**Министерство образования Иркутской области**

Государственное бюджетное профессиональное образовательное учреждение

Иркутской области

«Иркутский Авиационный техникум»

(ГБПОУИО «ИАТ»)

|  |
| --- |
| **ДП.09.02.01.21.172.24.ПЗ** УТВЕРЖДАЮ |
| Зам. директора по УР, к.т.н. |
| \_\_\_\_\_\_\_\_\_ Коробкова Е.А. |

ПОРТАТИВНЫЙ МЕДИАПЛЕЕР НА МИКРОКОНТРОЛЛЕРЕ ATMEGA

|  |  |  |
| --- | --- | --- |
| Нормоконтролер: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (А.Э. Кондратенко) |
| Консультант по экономической части | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (А.А. Белова) |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (Д.В. Шатурский) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.В. Шиверский) |

СОДЕРЖАНИЕ

Список условных сокращений 3

Введение 4

1 Общая часть 6

1.1 Обзор и классификация существующих решений 6

1.2 Обоснование создаваемого проекта 9

1.3 Алгоритм работы устройства и обоснование проекта 10

1.4 Выбор элементной базы 11

2 Специальная часть 14

2.1 Проектирование схемы электрической принципиальной 14

2.2 Проектирование печатной платы устройства 27

2.3 Изготовление устройства 31

3. Технологическая часть 39

3.1 Программирование устройства 39

3.2 Техническое рувоводство 41

4 Экономическая часть 43

4.1 Определение полной себестоимости 43

4.2 Расчет заработной платы 45

4.3 Расчет расходов на содержание и эксплуатацию оборудования 47

Заключение 49

Список информационных источников 50

Приложение А – Листинг программного кода 51

СПИСОК УСЛОВНЫХ СОКРАЩЕНИЙ

DIP – dual in-line package.

EDA – electronic design automation.

EEPROM – electrically erasable programmable read-only memory.

I2C(IIC) – inter-integrated circuit.

ISP – in-system programming.

LCD – liquid crystal display.

SIP – single in-line package.

SMD – surface mount device.

UART – universal asynchronous receiver-transmitter.

ЛУТ – лазерно-утюжная технология.

ОУ – операционный усилитель.

ПК – персональный компьютер.

УБО – условное буквенное обозначение.

УГО – условное графическое обозначение.

ШИМ – широтно-импульсная модуляция.

ВВЕДЕНИЕ

Закономерное эволюционирование аудио и видеопроигрывателей привело к появлению медиаплееров. Их главное отличие от своих предшественников – отсутствие движущихся механизмов. Фактически, любой мультимедийный плеер, это полностью цифровое устройство, которое втиснулось в нишу между ноутбуками, смартфонами и телевизорами.

Само понятие «Медиа» в переводе с латинского обозначает посредник. Данный термин активно используется в современном мире для обозначения различных явлений, связанных с передачей информации. В терминологии компьютерных технологий данное понятие связано с многообразием и означает, что предмет имеет несколько функций, режимов или форматов. В частности, к портативным медиаплеерам могут относиться отдельные устройства для воспроизведения звука – аудиаплееры, так же есть устройства, которые могут воспроизводить не только звук, но и видео и даже иметь выход в интернет. Несмотря это, людям привычнее пользоваться смартфонами, так как они уже имеют перечисленные возможности, поэтому портативные видеоплееры не особо распространены, а вот аудио напротив. Дело в том, что они имеют ряд преимуществ, по сравнению со смартфоном, которые делают их более привлекательными:

– размер;

– цена;

– качество звука;

– время автономной работы.

Использование флэш-памяти предоставляет наибольшую свободу разработчикам при создании медиаплееров, ведь в этом случае нет таких жестких ограничений по минимальным габаритам и весу, которые в значительной степени усложняют задачу для создателей аналогичных устройств на базе жестких дисков или оптических приводов.

В первую очередь стоит отметить, что все больше плееров оснащается исключительно встроенной флэш-памятью.

Благодаря продолжающемуся снижению цен на флэш-память увеличился максимальный объем встраиваемой памяти.

Целью дипломного проекта является разработка портативного медиаплеера на микроконтроллере Atmega, согласно заданию.

Задачи дипломного проекта:

– обзор и анализ существующих решений;

– разработка функциональной блок-схемы устройства;

– разработка электрической принципиальной схемы;

– разработка схемы монтажной платы;

– изготовление и программирование устройства;

– экономическое обоснование проекта.

1 Общая часть

1.1 Обзор и классификация существующих решений

Единого стандарта для медиаплееров не существует, поэтому есть базовые функции, которые присутствуют во всех устройствах этой группы. Но большинство производителей стремятся наделить свою продукцию чем-то особенным, или хотя бы максимально расширить стандартный функционал.

1. Стационарные ТВ-приставки. Эта категория устройств, которая применяется для расширения функциональных возможностей телевизоров. Например, для добавления функции, схожей со SmartTV, если изначально данная модель этого не поддерживала. После подключения медиаплеера к телевизору могут добавиться такие функции как:

– возможность подключения USB-носителей;

– возможность просмотра фото и видео, а также прослушивание цифровых аудиофайлов;

– возможность выхода в интернет и другое.

На рисунке 1.1 представлен внешний вид стационарного медиплеера Dune HD SmartBox 4K.



Рисунок 1.1 – Стационарный медиаплеер (ТВ-приставка)

2. Портативные ТВ-приставки. Данные устройства по функционалу не сильно отличаются от предыдущего типа, единственное их отличие это размер и возможность быстрого подключения и отключения. Физически могут напоминать USB флеш-накопитель (рисунок 1.2).



Рисунок 1.2 – Портативная ТВ-приставка

Данное устройство не может работать самостоятельно без телевизора, для этого существует отдельная категория портативный медиаплееров.

3. Портативные медиаплееры. Изначально эта категория устройств в основном поддерживала только формат MP3, поэтому такие устройства назывались MP3-плееры. Что касается современных аудиоплееров, то они могут поддерживать множество различных форматов цифрового звука и могут по праву называться медиаплеерами (рисунок 1.3).



Рисунок 1.3 – Портативный музыкальный медиаплеер

Некоторые производители со временем стали добавлять цветной экран с поддержкой воспроизведения видео и просмотра картинок (рисунок 1.4).



Рисунок 1.4 – Портативный медиаплеер с поддержкой воспроизведения фото и видеофайлов

В настоящее время такие устройства не особо распространены, так как в этом сегменте конкуренцию им составили смартфоны и планшеты с большим функционалом и качеством видео. В связи с этим производители современных медиаплееров сделали упор на качество звука, не акцентирую при этом внимание на видео, так как трудно составить конкуренцию широкоформатному планшету. Такие медиаплееры получили название Hi-Fi, что расшифровывается как «высокая четкость». Увеличения качества стало возможно благодаря применению новых цифро-аналоговых преобразователей с большей частотой дискретизации и разрядностью. Сама технология получила название Hi-Res Audio. Для этого были созданы новые аудиокодеки, например FLAC. В отличие от аудиокодеков, обеспечивающих сжатие с потерями (MP3, AAC, WMA, Ogg Vorbis, Opus), FLAC, как и любой другой lossless-кодек, не удаляет никакой информации из аудиопотока и подходит как для прослушивания музыки на высококачественной звуковоспроизводящей аппаратуре, так и для архивирования аудио коллекции.

Для сравнения: максимальная частота дискретизации MP3 составляет 48КГц, а разрядность 16 бит, при этом у формата FLAC частота может составлять 192КГц с разрядностью 24 бита.

Минусом таких устройств, поддерживающие форматы высокой четкости, является стоимость, в разы превышающая стоимость обычных портативный медиаплееров. На рисунке 1.5. представлен внешний вид Hi-Fi медиаплеера от компании Sony.



Рисунок 1.5 – Hi-Fi медиаплеер

1.2 Обоснование создаваемого проекта

Учитывая многообразие медиаплееров для конкретных задач, можно сказать, что промышленные устройства являются востребованным оборудованием для современного рынка. Несмотря на функционал смартфонов и их растущие характеристики, у медиаплееров присутствует своя аудитория, которой требуются функции отсутствующие в смартфонах, либо реализованы не полностью. К ним относится качество звука, громкость, время автономной работы, защита корпуса и другие.

Медиаплеер достаточное сложное устройство, которое предполагает, как аппаратную, так программную часть, поэтому данную тему можно рассмотреть в качестве дипломного проекта, так как ее реализация непосредственно демонстрирует профессиональные компетенции специалиста компьютерных систем и комплексов.

1.3 Алгоритм работы устройства и обоснование проекта

Учитывая не особую популярность портативных медиаплееров с функцией воспроизведения видео, было принято решение отказаться от данной функции. Чтобы обеспечить возможность прослушивания аудио без наушников, в медиаплеер будет встроен динамик, это будет одной из особенностей данного устройства по сравнению с обычными аудиоплеерами.

На первом этапе проектирования необходимо составить блок-схему будущего устройства (рисунок 1.6).

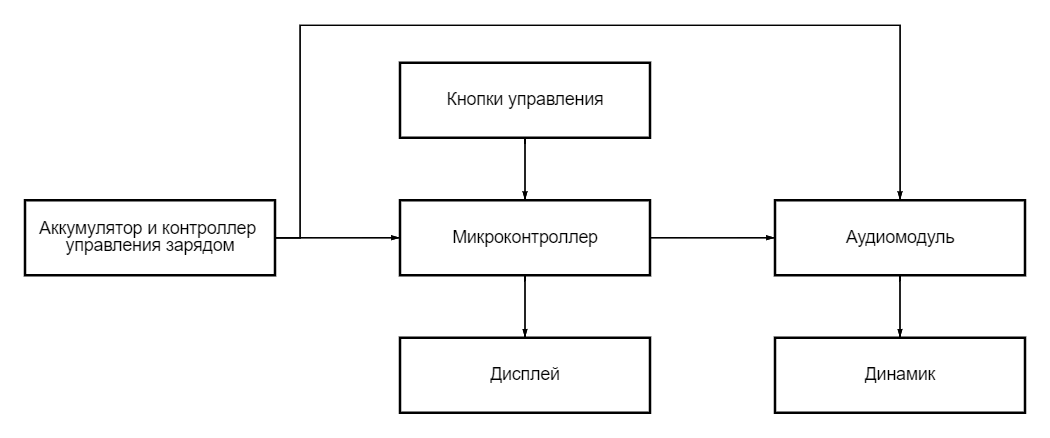


Рисунок 1.6 – Блок-схема портативного медиаплеера

Аккумулятор и контроллер управления зарядом необходимы для обеспечения портативности и возможности зарядки устройства. В тот момент, когда аккумулятор достигнет 100% заряда контроллер управления зарядом отключит цепь питания, тем самым исключит возможность выхода из строя аккумулятора при избытке энергии.

Управлять работой устройства будет микроконтроллер компании AVR, согласно заданию.

Чтобы управлять устройством и отображать информацию требуются кнопки и дисплей.

Для воспроизведения звука необходим отдельный аудиомодуль, включающий в себя ЦАП и возможность подключения карт памяти, на которых будут храниться аудиофайлы.

Встроенный динамик позволит воспроизводить звук без подключения дополнительного оборудования.

1.4 Выбор элементной базы

Детальный выбор радиокомпонентов будет произведен при проектировании схемы электрической принципиальной, однако основные элементы можно определить на данном этапе.

1. Микроконтроллер. Для управления устройством требуется определенный набор микросхемой логики. Учитывая функциональные возможности и задание дипломного проекта, для этого подойдет микроконтроллер серии AT компании AVR. В данной серии присутствуют множество микроконтроллеров с различными характеристиками, отучающиеся по цене.

ATmega ATtiny – это упрощенная серия микроконтроллеров ATmega, они дешевле и немного уступают по характеристикам, однако для данного проекта этого будет достаточно. Так же стоит отметить, что микроконтроллеры данной серии имеют меньшее количество контактов, соответственно их проще и дешевле монтировать в управляющую плату.

Основными представителями данной серии являются ATmega ATtiny 25, 45 и 85. Отличие заключается в объеме ПЗУ, но аппаратно данные микроконтроллеры полностью совместимы. Чтобы учесть возможность отладки программного кода большео объема был выбран ATmega ATtiny 85 (рисунок 1.7).

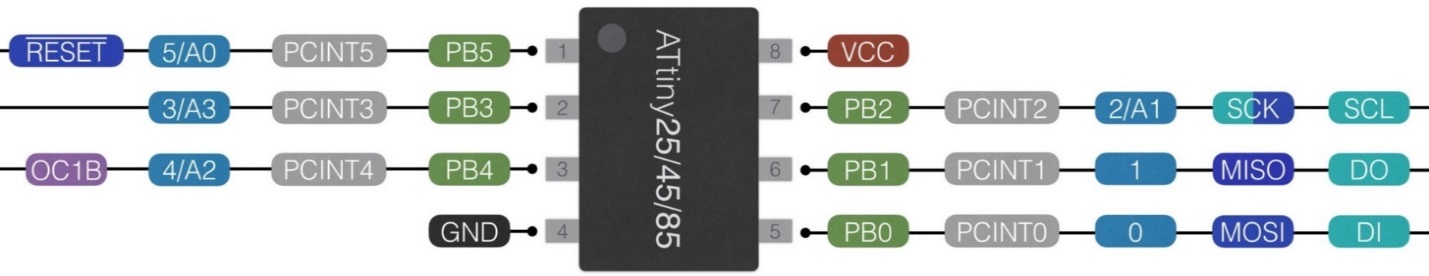


Рисунок 1.7 – Обозначение контактов микроконтроллеров ATmega ATtiny 25/45/85

2. Аудиомодуль. Для реализации необходимых функций будет использована модель: DFR0299 (рисунок 1.8).

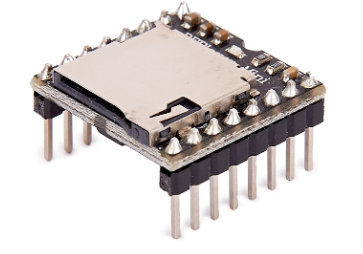


Рисунок 1.8 – Аудиомодуль DFR0299

Характеристики DFR0299:

* тип накопителя: SD-карта;
* поддерживаемые форматы аудио: MP3, WAV И WMA;
* поддерживаемые файловые системы: FAT16, FAT32;
* максимальный объём накопителя: до 32 гб;
* количество каталогов (папок): до 100;
* количество композиций в каталоге: до 255;
* линейный выход для внешнего усилителя (стерео);
* встроенный усилитель 3 вт для динамика (моно);
* уровни громкости: 30;
* динамический диапазон: до 90 дб;
* отношение сигнал/шум (snr): 85 дб;
* разрядность цап: 24 бита;
* частоты дискретизации: 8 / 11,025 / 12 / 16 / 22,05 / 24 / 32 / 44,1 / 48 кгц;
* режимы эквалайзера: normal, pop, rock, jazz, classic, base;
* аппаратный интерфейс цифровых портов i/o;
* аппаратный интерфейс аналоговых портов adkey;
* программный последовательный интерфейс uart;
* напряжение питания: 3,3–5 в;
* напряжение логических уровней: 3,3 в;
* потребляемый ток: до 250 ма;
* диапазон рабочих температур: −40…+80 °c;
* габариты: 20×20×10 мм.

3. LCD I2C. Для отображения информации и состояния устройства, следует предусмотреть разъем для установки дисплея. С учетом размеров устройства, а также количеству используемых линий для этого дела хорошо подойдет OLED дисплей с разрешением 128х64 i2c (рисунок 1.9).

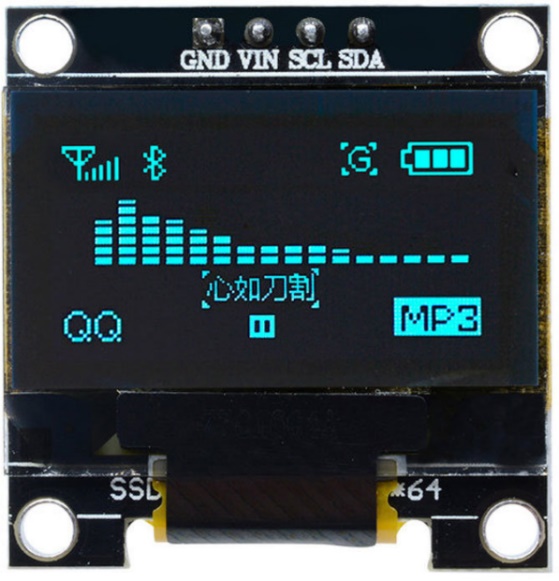


Рисунок 1.9 - OLED дисплей

2 Специальная часть

2.1 Проектирование схемы электрической принципиальной

Проектирование электрических принципиальных схем - это неотъемлемый этап разработки современных электронных устройств. Разумеется, можно сразу начать делать печатную плату устройства, чтобы не тратить время на разработку принципиальной схемы, однако данный способ не подходит для сложных устройств, так как будет очень трудно отследить правильные связи всех компонентов и будет риск неправильного подключения. Современные САПР для проектирования электронных устройств работают следующим образом: вначале строится схема электрическая принципиальная, на которой четко видны связи всех компонентов, затем схема проверяется на ошибки визуально и при помощи встроенного компилятора. Если ошибок не обнаружено, то принципиальная схема конвертируется в посадочные места радиоэлектронных компонентов и NET-лист, который показывает связь компонентов, не давая подключить их неправильно. Таким образом, риск допустить ошибку уменьшается. Для разработки подобных схем в настоящее время существует множество различных сред. Есть платные среды, такие как Altuim designer и Picad, а есть бесплатные: Kicad и Easy EDA. Платные среды в основном используются для проектирования сложных микропроцессорных устройств, которые работают на высоких частотах, и создаются для промышленности. Стоимость таких сред может составлять несколько сотен тысяч рублей, в зависимости от функций. Высокая цена обусловлена точностью расчета и автоматической трассировкой схем, а также инструментами тестирования, которые могут проверить устройство еще на этапе разработки в виртуальном виде на предмет ошибок. В данном случае для создания устройства такие функции не требуются, поэтому можно ограничиться бесплатной средой.

В настоящее время активно используются среды, которые работают в режиме онлайн через веб-интерфейс. Примером такой среды является Easy EDA, для работы в которой достаточно зарегистрироваться на сайте, затем перейти в окно редактора и можно сразу начать создавать свой проект. Преимуществом данной среды является то, что ее не нужно устанавливать на ПК, хотя для тех, у кого нет доступа к интернету, существует и установочная версия. Она практически ничем не отличается от онлайн редактора инструментами, однако, ней присутствует возможность сохранять проекты на компьютере, а так же нет необходимости авторизоваться каждый раз при входе. В остальных случаях редактор выводится в веб-интерфейс и все проекты сохраняются в облачное хранилище. Получить к нему доступ можно с любого ПК, для этого просто нужно войти в созданный профиль.

Еще одним достоинством среды является то, что можно создавать собственные УГО, посадочные места и схемы для электронных элементов, а затем открывать к ним свободный доступ. То есть любой пользователь может делиться своими разработками с тем кругом лиц, который определил сам. Благодаря данному подходу библиотеки компонентов постоянно пополняются. Так же стоит отметить, что Easy EDA имеет сразу оба типа условных графических обозначений согласно европейским и американским стандартам.

На рисунке 2.1 показана главная страница сайта среды EasyEda.



Рисунок 2.1 - Главная страница сайта EasyEda

Для того чтобы начать проектировать схему нужно нажать на кнопку «EasyEda Designer». Можно в начале зарегистрироваться, однако это не обязательно.

На рисунке 2.2. представлен внешний вид среды Easy EDA.

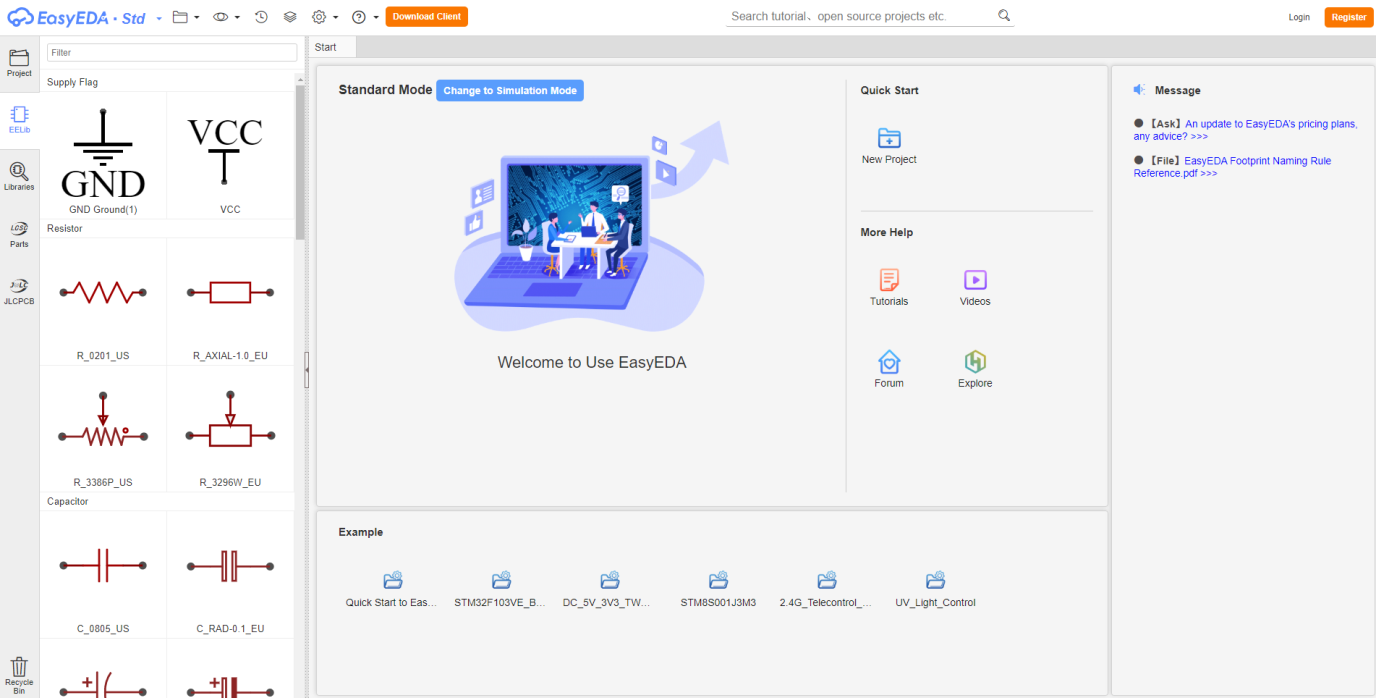


Рисунок 2.2 – Среда Easy EDA

Чтобы начать создавать проект необходимо нажать на иконку папки, затем выбрать «Новый», затем «Проект» (рисунок 2.3).

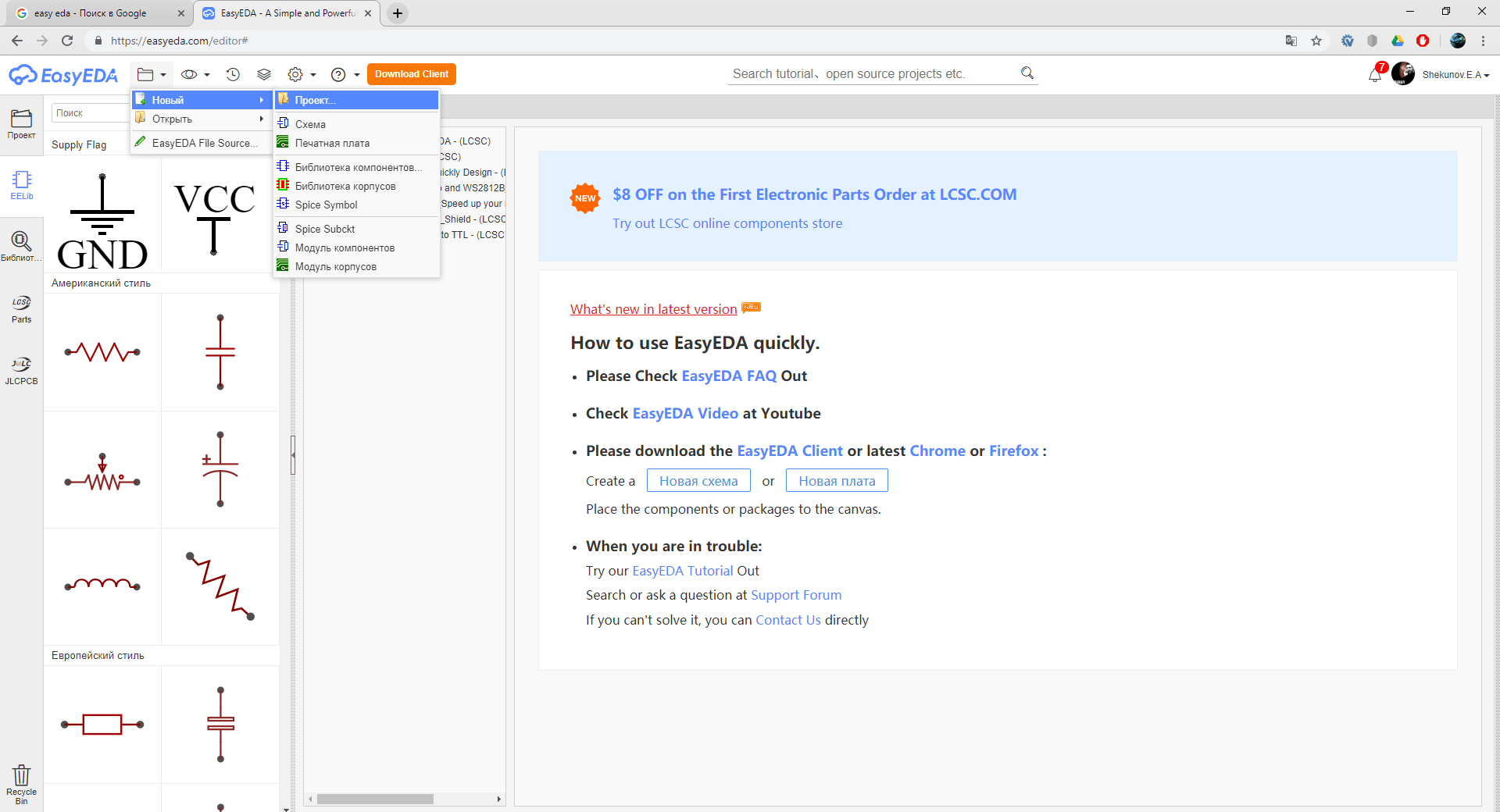


Рисунок 2.3 – Создание проекта в среде Easy EDA

Откроется окно проекта, в котором предложено ввести некоторую информацию о проекте (рисунок 2.4).

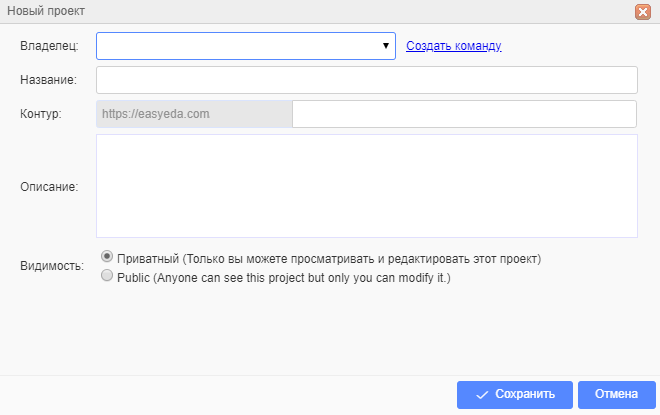


Рисунок 2.4 – Описание проекта

После создания проекта откроется редактор схем электрических принципиальных (рисунок 2.5).

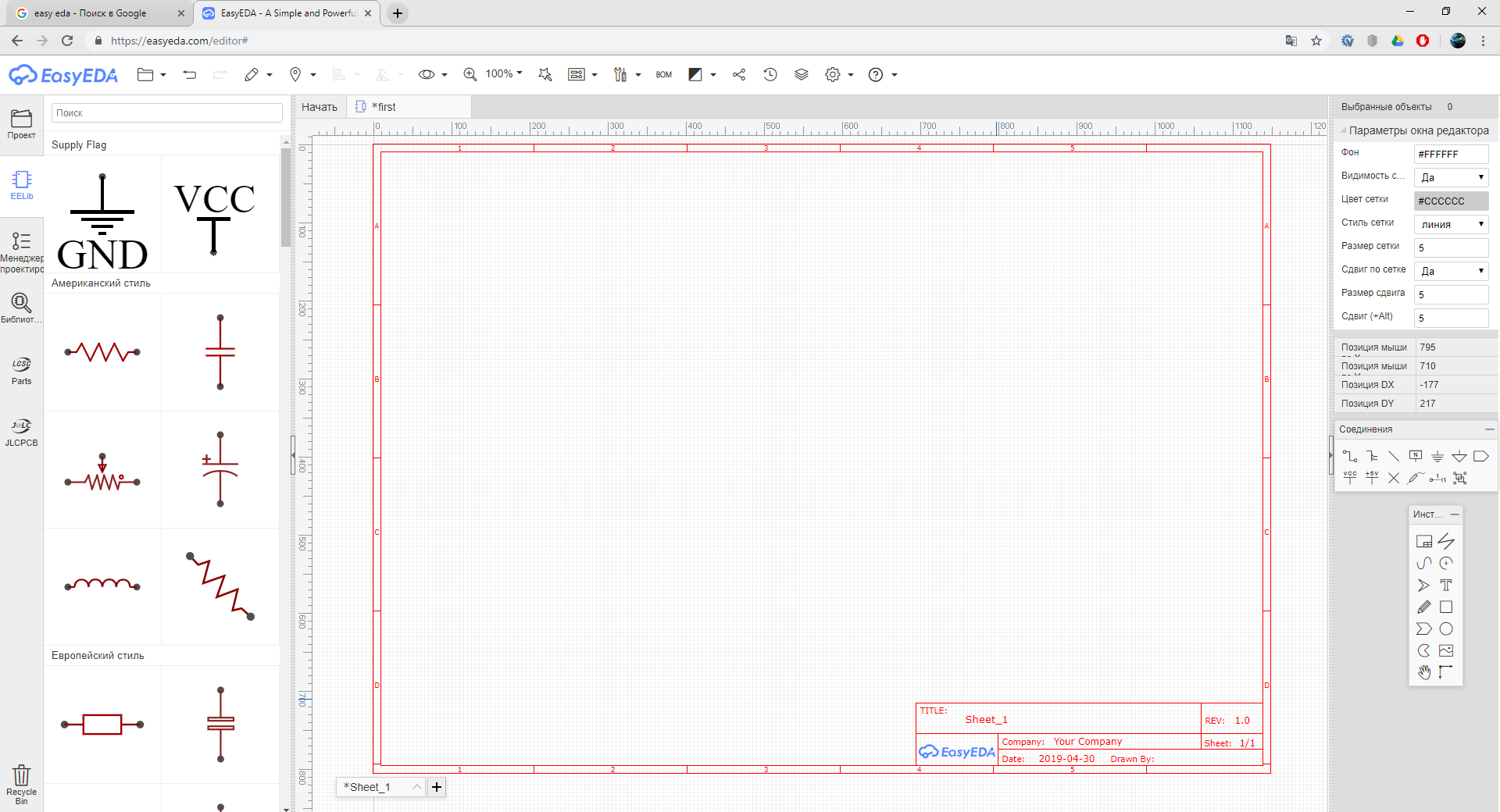


Рисунок 2.5 – Внешний редактора схем электрических принципиальных

Изначально есть рабочее поле с инженерной рамкой формата А4, при необходимости формат можно изменить. Слева находятся несколько функциональных иконок. Первая иконка под названием «EElib» открывает панель с часто используемыми УГО. Справа находятся наборы инструментов для создания и редактирования схем. Чтобы открыть всю библиотеку компонентов необходимо нажать на следующую иконку «Библиотека» и откроется окно поиска компонентов. Чтобы найти нужный, следует ввести его название, например «ATmega ATtiny 85», затем результаты поиска отобразятся в виде списка (рисунок 2.6).

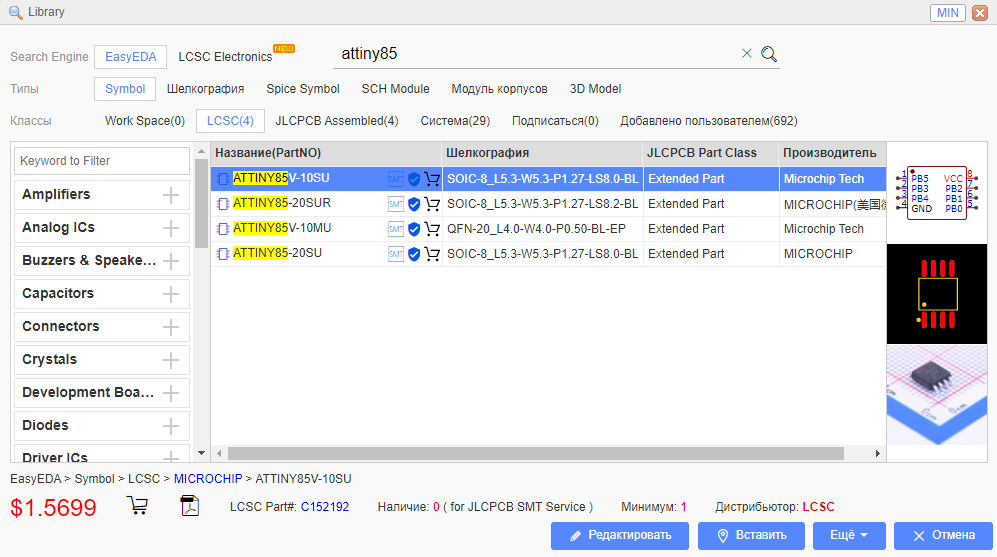


Рисунок 2.6 – Поиск компонентов

Справа можно увидеть УГО компонента, его посадочное место и в некоторых случаях фотографию внешнего вида, но стоит отметить, что так бывает не всегда. Например, пользователь создал своё УГО, но не стал создавать посадочное место, в таком случае, при создании монтажной схемы, его нужно будет рисовать самостоятельно. Так же стоит учитывать, что одни и те же компоненты с одинаковым УГО могут иметь несколько разных типов корпусов. В основном все корпуса разделяются на два типа: пространственные (SIP, DIP, ZIP) и с поверхностным монтажом (SMD). Корпуса пространственных типов имеют большие габариты, чем поверхностные и требуют наличие технологических отверстий с металлизацией для монтажа.

Поверхностный монтаж предполагает установку компонента на одну сторону, что в некотором случае облегчает монтаж. Однако сами компоненты могут быть настолько маленькими, что ручная установка будет невозможной. Такой вариант применим, если устройство имеет промышленное назначение и будет изготовлено на заводе. В данном случае, хоть некоторые компоненты и будут иметь SMD-корпус, первый прототип можно будет изготовить вручную.

После того как компоненты был найден, его нужно разместить в области прямоугольной рамки, желательно оставить место со всех сторон для подключения его контактов (рисунок 2.7).

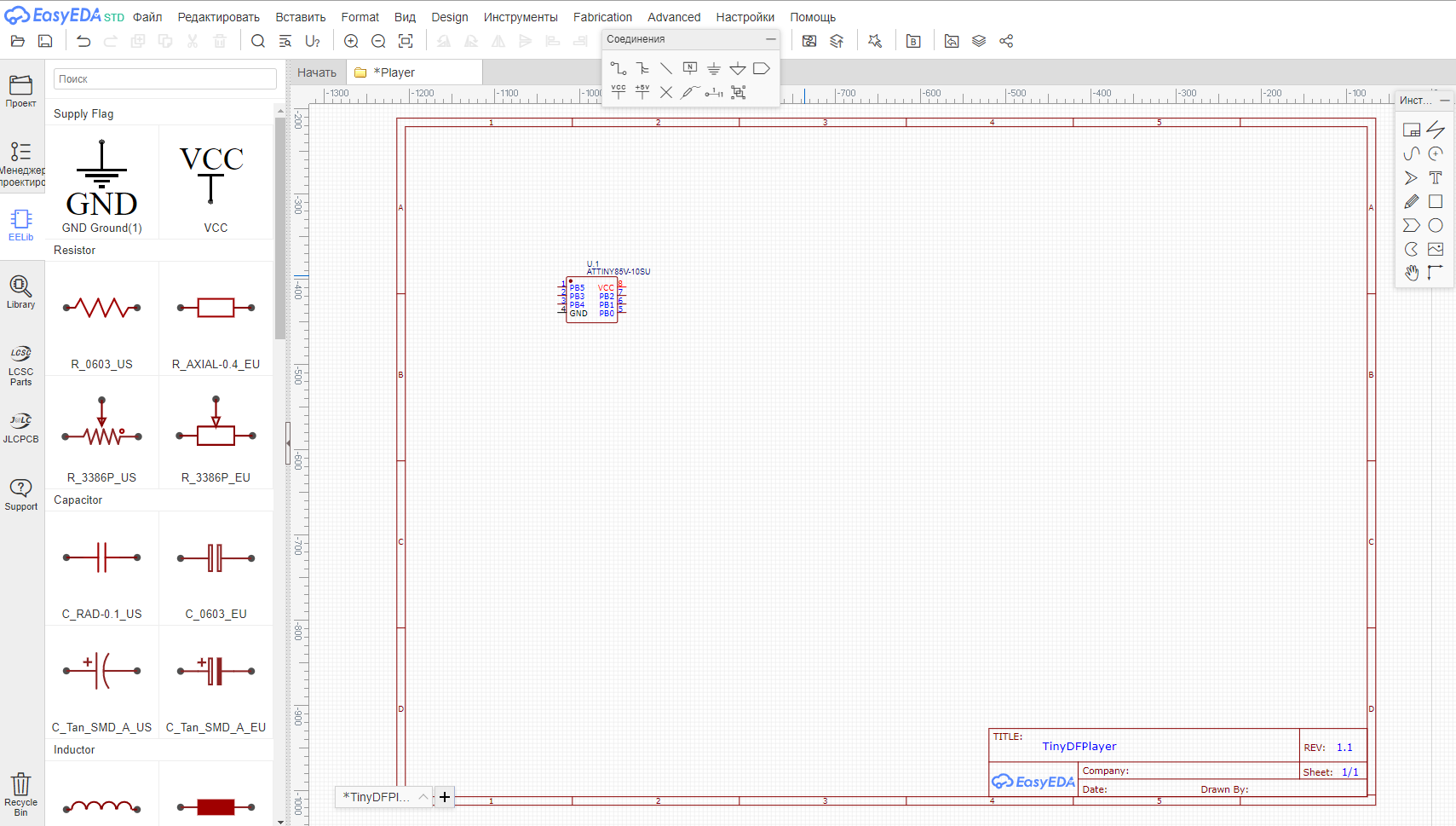


Рисунок 2.7 – Расположение микроконтроллера ATmega ATtiny 85

В случае чего его можно будет передвинуть. Далее необходимо подключить контакты питания и передачи данных. Делается это при помощи специальной панели инструментов, которая представлена на рисунке 2.8 и называется «Соединения».

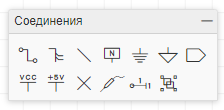


Рисунок 2.8 – Панель инструментов «Соединения»

Перечень инструментов слева направо, сверху вниз:

– одиночное соединение (Wire);

– шина (Bus);

– вход Шины (Bus Entry);

– название метки (Net Label);

– первый вид метки заземления (GND);

– второй вид метки заземления (GND);

– произвольная метка (NetPort);

– положительное напряжение (VCC);

– напряжение +5В (+5V);

– неподключенный контакт (Х);

– замер напряжения (U);

– контакт ножки (pin);

– группировка и разгруппировка компонентов (group).

Как видно из панели инструментов, создание связей в основном рассчитано на ссылки между нужными контактами, взамен старого способа: разводки каждой линии от контакта до контакта. Метка просто указывает название контакта, к которому она будет подключена. Это намного удобнее, так как не происходит нагромождения схемы и ее становиться проще читать. К тому же среда Easy EDA позволяет подсветить подключенные контакты, что облегчает их поиск.

Стоит отметить, что все метки и порты можно изменять, например вместо «+5V» указать «+12V» или вместо «VCC» указать «VSS». Тем самым получаются новые метки, которые будут связаны между собой.

После изучения набора инструментов, можно приступить к подключению контактов. На рисунке 2.9 представлен микроконтроллер ATmega ATtiny 85 с подключенной обвязкой.

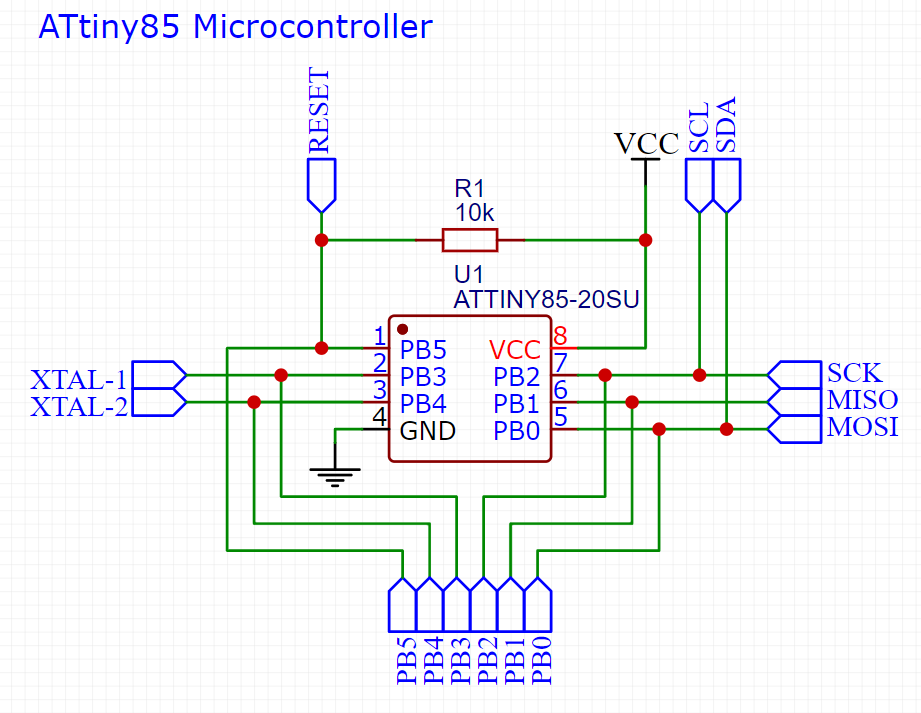


Рисунок 2.9 – Микроконтроллер с подключенной обвязкой

Для удобства каждый блок устройства можно подписывать, например, как показано на рисунке 2.9 «ATmega ATtiny 85 Microcontroller». Чтобы не запутаться в контактах, сами метки лучше называть так же, как и сигналы микроконтроллера, либо устройства, которое будет подключено к нему, например для кварцевого резонатора удобнее будет сделать название меток «XTAL1» и «XTAL2». Так же стоит отметить, что некоторые компоненты все же можно подключить, если они не нагромождают схему и не мешают другим линиям, например резистор R1. Это подтягивающий резистор к сигналу «RESET» к сигналу «VCC». Он необходим для того, чтобы не возникало ложных сигналов перезагрузки устройства, которые могут возникать из-за электромагнитны помех, возникающих при работе устройства, либо дребезг контактов кнопки, происходящий в момент ее нажатия.

Сигналы «SCK», «MISO» и «MOSI» формируют интерфейс для программирования, приема и передачи данных SPI.

Сигналы «SCL» и «SDA» дублируются на сигналы «SCK» и «MOSI», но при определенной программе они становятся отдельным интерфейсом для приема и передачи данных под названием I2С. Именно этот интерфейс использует OLED-дисплей.

Остальные сигналы вида «PB» являются сигналами ввода/вывода свободного назначения и будут использованы для некоторых элементов. Стоит отметить, что сигналы «PB0», «PB1» и «PB2» так же дублируют контакты вышеописанных интерфейсов. Всё это делается для удобства, чтобы было видно, сколько назначение может принимать одна ножка микроконтроллера.

На следующем этапе необходимо подключить контакты микросхемы аудиомодуля (рисунок 2.10).

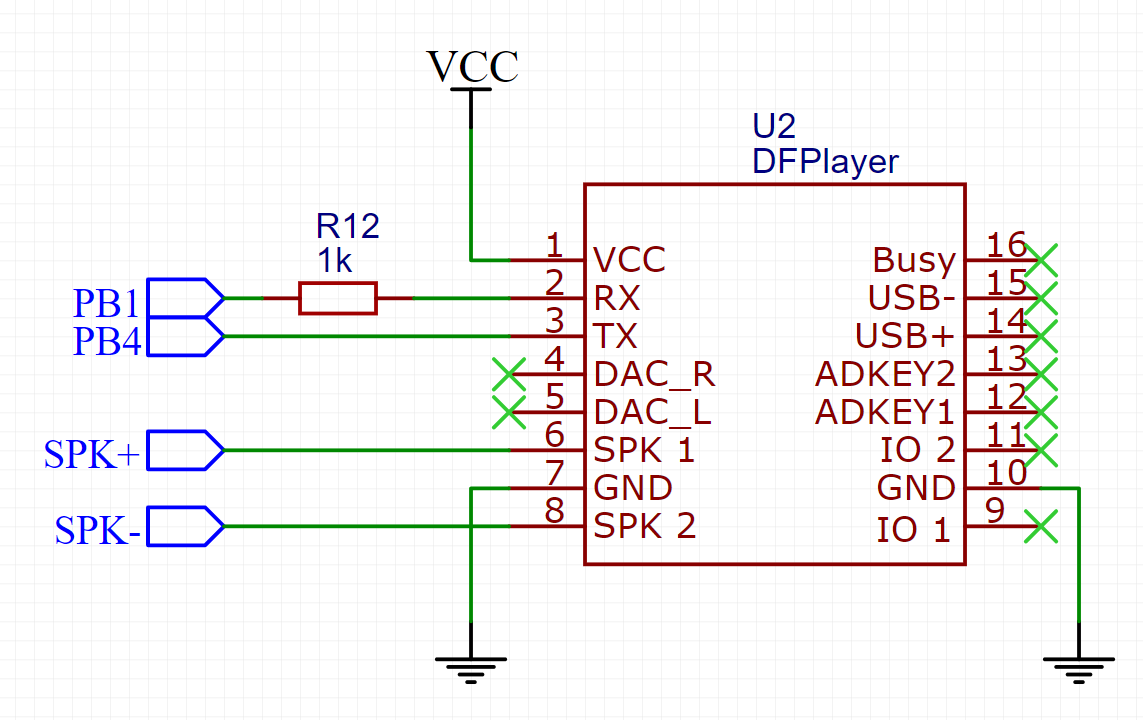


Рисунок 2.10 – Подключение аудиомодуля

Для связи с микроконтроллером данный модуль будет использовать сигналы «RX» и «TX», которые в свою очередь будут подключены к «PB1» и «PB4». Стоит отметить, что для сигнала «RX», в цепь стоит установить резистор номиналом в 1КОм, это нужно для ограничения напряжения, которое будет поступать на микроконтроллер от аудиомодуля, в противном случае микросхема ATmega ATtiny 85 может сгореть.

Сигналы «SPK+» и «SPK-» подключаются напрямую к динамику, так как в данном аудиомодуле уже есть встроенный усилитель.

Сигналы «VCC» и «GND» обозначают контакты питания. Остальные сигналы аудиомодуя остаются незадействованными, так как они служат для управления звуковыми дорожками (плей, пауза, переключение, громком), а в данном проекте управлять всем этим будет микроконтроллер ATmega ATtiny по интерфейсу UART (сигналы «RX» и «TX»).

Так как пор заданию устройство должно быть портативным, для этого необходима аккумуляторная батарея, разумеется, можно использовать одноразовые батарейки, но в 2021 году для подобных устройств – это не актуально, следовательно требуется аккумуляторная батарея и возможность ее подзарядки, для этого была спроектирована схема заряда АКБ (рисунок 2.11).

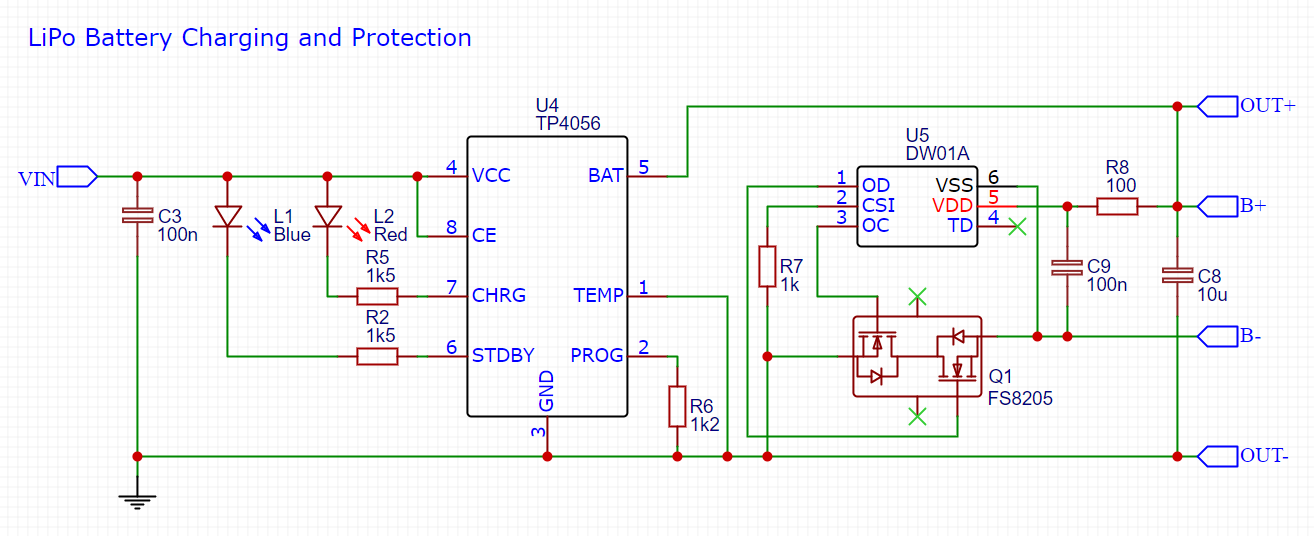


Рисунок 2.11 – Схема зарядки и защиты литий-ионных АКБ

Основой данной схемы является контроллер TP4056. Стоит отметить, что данный контроллер уже является контроллером заряда литий-ионных АКБ и для его полноценной работы требуется лишь подключить его согласно сигналам и добавить резистор на 1.2КОм к сигналу «PROG» для осуществления обратной связи с аккумуляторной батареей: чтобы микроконтроллер отключал ее, когда зарядка будет окончена. Что касается остальных компонентов схемы, то конденсатор «С3» является сглаживающим фильтром, однако он не обязателен, если зарядка осуществляется от USB-порта, светодиоды «L1» и «L2» вместе с подключенными к ним резисторами «R2» и «R5» нужны только для индикации а зарядке и ее завершении, разумеется, они необходимы, чтобы пользователь понимал, когда устройство будет полностью заряжено, но на саму работу они никак не влияют.

Микросхемы DW01A и FS8205 представляют из себя сборки по два полевых транзистора. Их назначение заключается в защита от короткого замыкания и переполюсовки. На самом деле для данной схемы они не обязательны, если устройство собирать аккуратно и внимательно. Они больше подходят для отдельных плат, предназначенных для заряда литий-ионных АКБ при постоянном подключении и отключении последних (рисунок 2.12).

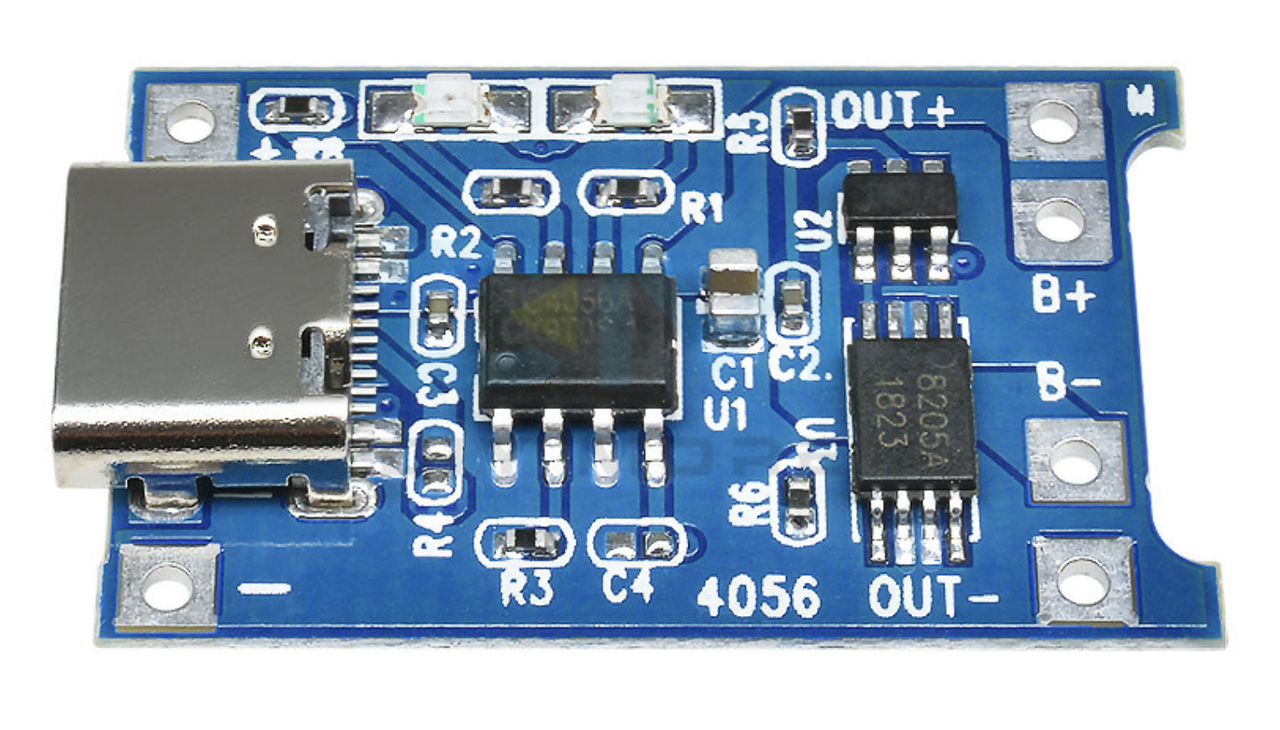


Рисунок 2.12 – Готовое устройство для контроля зарядки АКБ

Резисторы «R7» и «R8» являются частью обвязки данного участка схемы, конденсаторы «C8» и «C9» - сглаживающими выходными фильтрами.

Подключение дисплея показано на рисунке 2.13.

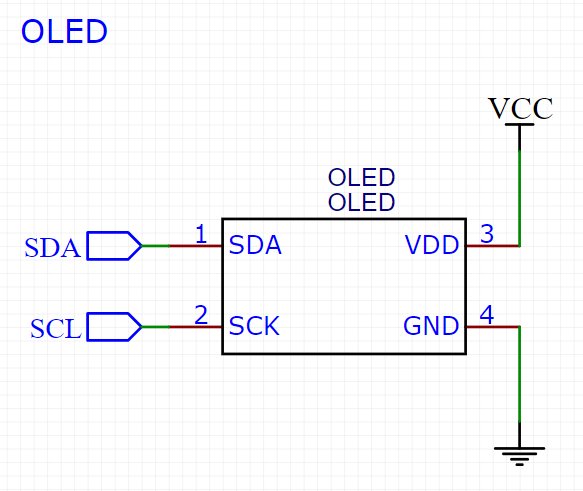


Рисунок 2.13 – Подключение OLED-дисплея

Дисплей использует интерфейс I2С и подключается напрямую к микроконтроллеру ATmega ATtiny 85.

Далее необходимо подключить кнопки (рисунок 2.14).

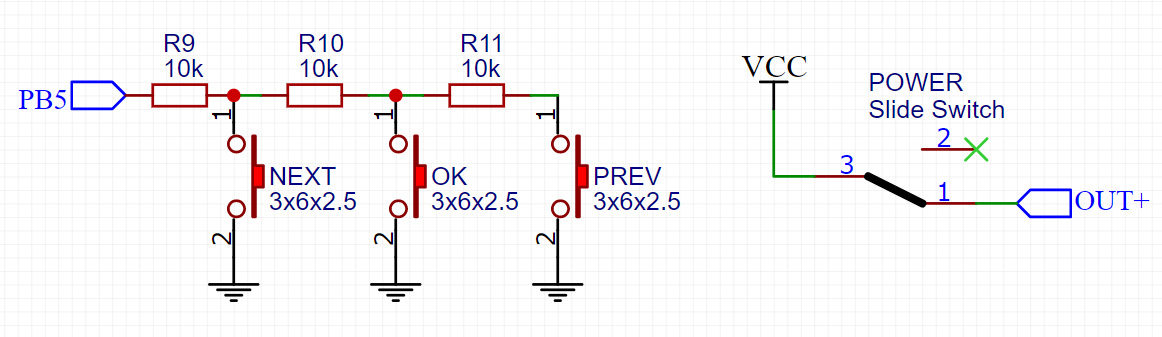


Рисунок 2.14 – Подключение кнопок

Кнопка «POWER» – необходима для включения и выключения устройства, а для переключения и воспроизведения композиций предусмотрены три кнопки: «NEXT», «OK» и «PREV», как видно из схемы они подключены одному и тому же контакту «PB5», но при замыкании контакта получается, что микроконтроллер будет принимать сигнал через разное сопротивление:

– «NEXT» - 10КОм;

– «OK» - 20КОм;

– «PREV» - 30Ком.

При программировании, сигнал «PB5» будет являться аналоговым входом, таким образом при нажатии кнопок, каждый сигнал будет иметь отдельные значения, и они будут отличаться, даже несмотря на то, что подключены к одному и тому же контакту.

На следующем этапе необходимо будет подключить все коннекторы (рисунок 2.15).

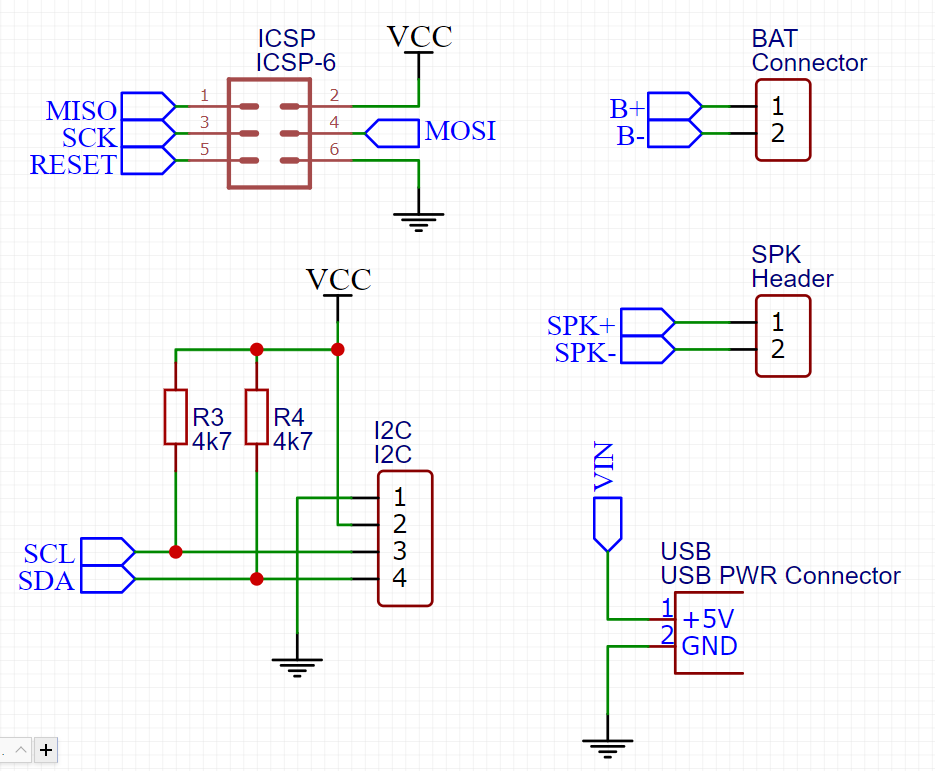


Рисунок 2.15 – Подключение коннекторов

Разъем «ICSP» используется для встроенного программирования по интерфейсу SPI, «BAT Connector» - для подключения аккумуляторной батареи. «SPK Header» - разъем для подключения динамика, «USB» - интерфейс для зарядки аккумуляторной батареи. I2С интерфейс служит для подключения дисплея или дополнительного устройства с данным интерфейсом. Стоит отметить, что для его корректной работы следует установить два подтягивающих резистора «R3» и «R4», в противном случае могут произойти помехи, которые будут способствовать зависанию дисплея.

Заключительным этапом проектирования схемы электрической принципиальной является подключение потенциометра на 10КОм для регулировки громкости медиаплеера (рисунок 2.16).

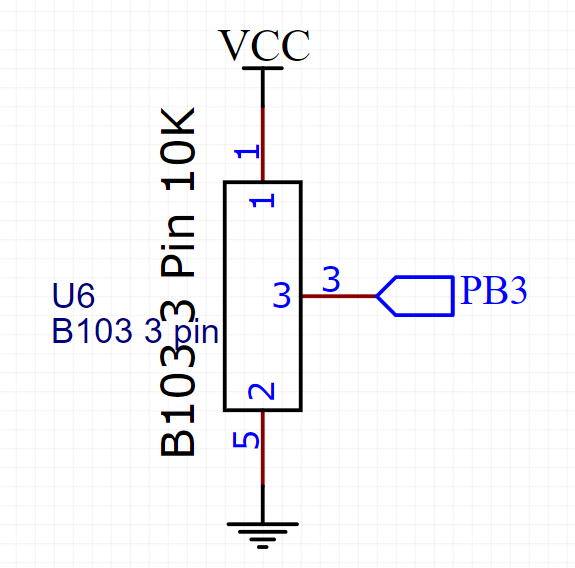


Рисунок 2.16 – Потенциометр для регулировки громкости

На этом проектирование схемы электрической принципиальной закончено.

2.2 Проектирование печатной платы устройства

На следующем этапе необходимо будет спроектировать схему монтажной платы. Для этого следует нажать на иконку вкладки, показанной на рисунке 2.17 и выбрать пункт «Конвертировать в печатную плату».



Рисунок 2.17 – Конвертирование в печатную плату

После данного действия откроется редактор печатных плат, в котором отобразятся все посадочные места компонентов электрической принципиальной схемы (рисунок 2.18).

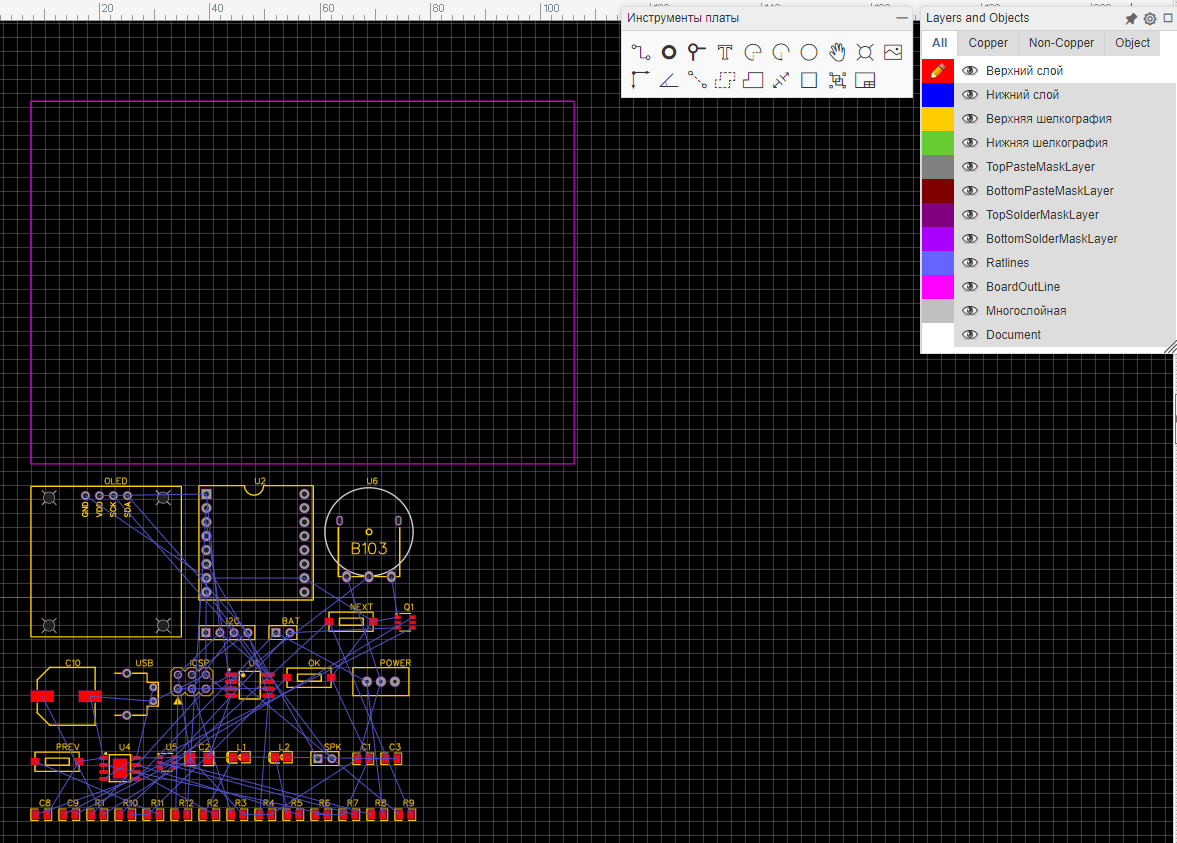


Рисунок 2.18 – Получение посадочных мест из принципиальной схемы

Цвет элементов имеет значение и впоследствии сохраняется как отдельный слой.

Перечень слоев и их цвета:

– верхний слой (красный);

– нижний слой (тёмно-синий);

– верхняя шелкография (желтый);

– нижняя шелкография (зеленый);

– верхний слой маски контактных площадок (тёмно-серый);

– нижний слой маски контактных площадок (тёмно-красный);

– верхний слой паяльной маски (тёмно-фиолетовый);

– нижний слой паяльной маски (фиолетовый);

– связи (синий);

– контур платы (пурпурный);

– сквозные отверстия (серый);

– описания (белый).

Размеры и форму контура платы можно редактировать. Среда Easy EDA по умолчанию устанавливает размер схемы, исходя из количества элементов и их размеров, однако, это можно изменить в большую или меньшую сторону.

Посадочные площадки компонентов так же можно изменять в процессе создания схемы печатной платы, для этого необходимо вернуться в редактор схем электрических принципиальных, выбрать нужный элемент и в его свойствах выбрать другой тип корпуса, затем сохранить документ и редакторе печатных плат нажать на кнопку «Импортировать изменения» (рисунок 2.19).

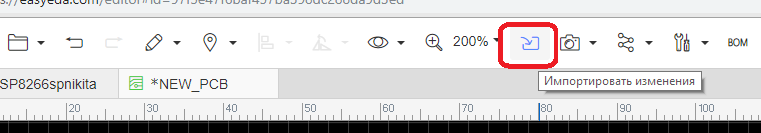


Рисунок 2.19 – Импорт изменений

Данная функция будет полезной в том случае, когда печатная плата уже частично спроектирована, но некоторые элементы не подходят по размеру или функционалу.

Теперь необходимо перенести все компоненты в область схемы печатной платы так, чтобы контакты не выступали за границы данного контура (рисунок 2.20).

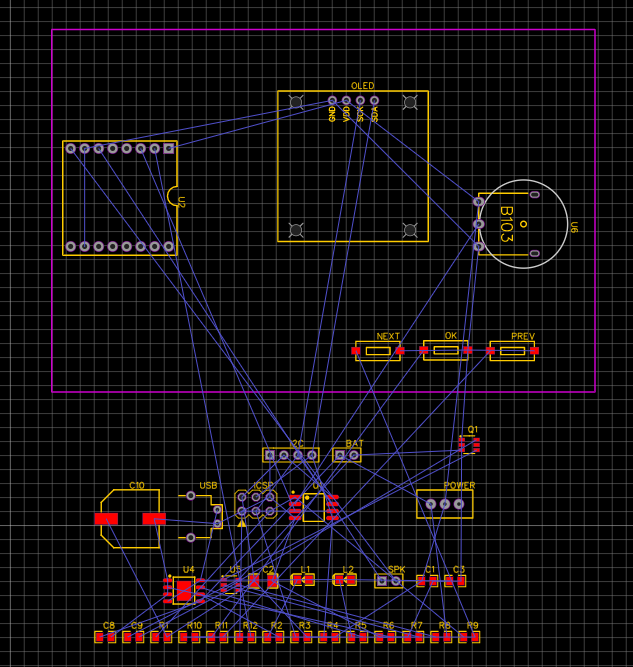


Рисунок 2.20 – Расположение посадочных мест

Синим цветом отображаются связи компонентов между собой, согласно схеме электрической принципиальной. Существует функция автотрассировки, но она не всегда работает корректно, поэтому лучше сделать ее вручную. Так как впоследствии плата будет заказана на заводе, можно установить минимальную толщину дорожки, однако это сделает устройство менее надежным и стойким к механическим повреждениям. Оптимальным выбором является толщина от 0,6 мм до 0,8. Данные значения будут использованы для сознания схемы печатной платы.

Готовая схема печатной платы показана на рисунке 2.21.

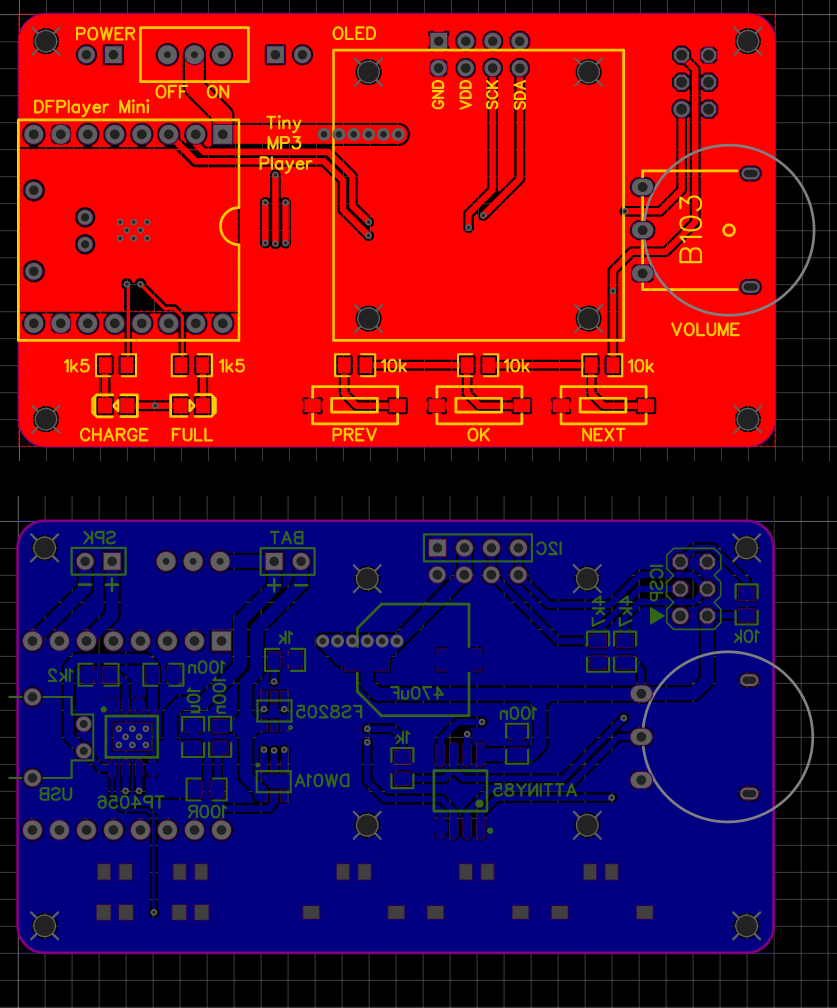


Рисунок 2.21 – Готовая схема печатной платы

2.3 Изготовление устройства

Для первого прототипа устройства было принято решение не заказывать его сразу на заводе, а изготовить вручную. Изготовление печатной платы будет производиться при помощи лазерно-утюжной технологии (ЛУТ). Данный метод широко используется при домашнем изготовлении, потому как прост и достаточно качествен. Принцип технологии заключается в следующем:

а) шаблон устройства распечатывается на бумаге (обычная, глянцевая или термотрансферная) при помощи лазерного принтера, стоит учесть, что изображение нужно отразить зеркально по горизонтали, если это не было просчитано в самом шаблоне;

б) распечатанный шаблон необходимо приложить к медной стороне фольгированного текстолита и нагреть при помощи утюга, время нагрева зависит от площади платы, температуры нагрева и типа бумаги;

в) после нагрева тонер с бумаги приливает к медной фольге, необходимо подождать пока плата остынет и, затем, удалить бумагу;

г) после удаления бумаги плата проверяется на предмет целостности трафарета и помещается в коррозийный раствор, это может быть хлорное железо, персульфат аммония, перекись водорода с лимонной кислотой, в зависимости от предпочтения разработчика;

д) коррозийный раствор начинает разъедать медь, незащищенную трафаретом из тонера и после небольшого промежутка времени на плате остаются только дорожки трафарета;

е) в завершении, плату необходимо очистить и промыть спиртосодержащим раствором.

Определив этапы, можно приступить к созданию печатной платы.

Для распечатки шаблона была использована специальная термотрансферная бумага (рисунок 2.22).

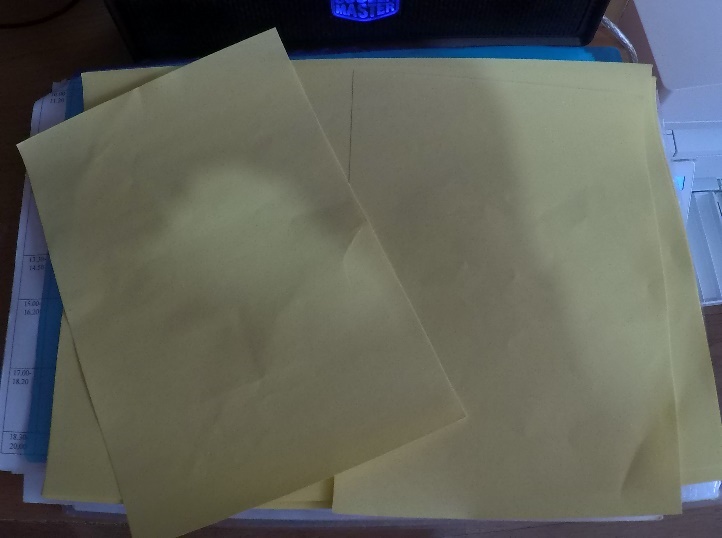


Рисунок 2.22 – Термотрансферная бумага

Данный тип бумаге создан специально для переноса тонера на другую поверхность. На рисунке 2.23 изображен распечатанный шаблон.

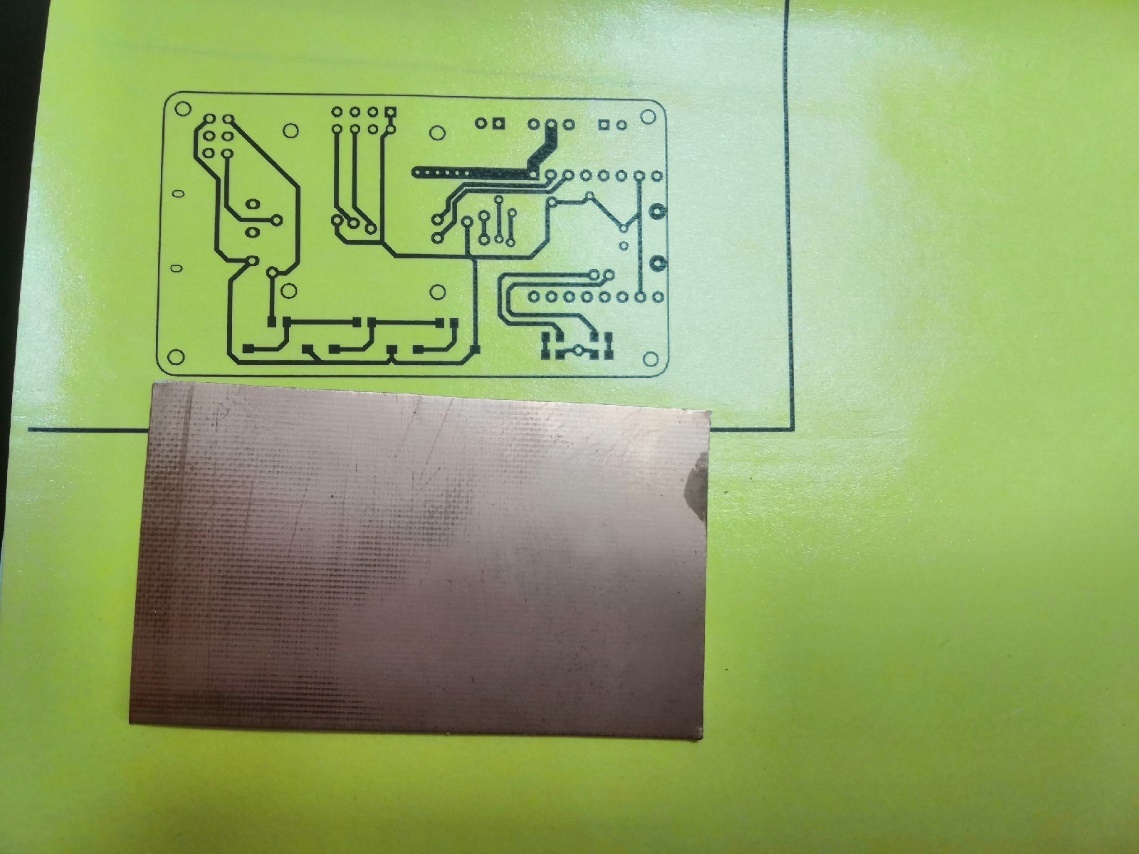


Рисунок 2.23 – Распечатанный шаблон

Структура бумаги не позволяет тонеру въедаться в текстуру и плотно прилипать к ней, поэтому любой шаблон переносится легко и целиком. Нужно поместить шаблон на текстолит и плотно прогладить утюгом (рисунок 2.24).

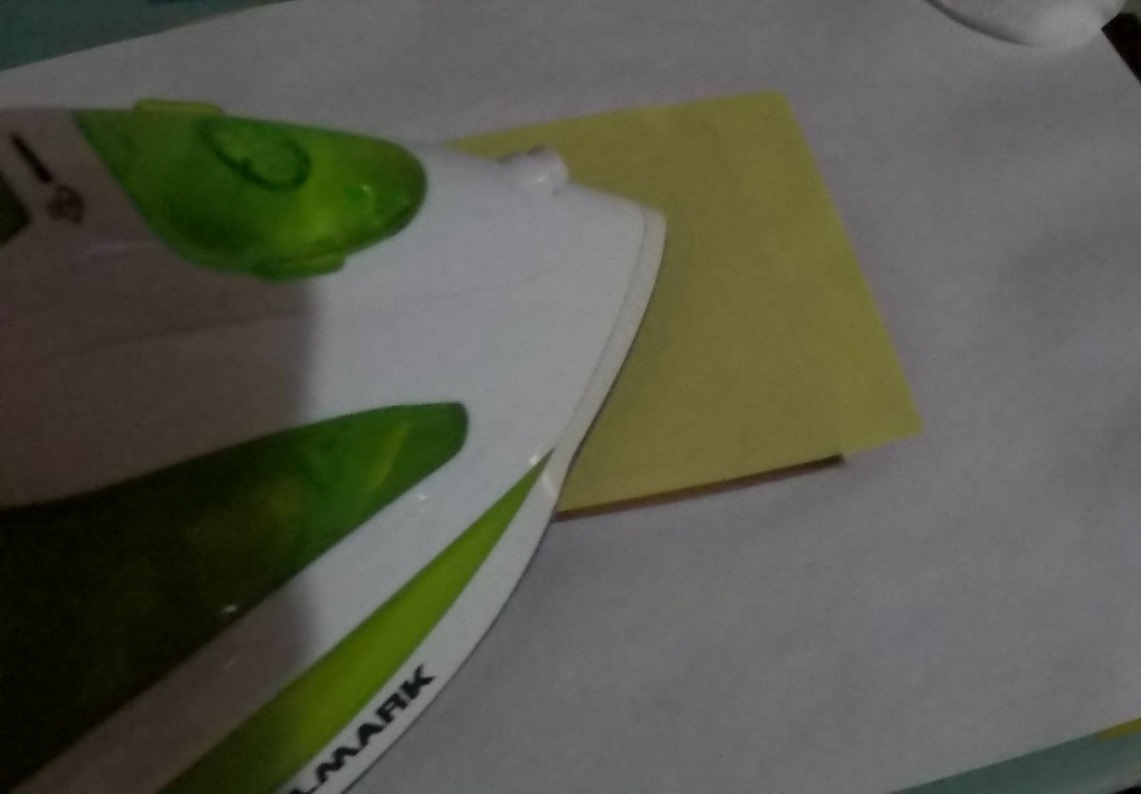


Рисунок 2.24 – Перенос трафарета путем нагрева

После того как плата остынет следует аккуратно удалить бумагу (рисунки 2.25 и 2.26).

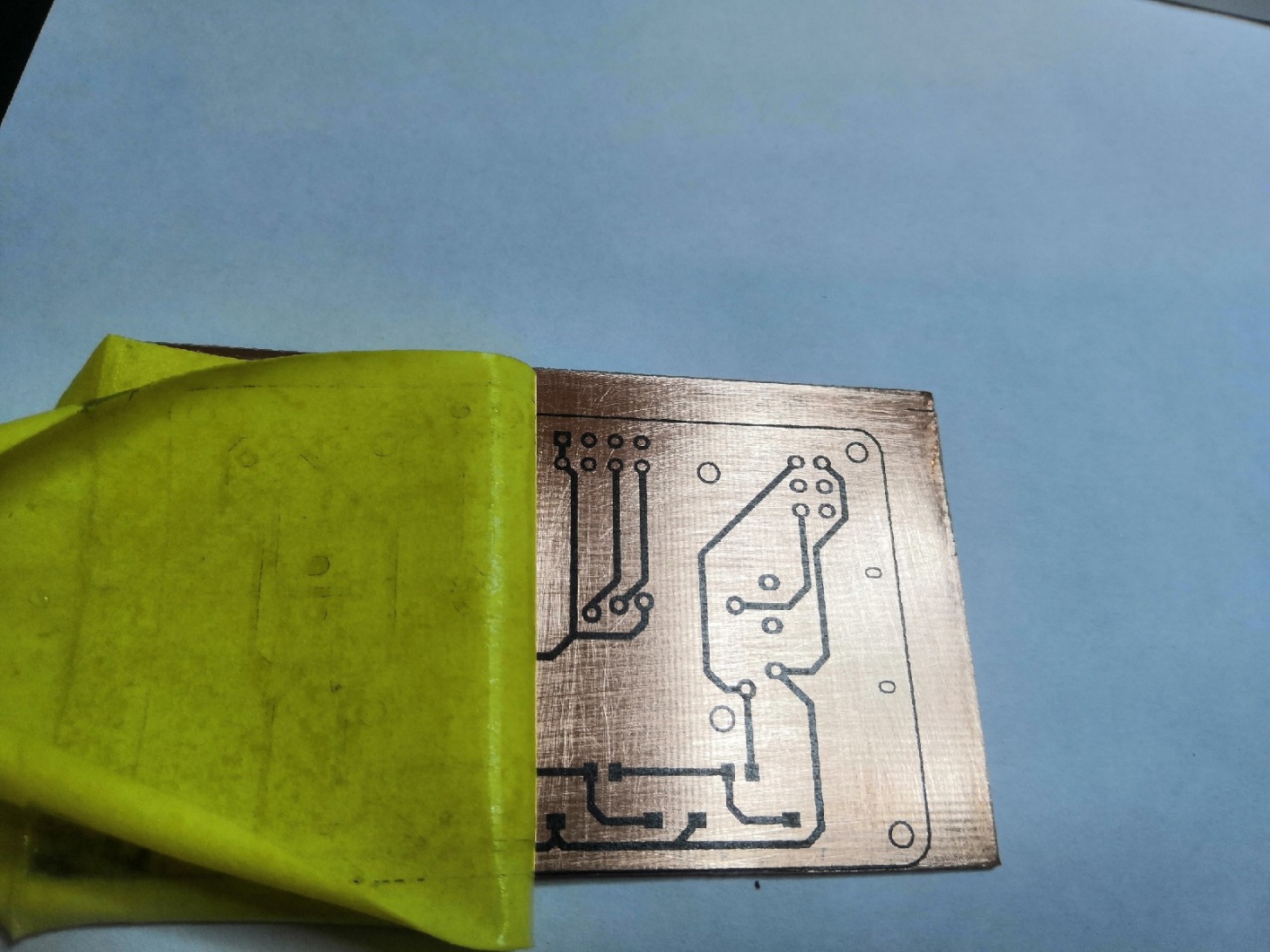


Рисунок 2.25 – Удаление бумаги

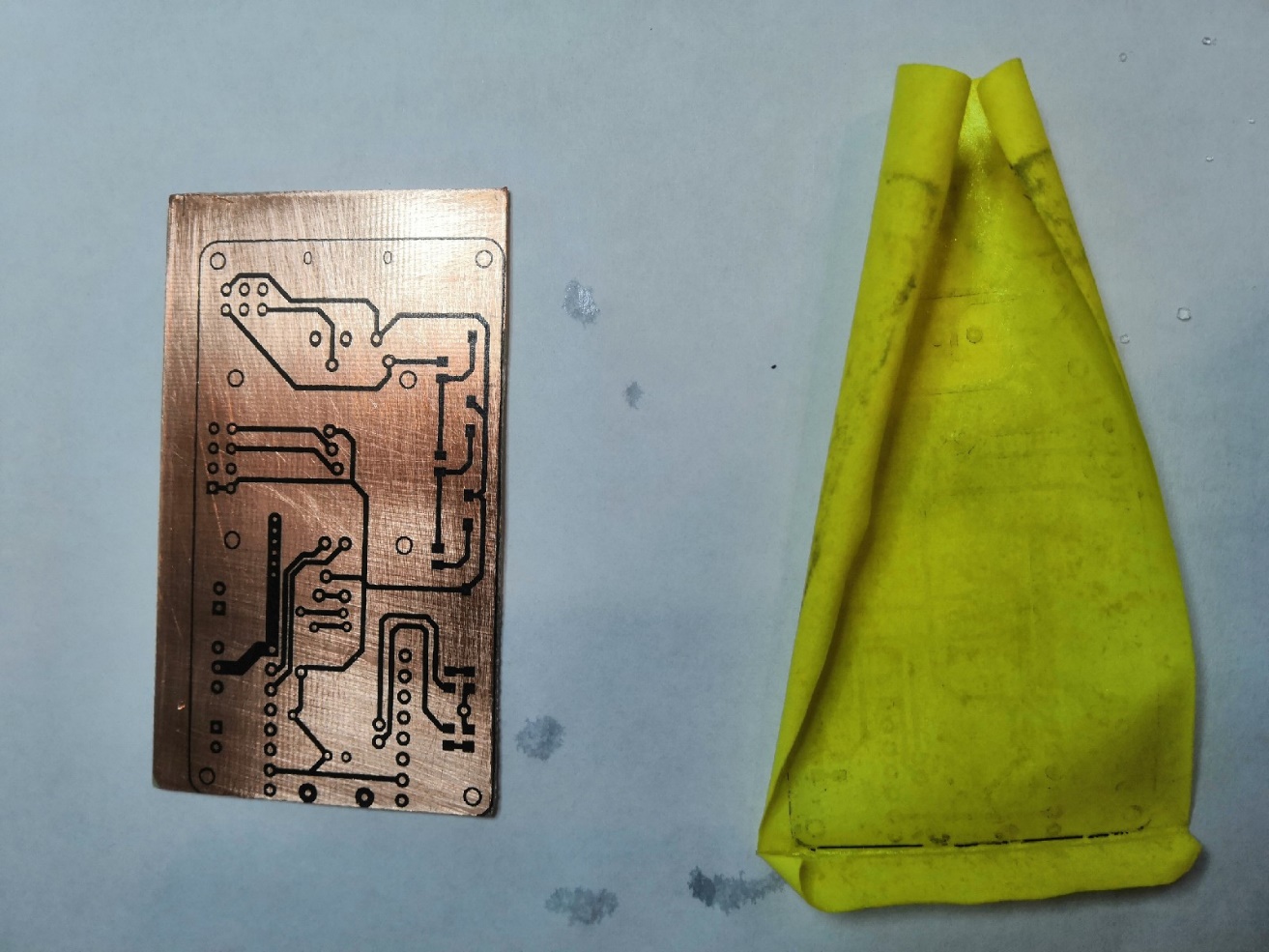


Рисунок 2.26 – Переведенный трафарет

Так как плата будет двухсторонняя, вторую сторону необходимо заклеить скотчем, чтобы раствор не разъел ее. Теперь необходимо поместить плату в коррозийный раствор. Самым популярным долгое время являлся раствор хлорного железа, однако он оставляет после себя следы на посуде, мебели и одежде, если случайно пролить. Так же он выделяет едкий запах и травление платы должно осуществляться в хорошо проветриемом помещении. Исходя из этого, был выбран трехпроцентный раствор перекиси водорода, соли и лимонной кислоты. Все материалы стоят дешево и продаются в обычных магазинах и аптеках. Необходимо смешать: полчайной ложки соли, 50 грамм лимонной кислоты и 100мЛ перекиси водорода в пластиковом контейнере (рисунок 2.27).

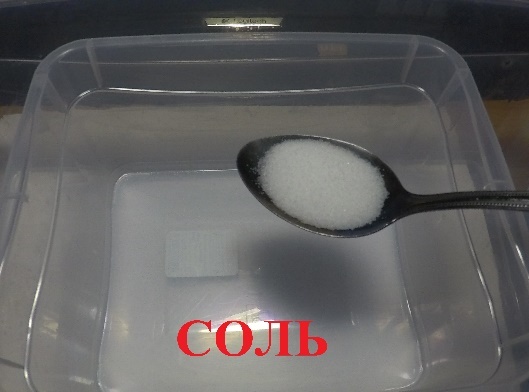
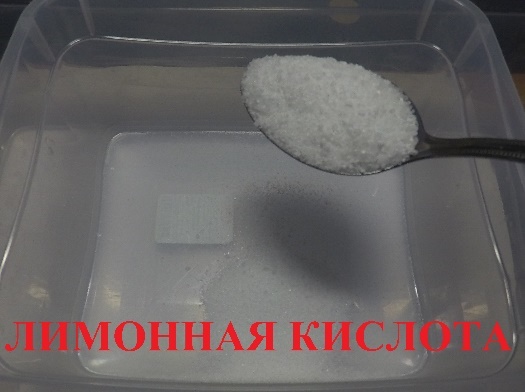
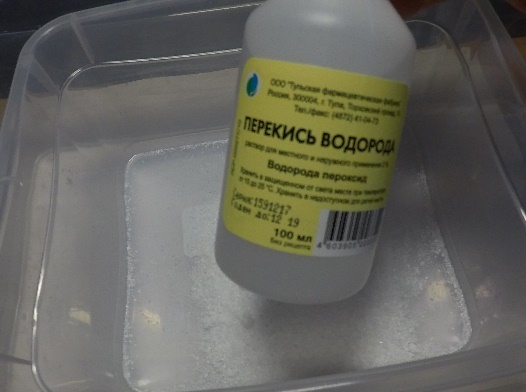
  

Рисунок 2.27 – Создание коррозийного раствора

После смешивания необходимо хорошо всё перемешать и дать настояться пять минут. Перекись и лимонная кислота образуют коррозийное вещество. Соль является катализатором, ускоряющим реакцию. Так же для уменьшения времени травления можно пообгореть раствор или использовать «водяную баню». После приготовления раствора необходимо поместить в него плату (рисунок 2.28).

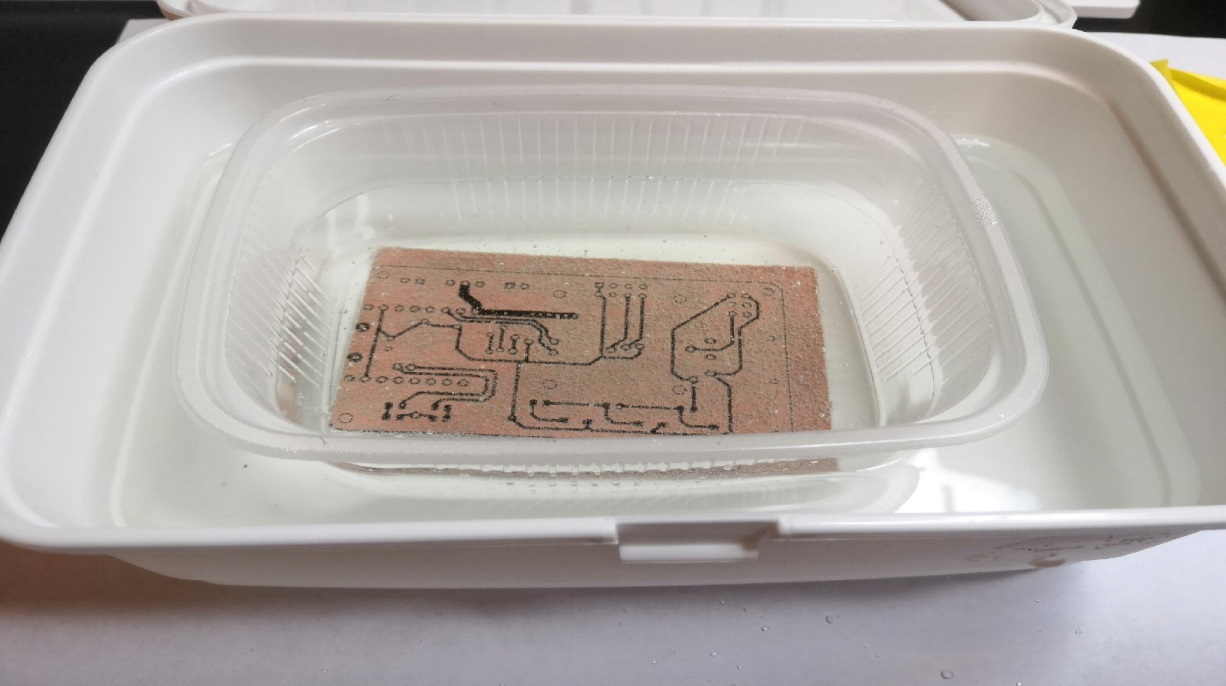


Рисунок 2.28 – Помещение платы в коррозийный раствор

Спустя десять минут цвет раствора изменился и слой меди частично растворился (рисунок 2.29).



Рисунок 2.29 – Травление платы

Спустя еще десять минут оставшийся слой незащищенной меди полностью разъел раствор, плату можно вынимать (рисунок 2.30).

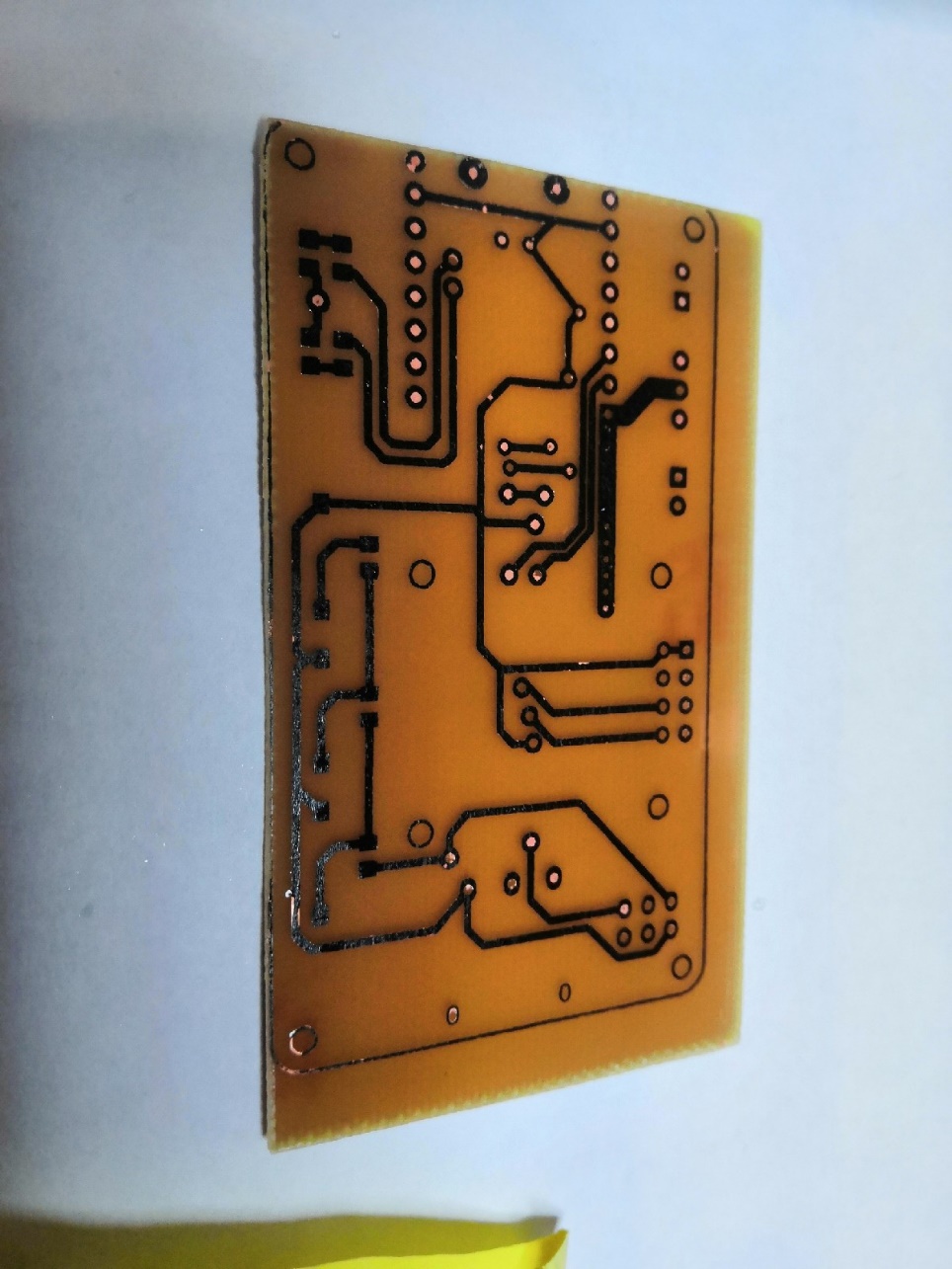


Рисунок 2.30 – Вытравленная плата

Следующим шагом будет очистка платы от тонера и сверление отверстий (рисунок 2.31).

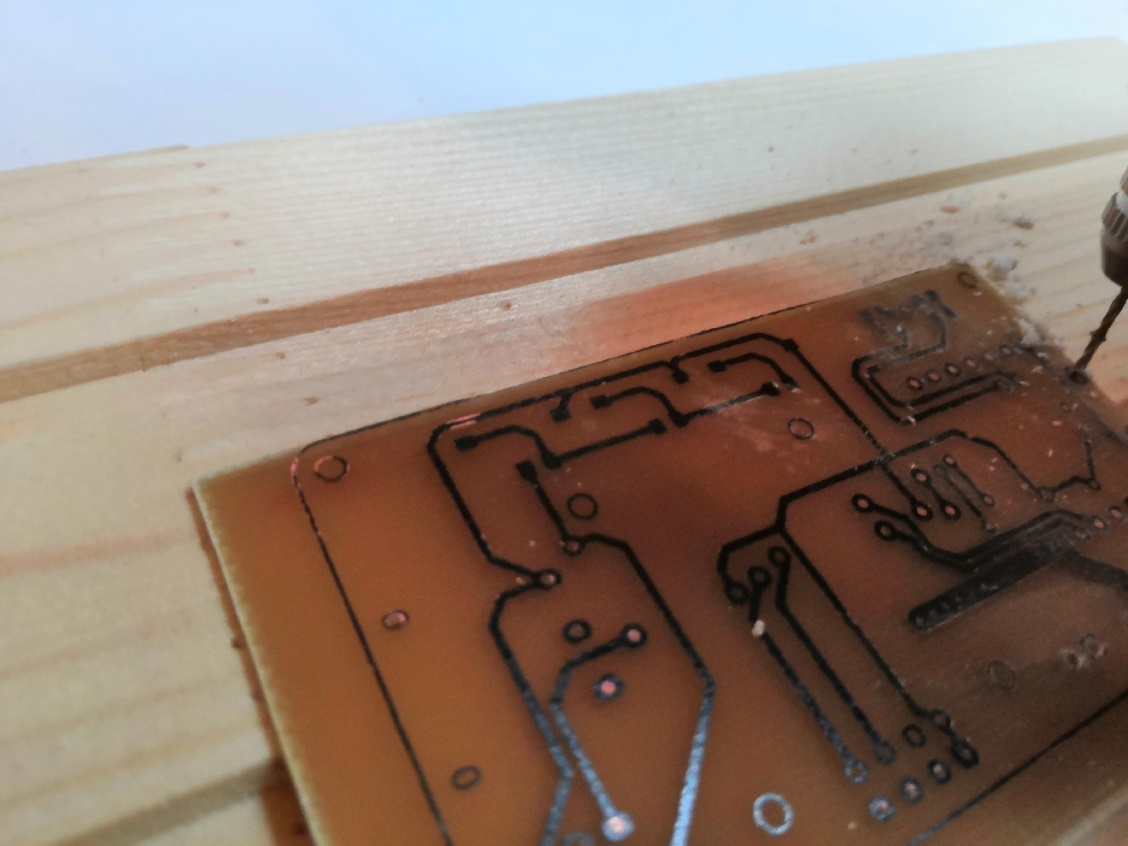


Рисунок 2.31 – Сверление отверстий

Затем, повторяя метод ЛУТ, на обратную сторону платы наносится второй шаблон, и процедура повторяется. Стоит учесть, что в данном случае уже вытравленную сторону следует заклеить, чтобы раствор не разъел ее. Остается обрезать плату по контуру и очистить ее спиртовым раствором (рисунок 2.32).

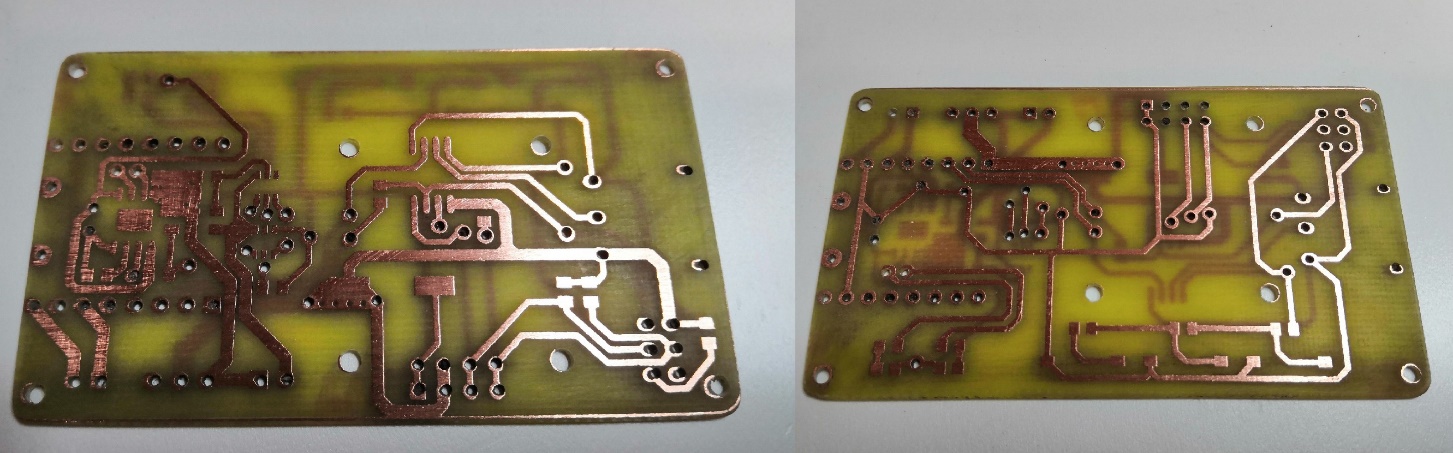


Рисунок 2.32 – Внешний вид готовой платы

Печатная плата готова, теперь необходимо распаять все имеющиеся компоненты. Для пайки можно использовать любой паяльник мощностью до 25Вт или паяльную станцию с регулировкой температуры (как в данном проекте). Температура паяльника должна быть не более 350 градусов, иначе появляется вероятность перегреть дорожку или радиокомпонент. Распайка радиоэлектронных элементов показана на рисунке 2.33.

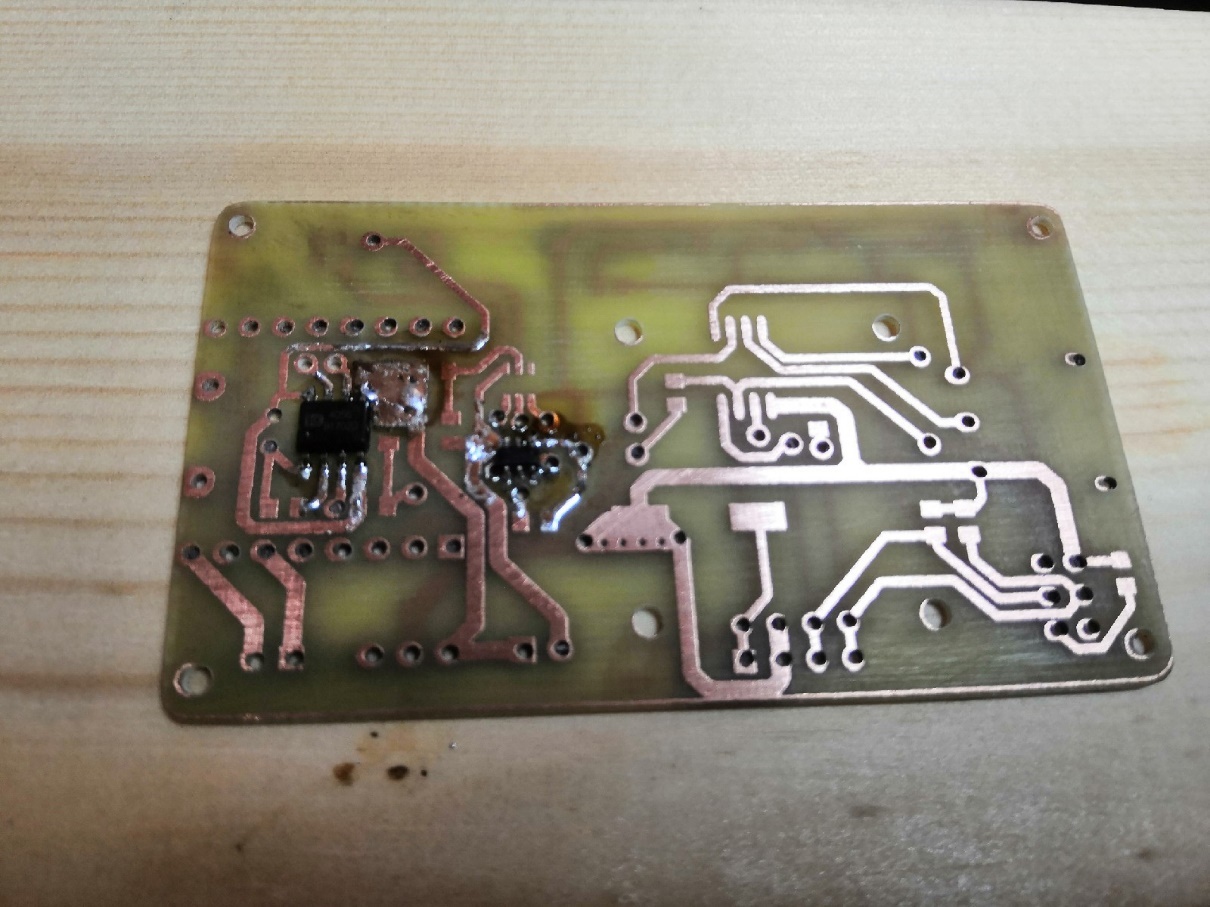


Рисунок 2.33 – Пайка компонентов

После распайки компонентов, необходимо запрограммировать устройство.

3. Технологическая часть

3.1 Программирование устройства

Заключительным этапом создания устройства является программирование. Программный код будет написан на языке C++ в среде Arduino IDE, которая распространяется бесплатно и доступна для скачивания с официального сайта www.arduino.cc. Изначально устройство намеренно задумывалось для совместимости со средой Arduino IDE, что позволит легко отладить проект, заказать следующий прототип на заводе и увидеть его работоспособность. Однако, в дальнейшем, для промышленного образца будет необходимо использовать более гибкие настройки и оптимизированный код, который будет занимать меньше места в памяти микроконтроллера, а его выполнение будет осуществляться быстрее, но эти задачи выходят за рамки дипломного проекта.

Внешний вид среды Arduino IDE показан на рисунке 3.1.



Рисунок 3.1 – Внешний вид среды Arduino IDE

Так выглядит интерфейс программы. В верхней части окна присутствует привычное меню, где можно открыть файл, настройки, выбрать плату, с которой осуществляется работа (Uno, Nano и другие), а также открыть проекты с готовыми примерами кода. Набор кнопок для работы с программой и их назначение представлены на рисунке 3.2.

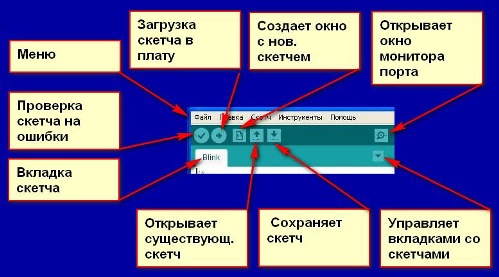


Рисунок 3.2 – Назначение кнопок среды Arduino IDE

В нижней части окна – область для вывода информации о проекте, о состоянии кода, прошивки и наличии ошибок.

Полный код программы для устройства электронный частотомер представлен в приложении А.

Для загрузки программы в микроконтроллер необходимо устройство к программатору, либо можно использовать плату Arduino, подключить к ней микроконтроллер и загрузить программу в него. Стоит отметить, что для работы с микроконтроллером ATmega ATtiny 85 в среде Arduino IDE, необходимо скачать для него соответствующую библиотеку с официального сайта arduino.cc Далее нужно выбрать используемый тип программатора и нажать кнопку «загрузить» или же сочетание клавиш «ctrl+u». После проверки программы на ошибки, будет выполнена загрузка. Теперь контроллер запрограммирован и устройство готово к использованию.

3.2 Техническое рувоводство

Разработанное устройствопредставляет из себя прототип портативного медиаплеера со следующими характиристиками:

– габариты 91х99х20мм;

– напряжение питание (зарядки) +5В;

– напряжение питания от АКБ +3.7В;

– микроконтроллер ATmega ATtiny 85;

– количество динамиков 1шт;

– количество кнопок управления 3шт и 1 колесико регулировки громкости;

– отображение информации на OLED-дисплей;

– присутствует возможность обновление прошивки;

– время автономной работы ~5 часов (зависит от установленного АКБ).

Внешний вид устройства представлен на рисунке 3.3

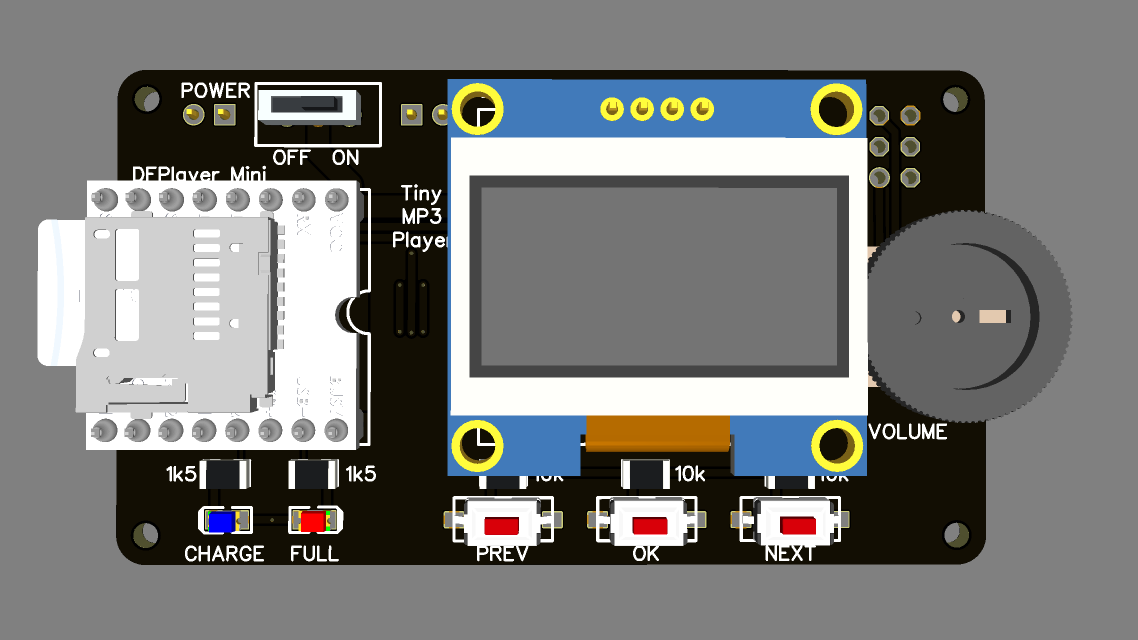


Рисунок 3.3 – Внешний вид устройства

Назначение элементов:

– клемма POWER – подключение АКБ;

– dfplayer mini – аудиомодуль с поддержкой SD-карт;

– переключатель ON-OFF – включение отключение питания;

– светодиод CHARGE – горит во время зарядки;

– светодиод FULL – горит, когда зарядка завершена;

– кнопка PREV – переключает на предыдущую аудиодорожку;

– кнопка OK – пауза/воспроизведение аудиодорожки;

– кнопка NEXT – переключает на следующую аудиодорожку;

– регулятор VOLUME – регулирует громкость аудиодорожки;

– oled-дисплей – отображает информацию о текущей аудиодорожки.

Для прослушивания музыки на устройстве необходимо скачать аудиофайлы на карту microSD, затем установить ее в разъем аудиомодуля и включить устройство. Воспроизведение начнется автоматически.

4 Экономическая часть

Основной целью экономической части является определение стоимости капиталовложений в проект.

Для успешной работы над организационно-экономической частью дипломного проекта необходимо перед началом работы иметь следующие данные:

1) перечень необходимого оборудования для реализации проекта;

2) перечень программного обеспечения для реализации проекта (если требуется условием проекта);

3) количество затраченных часов, необходимых для реализации проекта.

4.1 Определение полной себестоимости

В полную себестоимость включаются все затраты на оказание услуг: расходы на материалы, аренду помещения, оплату труда специалистов, содержание и эксплуатация оборудования, управленческие расходы.

Себестоимость рассчитывается по формуле

Сполн = Зм + ЗПосн + Надб + Осн , (4.1)

где Сполн – полная себестоимость, рублей;

Зм – материальные затраты, рублей;

ЗПосн – основная заработная плата, рублей;

Надб – надбавка, рублей;

Осн – отчисления на социальные нужды, рублей;

Ам(время) – амортизационные отчисления, рублей;

Сполн = 1033 + 9200 + 5520 + 3827 + 395 = 19975 рублей.

Материальные затраты – это затраты на приобретение основных материалов, необходимых в процессе оказания данных услуг.

Материальные затраты определяются по формуле

Зм = Нр × Цо, (4.2)

где Зм – материальные затраты, рублей за единицу продукции;

Нр – нормы расхода, штук;

Цо – оптовая цена на материал, рублей.

Расчет затрат на приобретение материалов, комплектующих изделий производится по формуле (4.2) и сведен в таблицу 4.1.

Таблица 4.1 – Затраты на материалы.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование материала | Ед. измерения | Кол.-во | Цена, руб. | Стоимость, руб. |
| ATmega ATtiny 85 | шт | 1 | 63 | 63 |
| Аудиомодуль | шт | 1 | 220 | 220 |
| OLED-дисплей | шт | 1 | 300 | 300 |
| Динамик | шт | 1 | 100 | 100 |
| Корпус | шт | 1 | 50 | 50 |
| Потенциометр | шт | 1 | 40 | 40 |
| Кварцевый резонатор 16МГц | шт | 1 | 12 | 12 |
| Конденсаторы SMD 100nf | шт | 2 | 4 | 8 |
| Конденсаторы SMD 10mf | шт | 1 | 6 | 6 |
| Резисторы SMD | компл | 1 | 60 | 60 |
| Разъемы 2.54мм | шт | 1 | 1 | 12 |
| Клеммы | шт | 4 | 9 | 36 |
| Светодиоды SMD | шт | 2 | 5 | 10 |
| Тактовые кнопки SMD | шт | 3 | 2 | 6 |
| Разъем питания | шт | 1 | 10 | 10 |
| Контроллер заряда АКБ | шт | 1 | 50 | 50 |
| Печатная плата | шт | 1 | 50 | 50 |
| Итого | | | | 1033 |

Общая стоимость проекта составила 1033 рубля.

4.2 Расчет заработной платы

Основная заработная плата – заработная плата производственных рабочих, инженерно-технического персонала и работников других категорий, непосредственно участвующих в изготовлении (установке, настройке, отладке) данной продукции.

Заработная плата определяется исходя из трудоемкости проекта и тарифной ставки (почасовой).

Основная заработная плата определяется по формуле

ЗПосн = Тст × tобщ , (4.3)

где ЗПосн – основная заработная плата, рублей;

Тст – часовая тарифная ставка руб./час;

tобщ – общая трудоемкость работ, час.

Расходы по основной заработной платы заносятся в таблицу 4.2.

Таблица 4.2 – Расходы по основной заработной плате

|  |  |  |
| --- | --- | --- |
| Наименование работ | Тарифная ставка,  руб./час | Трудоемкость,  час |
| Постановка задачи | 100 | 10 |
| Разработка алгоритма | 100 | 10 |
| Выполнение основной работы | 100 | 50 |
| Тестирование систем | 100 | 20 |
| Сдача заказчику | 100 | 2 |
| Итого | 100 | 92 |

Расходы по основной заработной плате составили: 100х92 = 9200 рублей.

Надбавка к заработной плате вычисляется формуле (4.4):

Надб = ЗПосн × (Ксев + Кр), (4.4)

где ЗПобщ – общая заработная плата, рублей;

Ксев – северный коэффициент;

Кр – районный коэффициент.

Районный коэффициент равен 30%, северный коэффициент равен 30%. Исходя из этого, по формуле (4.4) можно определить надбавку к заработной платы: 9200 х 0,6 = 5520 рублей.

Отчисления на социальные нужды - выплаты производимые предприятием во внебюджетные фонды. В эту статью включаются отчисления на социальные нужды по установленным нормам от суммы основной заработной платы и надбавки специалиста.

Отчисления на социальные нужды определяются по формуле

Осн = (ЗПосн + Надб) × Но, (4.5)

где Осн – отчисления на социальные нужды, рублей;

ЗПосн – основная заработная плата, рублей;

Набд – Надбавка, рублей;

Но – норма отчислений на социальные нужды, имеет значение 26%.

Исходя из формулы (4.5), отчисления на социальные нужды составляют: (9200 + +5520)х0,26 = 3827 рублей.

Полная заработная плата включает в себя основную и надбавку, а также отчисления на социальные нужды.

Полная заработная плата находится по формуле

ЗПполн = ЗПосн + Надб+ Осн, (4.6)

где ЗПполн – полная заработная плата, рублей;

ЗПосн – основная заработная плата, рублей;

Надб – Надбавка, рублей;

Осн – отчисления на социальные нужды, рублей.

Исходя из формулы (4.6), полная заработная плата составляет: 9200+5520+3827 = = 18547 рублей.

4.3 Расчет расходов на содержание и эксплуатацию оборудования

Амортизацией называют плановое перенесение стоимости основных фондов по мере их износа на производимую с их помощью продукцию.

Амортизационные отчисления исчисляются по каждой группе с учетом норм амортизации. Под нормой амортизации понимаются установленный в плановом порядке годовой процент стоимости основных фондов.

Амортизационные отчисления – это часть стоимости основных фондов, переносимое на единицу годовой продукции и включаемое в ее себестоимость.

Для реализации дипломного проекта был использован персональный компьютер общей стоимостью 20000р, набор электронных компонентов и материалов 2700р, принтер стоимостью 9800р и паяльник стоимостью 500р. Этого оборудования достаточно для создания прототипа устройства.

Амортизационные отчисления рассчитываются по формуле

Ам = Св × Нм, (4.7)

где Св – восстановительная стоимость оборудования, рублей;

Нм – годовая норма амортизации (для оборудования 30% годовых), проценты. Амортизационные отчисления составляют: (20000+2700+9800+500)х0,3 = 9900 р.

Амортизационные отчисления на период работ составляют

Ам(время) = (Ам/12/24/8) × tраб, (4.8)

где Ам(время) – амортизационные отчисления на период работы над проектом, рублей;

tраб – часы на работу, часов.

По формуле (4.8) определяется сумма амортизационных отчислений, которая составила: (9900/12/24/8) х 92 = 395 рублей, исходя из того, что полная трудоемкость составила 92 часа.

Смета затрат (таблица 4.3) – полный перечень затрат на производство продукции, выполнение работ, услуг.

Таблица 4.3 - Смета затрат

|  |  |  |
| --- | --- | --- |
| Статья калькуляции | Затраты, (руб) | В % к полной себестоимости |
| Расчет материальных затрат | 1033 | 5,1 |
| Расчет основной заработной платы | 9200 | 46 |
| Расчет надбавки | 5520 | 27,6 |
| Расчет отчислений на социальные нужды | 3827 | 19,1 |
| Расчет расходов на содержание и эксплуатацию оборудования | 395 | 2,2 |
| Итого: | 19975 | 100 |

Общие затраты на разработку проекта, расходы на содержание и эксплуатацию оборудования, расходы на заработную плату, социальные отчисления и содержание оборудования составили 19975 рублей. Эта сумма, которая может показаться несколько значительной, вполне оправдывает себя, так как производство подобных устройств может стоить значительно выше и уступать по качеству, судя по анализу, сделанному в ходе дипломного проекта.

Стоимость промышленных вариантов начинается от 2500, в то время как материалы для прототипа стоят 1033 рубля. Эту сумму можно снизить при оптовых заказах. Исходя из вышесказанного, разработку данного проекта можно считать актуальной и выгодной.

ЗАКЛЮЧЕНИЕ

В заключении стоит отметить, что многообразие медиаплееров для конкретных задач, можно сказать, что промышленные устройства являются востребованным оборудованием для современного рынка. Несмотря на функционал смартфонов и их растущие характеристики, у медиаплееров присутствует своя аудитория, которой требуются функции отсутствующие в смартфонах, либо реализованы не полностью. Медиаплеер достаточное сложное устройство, которое предполагает, как аппаратную, так программную часть, поэтому реализация данного проекта непосредственно демонстрирует профессиональные компетенции специалиста компьютерных систем и комплексов.

Общие затраты на разработку дипломного проекта составили 19975 рублей. Из них 1033 – стоимость материалов для самого устройства. Это сумма может показаться несколько значительной, вполне оправдывает себя, так как производство подобных устройств может стать значительно выше и уступать по качеству, судя по анализу, сделанному в ходе дипломного проекта.

Все поставленные задачи дипломного проекта, связанные с анализом существующих робототехнических решений, разработкой и изготовлением универсальной платформы, а так же описанием принципа ее работы выполнены. Цель дипломного проекта достигнута.

СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1 ГОСТ 2.105 - 95. ЕСКД. Общие требования к текстовым документам; Введ. 01.07.96. – М., 1995. – 38 с.

2 Приченко П.Г. Цифровая электроника. – СПб.: БХВ-Петербург, 2019 – 123с.

3 Проекты на Arduino, микроэлектроника [Электронный ресурс]. – Режим доступа: http://zelectro.cc/ - (Дата обращения: 01.05.2021).

4 Сара Л. Харрис Цифровая схемотехника и архитектура компьютера / Дэвид М. Харрис Сара Л. Харрис. - 2-е изд., издательство Morgan Kaufman, 2016 – 354с.

5 Сообщество Zelectro [Электронный ресурс] – Режим доступа: http://zelectro.cc (Дата обращения: 05.05.21).

6 Среда проектирования схем электрических принципиальных и печатных плат Easy EDA [Электронный ресурс] – Режим доступа: https://easyeda.com (Дата обращения: 11.05.21).

7 Форум по радиоэлектронике Espec [Электронный ресурс] – Режим доступа: http://monitor.espec.ws/section22/topic176109.html (Дата обращения: 11.05.21).

ПРИЛОЖЕНИЕ А – ЛИСТИНГ ПРОГРАММНОГО КОДА

#include <SoftwareSerial.h>

#include <Tiny4kOLED.h>

#include <DFRobotDFPlayerMini.h>

#include <avr/pgmspace.h>

#include <avr/sleep.h>

#include <avr/interrupt.h>

#include <avr/power.h>

#include <avr/wdt.h>

// Pin assignments

#define TXPIN 1 // connect to RX on DFPlayer via 1k resistor

#define RXPIN 4 // connect to TX on DFPlayer

#define POTIPIN A3 // connect to wiper of potentiometer

#define BUTTONS A0 // connect to buttons, make shure voltage

#define NEXT 552

#define OK 712

#define PREV 790

// OLED contrast levels

#define BRIGHT 127

#define DIMM 50

// Text strings stored in program memory

const char Header[] PROGMEM = "-- Tiny MP3 Player --";

const char Error1[] PROGMEM = "!!! ERROR !!!";

const char Error2[] PROGMEM = "Check SD-Card and";

const char Error3[] PROGMEM = "start again !";

const char Empty1[] PROGMEM = "!!! Battery empty !!!";

const char Empty2[] PROGMEM = "Please recharge !";

// Variables

uint8\_t filecounts; // total number of files in current folder

uint8\_t foldercounts; // total number of folders on sd-card

uint8\_t batlevel; // current battery level in percent

uint8\_t volume = 20; // current volume (0 .. 30)

uint8\_t folder = 1; // current sd-card folder

uint8\_t file = 1; // curent file in current folder

boolean pause = false; // true when player is paused

uint16\_t batcounter; // used to time battery level readings

uint16\_t lastpoti; // last value of potentiometer

SoftwareSerial tinySerial(RXPIN,TXPIN); // init serial communication

DFRobotDFPlayerMini tinyPlayer; // init DFPlayerMini

void setup() {

// reset watchdog timer

resetWatchdog (); // do this first in case WDT fires

// setup and disable ADC for energy saving

ADCSRA = bit (ADPS1) | bit (ADPS2); // set ADC clock prescaler to 64

ADCSRA |= bit (ADIE); // enable ADC interrupt

interrupts (); // enable global interrupts

power\_adc\_disable(); // turn off ADC

// init pins

pinMode (POTIPIN, INPUT);

pinMode (BUTTONS, INPUT);

// prepare and start OLED

oled.begin();

oled.setFont(FONT6X8);

oled.setContrast(BRIGHT);

oled.clear();

oled.on();

oled.switchRenderFrame();

// write start screen

oled.clear();

oled.setCursor(0, 0);

printP(Header);

oled.setCursor(0, 1);

oled.print("starting...");

oled.switchFrame();

// start communication with DFPlayer mini

tinySerial.begin(9600);

if (!tinyPlayer.begin(tinySerial)) {

oled.clear();

oled.setCursor(0, 0);

printP(Error1);

oled.setCursor(0, 2);

printP(Error2);

oled.setCursor(0, 3);

printP(Error3);

oled.switchFrame();

oled.setContrast(DIMM);

tinyPlayer.sleep();

while(true) sleep();

}

// check battery level

checkBatLevel();

// start the player

tinyPlayer.setTimeOut(500);

tinyPlayer.volume(volume);

foldercounts = tinyPlayer.readFolderCounts();

startFolderPlay();

}

void loop() {

// check player status

if (tinyPlayer.available()) {

uint8\_t type = tinyPlayer.readType();

int value = tinyPlayer.read();

switch (type) {

case DFPlayerPlayFinished:

if (file < filecounts) {

file++;

tinyPlayer.playFolder(folder, file);

updateOLED();

}

break;

default:

break;

}

}

// get, debounce and set volume

uint16\_t getpoti = denoiseAnalog(POTIPIN);

if ( (getpoti < (lastpoti - 8)) || (getpoti > (lastpoti + 8)) ) lastpoti = getpoti;

uint8\_t getvol = lastpoti / 34;

if (getvol != volume) {

volume = getvol;

tinyPlayer.volume(volume);

updateOLED();

}

// check buttons

uint16\_t buttons = getAnalog(BUTTONS);

if (buttons < 1000) {

if ( (buttons < (PREV + 20)) && (buttons > (PREV - 20)) ) {

if (folder < foldercounts) folder++; else folder = 1;

file = 1;

startFolderPlay();

}

if ( (buttons < (NEXT + 20)) && (buttons > (NEXT - 20)) ) {

if (file < filecounts) file++; else file = 1;

startFolderPlay();

}

if ( (buttons < (OK + 20)) && (buttons > (OK - 20)) ) {

pause = !pause;

if (pause) tinyPlayer.pause();

else tinyPlayer.start();

updateOLED();

}

while(getAnalog(BUTTONS) < 1000);

}

// check battery level every now and then

if ((--batcounter) == 0) checkBatLevel();

// sleep for a short time to save some energy

sleep();

}

// starts to play the actual file in the actual folder

void startFolderPlay() {

filecounts = tinyPlayer.readFileCountsInFolder(folder);

tinyPlayer.playFolder(folder, file);

pause = false;

updateOLED();

}

// prints the actual values on the OLED

void updateOLED() {

oled.clear();

oled.setCursor(0, 0);

printP(Header);

oled.setCursor(0, 1);

oled.print("Folder: ");

printDigits(folder);

oled.print(" of ");

printDigits(foldercounts);

oled.setCursor(0, 2);

oled.print("File: ");

printDigits(file);

oled.print(" of ");

printDigits(filecounts);

oled.setCursor(0, 3);

if (pause) oled.print("< Pause >");

else {

oled.print("Volume: ");

printDigits(volume);

}

oled.setCursor(78, 3);

printBatLevel();

oled.switchFrame();

}

// prints a string from progmem on the OLED

void printP(const char\* p) {

char ch = pgm\_read\_byte(p);

while (ch != 0)

{

oled.print(ch);

++p;

ch = pgm\_read\_byte(p);

}

}

// converts number to 2-digits and prints it on OLED

void printDigits(uint8\_t digits) {

if(digits < 10) oled.print(" ");

oled.print(digits);

}

// prints battery charging level on OLED

void printBatLevel() {

oled.print("Bat:");

if (batlevel <100) oled.print(" ");

if (batlevel <10) oled.print(" ");

oled.print(batlevel);

oled.print("%");

}

// checks and updates battery level; stops player if bat is empty

void checkBatLevel() {

batlevel = getBatLevel();

// if battery is empty wait for recharging

if (batlevel == 0) {

oled.setContrast(DIMM);

oled.clear();

oled.setCursor(0, 0);

printP(Empty1);

oled.setCursor(15, 2);

printP(Empty2);

oled.switchFrame();

if (!pause) tinyPlayer.pause();

while(getBatLevel() < 10) sleep();

oled.setContrast(BRIGHT);

batlevel = getBatLevel();

if (!pause) tinyPlayer.start();

}

updateOLED();

batcounter = 600;

}

// take an ADC reading in sleep mode

uint16\_t getAnalog (byte port) {

power\_adc\_enable(); // power on ADC

ADCSRA |= bit (ADEN) | bit (ADIF); // enable ADC, turn off any pending interrupt

if (port >= A0) port -= A0; // set port and

ADMUX = (port & 0x07); // reference to AVcc

set\_sleep\_mode (SLEEP\_MODE\_ADC); // sleep during sample for noise reduction

sleep\_mode(); // go to sleep while taking ADC sample

while (bitRead(ADCSRA, ADSC)); // make sure sampling is completed

uint16\_t result = ADC; // read ADC sample result

bitClear (ADCSRA, ADEN); // disable ADC

power\_adc\_disable(); // and save some energy

return result; // return value

}

// average several ADC readings to denoise

uint16\_t denoiseAnalog (byte port) {

uint16\_t result = 0;

power\_adc\_enable(); // power on ADC

ADCSRA |= bit (ADEN) | bit (ADIF); // enable ADC, turn off any pending interrupt

if (port >= A0) port -= A0; // set port and

ADMUX = (port & 0x07); // reference to AVcc

set\_sleep\_mode (SLEEP\_MODE\_ADC); // sleep during sample for noise reduction

for (uint8\_t i=0; i<8; i++) { // get 8 readings

sleep\_mode(); // go to sleep while taking ADC sample

while (bitRead(ADCSRA, ADSC)); // make sure sampling is completed

result += ADC; // add them up

}

bitClear (ADCSRA, ADEN); // disable ADC

power\_adc\_disable(); // and save some energy

return (result >> 3); // devide by 8 and return value

}

// get the battery charging level in percent by reading 1.1V reference against AVcc

uint8\_t getBatLevel() {

power\_adc\_enable(); // power on ADC

ADCSRA |= bit (ADEN) | bit (ADIF); // enable ADC, turn off any pending interrupt

ADMUX = bit (MUX3) | bit (MUX2); // set Vcc measurement against 1.1V reference

delay(2); // wait for Vref to settle

set\_sleep\_mode (SLEEP\_MODE\_ADC); // sleep during sample for noise reduction

sleep\_mode(); // go to sleep while taking ADC sample

while (bitRead(ADCSRA, ADSC)); // make sure sampling is completed

uint32\_t vcc = ADC; // read ADC sample result

bitClear (ADCSRA, ADEN); // disable ADC

power\_adc\_disable(); // and save some energy

vcc = 1125300L / vcc; // calculate Vcc in mV; 1125300 = 1.1\*1023\*1000

vcc = constrain(vcc, 3200, 4100); // 3200mV - bat empty, 4100mV - bat fully charged

uint8\_t result = (vcc - 3200) / 9; // calculate bat level in percent

return result; // return percentage value

}

// go to sleep in order to save energy, wake up again by watchdog timer

void sleep() {

set\_sleep\_mode (SLEEP\_MODE\_PWR\_DOWN); // set sleep mode to power down

bitSet (GIFR, PCIF); // clear any outstanding interrupts

power\_all\_disable (); // power off ADC, Timer 0 and 1, serial interface

noInterrupts (); // timed sequence coming up

resetWatchdog (); // get watchdog ready

sleep\_enable (); // ready to sleep

interrupts (); // interrupts are required now

sleep\_cpu (); // sleep

sleep\_disable (); // precaution

power\_all\_enable (); // power everything back on

power\_adc\_disable(); // except ADC

}

// reset watchdog timer

void resetWatchdog () {

MCUSR = 0; // clear various "reset" flags

WDTCR = bit (WDCE) | bit (WDE) | bit (WDIF); // allow changes, disable reset,

WDTCR = bit (WDIE) | bit (WDP1); // set interval to 64 milliseconds

wdt\_reset(); // pat the dog

}