работы была поставлена цель — разработать встроенную систему на базе одноплатного микрокомпьютера Orange Pi 3 LTS, включающую два основных компонента: аппаратную часть (подсистема питания) и программную (минималистичная операционная система на базе Linux). Данная цель реализуется через решение ряда взаимосвязанных задач, охватывающих как схемотехническое проектирование, так и работу с низкоуровневым программным обеспечением.

На первом этапе была решена задача анализа требований к системе электропитания Orange Pi 3 LTS. Проведён разбор документации, выявлены номиналы питающих напряжений и необходимые токи нагрузки для различных участков платы. Установлено, что платформа требует нескольких независимых и стабилизированных напряжений питания, среди которых выделяются линии для процессора, оперативной памяти, логических уровней и периферийных интерфейсов. В качестве основы подсистемы питания была выбрана микросхема AXP805, представляющая собой специализированную интегральную схему управления питанием (PMIC), обеспечивающую пять каналов регулирования, интерфейс конфигурации и встроенные защитные механизмы.

Следующей задачей было проектирование аппаратной части в ECAD-среде Altium Designer. На этом этапе выполнены все стандартные работы, включая:

Создание библиотеки компонентов: УГО, посадочные места, 3D-модели.

Построение принципиальной схемы на основе типовой конфигурации из технической документации.

Размещение компонентов на плате с соблюдением технологических рекомендаций.

Разработка топологии печатной платы, включая трассировку всех цепей и заливку земляных полигонов.

Настройка системы правил проектирования (Design Rules) и выполнение верификации схемы и платы с помощью автоматических средств проверки (DRC, ERC).

Печатная плата выполнена с учётом всех стандартов проектирования, в том числе требований к ширине проводников, допустимым зазорам, размерам контактных площадок и технологическим допускам. Использовались положения стандартов IPC-7351 и ГОСТ Р 55693-2013. Все параметры проектирования были предварительно уточнены с учётом типовых требований производителей печатных плат.

На программном этапе была решена задача подготовки минимального дистрибутива Linux, адаптированного для платформы Orange Pi 3 LTS. Работа велась в несколько этапов:

Сбор исходных кодов всех компонентов системы: загрузчика U-Boot, ядра Linux, библиотеки glibc и утилит BusyBox.

Настройка и запуск среды сборки на основе Docker-контейнера, что позволило изолировать окружение, избежать конфликтов зависимостей и обеспечить воспроизводимость процесса.

Конфигурация и компиляция всех компонентов с использованием кросс-компилятора под архитектуру ARM64.

Сборка финального загрузочного образа, пригодного для записи на SD-карту и загрузки на целевом оборудовании.

В результате получен собственный образ операционной системы, включающий только необходимые компоненты и не содержащий избыточного функционала. Это соответствует принципам построения минималистичных встраиваемых решений и позволяет снизить требования к ресурсам системы, ускорить загрузку и упростить обслуживание.

Файл Dockerfile, используемый в процессе сборки, содержит последовательность команд и конфигурационных шагов, что позволяет воспроизводить весь процесс в будущем без дополнительных действий. Кроме того, все шаги сборки и настройки задокументированы в составе дипломной работы, что обеспечивает прозрачность и проверяемость результатов.

Финальным этапом стала интеграция аппаратной и программной части: протестирована совместимость схемы питания с загрузчиком и операционной системой. В ходе проверок особое внимание уделялось стабильности подачи питающих напряжений в процессе старта системы, работе регуляторов под нагрузкой и корректной инициализации памяти и периферии со стороны ядра Linux. Система успешно проходит этап загрузки, инициализирует основные подсистемы и предоставляет доступ к командной строке.

Таким образом, в рамках дипломного проекта реализован полный цикл создания встроенной системы, начиная с анализа требований и заканчивая готовым работоспособным решением. Все поставленные задачи выполнены:

Подсистема питания спроектирована, оформлена и проверена на соответствие требованиям.

Операционная система собрана вручную из исходных компонентов и адаптирована под конкретное оборудование.

Создан загрузочный образ и подтверждена его работоспособность на целевой платформе.

Результаты работы соответствуют цели дипломной работы. Проект охватывает ключевые аспекты разработки встроенных систем: схемотехнику, проектирование печатных плат, выбор элементной базы, использование ECAD-инструментов, сборку Linux-дистрибутива и работу с загрузчиком. Отдельное внимание уделено автоматизации сборки, документированию этапов и соблюдению требований к надёжности и воспроизводимости решений.

Полученный результат может служить основой для расширения проекта: добавления пользовательского интерфейса, интеграции с внешними сенсорами, реализации сетевых функций или построения устройства на базе разработанной системы. Все наработки дипломной работы сохраняют применимость для последующих технических разработок.

# Список использованных источников

1. Which PCB Design Software is The Best? The Top 4 Are ... // YouTube URL: https://youtu.be/8IZnj4Z9CHU (дата обращения: 06.04.2025).
2. Как правильно проектировать электронику // Хабр URL: https://habr.com/ru/companies/lanit/articles/733092/ (дата обращения: 06.04.2025).
3. AXP805 X-Powers - Battery Management // Octopart URL: https://octopart.com/axp805-x-powers-109675085 (дата обращения: 06.04.2025).
4. Reference designator // Wikipedia URL: https://en.wikipedia.org/wiki/Reference\_designator (дата обращения: 06.04.2025).
5. AXP805 Datasheet // SUNXI URL: https://linux-sunxi.org/images/b/bc/AXP805\_Datasheet\_V1.0\_en.pdf (дата обращения: 06.04.2025).
6. How to Draw Schematic & Tips to Improve Schematic // YouTube URL: https://youtu.be/lF0UgVQnZ5M (дата обращения: 06.04.2025).
7. QFN Layout Guidelines // TI.com URL: https://www.ti.com/lit/an/sloa122/sloa122.pdf (дата обращения: 06.04.2025).
8. IPC-7351 Table of Contents // IPC International, Inc. URL: https://www.ipc.org/TOC/IPC-7351.pdf (дата обращения: 06.04.2025).
9. Horowitz P., Hill W. The Art of Electronics. - 3-е изд. - New York: Cambridge University Press, 2015. - 1192 с.
10. Footprint dimensions in mm // IOElectro URL: https://ioelectro.ir/blog/upload/smt/smt\_notes.pdf (дата обращения: 08.04.2025).
11. EST016 Ground Loops Handout // CMA4CH URL: http://www.cma4ch.org/chemo/ftp/EST016\_Ground\_Loops\_handout.pdf (дата обращения: 15.04.2025).
12. Vasjanov, A., & Barzdenas, V. (2023). Surface-Mount Zero-Ohm Jumper Resistor Characterization in High-Speed Controlled Impedance Transmission Lines. Sensors, 23, 10223. https://doi.org/10.3390/s232010223
13. What are Castellated Holes on a PCB? // PCBasic URL: https://www.pcbasic.com/blog/castellated\_pcb.html (дата обращения: 10.05.2025).
14. 1.2mm vs 1.6mm PCB: What PCB Thickness to Choose? // Best Technology URL: https://www.bestpcbs.com/blog/2024/09/1-2mm-vs-1-6mm-pcb-what-pcb-thickness-to-choose/ (дата обращения: 10.05.2025).
15. Purwanto, R. E., Sulistyono, & Mandayatma, E. (2021). Optimization of printed circuit board components placement. IOP Conference Series: Materials Science and Engineering, 1073, 012022. https://doi.org/10.1088/1757-899X/1073/1/012022
16. Y. Zhu, L. Lin, and Y. Liu, "Optimization of Decoupling Capacitors Placement in Power Distribution Network," *Proceedings of the 2017 IEEE International Conference on Applied Superconductivity and Electromagnetic Devices (ASEMD)*, pp. 128-132, 2017.
17. Simple Strategies for Double Sided 2-Layer PCBs // Cadence URL: https://resources.pcb.cadence.com/blog/simple-strategies-for-double-sided-2-layer-pcbs (дата обращения: 10.05.2025).
18. Mitra, S., & Kumar, A. (2017). A Study of Routing Algorithms for PCB Design. Proceedings of the International Conference on Circuits, Communication, and Computing (ICCCC), 3(1), 1-5.
19. ГОСТ Р 55693-2013. Национальный стандарт Российской Федерации. Платы печатные жесткие. Технические требования // МЕГАНОРМ URL: https://meganorm.ru/mega\_doc/norm/gost-r\_gosudarstvennyj-standart/16/gost\_r\_55693-2013\_natsionalnyy\_standart\_rossiyskoy.html (дата обращения: 10.05.2025).
20. Cook, S. A. (1971). The complexity of theorem-proving procedures. *Proceedings of the Third Annual ACM Symposium on Theory of Computing (STOC)*, 151-158. https://doi.org/10.1145/800157.805047
21. Правила проектирования PCB // Altium URL: https://resources.altium.com/ru/p/pcb-layout-guidelines (дата обращения: 10.05.2025).
22. How to Achieve Proper Grounding - Rick Hartley - Expert Live Training // YouTube URL: https://www.youtube.com/live/ySuUZEjARPY (дата обращения: 10.05.2025).
23. Embedded Linux system development training // bootlin URL: https://bootlin.com/doc/training/embedded-linux/embedded-linux-slides.pdf (дата обращения: 10.05.2025).
24. Getting started with U-Boot // Mike's homepage URL: https://krinkinmu.github.io/2023/08/12/getting-started-with-u-boot.html (дата обращения: 10.05.2025).
25. Сборка прошивки из исходников для Orange PI i96(Orange PI 2g-iot) // Habr URL: https://habr.com/ru/articles/763996/ (дата обращения: 10.05.2025).
26. Minimal requirements to compile the Kernel // The Linux Kernel Archives URL: https://www.kernel.org/doc/html/next/process/changes.html (дата обращения: 10.05.2025).
27. Arm GNU Toolchain Downloads 9.2-2019.12 // Arm Developer URL: https://developer.arm.com/downloads/-/gnu-a/9-2-2019-12 (дата обращения: 10.05.2025).
28. linux-orangepi // GtitHub URL: https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-5.10-media (дата обращения: 10.05.2025).
29. ARM Trusted Firmware // GitHub URL: https://github.com/ARM-software/arm-trusted-firmware/tree/v2.2 (дата обращения: 10.05.2025).
30. Filesystem Hierarchy Standard // Linux Foundation URL: https://refspecs.linuxfoundation.org/FHS\_3.0/fhs-3.0.html (дата обращения: 10.05.2025).

# Приложение A

Таблица A.1

Компоненты, используемые в проекте.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Дезигнатор | Серийный номер производителя | Описание с сайта поставщика | Метка | Корпус |
| C1 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C2 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C3 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C4 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C5 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C6 | CC0402KRX5R5BB105 | Multilayer Ceramic Capacitor, 1 uF, 100 V, ï¿½ 10%, X7R, 0402 [1005 Metric] | 1u / 6.3V | 0402 |
| C7 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C8 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C9 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C10 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C11 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C12 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C13 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C14 | C1608X5R1A475K080AE | C Series 0603 4.7 uF 10 V ±10 %Tolerance X5R SMT Multilayer Ceramic Capacitor | 4.7u / 10V | — |
| C15 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C16 | C1608X5R1A475K080AE | C Series 0603 4.7 uF 10 V ±10 %Tolerance X5R SMT Multilayer Ceramic Capacitor | 4.7u / 10V | — |
| C17 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C18 | C1608X5R1A475K080AE | C Series 0603 4.7 uF 10 V ±10 %Tolerance X5R SMT Multilayer Ceramic Capacitor | 4.7u / 10V | — |
| C19 | C1608X5R1A475K080AE | C Series 0603 4.7 uF 10 V ±10 %Tolerance X5R SMT Multilayer Ceramic Capacitor | 4.7u / 10V | — |
| C20 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C21 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C22 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C23 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C24 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| C25 | C1608X5R1A106K080AC | C Series 0603 10 uF 10 V ±10 % Tolerance X5R SMT Multilayer Ceramic Capacitor | 10u / 10V | 0603 |
| D1 | KG EELP41.22-PHRH-35-A8J8-20-R18 | Led Green Diffused 0603 SMD | Green | — |
| L1 | SPM3015T-1R5M-LR | Inductor Power Shielded Wirewound 1.5uH 20% 100KHz Metal 3.3A 0.0624Ohm DCR T/R | 1.5u / 3A | Nonstandard |
| L2 | SPM3015T-1R5M-LR | Inductor Power Shielded Wirewound 1.5uH 20% 100KHz Metal 3.3A 0.0624Ohm DCR T/R | 1.5u / 3A | Nonstandard |
| L4 | SPM3015T-1R5M-LR | Inductor Power Shielded Wirewound 1.5uH 20% 100KHz Metal 3.3A 0.0624Ohm DCR T/R | 1.5u / 3A | Nonstandard |

Продолжение таблицы A.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| L3 | SPM3015T-1R5M-LR | Inductor Power Shielded Wirewound 1.5uH 20% 100KHz Metal 3.3A 0.0624Ohm DCR T/R | 1.5u / 3A | Nonstandard |
| L5 | SPM3015T-1R5M-LR | Inductor Power Shielded Wirewound 1.5uH 20% 100KHz Metal 3.3A 0.0624Ohm DCR T/R | 1.5u / 3A | Nonstandard |
| R1 | RC0402BR-072KL | General Purpose Chip Resistor Thick Film 0402 2kOhm 0.1% Paper T/R | 2k | — |
| R2 | RC0402BR-072KL | General Purpose Chip Resistor Thick Film 0402 2kOhm 0.1% Paper T/R | 2k | — |
| R3 | RC0402BR-072KL | General Purpose Chip Resistor Thick Film 0402 2kOhm 0.1% Paper T/R | 2k | — |
| R4 | RC0402BR-072KL | General Purpose Chip Resistor Thick Film 0402 2kOhm 0.1% Paper T/R | 2k | — |
| R5 | PA0402-R-070RL | Zero Ohm Resistor, Jumper, 0402 [1005 Metric], Metal Film, 63 mW, 11 A, Surface Mount | 0R | — |
| R6 | PA0402-R-070RL | Zero Ohm Resistor, Jumper, 0402 [1005 Metric], Metal Film, 63 mW, 11 A, Surface Mount | 0R | — |
| R7 | PA0402-R-070RL | Zero Ohm Resistor, Jumper, 0402 [1005 Metric], Metal Film, 63 mW, 11 A, Surface Mount | 0R | — |
| R8 | PA0402-R-070RL | Zero Ohm Resistor, Jumper, 0402 [1005 Metric], Metal Film, 63 mW, 11 A, Surface Mount | 0R | — |
| R9 | PA0402-R-070RL | Zero Ohm Resistor, Jumper, 0402 [1005 Metric], Metal Film, 63 mW, 11 A, Surface Mount | 0R | — |
| R10 | PA0402-R-070RL | Zero Ohm Resistor, Jumper, 0402 [1005 Metric], Metal Film, 63 mW, 11 A, Surface Mount | 0R | — |
| R11 | RC0402FR-13560RL | SMD Chip Resistor, 560 Ohm, ± 1%, 63 mW, 0402 [1005 Metric], Thick Film, General Purpose | 560R | — |
| U1 | AXP805 | PMIC for Multi-Core High-Performance System | AXP805 | — |

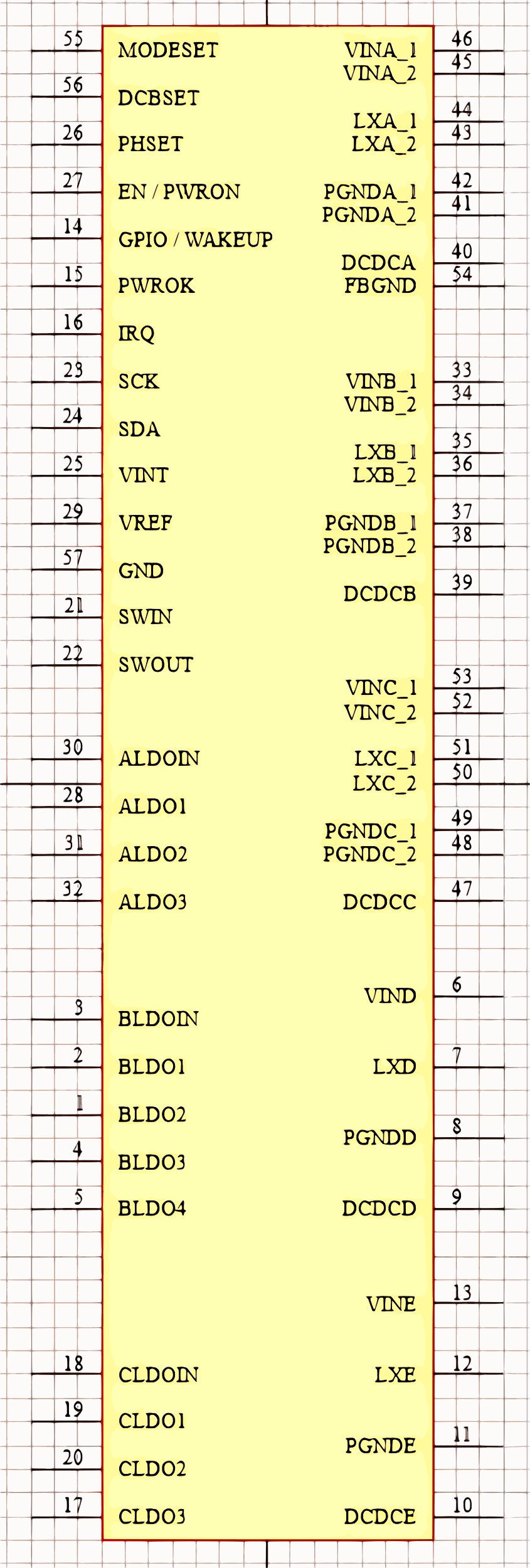


Рисунок A.1. Условно графическое обозначение AXP 805

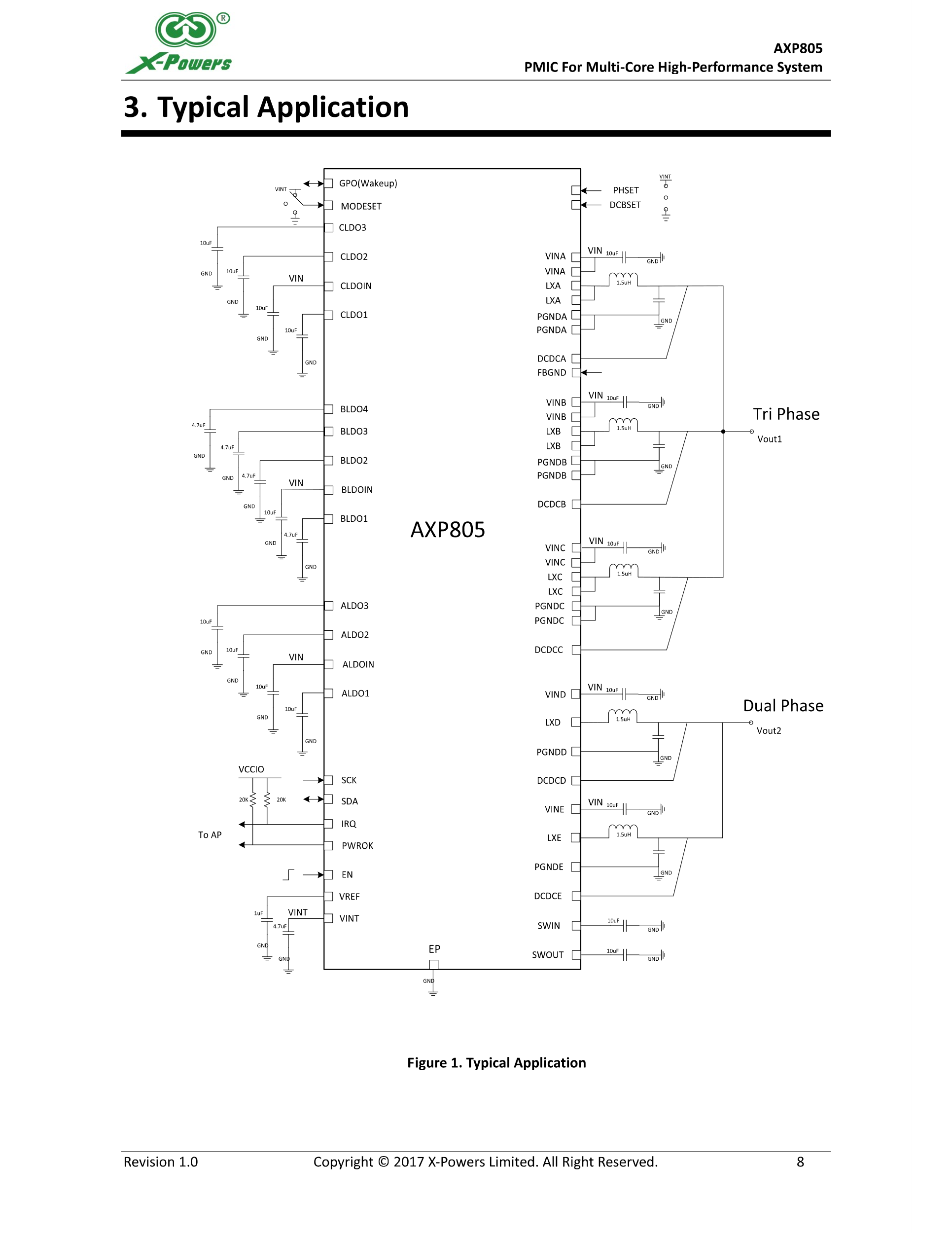


Рисунок A.2. Типовая схема применения AXP 805



Рисунок A.3. Принципиальная схема проекта

# Приложение B

Листинг 1

Система сборки образа операционной системы для Orange Pi 3 LTS

# syntax=docker/dockerfile:1-labs

FROM alpine:3.20 AS bootstrap

WORKDIR /tmp

ADD --checksum=sha256:02c7697e9a5351683bc56405b8e076f2acf0efb4dd49ce1b5a0dcdd398d5ca39 \

https://github.com/orangepi-xunlong/linux-orangepi/archive/refs/heads/orange-pi-5.10-media.zip \

linux-orange-pi-5.10-media.zip

ADD --checksum=sha256:703b7e5546b9db19f75b205f5c8d40f22238fa281e05f087aacb0da36540a928 \

https://github.com/orangepi-xunlong/u-boot-orangepi/archive/refs/heads/v2021.10-sunxi.zip \

u-boot-v2021.10-sunxi.zip

ADD --checksum=sha256:9551a51cc14943f238af8b7d3a212a518e9289c0517c2fa8421ace17fe2320b4 \

https://github.com/ARM-software/arm-trusted-firmware/archive/refs/tags/v2.2.zip \

arm-trusted-firmware-2.2.zip

ADD --checksum=sha256:9b368ce57f3e0aff7a4bfecfc237879bfc698f7a58f73dc6676a20b6d3b4e76d \

https://ftp.gnu.org/gnu/glibc/glibc-2.40.tar.bz2 \

glibc-2.40.tar.bz2

ADD --checksum=sha256:3311dff32e746499f4df0d5df04d7eb396382d7e108bb9250e7b519b837043a4 \

https://busybox.net/downloads/busybox-1.37.0.tar.bz2 \

busybox-1.37.0.tar.bz2

ADD --checksum=sha256:8dfe681531f0bd04fb9c53cf3c0a3368c616aa85d48938eebe2b516376e06a66 \

https://mirrors.dotsrc.org/armbian-dl/\_toolchain/gcc-arm-9.2-2019.12-x86\_64-aarch64-none-linux-gnu.tar.xz \

gcc-arm-9.2-2019.12-x86\_64-aarch64-none-linux-gnu.tar.xz

RUN unzip -qq linux-orange-pi-5.10-media.zip && rm linux-orange-pi-5.10-media.zip \

&& unzip -qq u-boot-v2021.10-sunxi.zip && rm u-boot-v2021.10-sunxi.zip \

&& unzip -qq arm-trusted-firmware-2.2.zip && rm arm-trusted-firmware-2.2.zip \

&& tar -xjf glibc-2.40.tar.bz2 && rm glibc-2.40.tar.bz2 \

&& tar -xjf busybox-1.37.0.tar.bz2 && rm busybox-1.37.0.tar.bz2 \

&& tar -xf gcc-arm-9.2-2019.12-x86\_64-aarch64-none-linux-gnu.tar.xz \

&& rm gcc-arm-9.2-2019.12-x86\_64-aarch64-none-linux-gnu.tar.xz

RUN sed -i 's/u-boot-\$(UBOOTVERSION)/\$(UBOOTVERSION)/g' \

u-boot-orangepi-2021.10-sunxi/scripts/dtc/pylibfdt/Makefile

# ---

FROM alpine:3.20 AS builder

WORKDIR /software

ENV PATH="$PATH:/opt/gcc-arm-9.2-2019.12-x86\_64-aarch64-none-linux-gnu/bin"

ENV TARGET\_TRIPLET="aarch64-none-linux-gnu"

COPY --from=bootstrap /tmp/linux-orangepi-orange-pi-5.10-media kernel

COPY --from=bootstrap /tmp/u-boot-orangepi-2021.10-sunxi bootloader

COPY --from=bootstrap /tmp/arm-trusted-firmware-2.2 firmware

COPY --from=bootstrap /tmp/glibc-2.40 glibc

COPY --from=bootstrap /tmp/busybox-1.37.0 busybox

COPY --from=bootstrap /tmp/gcc-arm-9.2-2019.12-x86\_64-aarch64-none-linux-gnu /opt/gcc-arm-9.2-2019.12-x86\_64-aarch64-none-linux-gnu

RUN apk update --no-cache \

&& apk add --no-cache \

build-base gcompat bison flex dtc python3 \

python3-dev openssl-dev py3-elftools py3-setuptools swig \

git linux-headers findutils perl tar xz \

openssl u-boot-tools gawk grep rsync

# TODO(annad): Debug build

# RUN cd /software/firmware \

# && make ENABLE\_BACKTRACE=1 PLAT=sun50i\_h6 DEBUG=0 bl31 -j32 CROSS\_COMPILE=$TARGET\_TRIPLET- \

RUN cd /software/firmware \

&& make -j`nproc` PLAT=sun50i\_h6 CROSS\_COMPILE=$TARGET\_TRIPLET- bl31

RUN cd /software/bootloader \

&& make -j`nproc` orangepi\_3\_lts\_defconfig CROSS\_COMPILE=$TARGET\_TRIPLET- \

&& make -j`nproc` CROSS\_COMPILE=$TARGET\_TRIPLET- SCP=/dev/null BL31=/software/firmware/build/sun50i\_h6/release/bl31.bin

RUN cd /software/kernel \

&& make -j`nproc` orangepi\_defconfig ARCH=arm64 CROSS\_COMPILE=$TARGET\_TRIPLET- \

&& make -j`nproc` ARCH=arm64 CROSS\_COMPILE=$TARGET\_TRIPLET-

RUN cd /software/glibc && mkdir build && cd build \

&& ../configure --prefix=/usr --host=$TARGET\_TRIPLET --disable-mathvec CFLAGS="-O2" CC=$TARGET\_TRIPLET-gcc AR=$TARGET\_TRIPLET-ar CXX=$TARGET\_TRIPLET-g++ \

&& make -j`nproc` CC=$TARGET\_TRIPLET-gcc AR=$TARGET\_TRIPLET-ar CXX=$TARGET\_TRIPLET-g++

RUN cd /software/busybox \

&& make -j`nproc` CROSS\_COMPILE=$TARGET\_TRIPLET- defconfig \

&& sed -i 's/CONFIG\_SHA1\_HWACCEL=y/CONFIG\_SHA1\_HWACCEL=n/' .config \

&& sed -i 's/CONFIG\_SHA256\_HWACCEL=y/CONFIG\_SHA256\_HWACCEL=n/' .config \

&& sed -i 's/CONFIG\_PREFIX=".\/\_install"/CONFIG\_PREFIX="\/software\/fs\/ram"/' .config \

&& make -j`nproc` CROSS\_COMPILE=$TARGET\_TRIPLET-

RUN mkdir -p /software/fs/ram && cd /software/fs/ram \

&& mkdir -p bin lib lib64 sbin usr sys dev proc mnt/root

RUN cd /software/kernel \

&& make -j`nproc` ARCH=arm64 CROSS\_COMPILE=$TARGET\_TRIPLET- INSTALL\_MOD\_PATH=/software/fs/ram modules\_install \

&& make -j`nproc` ARCH=arm64 CROSS\_COMPILE=$TARGET\_TRIPLET- INSTALL\_HDR\_PATH=/software/fs/ram/usr headers\_install

RUN cd /software/glibc/build \

&& make DESTDIR=/software/fs/ram install \

&& cd /software/fs/ram && rm usr/lib64/\*.a usr/lib64/\*.o

RUN cd /software/busybox && make -j`nproc` CROSS\_COMPILE=$TARGET\_TRIPLET- install

# NOTE(annad): Rootfs fork

RUN cp -rf /software/fs/ram /software/fs/root \

&& cd /software/fs/root && mkdir -p boot \

&& cp -rf /software/kernel/arch/arm64/boot/Image /software/fs/root/boot \

&& cp -rf /software/kernel/arch/arm64/boot/dts/allwinner/sun50i-h6-orangepi-3-lts.dtb /software/fs/root/boot

# ---

FROM alpine:3.20 AS assembler

WORKDIR /software

COPY --from=builder /software/fs fs

COPY --from=builder /software/bootloader/u-boot-sunxi-with-spl.bin bootloader/u-boot-sunxi-with-spl.bin

RUN apk update --no-cache \

&& apk add u-boot-tools e2fsprogs

COPY --chmod=755 ./init /software/fs/ram/init

COPY --chmod=644 ./boot.cmd /software/fs/root/boot

RUN cd /software/fs && dos2unix ram/init root/boot/boot.cmd

RUN cd /software/fs/ram \

&& find . | cpio -H newc -ov --owner root:root > ../initramfs.cpio \

&& cd .. && gzip initramfs.cpio \

&& mkimage -A arm -T ramdisk -n uInitrd -d initramfs.cpio.gz uInitrd

RUN cd /software/fs/root/boot \

&& mkimage -C none -A arm -T script -d boot.cmd boot.scr \

&& cp /software/fs/uInitrd .

RUN dd if=/dev/zero bs=1M count=2 of=DISK.ISO \

&& dd conv=notrunc if=bootloader/u-boot-sunxi-with-spl.bin bs=1024 seek=8 of=DISK.ISO

RUN dd if=/dev/zero bs=1M count=510 of=DISK1.ISO \

&& mkfs.ext2 -L BOOT DISK1.ISO

RUN --security=insecure mount DISK1.ISO /tmp && cp -rf /software/fs/root/\* /tmp && umount /tmp

RUN dd if=DISK1.ISO conv=notrunc oflag=append bs=1M seek=2 of=DISK.ISO

# TODO(annad): Change this part

RUN printf "o\nn\np\n1\n4096\n\nw\n\n" | fdisk DISK.ISO || true

# ---

FROM scratch

COPY --from=assembler /software/DISK.ISO FLAMME.ISO