Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра проектирования информационно-компьютерных систем

Дисциплина «Мобильные вычислительные системы»

«К защите допустить»

Руководители курсового проекта

Доцент кафедры ПИКС *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*В.С. Колбун

*\_\_*.*\_\_*.20*\_\_*

Доцент кафедры ПИКС *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*О.Ч. Ролич

*\_\_*.*\_\_*.20*\_\_*

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту

на тему

**«МОБИЛЬНЫЙ ЭЛЕКТРОННЫЙ ЦЕННИК   
С РАДИОКАНАЛОМ СВЯЗИ»**

БГУИР КП 1–39 03 02 022 ПЗ

|  |  |  |
| --- | --- | --- |
| Студент | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*  (подпись студента) | Н.Н. Романович |
|  |  | Курсовой проект представлен на проверку *\_\_*.*\_\_*.20*\_\_*  *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*  (подпись студента) |

Минск 2020

**РЕФЕРАТ**

**Романович, Н.Н.** Мобильный электронный ценник с радиоканалом связи: курсовой проект / Н.Н. Романович – Минск: БГУИР, 2020, – п.з. – 36 с., чертежей – 4 л. формата А3, плакатов – 2 л. формата А3.

Цель проектирования: проектирование устройства, предназначенного для использования в качестве портативного модуля индикации в системе динамического управления ценообразованием

Методология проведения работы: в процессе решения поставленных задач использованы принципы системного подхода, теория проектирования сверхнизкопотребляющих автономных устройств, теория проектирования печатных плат и программирования микроконтроллеров семейства *ARM*.

Результаты работы: выполнен анализ предметной области   
и потребностей пользователей, спроектировано мобильное автомномное устройство, предназначенное для работы в системе динамического ценообразования, разработано демонстративное программное обеспечение для работы устройства.

Устройство обеспечивает отображение текстовой (цены, сроки годности, акции, скидки и другое) и графической информации (двухмерные коды, бихромные изображения товаров).

**СОДЕРЖАНИЕ**

[Введение..................................... .. 7](#_Toc39785971)

[1 Общетехническое обоснование разработки устройства 9](#_Toc39785972)

[1.1 Анализ исходных данных 9](#_Toc39785973)

[1.2 Теоретические сведения и принципы функционирования   
отдельных узлов устройства 9](#_Toc39785974)

[1.2.1 Архитектурные особенности современных систем управления ценообразованием, общие алгоритмы функционирования 9](#_Toc39785975)

[1.2.2 Сравнительный анализ современных микроконтроллеров   
по токопотреблению 10](#_Toc39785976)

[1.2.3 Структура микроконтроллера *nRF52832* 11](#_Toc39785977)

[1.2.4 Регистровая модель портов ввода-вывода общего назначения микроконтроллерного ядра *ARM Cortex-M4F* 12](#_Toc39785978)

[1.2.5 Физический и канальный уровни интерфейсов *SPI* и *I2C* 12](#_Toc39785979)

[1.2.6 Принцип действия дисплеев типа *ePD* «электронная бумага» 16](#_Toc39785980)

[1.2.7 Логика работы микросхемы *M41T62* «часов реального времени» 17](#_Toc39785981)

[1.2.8 Принцип действия, регистровая модель и логика функционирования   
*MEMS*-акселерометра *LIS3DH* 18](#_Toc39785982)

[1.2.9 Принцип действия пьезоэлектрического излучателя типа «*buzzer*», способы программного управления в формировании звуковых эффектов 19](#_Toc39785983)

[2 Разработка структурной электрической схемы мобильного электронного ценника с радиоканалом связи............... 20](#_Toc39785984)

[2.1 Обоснование базовых блоков структурной схемы электронного ценника 20](#_Toc39785985)

[2.2 Обоснование связей структурной схемы электронного ценника 20](#_Toc39785986)

[3 Разработка принципиальной электрической схемы электронного ценника 22](#_Toc39785987)

[3.1 Обоснование выбора САПР для разработки   
принципиальной электрической схемы 22](#_Toc39785988)

[3.2 Описание используемых библиотечных элементов и процесса их создания 22](#_Toc39785989)

[3.3 Обоснование выбора базовых компонентов принципиальной схемы   
электронного ценника с радиоканалом связи 23](#_Toc39785990)

[3.4 Обоснование связей принципиальной электрической схемы   
электронного ценника с радиоканалом связи 24](#_Toc39785991)

[3.5 Анализ и обоснование принципиальной электрической схемы питания   
электронного ценника 25](#_Toc39785992)

[4 Разработка модели и алгоритма функционирования   
электронного ценника с радиоканалом связи 26](#_Toc39785993)

[5 Разработка конструкции проектируемого изделия 27](#_Toc39785994)

[5.1 Выбор и обоснование элементной базы 27](#_Toc39785995)

[5.2 Выбор и обоснование конструктивных элементов и установочных изделий 27](#_Toc39785996)

[6 Расчёт конструктивно-технологических параметров проектируемого изделия 29](#_Toc39785997)

[6.1 Проектирование печатного модуля 29](#_Toc39785998)

[6.2 Выбор и обоснование материалов конструкции и защитных покрытий, маркировки деталей и сборочных единиц 30](#_Toc39785999)

[7 Применение средств автоматизированного проектирования   
при разработке устройства.... 32](#_Toc39786000)

[Заключение................................. 33](#_Toc39786001)

[Список использованных источников 34](#_Toc39786002)

[Приложение А (обязательное) Визуализированная трехмерная модель устройства 35](#_Toc39786003)

[Приложение Б (обязательное) Листинги программного кода 36](#_Toc39786004)

[Приложение В (обязательное) Результат проверки проекта на плагиат 52](#_Toc39786005)

[Ведомость документов.............. 53](#_Toc39786006)

# Введение

Стремительно развивающиеся инновационные технологии не могли не коснуться такого обширного сектора, как розничная торговля. Еще лет пятнадцать тому назад товары в магазинах отпускались в основном по традиционному методу продаж – через прилавок. При этом не применялись какие-либо инновационные или передовые технические решения. Современная действительность уже немыслима без магазинов самообслуживания, где с каждым днем внедряется все больше технических решений, облегчающих жизнь и персоналу, и покупателям.

*C* ростом числа торговых сетей и, как следствие, конкуренции все большую значимость для эффективного управления предприятием приобретает внедрение современных инновационных технологий и систем в оптовой и розничной торговле, начиная с поставки необходимого предприятию торгового оборудования и специализированного программного обеспечения до ввода его в эксплуатацию и осуществления дальнейшего его сервисного сопровождения. Современные информационные системы дают возможность оптимизации управления складскими запасами и ассортиментом товаров, предлагают механизмы гибкого ценообразования, системы повышения лояльности клиентов. В конечном итоге вложения в инновации оборачиваются для владельцев торговых предприятий полным контролем над процессами торговли и повышением финансовых результатов.

Спектр применяемых технологий и оборудования обширен: начиная с обычных ручных сканеров, заканчивая автоматическими и полуавтоматическими упаковщиками. В последнее время все большую заинтересованность клиентов вызывают электронные ценники. Внедрение электронных ценников на предприятии – это уход от бумажных технологий, минимизация ошибок и экономия времени при актуализации цен.

Существует множество разновидностей электронных ценников. Малогабаритное беспроводное устройство с экраном не только демонстрирует покупателю актуальную цену на товар, но и напрямую связывается посредством радиосвязи с весами, являясь неким их продолжением, «удаленной» кнопкой.

Применение данных устройств позволяет оптимизировать труд персонала. Сотрудник уже не работает с бумагой и принтером, ему не требуется раз за разом распечатывать бумажные ценники при изменении цены на товар.

Достаточно поставить устройство к определенному виду продукции – и при изменении цены товара она автоматически изменится на электронном экране ценника. Таким образом, с одной стороны, вся предоставляемая клиенту информация является актуальной, с другой – минимизируются ошибки персонала, ускоряются процессы выбора нужного товара на весах и своевременного обновления цен. Информирование покупателя об изменении цены на товар уже не зависит от скорости работы персонала. Электронные ценники просты в использовании и легко поддаются влажной обработке и дезинфекции. Они имеют высокий класс защиты от влаги и пыли, их можно легко мыть под струей проточной воды. Это, как показывает практика, не только удобно, но и экономично.

Таким образом целью данного курсового проекта является разработка печатной платы мобильного электронного ценника с радиоканалом связи, создание чертежей для последующего производства печатной платы, а также описание работы данного устройства в виде диаграммы состояния и схемы алгоритма. В будущем, запрограммировав устройство будет возможность использовать его по предназначению.

Тогда основными задачами в процессе проектирования будут: разработка структурной электрической схемы устройства, принципиальной электрической, модели и алгоритма функционирования, проектирование и разработка конструкции изделия с расчетом конструктивно-технологических параметров.

Курсовой проект выполнен самостоятельно, проверен в системе «Антиплагиат». Процент оригинальности составляет 74,25%. Цитирования обозначены ссылками на публикации, указанными в «Списке использованных источников».

Скриншот приведен в приложении.

# Общетехническое обоснование разработки устройства

## Анализ исходных данных

Разработка электронного устройства, предназначенного для использования в качестве портативного модуля индикации в системе динамического управления ценообразованием – мобильного электронного ценника с радиоканалом связи.

Размеры устройства: не более 80 × 30 × 10 мм.

Масса – не более 0,1 кг.

Потребляемый ток: средний – 10 мкА, пиковый – не более 20 мА.

Основное питание от аккумуляторной батареи типа 2032 напряжением 3В (емкость 220 мАч).

## Теоретические сведения и принципы функционирования отдельных узлов устройства

### Архитектурные особенности современных систем управления ценообразованием, общие алгоритмы функционирования

Система управления ценообразованием состоит из большого количества автономных электронных ценников и базовой станции управления, которая интегрируется в кассовую систему магазина.

Автономный электронный ценник периодически, либо по сигналу базовой станции связывается с ней для обновления данных и отправки информации о состоянии, например, о заряде батареи.

Для защиты от несанкционированного воздействия используются криптографические средства защиты информации: шифрование передаваемых данных, проверка их целостности посредством цифровой подписи. Для защиты от физического воздействия в мобильном электронном ценнике имеется акселерометр, который запускает звуковой сигнал в случае физического воздействия, а также сообщает об этом воздействии на базовую станцию для своевременного уведомления сотрудников о произошедшем.

### Сравнительный анализ современных микроконтроллеров по токопотреблению

В качестве основы устройства целесообразно использовать решения в виде системы на одном кристалле (далее – *SoC*), включающие в одной микросхеме ядро микроконтроллера и радиоинтерфейс. В качестве протокола для радиообмена целесообразно использовать *Bluetooth* *Low* *Energy*(далее – *BLE*), разработанный специально для устройств класса «интернет вещей» – устройств, характеризующихся низким энергопотреблением и долгим временем работы.

На рынке представлено множество *SoC* с ультранизким энергопотреблением, имеющих в своем составе микроконтроллер *Cortex-M0+* или *Cortex-M4F*, радиоинтерфейс и стек протоколов *Bluetooth* *Low* *Energy*. Например, серия *BlueNRG* от *STMicroelectronics*, *CC2541* и *CC2640* от *Texas* *Instruments*, серия *nRF52* от *Nordic* *Semiconductors*.

Серия *nRF52* от *Nordic* *Semiconductors* является наиболее перспективной для данного проекта, в качестве базы устройства будет использован контроллер *nRF52832*.

Чипсет *nRF52832* - новая мощная, малопотребляющая и гибкая система на кристалле с поддержкой протоколов *Bluetooth* *Smart*, *ANT* и проприетарных стеков собственной разработки пользователя в частотном диапазоне 2.4ГГц. *nRF52832* построен на 32-разрядном процессорном ядре *ARM Cortex-M4F* с 512 КБ *Flash* и 64 КБ *RAM* памяти. *nRF52832* является сверхмалопотребляющей системой на кристалле, способной использовать питание в диапазоне от 1.7В до 3.6В. Вся периферия и тактовые генераторы могут оперативно отключаться в автоматическом режиме с целью сведения энергопотребления до минимального уровня [1].

Встроенное процессорное ядро имеет автоматизированную адаптивную систему управления энергопотреблением. Энергопотребление автоматически регулируется в зависимости от операций, выполняемых в данный момент процессорным ядром при обращении к периферийным устройствам таким образом, чтобы работали только те составные части системы, которые необходимы для выполнения текущей задачи.

Кроме того, преимуществом контроллеров серии *nRF52* является наличие модуля *PPI* (*Programmable* *peripheral* *interconnect*). *PPI* модуль необходим для передачи данных о событии или прерывании одной периферии в другую без участия ядра контроллера. Например, прерывание по таймеру может без участия ядра контроллера запустить любую единицу периферии. Кроме того, общение через модуль *PPI* между периферийными устройствами может происходить при спящем или отключенном *CPU*.

### Структура микроконтроллера *nRF52832*

*nRF52832* содержит мощное процессорное ядро *ARM* *Cortex*-*M4F*, соответствующее требованиям многих требовательных к вычислительной производительности, но при этом компактных приложений, способных работать на одном кристалле (см. рисунок 1.1). Ядро способно решать задачи цифровой обработки сигналов, выполнять операции с плавающей точкой, выполнять операции с мультиплексированием и накоплением в пределах одного рабочего цикла, аппаратно поддерживает энергетически эффективно реализованную операцию деления в вещественной и комплексной формах[1].

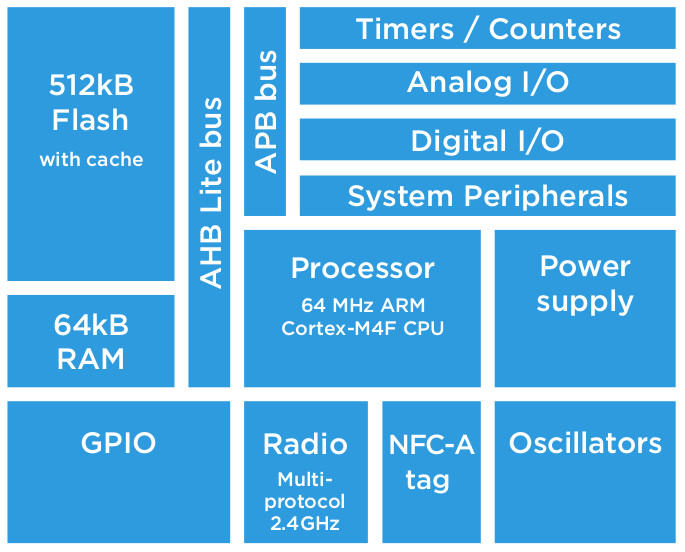


Рисунок 1.1 – Структура микроконтроллера *nRF52832*

Радиочасть 2.4 ГГц совместима со стеками протоколов *Bluetooth* *Smart*, *ANT* и любыми проприетарными стеками. Радио имеет встроенный регистр *RSSI* высокого разрешения с возможностью автоматической работы в режиме *EasyDMA* для прямого доступа к памяти в момент приема и передачи данных по радиоканалу. Производитель предоставляет стеки протоколов *Bluetooth* *Smart*, *ANT* и *Gazel* (2.4ГГц) в качестве загружаемого контента на своем официальном веб-сайте.

### Регистровая модель портов ввода-вывода общего назначения микроконтроллерного ядра *ARM Cortex-M4F*

Каждый из выводов *GPIO* может быть настроен программным обеспечением как выход (двухтактный или открытый сток, с или без подтягивания), как вход (плавающий, с подтягиванием или без) или как периферийная альтернативная функция. Большинство выводов *GPIO* используются совместно с цифровыми или аналоговыми альтернативными функциями. У всех портов *GPIO* высокий ток и есть выбор скорости для лучшего управления внутренним шумом, потреблением энергии и электромагнитным излучением. Конфигурация входов / выходов может быть заблокирована путем выполнения определенной последовательности. Это делается для того, чтобы избежать ложной записи в регистры ввода/вывода.

### Физический и канальный уровни интерфейсов *SPI* и *I*2*C*

*SPI* (*Serial* *Peripheral* *Interface*) – последовательный синхронный стандарт передачи данных в режиме полного дуплекса, предназначенный для обеспечения простого и недорогого высокоскоростного сопряжения микроконтроллеров и периферии. *SPI* также иногда называют четырёхпроводным интерфейсом [2].

*SPI* является синхронным интерфейсом, в котором любая передача синхронизирована с общим тактовым сигналом, генерируемым ведущим устройством (процессором). Принимающая (ведомая) периферия синхронизирует получение битовой последовательности с тактовым сигналом. К одному последовательному периферийному интерфейсу ведущего устройства-микросхемы может присоединяться несколько микросхем. Ведущее устройство выбирает ведомое для передачи, активируя сигнал «выбор кристалла» (*chip* *select*) на ведомой микросхеме. Периферия, не выбранная процессором, не принимает участия в передаче по *SPI*.

В *SPI* используются четыре цифровых сигнала:

* *MOSI* – выход ведущего, вход ведомого (*Master* *Out* *Slave* *In*). Служит для передачи данных от ведущего устройства ведомому;
* *MISO* – вход ведущего, выход ведомого (*Master* *In* *Slave* *Out*). Служит для передачи данных от ведомого устройства ведущему;
* *SCLK* – последовательный тактовый сигнал (*Serial* *Clock*). Служит для передачи тактового сигнала для ведомых устройств;
* *CS* или *SS* – выбор микросхемы, выбор ведомого (*Chip* *Select*, *Slave* *Select*).

Частота следования битовых интервалов в линиях передачи данных определяется синхросигналом *SCK*, который генерирует ведущее устройство, ведомые устройства используют синхросигнал для определения моментов изменения битов на линии данных, при этом ведомые устройства никак не могут влиять на частоту следования битовых интервалов.

Так как действия ведущего и ведомого устройства тактируются одним и тем же сигналом, то к стабильности этого сигнала не предъявляется никаких требований, за исключением ограничения на длительность полупериодов, которая определяется максимальной рабочей частотой более медленного устройства. Это позволяет использовать *SPI* в системах с низкостабильной тактовой частотой.

Передача осуществляется пакетами. Длина пакета, как правило, составляет 1 байт (8 бит), при этом известны реализации *SPI* с иной длиной пакета. Ведущее устройство инициирует цикл связи установкой низкого уровня на выводе выбора подчиненного устройства (*SS*) того устройства, с которым необходимо установить соединение.

Подлежащие передаче данные ведущее и ведомое устройства помещают в сдвиговые регистры. После этого ведущее устройство начинает генерировать импульсы синхронизации на линии *SCLK*, что приводит к взаимному обмену данными. Передача данных осуществляется бит за битом от ведущего по линии *MOSI* и от ведомого по линии *MISO*.

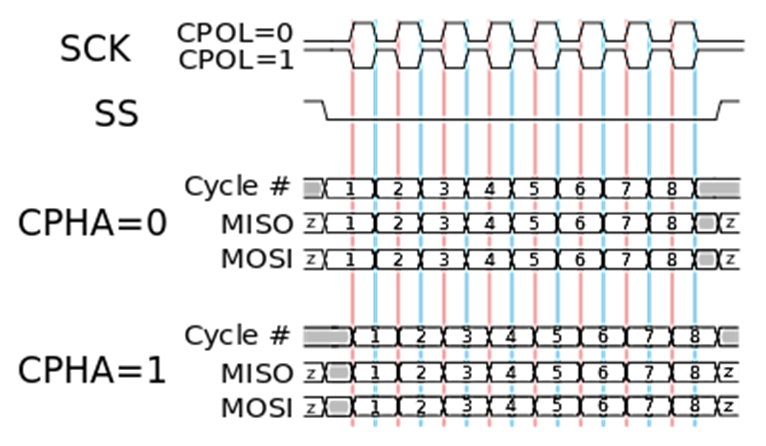


Рисунок 1.2 – Временные диаграммы работы интерфейса *SPI* при *CPHA* = 0 и *CPHA* = 1

Возможны четыре комбинации фазы (*CPHA*) и полярности (*CPOL*) сигнала *SCLK* по отношению к сигналам данных. Режимы работы определяются комбинацией бит *CPHA* и *CPOL*:

* *CPOL* = 0 – сигнал синхронизации начинается с низкого уровня;
* *CPOL* = 1 – сигнал синхронизации начинается с высокого уровня;
* *CPHA* = 0 – выборка данных производится по переднему фронту сигнала синхронизации;
* *CPHA* = 1 – выборка данных производится по заднему фронту сигнала синхронизации.

В простейшем случае к ведущему устройству подключено единственное ведомое устройство и необходим двусторонний обмен данными. В таком случае используется трехпроводная схема подключения. Интерфейс *SPI* позволяет подключать к одному ведущему устройству несколько ведомых устройств, причем подключение может быть осуществлено несколькими способами.

Первый способ позволяет реализовать радиальную структуру связи (см. рисунок 1.3), его принято считать основным способом подключения нескольких ведомых устройств.

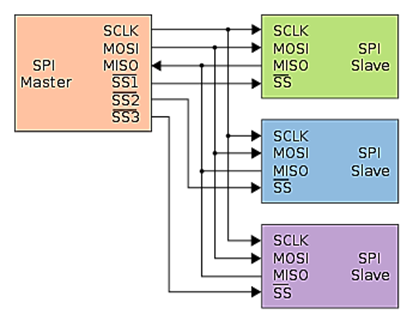


Рисунок 1.3 – Радиальная структура связи с несколькими ведомыми устройствами

В данном случае для обмена более чем с одним ведомым устройством ведущее устройство должно формировать соответствующее количество сигналов выбора ведомого устройства (*SS*). При обмене данными с ведомым устройством, соответствующий ему сигнал *SS* переводится в активное (низкое) состояние, при этом все остальные сигналы *SS* находятся в неактивном (высоком) состоянии. Выводы данных *MISO* ведомых устройств соединены параллельно, при этом они находятся в неактивном состоянии, а перед началом обмена один из выходов (выбранного ведомого устройства) переходит в активный режим.

Второй способ позволяет реализовать структуру связи типа «кольцо» (см. рисунок 1.4). В данном случае для активации одновременно нескольких ведомых устройств используется один сигнал *SS*, а выводы данных всех устройств соединены последовательно и образуют замкнутую цепь.

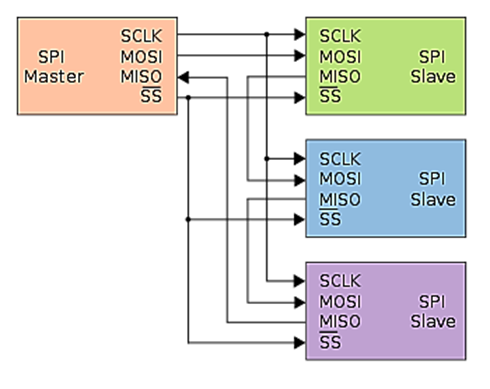


Рисунок 1.4 – Кольцевая структура связи с несколькими ведомыми устройствами

При передаче пакета от ведущего устройства этот пакет получает первое ведомое устройство, которое, в свою очередь, транслирует свой пакет следующему ведомому устройству и так далее. Для того, чтобы пакет от ведущего устройства достиг определенного ведомого устройства, ведущее устройство должно отправить еще несколько пакетов.

*I*²*C* (англ. *Inter*-*Integrated* *Circuit*) – последовательная асимметричная шина для связи между интегральными схемами внутри электронных приборов. Использует две двунаправленные линии связи (*SDA* и *SCL*), применяется для соединения низкоскоростных периферийных компонентов с процессорами и микроконтроллерами (например, на материнских платах, во встраиваемых системах, в мобильных телефонах).

Данные передаются по двум проводам – проводу данных и проводу тактов. Есть ведущий (*master*) и ведомый (*slave*), такты генерирует *master*, ведомый лишь «поддакивает» при приёме байта. Всего на одной двупроводной шине может быть до 127 устройств.

*I*²*C* использует две двунаправленные линии, подтянутые к напряжению питания и управляемые через открытый коллектор или открытый сток – последовательная линия данных (*SDA*, англ. *Serial* *DAta*) и последовательная линия тактирования (*SCL*, англ. *Serial* *CLock*). Стандартные напряжения +5 В или +3,3 В, однако допускаются и другие.

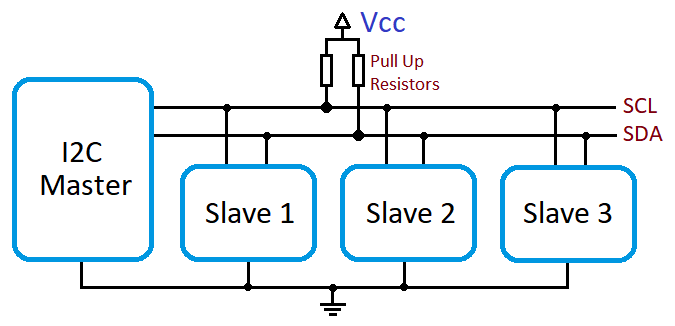


Рисунок 1.5 – Пример с одним ведущим и несколькими ведомыми устройствами

Классическая адресация включает 7-битное адресное пространство с 16 зарезервированными адресами. Это означает, что разработчикам доступно до 112 свободных адресов для подключения периферии на одну шину.

Основной режим работы 100 кбит/с; 10 кбит/с в режиме работы с пониженной скоростью. Также немаловажно, что стандарт допускает приостановку тактирования для работы с медленными устройствами.

### Принцип действия дисплеев типа *ePD* «электронная бумага»

Электронная бумага – технология отображения информации, разработанная для имитации обычной печати на бумаге и основанная на явлении электрофореза. В отличие от традиционных плоских жидкокристаллических дисплеев, в которых используется просвет матрицы для формирования изображения, электронная бумага формирует изображение в отражённом свете как обычная бумага и может хранить изображение текста и графики в течение достаточно длительного времени, не потребляя при этом электрической энергии и затрачивая её только на изменение изображения. В отличие от традиционной бумаги, технология позволяет произвольно изменять записанное изображение.

Преимуществом можно назвать большее время автономной работы, которое отличается в лучшую сторону по сравнению с прочими электронными устройствами с дисплеями. Экран на основе электронной бумаги потребляет энергию при изменении отображаемой информации (например, перелистывании страниц), тогда как типичный ЖК экран потребляет энергию постоянно.

### Логика работы микросхемы *M*41*T*62 «часов реального времени»

В семейство *M41T6x* входят микросхемы *RTC* с низким энергопотреблением и встроенным кварцевым генератором на частоте 32,768 кГц. Восемь регистров, использующиеся для функции часы/календарь, конфигурируются в двоично-десятичном коде (*BCD*). Дополнительные восемь регистров обеспечивают функции состояния/управления будильником, управляют выходной частотой 32 кГц, осуществляют калибровку и функцию сторожевого таймера. Обмен данными происходит по интерфейсу *I2C* (400 кГц). Адресный регистр автоматически инкрементируется после записи или чтения байта данных. Блок-схема представителя семейства *M41T6x* – *M41T62* показана на рисунке 1.6.

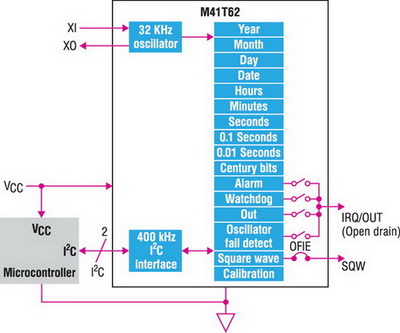


Рисунок 1.6 – Структура микросхемы часов реального времени *M41T62*

Функции, доступные пользователю, включают время суток/календарь, прерывания по срабатыванию будильника, выход частоты 32 кГц, выход прямоугольного сигнала с программируемой частотой. По восьми адресам содержатся значения столетия, года, месяца, даты, дня недели, часов, минут, секунд и десятых/сотых долей секунды в 24-часовом *BCD*-формате. Коррекция продолжительности месяца в 28, 29 (февраль високосного года), 30 или 31 день осуществляется автоматически [3].

Микросхемы *M41T62* выпускаются в двух типах корпусов миниатюрного размера: *QFN* с 16 выводами размером 3×3 мм, для которого требуется внешний кристалл частотой 32 кГц, и *LCC* 1,5×3,2 мм со встроенным кристаллом.

### Принцип действия, регистровая модель и логика функционирования *MEMS*-акселерометра *LIS*3*DH*

В общем случае в состав *MEMS*-акселерометра (рисунок 3) входит подвижная трехмерная структура, которую можно представить в виде набора конденсаторов переменной емкости, усилителя заряда, связанного с мультиплексором, подающим сигнал на вход усилителя, и демультиплексором, выдающим сигналы для сигма-дельта АЦП. После АЦП данные фильтруются и поступают в регистры хранения. Доступ к данным и управление настройками акселерометра осуществляются по *I2C*- или *SPI*-интерфейсу, отдельный блок управляет работой акселерометра и формированием прерываний [4].

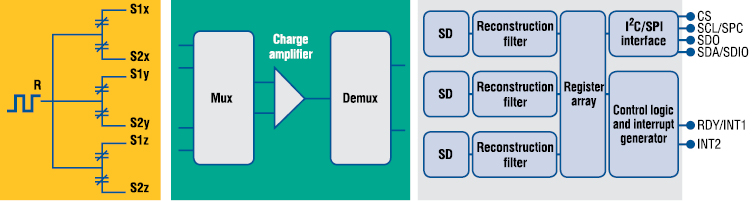


Рисунок 1.7 – Структурная схема трехосевого акселерометра

*LIS3DH* является высокопроизводительным трехосевым цифровым акселерометром с ультранизким энергопотреблением. *LIS3DH* имеет два режима работы – нормальный, обеспечивающий высокую производительность, и режим пониженного потребления.

Динамический диапазон измерений датчика может выбираться пользователем и лежит в пределах ±2*g*/±4*g*/±8*g*/±16*g*. Частота следования отсчетов – 0.001…5 кГц. Встроенная функция самотестирования позволяет проверить функционирование датчика в конечном устройстве.

Акселерометр может генерировать два независимых сигнала прерывания – срабатывание по преодолению порогового значения или по обнаружению свободного падения, а также по изменению положения устройства. Пороги и времена срабатывания прерываний могут быть заданы пользователем.

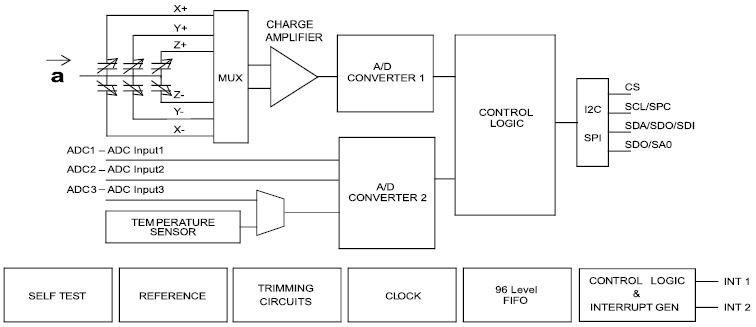


Рисунок 1.8 – Структурная схема акселерометра *LIS*3*DH*

На каждый канал измерений (ось) имеется собственный десятиразрядный *FIFO*-буфер на 32 значения.

В качестве «дополнительного бонуса» *LIS3DH* имеет три внешних канала АЦП (10 бит) для отслеживания внешних сигналов.

### Принцип действия пьезоэлектрического излучателя типа «*buzzer*», способы программного управления в формировании звуковых эффектов

Пьезоэлектрический излучатель – электроакустическое устройство, способное воспроизводить звук, либо излучать ультразвук, благодаря обратному пьезоэлектрическому эффекту.

Пьезоэлектрический излучатель состоит из металлической пластины, на которую нанесён слой пьезоэлектрика, имеющий на внешней стороне токопроводящее напыление. Пластина и напыление являются двумя контактами. Для увеличения громкости звука к металлической пластине может крепиться небольшой рупор в виде металлического или пластикового купола с отверстием. В качестве рупора также может использоваться углубление в корпусе устройства, в котором используется пьезоизлучатель.

Пьезоизлучатели приводятся в действие подачей переменного напряжения определённой частоты и амплитуды, обычно, 3…10 В. Частота, при которой звуковое давление максимально, может достигать 75 дБ на расстоянии 1 метр от излучателя. Резонансная частота для большинства пьезоизлучателей составляет 1…4 кГц. Этим обусловлен их характерный, узнаваемый звук.

# Разработка структурной электрической схемы мобильного электронного ценника с радиоканалом связи

## Обоснование базовых блоков структурной схемы электронного ценника

Основными составляющими структурной схемы являются:

1. Микроконтроллер. Используется для логического функционирования устройства, а также определяет допустимые параметры работы с устройством. Содержит в себе радиоинтерфейс, что позволяет обойтись без дополнительных элементов для обеспечения радиоканала связи.
2. Дисплей на электронных чернилах. Используется для отображения информации.
3. Часы реального времени. Используется для обеспечения алгоритмов работы устройства реальным текущим временем, а также для пробуждения устройства по расписанию.
4. Акселерометр. Используется для обеспечения минимальной физической защиты устройства – например, для реакции на удар по устройству.
5. Пьезокерамический излучатель. Используется для звуковой сигнализации для удобства поиска устройства обслуживающим персоналом, а также для сигнализации при физическом воздействии.
6. Светодиод. Используется для световой индикации.
7. Антенна.
8. Батарея.

## Обоснование связей структурной схемы электронного ценника

Микроконтроллер является входной точкой к практически всем элементам на структурной схеме, поскольку именно он отвечает за логику функционирования составных частей устройства, обрабатывая, анализируя и определяя их разрешения на тое или иное событие.

Аккумуляторная батарея подключена ко всем элементам на структурной схеме, поскольку она обеспечивает питание всех элементов для полноценного функционирования устройства.

Дисплей на электронных чернилах подключается к микроконтроллеру по стандартному интерфейсу *SPI* с дополнительными линиями согласно документации на дисплей.

Часы реального времени и акселерометр подключаются к микроконтроллеру посредством шины *I*2*C*. Дополнительно подключаются выводы прерываний для того, чтобы вышеперечисленные устройства могли сгенерировать аппаратное прерывание на контроллере, а также при необходимости вывести его из режима сна.

Пьезокерамический излучатель подключается к выводу общего назначения, который сконфигурирован для работы в режиме широтно-импульсной модуляции.

Светодиод подключается к выводу общего назначения, который сконфигурирован в режиме широтно-импульсной модуляции, что позволяет динамически управлять яркостью.

Порт прошивки и отладки подключается к микроконтроллеру через *SWD*.

# Разработка принципиальной электрической схемы электронного ценника

## Обоснование выбора САПР для разработки принципиальной электрической схемы

Для разработки принципиальной схемы используется   
*Altium* *Designer* – комплексная система автоматизированного проектирования (САПР) радиоэлектронных средств, разработанная австралийской компанией *Altium*. *Altium* *Designer* позволяет реализовывать проекты электронных средств на уровне схемы или программного кода с последующей передачей информации проектировщику ПЛИС или печатной платы [5]. Отличительной особенностью программы является проектная структура и сквозная целостность ведения разработки на разных уровнях проектирования.

Для оформления конструкторской документации в соответствии с требованиями ЕСКД, использовалось расширение для *Altium* *Designer* от компании «Главкон» – *Board* *Assistant* – программный комплекс, разработанный с целью автоматизировать процесс проектирования, а именно выпуск конструкторской документации по ЕСКД [6].

## Описание используемых библиотечных элементов и процесса их создания

Для большинства стандартных компонентов, таких как резисторы, конденсаторы, диоды, катушки индуктивности, корпуса и посадочные места являются стандартными. Поэтому нет необходимости в ручной отрисовке посадочных мест и условных графических обозначений, а возможно использовать готовую открытую библиотеку компонентов, разработанную в соответствии со стандартами ЕСКД [7].

Для остальных компонентов, таких как микросхемы, держатель батареи, антенна, библиотечные элементы создаются вручную, либо с использованием инструмента *IPC* *Compliant* *Footprint* *Wizard*, который позволяет автоматически создавать посадочные места стандартных типов корпусов на основе данных от производителя компонента.

Процесс создания библиотечных элементов проходит по общему алгоритму для каждого компонента, но различаться лишь размерами корпуса, существующих и представленных в официальном документе условных графических обозначений, а также количеством выводов:

1. Создание УГО. Сначала создается схематическое отображение элемента в *SchLib* т.е. его отображение, которое появляется при добавлении элемента на *SchDoc*. Основной задачей является правильно расставить выводы элемента, так чтобы на схеме было удобно подключать его к другим компонентам.
2. Создание посадочного места с прикреплением трехмерной модели. *Alitum* *Designer* предоставляет возможность создавать посадочные места в *PcbLib* вручную, либо же загружать их со встроенных, либо сторонних библиотек.

## Обоснование выбора базовых компонентов принципиальной схемы электронного ценника с радиоканалом связи

Принципиальная схема содержит следующие компоненты:

1. Разъем подключения дисплея *FPC*-*24*. Необходим для подключения шлейфа дисплея на электронных чернилах *Waveshare* *EPD* 2.13B.
2. Разъем прошивки и отладки *SWD*. Используется для загрузки прошивки на производстве. Представляет собой 4 контактных площадки, расположенные непосредственно на печатной плате.
3. Микроконтроллер *nRF52832-QFAA-R*. Микроконтроллер выбран в корпусе *VQFN-48*, который позволяет разместить микроконтроллер на двухсторонней печатной плате, не прибегая к разработке многослойных плат.
4. Часы реального времени *M41T62LC6F*. Микросхема выбрана в корпусе *LCC*-6, так как в данном корпусе данная микросхема содержит выход *SQW*, который может быть использован для тактирования основного микроконтроллера в режиме пониженного энергопотребления.
5. *MEMS*-акселерометр *LIS3DHTR* в корпусе *LGA-16*.
6. Держатель батареи типоразмера *CR2032* *BC*-*2003*.
7. Излучатель пьезокерамический *PKMCS0909E4000*-*R*.
8. Антенна 2,4ГГц *2450AT18B100E*. Использование чип-антенны позволяет избежать сложного расчета и подстройки *PCB*-антенн и сэкономить площадь печатной платы.

## Обоснование связей принципиальной электрической схемы электронного ценника с радиоканалом связи

Для подключения питания и антенны к микроконтроллеру использовалась схема (см. рисунок 3.1) и рекомендации от производителя. В качестве источника тактирования режима пониженного потребления вместо опционального кварцевого резонатора частотой 32,768кГц используется выход *SQW* микросхемы часов реального времени, который выдает прямоугольные импульсы той же частоты. Микроконтроллер поддерживает данный режим работы, при этом вывод *XL*2 необходимо подключить к общему проводу.

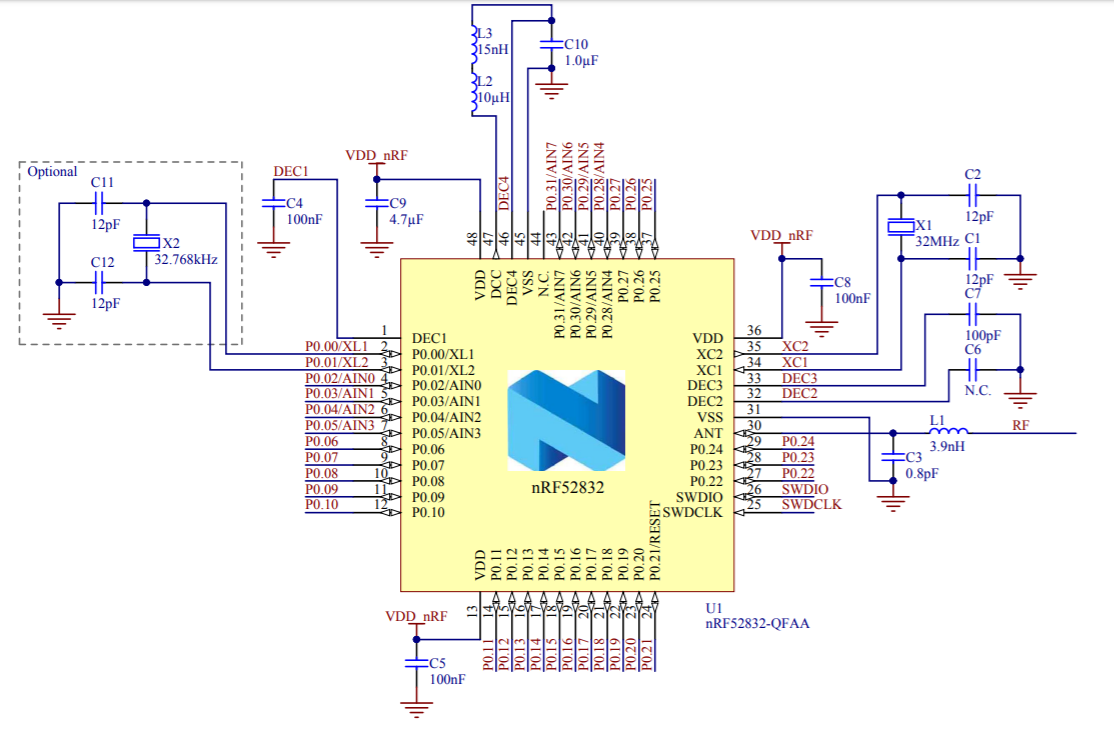


Рисунок 3.1 – Рекомендуемая схема подключения *nRF52832*

Микросхемы часов реального времени *M41T62LC6F* и акселерометра *LIS3DHTR* подключатся к источнику питания согласно рекомендациям производителя, а также к шине *I*2*C*, которая подключена к источнику питания через подтягивающие резисторы.

Излучатель пьезокерамический *PKMCS0909E4000*-*R* подключается к микроконтроллеру через транзистор во избежание превышения максимального тока через вывод микроконтроллера, а также во избежание повреждения микроконтроллера в результате прямого пьезоэффекта.

Дисплей на электронных чернилах *Waveshare* *EPD* *2.13B* подключается к питанию через рекомендуемую производителем схему, а линии данных подключаются напрямую к микроконтроллеру.

## Анализ и обоснование принципиальной электрической схемы питания электронного ценника

Так как все составные части устройства требуют напряжения питания в диапазоне 1.7 – 3.6 В, а рабочий диапазон батареи *CR2032* составляет 1.7 – 3.2 В, то отсутствует необходимость в дополнительных преобразователях для обеспечения питания устройства.

Для оценки уровня заряда аккумулятора используется встроенный в микроконтроллер аналогово-цифровой преобразователь.

# Разработка модели и алгоритма функционирования электронного ценника с радиоканалом связи

Устройство работает в двух основных режимах: основной режим, и режим пониженного энергопотребления, в котором микроконтроллер находится в отключенном состоянии.

После установки батареи устройство инициализируется и переходит в режим спаривания с базовой станцией: объявляет о себе по протоколу *BLE* и выводит на дисплей код для установки соединения, а затем ожидает соединения от базовой станции.

После установки соединения по протоколу *BLE*, устройство производит синхронизацию времени с базовой станцией, и переходит в режим нормальной работы.

В режиме нормальной работы устройство периодически обменивается с базовой станцией пакетами данных о состоянии устройства. В случае необходимости базовая станция передает на устройство данные, которые необходимо вывести на дисплей, пьезокерамический излучатель либо светодиодный индикатор.

Для увеличения времени работы от одной батареи предусмотрен режим пониженного энергопотребления, в котором устройство полностью отключается, за исключением часов реального времени, и на часах реального времени устанавливается таймер пробуждения. Подразумевается, что устройство находится в рабочем режиме во время работы условного магазина и отключается, когда магазин закрыт. Несмотря на то, что устройство отключено, информация на *EPD*-дисплее сохраняется.

Данный алгоритм отображен в схеме работы устройства, а соответствующие состояния на диаграмме состояний.

# Разработка конструкции проектируемого изделия

## Выбор и обоснование элементной базы

Выбор элементной базы проводится на основе схемы электрической принципиальной с учетом изложенных в техническом задании условий и требований. Эксплуатационная надежность элементной базы в основном определяется правильным выбором типа элементов при проектировании и при использовании в режимах, которые не превышают предельно допустимые.

Элементы обвязки, такие как конденсаторы и резисторы выбраны в *SMD* корпусах с типоразмерами 0402, 0201, 0603. Применение компонентов поверхностного монтажа заметно уменьшает размеры готовых печатных плат, уменьшается их вес, как следствие для этого устройства потребуется небольшой компактный корпус. Так можно собрать очень компактные и миниатюрные устройства.

## Выбор и обоснование конструктивных элементов и установочных изделий

Основные конструктивные элементы печатной платы – основание (подложка) и проводники. Круг второстепенных элементов несколько шире: контактные площадки, переходные металлизированные и монтажные отверстия, ламели для контактирования с разъемами, участки для осуществления теплоотвода и прочее.

В отличие от навесного монтажа, на печатной плате электропроводящий рисунок выполнен из фольги, целиком расположенной на твердой изолирующей основе. Печатная плата содержит монтажные отверстия и контактные площадки для монтажа выводных или планарных компонентов. Кроме того, в печатных платах имеются переходные отверстия для электрического соединения участков фольги, расположенных на разных слоях платы. С внешних сторон на плату обычно нанесены защитное покрытие («паяльная маска») и маркировка (вспомогательный рисунок и текст согласно конструкторской документации).

Основой печатной платы служит диэлектрик, наиболее часто используются такие материалы, как стеклотекстолит и гетинакс.

Также основой печатных плат может служить металлическое основание, покрытое диэлектриком (например, анодированный алюминий), поверх диэлектрика наносится медная фольга дорожек. Такие печатные платы применяются в силовой электронике для эффективного теплоотвода от электронных компонентов. При этом металлическое основание платы крепится к радиатору.

Проводники на печатной плате соответствуют классу точности 4. Обычные проводники имеют ширину 0.2 мм. Это связано с тем, что некоторые элементы на плате имеют довольно маленькую ширину контактных площадок, что при ширине проводников в более, чем 0.2 мм, невозможно будет произвести трассировку печатной платы.

При трассировке печатной платы было сделано 55 переходных металлизированных отверстий. Это связано с отсутствием или невозможностью провести проводник на одной стороне печатной платы, а также применением инструмента «*Vie* *stitching*» для улучшения связности силовых полигонов на обеих сторонах печатной платы.

Ширина проводников, связанных с питанием, нескольким больше, чем обычные проводники, поскольку через них течет больший ток, по сравнению с обычными проводниками.

# Расчёт конструктивно-технологических параметров проектируемого изделия

## Проектирование печатного модуля

Для проектирования устройства выбрана двусторонняя печатная плата – печатная плата, на обеих сторонах которой выполнены элементы проводящего рисунка и все требуемые соединения, в соответствии с электрической принципиальной схемой. Электрическая связь между сторонами осуществляется с помощью металлизированных отверстий. Преимущество двухсторонних печатных плат в наличии двух проводящих слоев, что позволяет плотнее компоновать элементы и увеличивает трассировочную способность печатной платы.

Изготовление печатных плат определенного класса точности (ГОСТ 23751-86) обеспечивают, применяя различные техническое оснащение и вспомогательные материалы. Печатные платы 1 и 2 классов точности наиболее просты в исполнении, надежны в эксплуатации и имеют минимальную стоимость; 3 класса – требуют использования высококачественных материалов, более точного инструмента и оборудования; 4 и 5 классов – специальных материалов, прецизионного оборудования, особых условий для изготовления.

Наименьшие номинальные значения основных размеров элементов конструкции печатной платы для узкого места в зависимости от классов точности приведены в таблице 6.1.1.

Таблица 6.1.1 – Классы точности печатных плат

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Условное обозначение | Номинальное значение основных размеров для класса точности | | | | |
| 1 | 2 | 3 | 4 | 5 |
| *t*, мм | 0,75 | 0,45 | 0,25 | 0,15 | 0,10 |
| *S*, мм | 0,75 | 0,45 | 0,25 | 0,15 | 0,10 |
| *b*, мм | 0,30 | 0,20 | 0,10 | 0,05 | 0,025 |
| *g*\* | 0,40 | 0,40 | 0,33 | 0,25 | 0,20 |

Исходя из чертежа печатной платы, выбран класс точности 4.

В соответствии с ГОСТ 10317-79, основной шаг координатной сети должен составлять 2.5 мм. При использовании шага координатной сетки менее основного, следует применять шаг, равный 1.25 мм или 0.625 мм. В нашем случае применялся шаг 0.625 мм, поскольку для таких размеров печатной платы этот шаг считается наиболее приемлемым.

При изготовлении печатной платы был выбран комбинированный позитивный метод. Выбор на этот метод изготовления пал потому, что в нем могут присутствовать металлизированные отверстия, он подходит для двусторонних печатных плат, обладает рядом преимуществ и малым количеством недостатков, а самое главное – минимальная ширина проводника должна составлять 0.1 мм, что подходит для данной печатной платы.

Основные параметры разработанной печатной платы:

* габаритные размеры: 65 × 30 мм;
* толщина печатной платы: 0.71 мм.

Таким образом были описаны основные параметры печатной платы для того, чтобы отдавать ее в производство.

## Выбор и обоснование материалов конструкции и защитных покрытий, маркировки деталей и сборочных единиц

В качестве материала для производства печатной платы выбран стеклотекстолит марки *FR*4 толщиной 0,71 мм с двухсторонним фольгированным слоем и толщиной печатного проводника равной 35 мкм для изготовления двухсторонних печатных плат. Достаточно малая толщина 0,71 мм обусловлена необходимостью соблюдения согласования импеданса проводников в высокочастотном радиотракте устройства.

Стеклотекстолит *FR*4 полностью соответствует ГОСТ 26246.5-89.

Для защиты поверхности платы и медных участков, не подлежащих нанесению финишного покрытия, на плату наносится защитная паяльная маска. Наиболее широко распространена жидкая двухкомпонентная фоточувствительная паяльная маска [8].

Для идентификации монтируемых компонентов большинство изготавливаемых печатных плат имеют маркировку. Маркировка наносится после проявления маски.

На открытые от маски участки меди наносится финишное покрытие для обеспечения качественной пайки. Среди наиболее распространенных защитных покрытий будем использовать *HASL* (*Hot* *Air* *Solder* *Leveling*) – покрытие припоем с выравниванием. Процесс заключается в погружении платы в емкость с расплавом олова и свинца с последующим удалением излишков пасты при помощи «воздушных ножей», которые обдувают горячим воздухом поверхность платы. Одно из очевидных преимуществ технологии *HASL* заключается в том, что плата подвергается температурному воздействию в 265°*C*, которое выявит такую проблему как расслоение, до того, как компоненты будут напаяны на плату.

# Применение средств автоматизированного проектирования при разработке устройства

В данном курсовом проекте использовалась САПР *Altium* *Disigner* *c* расширением *Board* *Assistant* для оформления конструкторской документации в соответствии со стандартами ЕСКД.

С помощью САПР *Altium* *Designer* были спроектированы библиотечные элементы для каждого компонента на принципиальной схеме, которые были найдены предварительно, далее созданы посадочные места, содержащие в себе контактные площадки либо отверстия, в зависимости от компонента, а также подключенные *STEP*-модели в 3*D*.

После сопоставления библиотечных элементов и посадочных мест была спроектирована принципиальная схема устройства.

Далее была создана печатная плата, заданы ее размеры и форма, и размещены компоненты.

Последним шагом разработки печатной платы являлась ее трассировка – процесс прокладывания проводящих дорожек на поверхности платы.

Во избежание ошибок и выхода за пределы заданного класса точности были настроены правила разработки печатной платы, которые задают ряд ограничений и не позволяют произвести размещение компонентов и трассировку дорожек, не соответствующих стандартам и технологическим нормам.

# Заключение

В ходе курсового проектирования была спроектирована плата мобильного устройства – электронного ценника с радиоканалом связи по протоколу *BLE*.

Первым этапом была разработка принципиальной схемы устройства и поиск необходимых компонентов.

Второй этап – разработка печатной платы. На этом этапе была создана библиотека используемых компонентов.

Далее была разработана вся необходимая документация по устройству: чертеж принципиальной схемы, структурная схема устройства, чертеж печатной платы, сборочный чертеж, перечень элементов, спецификация.

Все вышеуказанные этапы были полностью проделаны в САПР *Altium* *Designer* с расширением *Board* *Assistant*, формирующим документацию по ЕСКД.

Была создана программная часть для демонстрационных целей на языке программирования *C*.

В результате была спроектирована печатная плата размером 65 × 30 мм, со всей необходимой документацией для запуска в серийное производство.

Таким образом цели и задачи, поставленные в начале разработки можно считать выполненными.

# Список использованных источников

1. *nRF*52832 [Электронный ресурс]  / Ультран Электронные компоненты – Режим доступа : *http*://*ultran*.*ru*/*nrf*52832 – Дата доступа : 05.05.2020.
2. Последовательный интерфейс *SPI* [Электронный ресурс]. – Режим доступа : *http*://*www*.*gaw*.*ru*/*html*.*cgi*/*txt*/*interface*/*spi*/. Дата доступа: 05.05.2020.
3. Микросхемы часов реального времени (*RTC*) от *STMicroelectronics* [Электронный ресурс] / Компэл. – Режим доступа : *https*://*www*.*compel*.*ru*/*lib*/53975. Дата доступа: 05.05.2020.
4. Акселерометры *STMicroelectronics*: определяя любое движение [Электронный ресурс] / Компэл. – Режим доступа : *https*://*www*.*compel*.*ru*/*lib*/60820. Дата доступа : 05.05.2020.
5. *Altium* *Designer* 20 [Электронный ресурс] / *Altium* – Режим доступа : *https*://*www*.*altium*.*com*.*au*/*ru*/*altium*-*designer*/*features*. Дата доступа: 05.05.2020.
6. *Board* *Assistant* КД по ЕСКД в *Altium* [Электронный ресурс] / Главкон – Режим доступа : *https*://*glavkon*.*com*/*boas*. Дата доступа : 05.05.2020.
7. Библиотека компонентов для *Altium* *Designer* [Электронный ресурс] / *Gardarica* – Режим доступа : *https*://*github*.*com*/*gardarica*/*altium*-*library* – Дата доступа : 05.05.2020.
8. Технология производства печатных плат в картинках [Электронный ресурс] / Резонит – Режим доступа : *https*://*www*.*rezonit*.*ru*/*directory*/*baza*-*znaniy*/*tekhnologiya*-*izgotovleniya*-*pechatnykh*-*plat*-*v*-*kartinkakh*/*dvustoronnie*-*pechatnye*-*plat*/. Дата доступа : 05.05.2020.
9. Конструирование и технология электронных систем: пособие к курсовому проектированию для студ. спец. «Электронно-оптические системы и технологии» всех форм обуч. / А. А. Костюкевич [и др.] – Минск : БГУИР, 2011.

# (обязательное) Визуализированная трехмерная модель устройства

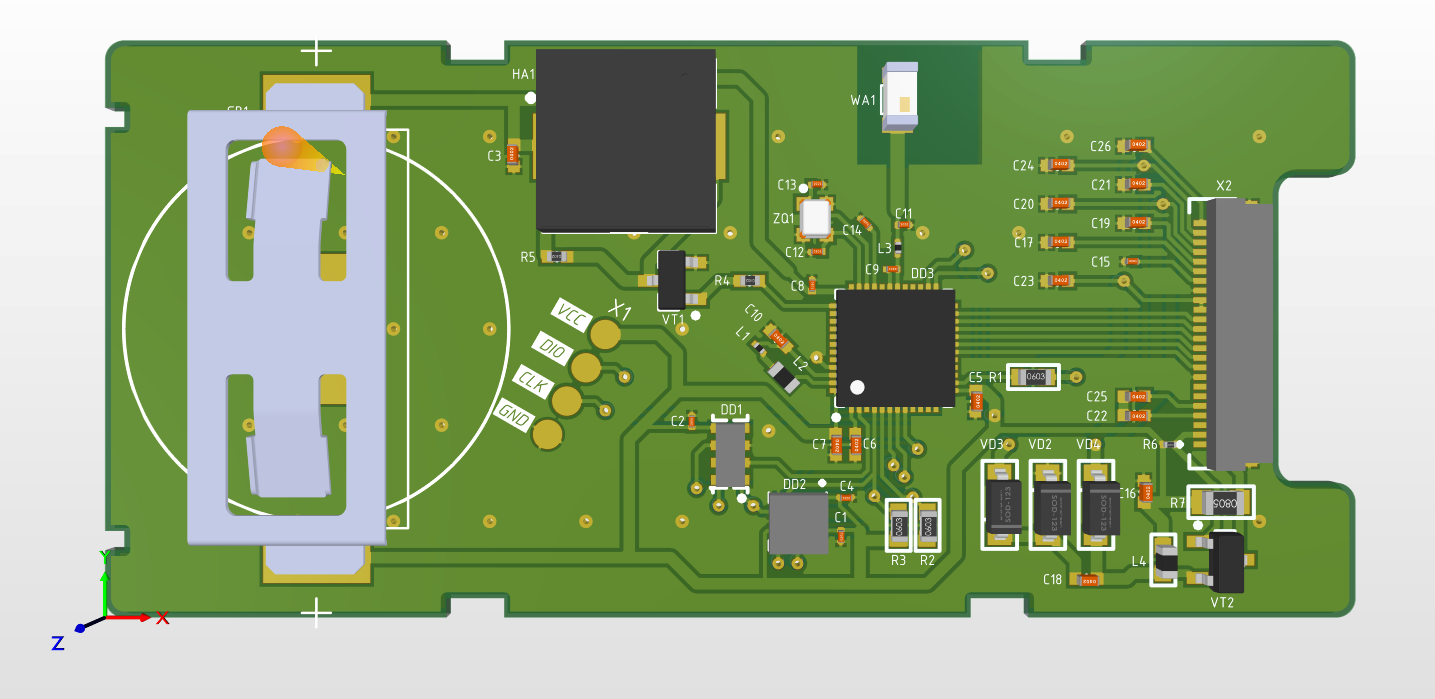


Рисунок А.1 – Лицевая сторона печатной платы

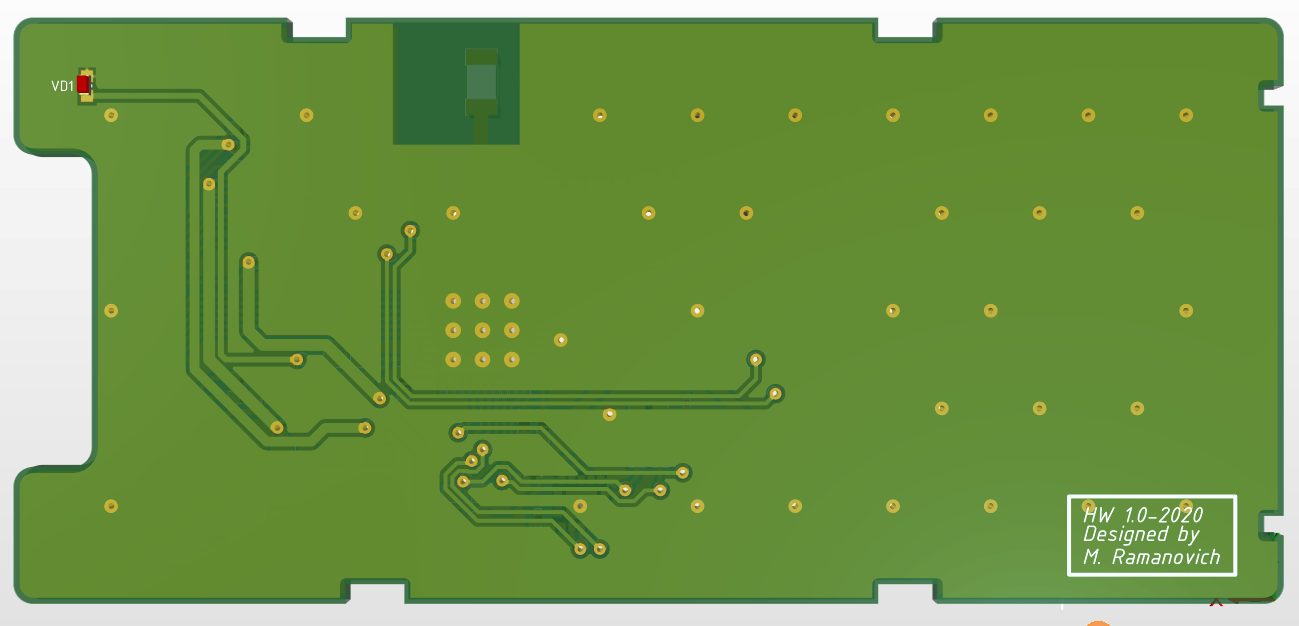


Рисунок А.2 – Тыльная сторона печатной платы

# (обязательное) Листинги программного кода

static uint16\_t m\_conn\_handle = BLE\_CONN\_HANDLE\_INVALID; /\*\*< Handle of the current connection. \*/

static nrf\_ble\_gatt\_t m\_gatt; /\*\*< GATT module instance. \*/

static ble\_eink\_t m\_eink;

static uint8\_t disptxt\_buffer[16] = "Hello Epaper\0\0\0\0";

static uint8\_t epaper\_pending;

static const nrf\_lcd\_t \* p\_lcd = &nrf\_lcd\_wsepd154;

extern const nrf\_gfx\_font\_desc\_t orkney\_24ptFontInfo;

static const nrf\_gfx\_font\_desc\_t \* p\_font = &orkney\_24ptFontInfo;

void assert\_nrf\_callback(uint16\_t line\_num, const uint8\_t \* p\_file\_name)

{

app\_error\_handler(DEAD\_BEEF, line\_num, p\_file\_name);

}

static void leds\_init(void)

{

// Only using LED2 for debug purposes

nrf\_gpio\_cfg\_output(19);

nrf\_gpio\_pin\_set(19);

}

static void timers\_init(void)

{

// Initialize timer module, making it use the scheduler

ret\_code\_t err\_code = app\_timer\_init();

APP\_ERROR\_CHECK(err\_code);

}

static void gap\_params\_init(void)

{

ret\_code\_t err\_code;

ble\_gap\_conn\_params\_t gap\_conn\_params;

ble\_gap\_conn\_sec\_mode\_t sec\_mode;

BLE\_GAP\_CONN\_SEC\_MODE\_SET\_OPEN(&sec\_mode);

err\_code = sd\_ble\_gap\_device\_name\_set(&sec\_mode,

(const uint8\_t \*)DEVICE\_NAME,

strlen(DEVICE\_NAME));

APP\_ERROR\_CHECK(err\_code);

memset(&gap\_conn\_params, 0, sizeof(gap\_conn\_params));

gap\_conn\_params.min\_conn\_interval = MIN\_CONN\_INTERVAL;

gap\_conn\_params.max\_conn\_interval = MAX\_CONN\_INTERVAL;

gap\_conn\_params.slave\_latency = SLAVE\_LATENCY;

gap\_conn\_params.conn\_sup\_timeout = CONN\_SUP\_TIMEOUT;

err\_code = sd\_ble\_gap\_ppcp\_set(&gap\_conn\_params);

APP\_ERROR\_CHECK(err\_code);

}

static void gatt\_init(void)

{

ret\_code\_t err\_code = nrf\_ble\_gatt\_init(&m\_gatt, NULL);

APP\_ERROR\_CHECK(err\_code);

}

static void advertising\_init(void)

{

ret\_code\_t err\_code;

ble\_advdata\_t advdata;

ble\_advdata\_t scanrsp;

ble\_uuid\_t adv\_uuids[] = {{EINK\_UUID\_SERVICE, m\_eink.uuid\_type}};

// Build and set advertising data

memset(&advdata, 0, sizeof(advdata));

advdata.name\_type = BLE\_ADVDATA\_FULL\_NAME;

advdata.include\_appearance = true;

advdata.flags = BLE\_GAP\_ADV\_FLAGS\_LE\_ONLY\_GENERAL\_DISC\_MODE;

memset(&scanrsp, 0, sizeof(scanrsp));

scanrsp.uuids\_complete.uuid\_cnt = sizeof(adv\_uuids) / sizeof(adv\_uuids[0]);

scanrsp.uuids\_complete.p\_uuids = adv\_uuids;

err\_code = ble\_advdata\_set(&advdata, &scanrsp);

APP\_ERROR\_CHECK(err\_code);

}

static void epaper\_demo\_text(void)

{

APP\_ERROR\_CHECK(nrf\_gfx\_init(p\_lcd));

nrf\_gfx\_rotation\_set(p\_lcd, NRF\_LCD\_ROTATE\_270);

nrf\_gfx\_point\_t text\_start = NRF\_GFX\_POINT(15, 70);

nrf\_gfx\_screen\_fill(p\_lcd, 0xff);

APP\_ERROR\_CHECK(nrf\_gfx\_print(p\_lcd, &text\_start, 0x00, (const char \*)disptxt\_buffer, p\_font, true));

nrf\_gfx\_display(p\_lcd);

nrf\_gfx\_uninit(p\_lcd);

}

static void epaper\_demo\_draw(void)

{

APP\_ERROR\_CHECK(nrf\_gfx\_init(p\_lcd));

nrf\_gfx\_rotation\_set(p\_lcd, NRF\_LCD\_ROTATE\_270);

nrf\_gfx\_point\_t text\_start = NRF\_GFX\_POINT(15, 70);

nrf\_gfx\_screen\_fill(p\_lcd, 0x00);

APP\_ERROR\_CHECK(nrf\_gfx\_print(p\_lcd, &text\_start, 0xff, DEVICE\_NAME, p\_font, true));

nrf\_gfx\_circle\_t circ\_1 = NRF\_GFX\_CIRCLE(25, 25, 10);

nrf\_gfx\_circle\_draw(p\_lcd, &circ\_1, 0xff, false);

circ\_1.x = 125;

nrf\_gfx\_circle\_draw(p\_lcd, &circ\_1, 0xff, true);

nrf\_gfx\_display(p\_lcd);

nrf\_gfx\_uninit(p\_lcd);

}

static void epaper\_demo\_imarray(void)

{

APP\_ERROR\_CHECK(nrf\_gfx\_init(p\_lcd));

wsepd154\_draw\_monobmp((const uint8\_t \*)gImage\_1in54);

nrf\_gfx\_display(p\_lcd);

nrf\_gfx\_uninit(p\_lcd);

}

static void demoscr\_write\_handler(uint16\_t conn\_handle, ble\_eink\_t \* p\_lbs, uint8\_t scrnum)

{

switch(scrnum)

{

case 0:

// Trigger writing of display text

epaper\_pending = 1;

break;

case 1:

// Trigger draw primitives

epaper\_pending = 2;

break;

case 2:

// Trigger image display

epaper\_pending = 3;

break;

case 32:

nrf\_gpio\_pin\_set(19);

NRF\_LOG\_INFO("Received LED OFF!\r\n");

break;

case 33:

nrf\_gpio\_pin\_clear(19);

NRF\_LOG\_INFO("Received LED ON!\r\n");

break;

case 64:

// Jump to bootloader

NRF\_LOG\_INFO("Setting DFU flag and disconnecting\r\n");

sd\_ble\_gap\_disconnect(m\_conn\_handle, BLE\_HCI\_CONN\_INTERVAL\_UNACCEPTABLE);

epaper\_pending = 255;

break;

}

}

static void disptxt\_write\_handler(uint16\_t conn\_handle, ble\_eink\_t \* p\_eink, uint8\_t len, uint8\_t \*data)

{

if((data[0] == 0xB0) && (data[1] == 0x07))

{

// Jump to DFU

NRF\_LOG\_INFO("Setting DFU flag and disconnecting\r\n");

sd\_ble\_gap\_disconnect(m\_conn\_handle, BLE\_HCI\_CONN\_INTERVAL\_UNACCEPTABLE);

epaper\_pending = 255;

}

else

{

char str\_data[8];

memcpy(str\_data, data, 8);

// Print received bytes as string

NRF\_LOG\_INFO("Received String: %s\r\n", nrf\_log\_push(str\_data));

}

}

static void services\_init(void)

{

ret\_code\_t err\_code;

ble\_eink\_init\_t init;

init.disptxt\_write\_handler = disptxt\_write\_handler;

init.demoscr\_write\_handler = demoscr\_write\_handler;

init.disptxt\_buffer = disptxt\_buffer;

err\_code = einksvc\_init(&m\_eink, &init);

APP\_ERROR\_CHECK(err\_code);

}

static void on\_conn\_params\_evt(ble\_conn\_params\_evt\_t \* p\_evt)

{

ret\_code\_t err\_code;

if (p\_evt->evt\_type == BLE\_CONN\_PARAMS\_EVT\_FAILED)

{

err\_code = sd\_ble\_gap\_disconnect(m\_conn\_handle, BLE\_HCI\_CONN\_INTERVAL\_UNACCEPTABLE);

APP\_ERROR\_CHECK(err\_code);

}

}

static void conn\_params\_error\_handler(uint32\_t nrf\_error)

{

APP\_ERROR\_HANDLER(nrf\_error);

}

static void conn\_params\_init(void)

{

ret\_code\_t err\_code;

ble\_conn\_params\_init\_t cp\_init;

memset(&cp\_init, 0, sizeof(cp\_init));

cp\_init.p\_conn\_params = NULL;

cp\_init.first\_conn\_params\_update\_delay = FIRST\_CONN\_PARAMS\_UPDATE\_DELAY;

cp\_init.next\_conn\_params\_update\_delay = NEXT\_CONN\_PARAMS\_UPDATE\_DELAY;

cp\_init.max\_conn\_params\_update\_count = MAX\_CONN\_PARAMS\_UPDATE\_COUNT;

cp\_init.start\_on\_notify\_cccd\_handle = BLE\_GATT\_HANDLE\_INVALID;

cp\_init.disconnect\_on\_fail = false;

cp\_init.evt\_handler = on\_conn\_params\_evt;

cp\_init.error\_handler = conn\_params\_error\_handler;

err\_code = ble\_conn\_params\_init(&cp\_init);

APP\_ERROR\_CHECK(err\_code);

}

static void advertising\_start(void)

{

ret\_code\_t err\_code;

ble\_gap\_adv\_params\_t adv\_params;

// Start advertising

memset(&adv\_params, 0, sizeof(adv\_params));

adv\_params.type = BLE\_GAP\_ADV\_TYPE\_ADV\_IND;

adv\_params.p\_peer\_addr = NULL;

adv\_params.fp = BLE\_GAP\_ADV\_FP\_ANY;

adv\_params.interval = APP\_ADV\_INTERVAL;

adv\_params.timeout = APP\_ADV\_TIMEOUT\_IN\_SECONDS;

err\_code = sd\_ble\_gap\_adv\_start(&adv\_params, BLE\_CONN\_CFG\_TAG\_DEFAULT);

APP\_ERROR\_CHECK(err\_code);

bsp\_board\_led\_on(ADVERTISING\_LED);

}

static void on\_ble\_evt(ble\_evt\_t \* p\_ble\_evt)

{

ret\_code\_t err\_code;

switch (p\_ble\_evt->header.evt\_id)

{

case BLE\_GAP\_EVT\_CONNECTED:

NRF\_LOG\_INFO("Connected\r\n");

bsp\_board\_led\_on(CONNECTED\_LED);

bsp\_board\_led\_off(ADVERTISING\_LED);

m\_conn\_handle = p\_ble\_evt->evt.gap\_evt.conn\_handle;

break; // BLE\_GAP\_EVT\_CONNECTED

case BLE\_GAP\_EVT\_DISCONNECTED:

NRF\_LOG\_INFO("Disconnected\r\n");

bsp\_board\_led\_off(CONNECTED\_LED);

m\_conn\_handle = BLE\_CONN\_HANDLE\_INVALID;

if(epaper\_pending == 255)

{

NRF\_LOG\_INFO("Doing system soft reset\r\n");

NRF\_POWER->GPREGRET = BOOTLOADER\_DFU\_START;

NVIC\_SystemReset();

}

else

{

advertising\_start();

}

break; // BLE\_GAP\_EVT\_DISCONNECTED

case BLE\_GAP\_EVT\_SEC\_PARAMS\_REQUEST:

err\_code = sd\_ble\_gap\_sec\_params\_reply(m\_conn\_handle,

BLE\_GAP\_SEC\_STATUS\_PAIRING\_NOT\_SUPP,

NULL,

NULL);

APP\_ERROR\_CHECK(err\_code);

break; // BLE\_GAP\_EVT\_SEC\_PARAMS\_REQUEST

case BLE\_GATTS\_EVT\_SYS\_ATTR\_MISSING:

// No system attributes have been stored.

err\_code = sd\_ble\_gatts\_sys\_attr\_set(m\_conn\_handle, NULL, 0, 0);

APP\_ERROR\_CHECK(err\_code);

break; // BLE\_GATTS\_EVT\_SYS\_ATTR\_MISSING

case BLE\_GATTC\_EVT\_TIMEOUT:

// Disconnect on GATT Client timeout event.

NRF\_LOG\_DEBUG("GATT Client Timeout.\r\n");

err\_code = sd\_ble\_gap\_disconnect(p\_ble\_evt->evt.gattc\_evt.conn\_handle,

BLE\_HCI\_REMOTE\_USER\_TERMINATED\_CONNECTION);

APP\_ERROR\_CHECK(err\_code);

break; // BLE\_GATTC\_EVT\_TIMEOUT

case BLE\_GATTS\_EVT\_TIMEOUT:

// Disconnect on GATT Server timeout event.

NRF\_LOG\_DEBUG("GATT Server Timeout.\r\n");

err\_code = sd\_ble\_gap\_disconnect(p\_ble\_evt->evt.gatts\_evt.conn\_handle,

BLE\_HCI\_REMOTE\_USER\_TERMINATED\_CONNECTION);

APP\_ERROR\_CHECK(err\_code);

break; // BLE\_GATTS\_EVT\_TIMEOUT

case BLE\_EVT\_USER\_MEM\_REQUEST:

err\_code = sd\_ble\_user\_mem\_reply(p\_ble\_evt->evt.gattc\_evt.conn\_handle, NULL);

APP\_ERROR\_CHECK(err\_code);

break; // BLE\_EVT\_USER\_MEM\_REQUEST

case BLE\_GATTS\_EVT\_RW\_AUTHORIZE\_REQUEST:

{

ble\_gatts\_evt\_rw\_authorize\_request\_t req;

ble\_gatts\_rw\_authorize\_reply\_params\_t auth\_reply;

req = p\_ble\_evt->evt.gatts\_evt.params.authorize\_request;

if (req.type != BLE\_GATTS\_AUTHORIZE\_TYPE\_INVALID)

{

if ((req.request.write.op == BLE\_GATTS\_OP\_PREP\_WRITE\_REQ) ||

(req.request.write.op == BLE\_GATTS\_OP\_EXEC\_WRITE\_REQ\_NOW) ||

(req.request.write.op == BLE\_GATTS\_OP\_EXEC\_WRITE\_REQ\_CANCEL))

{

if (req.type == BLE\_GATTS\_AUTHORIZE\_TYPE\_WRITE)

{

auth\_reply.type = BLE\_GATTS\_AUTHORIZE\_TYPE\_WRITE;

}

else

{

auth\_reply.type = BLE\_GATTS\_AUTHORIZE\_TYPE\_READ;

}

auth\_reply.params.write.gatt\_status = APP\_FEATURE\_NOT\_SUPPORTED;

err\_code = sd\_ble\_gatts\_rw\_authorize\_reply(p\_ble\_evt->evt.gatts\_evt.conn\_handle,

&auth\_reply);

APP\_ERROR\_CHECK(err\_code);

}

}

} break; // BLE\_GATTS\_EVT\_RW\_AUTHORIZE\_REQUEST

default:

// No implementation needed.

break;

}

}

static void ble\_evt\_dispatch(ble\_evt\_t \* p\_ble\_evt)

{

on\_ble\_evt(p\_ble\_evt);

ble\_conn\_params\_on\_ble\_evt(p\_ble\_evt);

ble\_eink\_on\_ble\_evt(&m\_eink, p\_ble\_evt);

nrf\_ble\_gatt\_on\_ble\_evt(&m\_gatt, p\_ble\_evt);

}

static void ble\_stack\_init(void)

{

ret\_code\_t err\_code;

nrf\_clock\_lf\_cfg\_t clock\_lf\_cfg = NRF\_CLOCK\_LFCLKSRC;

// Initialize the SoftDevice handler module.

SOFTDEVICE\_HANDLER\_INIT(&clock\_lf\_cfg, NULL);

// Fetch the start address of the application RAM.

uint32\_t ram\_start = 0;

err\_code = softdevice\_app\_ram\_start\_get(&ram\_start);

APP\_ERROR\_CHECK(err\_code);

// Overwrite some of the default configurations for the BLE stack.

ble\_cfg\_t ble\_cfg;

memset(&ble\_cfg, 0, sizeof(ble\_cfg));

ble\_cfg.gap\_cfg.role\_count\_cfg.periph\_role\_count = BLE\_GAP\_ROLE\_COUNT\_PERIPH\_DEFAULT;

ble\_cfg.gap\_cfg.role\_count\_cfg.central\_role\_count = 0;

ble\_cfg.gap\_cfg.role\_count\_cfg.central\_sec\_count = 0;

err\_code = sd\_ble\_cfg\_set(BLE\_GAP\_CFG\_ROLE\_COUNT, &ble\_cfg, ram\_start);

APP\_ERROR\_CHECK(err\_code);

// Enable BLE stack.

err\_code = softdevice\_enable(&ram\_start);

APP\_ERROR\_CHECK(err\_code);

// Subscribe for BLE events.

err\_code = softdevice\_ble\_evt\_handler\_set(ble\_evt\_dispatch);

APP\_ERROR\_CHECK(err\_code);

}

static void log\_init(void)

{

ret\_code\_t err\_code = NRF\_LOG\_INIT(NULL);

APP\_ERROR\_CHECK(err\_code);

}

/\*\*@brief Function for the Power Manager.

\*/

static void power\_manage(void)

{

ret\_code\_t err\_code = sd\_app\_evt\_wait();

APP\_ERROR\_CHECK(err\_code);

switch(epaper\_pending)

{

case 1:

NRF\_LOG\_INFO("Starting to display text\r\n");

epaper\_demo\_text();

epaper\_pending = 0;

break;

case 2:

NRF\_LOG\_INFO("Starting to draw demo\r\n");

epaper\_demo\_draw();

epaper\_pending = 0;

break;

case 3:

NRF\_LOG\_INFO("Starting to draw image\r\n");

epaper\_demo\_imarray();

epaper\_pending = 0;

default:

break;

}

}

/\*\*@brief Function for application main entry.

\*/

int main(void)

{

// Initialize.

leds\_init();

timers\_init();

log\_init();

ble\_stack\_init();

gap\_params\_init();

gatt\_init();

services\_init();

advertising\_init();

conn\_params\_init();

epaper\_pending = 0;

advertising\_start();

// Enter main loop.

for (;;)

{

if (NRF\_LOG\_PROCESS() == false)

{

power\_manage();

}

}

}

#define DRIVER\_OUTPUT\_CONTROL 0x01

#define BOOSTER\_SOFT\_START\_CONTROL 0x0C

#define GATE\_SCAN\_START\_POSITION 0x0F

#define DEEP\_SLEEP\_MODE 0x10

#define DATA\_ENTRY\_MODE\_SETTING 0x11

#define SW\_RESET 0x12

#define TEMPERATURE\_SENSOR\_CONTROL 0x1A

#define MASTER\_ACTIVATION 0x20

#define DISPLAY\_UPDATE\_CONTROL\_1 0x21

#define DISPLAY\_UPDATE\_CONTROL\_2 0x22

#define WRITE\_RAM 0x24

#define WRITE\_VCOM\_REGISTER 0x2C

#define WRITE\_LUT\_REGISTER 0x32

#define SET\_DUMMY\_LINE\_PERIOD 0x3A

#define SET\_GATE\_TIME 0x3B

#define BORDER\_WAVEFORM\_CONTROL 0x3C

#define SET\_RAM\_X\_ADDRESS\_START\_END\_POSITION 0x44

#define SET\_RAM\_Y\_ADDRESS\_START\_END\_POSITION 0x45

#define SET\_RAM\_X\_ADDRESS\_COUNTER 0x4E

#define SET\_RAM\_Y\_ADDRESS\_COUNTER 0x4F

#define TERMINATE\_FRAME\_READ\_WRITE 0xFF

#define WSEPD\_HEIGHT 200

#define WSEPD\_WIDTH 200

const unsigned char lut\_full\_update[] = {

0x02, 0x02, 0x01, 0x11, 0x12, 0x12, 0x22, 0x22,

0x66, 0x69, 0x69, 0x59, 0x58, 0x99, 0x99, 0x88,

0x00, 0x00, 0x00, 0x00, 0xF8, 0xB4, 0x13, 0x51,

0x35, 0x51, 0x51, 0x19, 0x01, 0x00

};

const unsigned char lut\_partial\_update[] = {

0x10, 0x18, 0x18, 0x08, 0x18, 0x18, 0x08, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x13, 0x14, 0x44, 0x12,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00

};

static lcd\_cb\_t wsepd154\_cb = {

.height = WSEPD\_HEIGHT,

.width = WSEPD\_WIDTH,

};

static const nrf\_drv\_spi\_t spi = NRF\_DRV\_SPI\_INSTANCE(WSEPD\_SPI\_INSTANCE);

uint8\_t screen\_buffer[((WSEPD\_WIDTH % 8 == 0)? (WSEPD\_WIDTH / 8 ): (WSEPD\_WIDTH / 8 + 1)) \* WSEPD\_HEIGHT];

static inline void spi\_write(const void \* data, size\_t size)

{

APP\_ERROR\_CHECK(nrf\_drv\_spi\_transfer(&spi, data, size, NULL, 0));

}

static inline void write\_command(uint8\_t c)

{

nrf\_gpio\_pin\_clear(WSEPD\_DC);

nrf\_gpio\_pin\_clear(WSEPD\_SPI\_CS);

spi\_write(&c, sizeof(c));

nrf\_gpio\_pin\_set(WSEPD\_SPI\_CS);

}

static inline void write\_data(uint8\_t c)

{

nrf\_gpio\_pin\_set(WSEPD\_DC);

nrf\_gpio\_pin\_clear(WSEPD\_SPI\_CS);

spi\_write(&c, sizeof(c));

nrf\_gpio\_pin\_set(WSEPD\_SPI\_CS);

}

static void set\_cursor(uint16\_t x, uint16\_t y)

{

write\_command(SET\_RAM\_X\_ADDRESS\_COUNTER);

write\_data((x >> 3) & 0xff);

write\_command(SET\_RAM\_Y\_ADDRESS\_COUNTER);

write\_data(y & 0xFF);

write\_data((y >> 8) & 0xFF);

}

static void set\_addr\_window(uint16\_t x\_0, uint16\_t y\_0, uint16\_t x\_1, uint16\_t y\_1)

{

ASSERT(x\_0 < x\_1);

ASSERT(y\_0 < y\_1);

write\_command(SET\_RAM\_X\_ADDRESS\_START\_END\_POSITION);

write\_data((x\_0 >> 3) & 0xff);

write\_data((x\_1 >> 3) & 0xff);

write\_command(SET\_RAM\_Y\_ADDRESS\_START\_END\_POSITION);

write\_data(y\_0 & 0xff);

write\_data((y\_0 >> 8) & 0xff);

write\_data(y\_1 & 0xff);

write\_data((y\_1 >> 8) & 0xff);

}

static void wait\_until\_idle(void)

{

NRF\_LOG\_INFO("epd busy\r\n");

while(nrf\_gpio\_pin\_read(WSEPD\_BUSY) == 1) { //LOW: idle, HIGH: busy

nrf\_delay\_ms(100);

}

NRF\_LOG\_INFO("epd busy release\r\n");

}

static void turn\_on\_display(void)

{

write\_command(DISPLAY\_UPDATE\_CONTROL\_2);

write\_data(0xc4);

write\_command(MASTER\_ACTIVATION);

write\_command(TERMINATE\_FRAME\_READ\_WRITE);

wait\_until\_idle();

}

static void wsepd\_reset(void)

{

nrf\_gpio\_pin\_set(WSEPD\_RST);

nrf\_delay\_ms(200);

nrf\_gpio\_pin\_clear(WSEPD\_RST);

nrf\_delay\_ms(200);

nrf\_gpio\_pin\_set(WSEPD\_RST);

nrf\_delay\_ms(200);

}

static void command\_list(void)

{

wsepd\_reset();

write\_command(DRIVER\_OUTPUT\_CONTROL);

write\_data((WSEPD\_HEIGHT - 1) & 0xFF);

write\_data(((WSEPD\_HEIGHT - 1) >> 8) & 0xFF);

write\_data(0x00);

write\_command(BOOSTER\_SOFT\_START\_CONTROL);

write\_data(0xD7);

write\_data(0xD6);

write\_data(0x9D);

write\_command(WRITE\_VCOM\_REGISTER);

write\_data(0xA8);

write\_command(SET\_DUMMY\_LINE\_PERIOD);

write\_data(0x1A);

write\_command(SET\_GATE\_TIME);

write\_data(0x08);

write\_command(DATA\_ENTRY\_MODE\_SETTING);

write\_data(0x03);

write\_command(WRITE\_LUT\_REGISTER);

for(uint8\_t i = 0; i < 30; i++)

{

// TODO: allow partial update mode

write\_data(lut\_full\_update[i]);

}

}

static ret\_code\_t hardware\_init(void)

{

ret\_code\_t err\_code;

nrf\_gpio\_cfg\_output(WSEPD\_DC);

nrf\_gpio\_cfg\_output(WSEPD\_RST);

nrf\_gpio\_cfg\_output(WSEPD\_SPI\_CS);

nrf\_gpio\_cfg\_input(WSEPD\_BUSY, NRF\_GPIO\_PIN\_NOPULL);

nrf\_drv\_spi\_config\_t spi\_config = NRF\_DRV\_SPI\_DEFAULT\_CONFIG;

spi\_config.sck\_pin = WSEPD\_SPI\_CLK;

spi\_config.miso\_pin = WSEPD\_SPI\_MISO;

spi\_config.mosi\_pin = WSEPD\_SPI\_MOSI;

err\_code = nrf\_drv\_spi\_init(&spi, &spi\_config, NULL, NULL);

return err\_code;

}

static ret\_code\_t wsepd154\_init(void)

{

ret\_code\_t err\_code;

err\_code = hardware\_init();

if (err\_code != NRF\_SUCCESS)

{

return err\_code;

}

command\_list();

// Clear screen buffer to white

memset(screen\_buffer, 0xff, sizeof(screen\_buffer));

return 0;

}

static void wsepd154\_uninit(void)

{

ASSERT(wsepd154\_cb.state != NRF\_DRV\_STATE\_UNINITIALIZED);

write\_command(DEEP\_SLEEP\_MODE);

write\_data(0x01);

nrf\_drv\_spi\_uninit(&spi);

}

static void wsepd154\_pixel\_draw(uint16\_t x, uint16\_t y, uint32\_t color)

{

uint16\_t x\_, y\_;

if(x > WSEPD\_WIDTH || y > WSEPD\_HEIGHT){

NRF\_LOG\_INFO("Exceeding display boundaries\r\n");

return;

}

switch(wsepd154\_cb.rotation)

{

case NRF\_LCD\_ROTATE\_90:

x\_ = WSEPD\_WIDTH - y - 1;

y\_ = x;

break;

case NRF\_LCD\_ROTATE\_180:

x\_ = WSEPD\_WIDTH - x - 1;

y\_ = WSEPD\_HEIGHT - y - 1;

break;

case NRF\_LCD\_ROTATE\_270:

x\_ = y;

y\_ = WSEPD\_HEIGHT - x - 1;

break;

default:

x\_ = x;

y\_ = y;

break;

}

if(x\_ > WSEPD\_WIDTH || y\_ > WSEPD\_HEIGHT){

NRF\_LOG\_INFO("Exceeding display boundaries\r\n");

return;

}

uint16\_t addr = x\_ / 8 + y\_ \* ((WSEPD\_WIDTH % 8) == 0 ? WSEPD\_WIDTH / 8 : WSEPD\_WIDTH / 8 + 1);

uint8\_t rdata = screen\_buffer[addr];

if(color == 0)

screen\_buffer[addr] = rdata & ~(0x80 >> (x\_ % 8));

else

screen\_buffer[addr] = rdata | (0x80 >> (x\_ % 8));

}

static void wsepd154\_rect\_draw(uint16\_t x, uint16\_t y, uint16\_t width, uint16\_t height, uint32\_t color)

{

uint16\_t x\_, y\_;

// TODO: This could be done about 8x faster with cheeky bitfield manipulations

// but it's really difficult to get it all perfect given the 6 conditions

for(y\_ = y; y\_ < (y + height); y\_ ++)

{

for(x\_ = x; x\_ < (x + width); x\_ ++)

{

wsepd154\_pixel\_draw(x\_, y\_, color);

}

}

}

static void wsepd154\_display(void)

{

uint16\_t width;

uint32\_t addr = 0;

uint16\_t i, j;

width = (WSEPD\_WIDTH % 8 == 0) ? (WSEPD\_WIDTH / 8) : (WSEPD\_WIDTH / 8 + 1);

set\_addr\_window(0, 0, WSEPD\_WIDTH, WSEPD\_HEIGHT);

for(i = 0; i < WSEPD\_HEIGHT; i++)

{

set\_cursor(0, i);

write\_command(WRITE\_RAM);

for(j = 0; j < width; j++)

{

addr = j + i \* width;

write\_data(screen\_buffer[addr]);

}

}

turn\_on\_display();

}

static void wsepd154\_rotation\_set(nrf\_lcd\_rotation\_t rotation)

{

// dummy function - actions are taken in the drawing primitives

}

static void wsepd154\_display\_invert(bool invert)

{

// dummy function - doesn't have feature built in

}

// Can only draw an image the size of screen or larger

void wsepd154\_draw\_monobmp(const uint8\_t \*image\_buffer)

{

uint16\_t x, y;

uint32\_t addr = 0;

uint8\_t byte\_width = (WSEPD\_WIDTH % 8 == 0) ? (WSEPD\_WIDTH / 8) : (WSEPD\_WIDTH / 8 + 1);

for (y = 0; y < WSEPD\_HEIGHT; y++) {

for (x = 0; x < byte\_width; x++)

{//8 pixel = 1 byte

addr = x + y \* byte\_width;

screen\_buffer[addr] = (uint8\_t)image\_buffer[addr];

}

}

}

const nrf\_lcd\_t nrf\_lcd\_wsepd154 = {

.lcd\_init = wsepd154\_init, // Done

.lcd\_uninit = wsepd154\_uninit, // Done

.lcd\_pixel\_draw = wsepd154\_pixel\_draw, // Done

.lcd\_rect\_draw = wsepd154\_rect\_draw, // TODO: reqd

.lcd\_display = wsepd154\_display,

.lcd\_rotation\_set = wsepd154\_rotation\_set,

.lcd\_display\_invert = wsepd154\_display\_invert,

.p\_lcd\_cb = &wsepd154\_cb

};

# (обязательное) Результат проверки проекта на плагиат

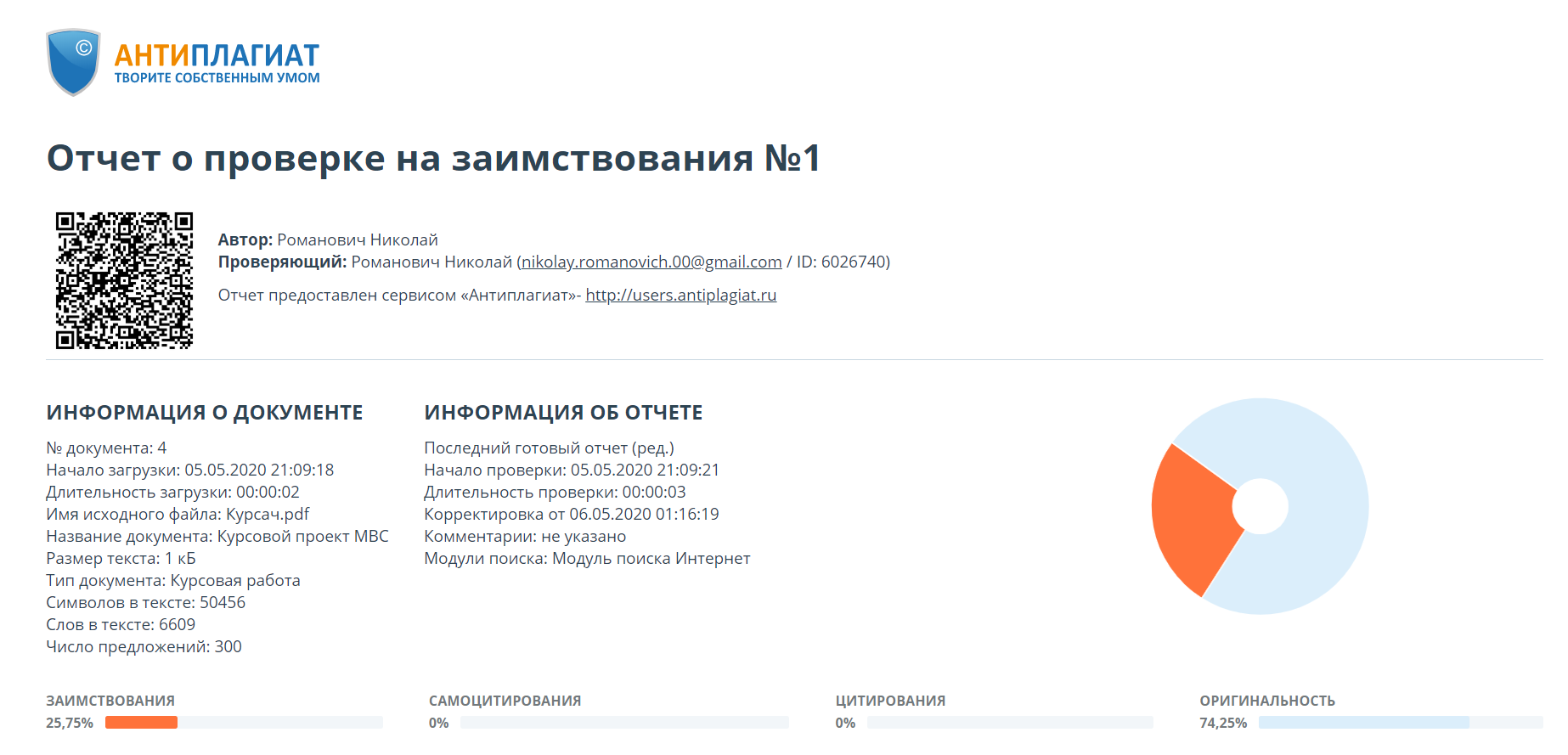


Рисунок В.1 – Отчет проверки на плагиат

# Ведомость документов

ВЫКИНУТЬ ЭТУ СТРАНИЦУ И ВСТАВИТЬ СЮДА ВЕДОМОСТЬ