МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования
«Гомельский государственный университет
имени Франциска Скорины»

Факультет физики и информационных технологий
Кафедра общей физики

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проекту
на тему

ЛОГИЧЕСКИЙ АНАЛИЗАТОР ПАКЕТОВ ИНТЕРФЕЙСА USB

ГГУ 1-39 03 02 КП

Исполнитель:
Студент группы МС-32: _____ Воевода А. А.

Научный руководитель:
Старший преподаватель
Кафедры общей физики _____ Подалов М. А.

Гомель 2023

СОДЕРЖАНИЕ

Введение

USB (Universal Serial Bus – «универсальная последовательная шина») – последовательный интерфейс для подключения периферийных устройств к вычислительной технике [1]. В настоящее время данный интерфейс есть практически в каждом устройстве.

Основными областями применения USB являются: устройства ввода (клавиатуры, мыши и т.п.), принтеры, сканеры, аудиоустройства (колонки, микрофоны и т.п.), устройства хранения (жесткие диски, флэш-карты и т.п.), игровые устройства (джойстики). Однако, это ещё не полный список сфер и устройств, в которых может применяться интерфейс USB [1].

Анализаторы пакетов интерфейса USB обычно используются разработчиками USB устройств и программистами для отладки оборудования и драйверов. В данном случае, вопрос анализа пакетов рассматриваемого интерфейса довольно актуален в силу его распространенности.

Цель: разработать электронное мобильное устройство, которое позволит анализировать, хранить и выводить данные о пакетах интерфейса USB.

Задачи:

– Произвести анализ предметной области. Рассмотреть особенности и структуру интерфейса USB;

– Рассмотреть перечень отладочных плат и микроконтроллеров, которые могут быть использованы для реализации логического анализатора;

– Произвести описание архитектуры и устройства выбранной отладочной платы или микроконтроллера, рассмотреть ее особенности.

– Разработать структурную схему логического анализатора;

– Разработать принципиальную электрическую схему логического анализатора;

– Разработать модель и алгоритм функционирования логического анализатора пакетов USB;

1 Анализ предметной области

1.1. Общая информация

1.1.1. Интерфейс USB

Как было сказано ранее, USB — это последовательный интерфейс для подключения периферийных устройств к вычислительной технике. Помимо обмена данными, интерфейс обеспечивает электропитание периферийного устройства.

Первые спецификации для USB 1.0 были представлены в 1994 – 1995 годах. Кабель USB (до 2.0 включительно) состоит из четырех медных проводников, их описание приведено в таблице 1.

Таблица 1 – Состав кабеля интерфейса USB 2.0

Провод	Сигнал	Цвет	Описание
1	VCC	красный	+5 В
2	D-	белый	Data-
3	D+	зеленый	Data+
4	GND	черный	Земля

При помощи кабеля формируется интерфейс между USB-устройством и USB-хостом. Хостом обычно является программно-управляемый USB-контроллер. За счет него обеспечивается функционирование всего интерфейса, так как USB-устройство не имеет возможности передавать данные, пока оно не будет опрошено USB-хостом [1].

Интерфейс USB 2.0 поддерживает работу в трех режимах:

- Высокая скорость (High Speed) – до 480 Мбит/с;
- Полная скорость (Full Speed) – до 12 Мбит/с;
- Низкая скорость (Low Speed) – до 1.5 Мбит/с [1].

Так как некоторым устройствам может требоваться питание от интерфейса USB, любому устройству гарантируется ток до 100 мА, а после того, как происходит согласование с хост-контроллером, ток увеличивается до 500 мА. Рассмотрим физическую реализацию подключения устройств помощью интерфейса USB 2.0 [2].

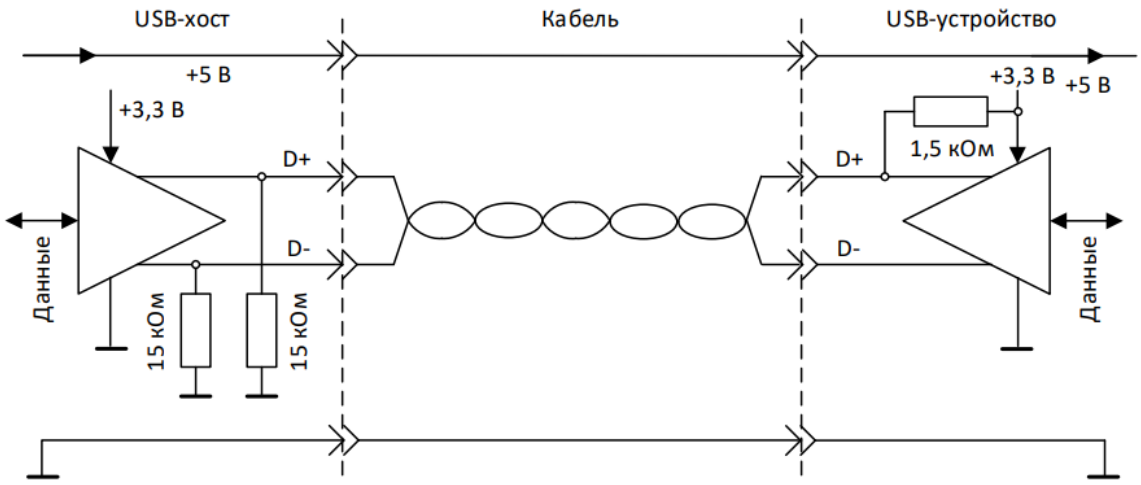


Рис. 1.1.1. Упрощенная схема подключения устройства к USB-хосту (режимы Full Speed и High Speed)

Интерфейс USB 2.0 предусматривает два режима электропитания подключаемых устройств: до 100 мА непосредственно после подключения устройства (Low-power), до 500 мА – после согласования с USB-хостом (High-power), при напряжении 5 В. Если к хосту ничего не подключено, D+ и D- подтянуты резисторами 15 кОм к минусу питания. При подключении устройства одна из линий подтягивается к +3.3 В через резистор 1.5 кОм: D+ – для Full Speed и High Speed, D- – для Low Speed [2].

Таблица 2 – Способы передачи данных интерфейса USB 2.0

Передаваемый дифференциальный сигнал	Уровень линии D+	Уровень линии D-
Diff0	< 0.3 В	> 2.8 В
Diff1	> 2.8 В	< 0.3 В
SE0	< 0.8 В	< 0.8 В

Таблица 3 – Сигналы для кодирования состояний шины в зависимости от скорости передачи данных

Состояние/Скорость	Low Speed	Full Speed	High Speed
J (data J state)	Diff0	Diff1	Diff1
K (data K state)	Diff1	Diff0	Diff0
Состояние покоя	Длительное состояние Diff0	Длительное состояние Diff1	SEO

Следует понимать, что при передаче данных по шине USB 2.0 используется принцип кодирования NRZI (Non-return-to-zero, без возврата к нулю с инверсией). Каждому нулевому биту входных данных соответствует изменение состояния шины (J→K, K→J), а при передаче единичного бита изменений нет. Для исключения потери синхронизации на длительных единичных последовательностях применяется принудительная вставка в поток данных нуля, на каждые 6 бит подряд. NRZI-кодировка снижает вероятность ошибки благодаря уменьшению изменений количества состояний, снижает энергопотребление, но усложняет сохранение синхронизации между хостами [2].

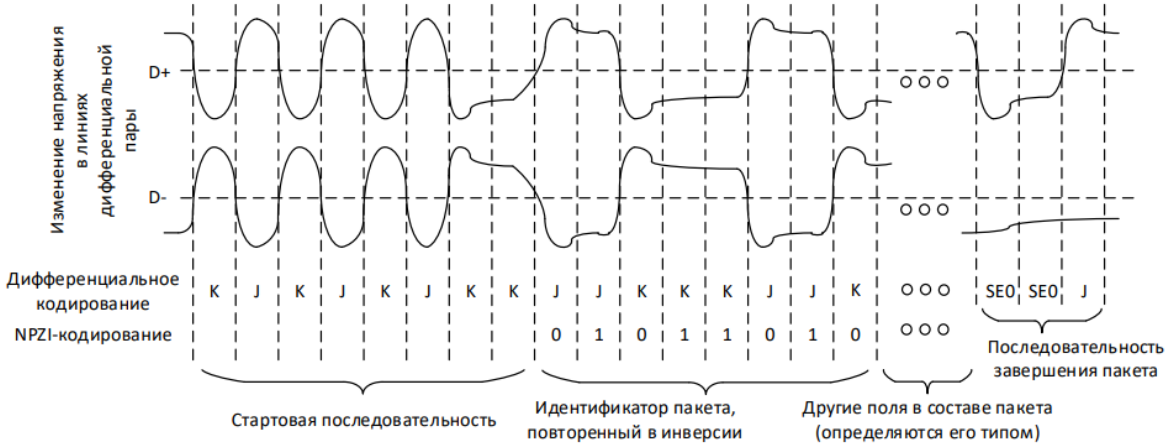


Рис. 1.1.2. Принцип формирования сигналов для передачи данных по стандарту USB 2.0 с использованием NRZI-кодировки

Если рассмотреть логический уровень интерфейса USB, то можно сказать о том, что на нем поддерживаются транзакции приема и передачи данных. Каждый такой пакет для каждой транзакции содержит номер оконечной точки (endpoint). Этот номер позволяет USB-хосту различать USB-устройства и организовывать передачу данных с несколькими USB-устройствами одновременно. Драйвера в ядре операционной системы хоста читают с устройства список оконечных точек и создают управляющие структуры данных.

Вся информация по интерфейсу USB передается кадрами, которые отправляются через равные промежутки времени.

Кадр 1	Кадр 2	…	Кадр N

Каждый кадр состоит из транзакций. Помимо транзакций, каждый кадр включает в себя пакет SOF (Start Of Frame) и EOF (End Of Frame):

SOF	Транзакция 1	Транзакция 2	…	Транзакция 3	EOF

Каждая транзакция имеет вид:

Token	Data	Status

Token-пакет содержит информацию об адресе устройства USB и номер конечной точки, которой предназначена транзакция, хранится информация о типе транзакции. Data-пакет – содержит в себе непосредственно данные, которые передает хост, либо конечное устройство. Status-пакет – предназначен для проверки успешности получения данных.

Перейдем к более подробному рассмотрению Token-пакета:

Sync	PID	Address	Endpoint	CRC	EOP

Поля Address и Endpoint – содержат адрес USB-устройства и номер конечной точки. Поле CRC – это контрольная сумма, которая позволит понять, что все данные были переданы успешно.

Таблица 2 – Типы Token-пакетов

Тип пакета	PID	Описание
IN	0001	Сообщает устройству, что хост готов принимать от него информацию.
OUT	1001	Сообщает хосту о готовности устройства передать данные.
SETUP	1101	Необходим для использования управляющих передач
SOF	0101	Используется для инициализации начала кадра

Следует отметить, что поле PID включает в себя 4 бита, но при передаче к ним добавляется еще 4 бита, которые были получены путем инвертирования первых 4-х бит.

Рассмотрим более подробно структура Data-пакета:

Sync	PID	Data	CRC	EOP

Структура схожа со структурой Token-пакета. В поле Data находятся передаваемые данные.

Структура Status-пакета выглядит следующим образом:

Sync	PID	EOP

В данном случае поле PID может принимать только два значения: если пакет принят корректно – PID = 0010; если при приеме пакета возникла ошибка – PID = 1010.

Осталось рассмотреть структуру Start Of Frame пакета:

Sync	PID	Frame	CRC	EOP

Структура схожа со структурами пакетов, рассмотренных выше. Отличием является лишь поле Frame, которое содержит в себе номер передаваемого кадра.

1.1.2. Логический анализатор

Логический анализатор – это электронный прибор, который может записывать и отображать последовательности цифровых сигналов. Такие устройства используют для тестирования и отладки цифровых электронных схем. Если сравнить логический анализатор с таким прибором, как осциллограф, то можно выявить ряд отличий: логический анализатор имеет значительно больше входов, но, в отличии от осциллографа, может показывать лишь два уровня сигнала «0» и «1» [3].

Функции и возможности логического анализатора пакетов USB:

1. Захват и отображение USB-трафика: Логический анализатор USB может записывать и отображать все USB-пакеты, передаваемые между устройствами USB и хост-системами, включая пакеты управления, данные и пакеты протокола USB.
2. Декодирование USB-протокола: Логический анализатор USB может декодировать USB-протокол, расшифровывая биты, байты и поля данных, а также анализируя структуру USB-пакетов и состояние линий данных USB.
3. Анализ и фильтрация USB-трафика: Логический анализатор USB может выполнять фильтрацию и анализ USB-трафика на основе определенных условий, таких как адрес устройства, endpoint, тип транзакции, данные и т. д., что позволяет разработчикам искать и анализировать определенные пакеты или события.
4. Тайминг и анализ времени: Логический анализатор USB может предоставлять информацию о времени передачи USB-пакетов, задержках, времени ожидания, переключении состояний и других тайминговых параметрах, что позволяет анализировать и оптимизировать производительность и временные характеристики USB-устройств.

5. Графическое представление и анализ: Логический анализатор USB может предоставлять графические интерфейсы и инструменты для визуализации и анализа USB-трафика, такие как временные диаграммы, диаграммы состояний, таблицы событий и другие инструменты, упрощающие анализ и отладку USB-протокола.

Рассмотрим некоторые параметры логических анализаторов [4].

Таблица 3 – Параметры логических анализаторов

Параметр	Описание
Частота дискретизации	Один из важнейших параметров. Он характеризует время между отсчетами логических состояний. Из него вытекает ограничение на максимальную возможную частоту анализируемого сигнала.
Количество каналов	Чем больше количество независимых входов для измерения логических состояний, тем лучше. Это может позволить производить более детальный и подробный анализ, охватывая множество цифровых сигналов. Однако, увеличение числа каналов может приводить к снижению частоты дискретизации.
Пропускная способность	Параметр, который так же может влиять на частоту дискретизации. Даже если измерения на каждом канале будет происходить синхронно и независимо, при слишком большом количестве каналов частота дискретизации будет снижаться из-за недостаточной пропускной способности.
Внутренняя память	Этот параметр позволяет записывать данные, полученные с каналов логического анализатора. Если при использовании логического анализатора требуется записывать длинные последовательности данных для дальнейшего анализа, то данный параметр будет являться одним из важнейших.
Наличие защитных цепей	Это может обезопасить выход микросхемы анализатора из строя в случае попадания высокого напряжения на входные разъемы в процессе анализа цифровых сигналов.
Программное обеспечение	Если логический анализатор не является портативным и не имеет собственного экрана для отображения информации, этот параметр будет играть значительную роль.

1.1.3. Протокол I2C

Протокол I2C (Inter-Integrated Circuit) является серийным синхронным протоколом связи между микроконтроллерами, микропроцессорами и другими интегральными схемами. Он был разработан компанией Philips (сейчас NXP Semiconductors) и предназначен для передачи данных между устройствами внутри одной платы или между разными платами, подключенными к одной шине [5].

Протокол I2C основан на двухпроводной архитектуре, где используется всего две линии для связи между устройствами: линия данных (SDA - Serial Data) и линия тактирования (SCL - Serial Clock). Он поддерживает множество устройств, подключенных к одной шине, каждое из которых имеет свой уникальный адрес.

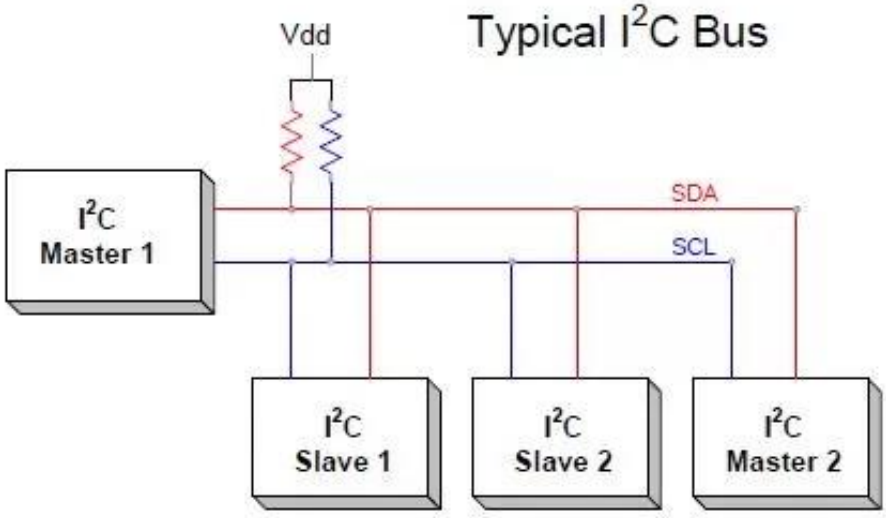


Рис.1.1.3. Схемотехника шины I2C

Основные характеристики протокола I2C:

1. Мастер-слейв архитектура: в протоколе I2C одно или несколько устройств выступают в роли мастера, который инициирует передачу данных, и одно или несколько устройств выступают в роли слейвов, которые отвечают на команды мастера или передают данные мастеру.

2. Битовая ориентация: данные передаются по одному биту за раз в последовательной форме. Биты передаются по линии данных (SDA) нарастающим фронтом тактового сигнала (SCL).

3. Синхронизация по тактовому сигналу: передача данных осуществляется синхронно по тактовому сигналу (SCL), который генерирует мастер. Все

устройства на шине синхронизируют свои операции по этому тактовому сигналу.

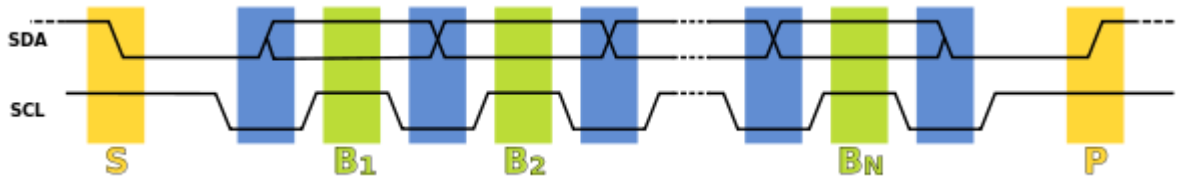


Рис. 1.1.4. Тактирование последовательности передачи данных

4. Адресация: каждое устройство на шине имеет свой уникальный адрес, который используется мастером для выбора конкретного устройства, с которым он хочет обмениваться данными. Каждый адрес идентифицируются уникальными 7-битными или 10-битными адресами, которые указывают на конкретное устройство на шине. Мастер-устройство инициирует передачу данных к определенному адресу слейва.

5. Различные режимы работы: протокол I2C поддерживает различные режимы передачи данных, включая режимы чтения, записи и чтения-записи.

6. Надежность: протокол I2C включает в себя механизмы обнаружения ошибок для индикации ошибки при приеме данных.

1.2. Обзор платформ для разработки

1.2.1. Отладочная плата Arduino на базе Atmega328

Отладочная плата Arduino на базе микроконтроллера Atmega328 представляет собой плату разработки, которая используется для программирования и отладки проектов на платформе Arduino [6]. Она оснащена микроконтроллером Atmega328, который является одним из наиболее распространенных микроконтроллеров, используемых в платформе Arduino.

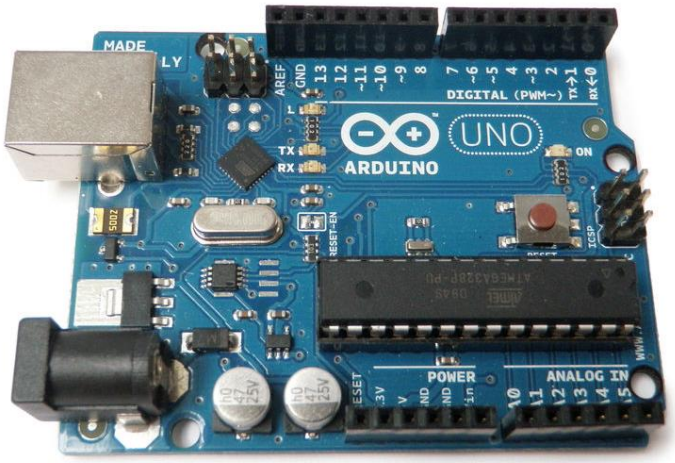


Рис. 1.2.1. Отладочная плата Arduino Uno

Отладочная плата Arduino на базе Atmega328 имеет следующие основные характеристики:

1. Микроконтроллер Atmega328: основной элемент платы - микроконтроллер Atmega328, который работает на частоте 16 МГц и имеет 32 килобайта внутренней флеш-памяти, 2 килобайта оперативной памяти (RAM) и 1 килобайт энергонезависимой памяти (EEPROM).
2. Входы/выходы: плата имеет ряд цифровых и аналоговых входов/выходов, которые могут использоваться для подключения различных устройств и датчиков. Всего на плате есть 14 цифровых пинов, из которых 6 можно использовать в качестве ШИМ-выходов, и 6 аналоговых входов.
3. Интерфейсы: плата оснащена различными интерфейсами, такими как UART (Serial), I2C и SPI, что позволяет подключать различные устройства,

такие как датчики, дисплеи, модули связи и другие, для взаимодействия с микроконтроллером.

4. Питание: отладочная плата питается от внешнего источника питания напряжением 5 В, которое может быть подано через разъем питания или подано напрямую на контакт Vcc. Также на плате есть регулятор напряжения, который позволяет подавать на плату более высокое напряжение (до 12 В), которое будет автоматически преобразовано в 5 В для питания микроконтроллера и других компонентов на плате.

5. Программирование: плата может быть программирована с использованием Arduino IDE (интегрированная среда разработки), которая предоставляет удобный и простой в использовании интерфейс для создания и загрузки программ на микроконтроллер.

1.2.2. Отладочная плата Discovery на базе STM32

Отладочная плата Discovery на базе STM32 (или просто STM32 Discovery) - это плата разработки, предназначенная для работы с микроконтроллерами STM32, которые производит компания STMicroelectronics. Она предоставляет разработчикам удобное и мощное средство для создания и отладки встраиваемых систем на базе STM32 микроконтроллеров.

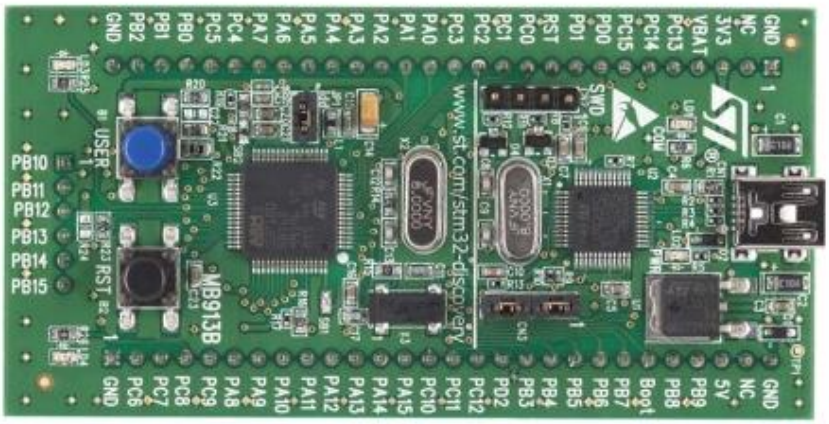


Рис. 1.2.2 Отладочная плата STM32 Discovery

Основные особенности отладочной платы Discovery на базе STM32 включают:

1. Микроконтроллер STM32: отладочная плата Discovery на базе STM32 обычно оснащена микроконтроллером STM32Fxxx серии, где xxx обозначает

различные модели микроконтроллеров STM32, такие как STM32F0, STM32F1, STM32F3, STM32F4 и т.д. В зависимости от модели микроконтроллера, отладочная плата может иметь различные характеристики, такие как частота работы, объем встроенной памяти, наличие периферийных устройств и т.д.

2. Встроенные модули и периферийные устройства: отладочная плата STM32 Discovery может содержать различные встроенные модули и периферийные устройства, такие как кнопки, светодиоды, дисплеи, датчики, интерфейсы связи (например, UART, SPI, I2C, USB) и т.д. Эти модули и устройства предоставляют разработчикам удобные средства для тестирования и отладки кода на микроконтроллере.

3. Отладочные интерфейсы: отладочная плата STM32 Discovery может иметь различные отладочные интерфейсы, такие как JTAG (Joint Test Action Group) или SWD (Serial Wire Debug), которые позволяют разработчикам осуществлять отладку микроконтроллера в реальном времени, мониторить и изменять его состояние, а также загружать и запускать программное обеспечение на микроконтроллере.

4. Поддержка интегрированной среды разработки: отладочная плата STM32 Discovery часто поддерживается интегрированной средой разработки (IDE) от STMicroelectronics, такой как IAR Embedded Workbench, Keil MDK, или STM32CubeIDE, что упрощает процесс разработки, отладки и тестирования кода для STM32 микроконтроллеров.

1.2.3. Отладочная плата ESP32

Отладочная плата ESP32 — это плата разработки, основанная на микроконтроллере ESP32, который является мощным и гибким решением для разработки приложений Интернета вещей (IoT). Отладочная плата ESP32 предназначена для упрощения процесса разработки и отладки приложений на базе ESP32.

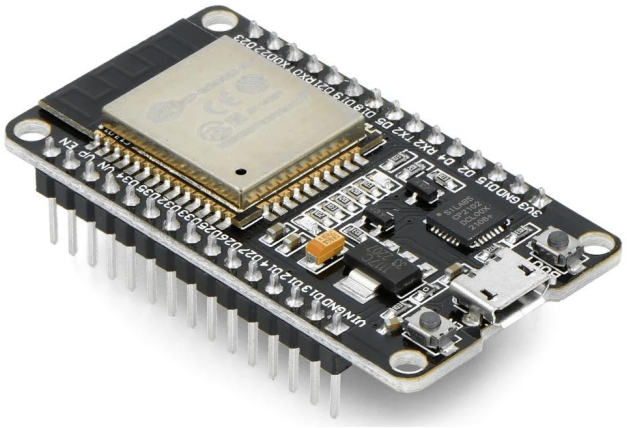


Рис. 1.2.3. Отладочная плата ESP32

Отладочная плата ESP32 обычно имеет следующие характеристики:

1. Микроконтроллер ESP32: отладочная плата содержит микроконтроллер ESP32, который предоставляет множество возможностей, таких как двухъядерный процессор, высокоскоростное подключение Wi-Fi и Bluetooth, множество цифровых и аналоговых входов/выходов, аппаратную поддержку протоколов шифрования и многое другое.

2. Разъемы ввода/вывода (I/O): отладочная плата ESP32 обычно содержит разъемы для подключения внешних устройств и датчиков через цифровые и аналоговые входы/выходы. Это позволяет разработчикам подключать различные периферийные устройства, такие как датчики температуры, давления, освещенности и т.д., для тестирования и отладки своих приложений.

3. Интерфейсы связи: отладочная плата ESP32 обычно имеет различные интерфейсы связи, такие как UART, SPI, I2C, GPIO и др., которые позволяют подключаться к другим устройствам и модулям, таким как дисплеи, сенсоры и другие.

4. Программатор/отладчик: отладочная плата ESP32 может быть оснащена встроенным программатором/отладчиком, который обеспечивает возможность загрузки программного кода на микроконтроллер ESP32, отладки и мониторинга его работы в режиме реального времени. Это облегчает процесс разработки и отладки приложений на ESP32.

2 Разработка электрических схем логического анализатора

2.1. Разработка структурной схемы логического анализатора

2.1.1. Обоснование базовых блоков структурной электрической схемы логического анализатора

Рассмотрим каждый из базовых блоков структурной электрической схемы, приведенной в Приложении А.

1. USB-хост является важной составляющей, так как благодаря ему возможен обмен данными через интерфейс USB. Он инициирует опрос USB-устройства, после чего начинается пересылка данных, которые и должен отслеживать логический анализатор пакетов интерфейса USB. В данном случае USB-хост не является конкретно определенным, важно лишь его наличие и исправная работа.

2. USB-устройство является второй важной составляющей. Оно подключается к USB-хосту и после того, как USB-хост установит связь, начинает принимать и отправлять данные по интерфейсу USB. USB-устройство так же может не являться каким-либо конкретным периферийным устройством. Однако, стоит отметить, что различные периферийные USB-устройства могут работать в разных скоростных режимах интерфейса USB, что будет влиять на анализ пакетов.

3. ESP32 – отладочная плата с микроконтроллером ESP-WROOM32. Будет использована для того, чтобы отслеживать изменения уровней напряжения на соответствующих проводах интерфейса USB, использующихся

для передачи данных: D+ и D-. Так как ESP32 работает при напряжении 3.3 В, этого будет достаточно для того, чтобы отследить изменения напряжения от 0.3 В до 2.8 В, которые будет происходить при передаче данных по интерфейсу USB.

4. I2C коннектор – базовый блок структурной схемы, который будет использоваться для связи отладочной платы ESP32 и LCD-дисплея, на который будет выводится информация. В рамках данного курсового проекта будет использоваться I2C коннектор PCF8574.

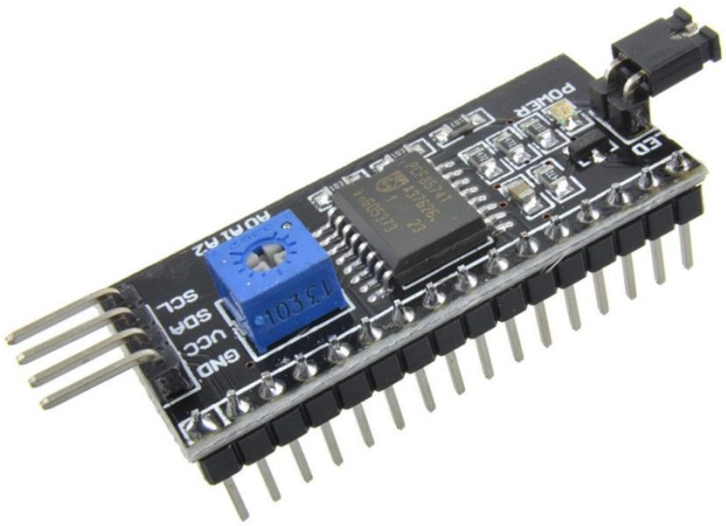


Рис. 2.1.1. I2C коннектор PCF8574

PCF8574 - это 8-битный расширитель портов ввода-вывода (I/O), который позволяет управлять до 8 периферийными устройствами через двухпроводную последовательную шину I2C. Кроме того, данный компонент имеет встроенный прерыватель, что обеспечивает более эффективную работу микроконтроллера в системах, где прерывания являются необходимым элементом.

PCF8574 имеет два I2C адреса: 0x20 и 0x38. Подключение к шине I2C происходит через два вывода: SDA (Serial Data) и SCL (Serial Clock). Коннектор I2C PCF8574 имеет 4 вывода: VCC, GND, SDA и SCL. Вывод VCC подключается к питанию, а GND - к земле. Выводы SDA и SCL подключаются к соответствующим выводам на плате контроллера или другого устройства, которое будет использоваться в качестве мастера на шине I2C.

После подключения PCF8574 к шине I2C и подачи питания можно начать работу с расширителем портов. При использовании данного компонента необходимо записывать данные в соответствующие регистры,

чтобы управлять состоянием выводов. Например, чтобы установить на выходе 1-й вывод значение логической единицы, необходимо записать байт со значением 0x01 в регистр PCF8574.

В целом, коннектор I2C PCF8574 - это удобный и простой способ управления до 8 периферийными устройствами через I2C-шину, что позволяет сократить количество проводов и упростить процесс подключения. Именно поэтому в нашем проекте будем использовать I2C коннектор для связи микроконтроллера и LCD-дисплея.

5. LCD-дисплей – будет использован для вывода информации. В качестве LCD-дисплея выбрана модель 1602. LCD-дисплей 1602 - это текстовый жидкокристаллический дисплей, который состоит из двух строк по 16 символов в каждой. Он может отображать текст и символы, а также имеет возможность управления яркостью подсветки.

Рис. 2.1.2. LCD-дисплей 1602

Для соединения LCD-дисплея с микроконтроллером используется пара проводов для передачи данных (Data) и управляющие сигналы (RS, RW, E):

- RS (Register Select) определяет режим передачи данных: командный или символьный.
- RW (Read/Write) определяет направление передачи данных - на запись или на чтение.
- E (Enable) используется для управления логическим уровнем и синхронизации передачи данных.

LCD-дисплей 1602 используется во многих электронных устройствах, таких как цифровые часы, термометры, вольтметры и другие приборы. Он легко программируется и может быть использован для отображения информации о состоянии устройства, результатах измерений, вывода текстовых сообщений и т.д.

2.1.2. Обоснование связей структурной электрической схемы логического анализатора.

Связей в структурной схеме логического анализатора не так много. Во-первых, нам необходимо связать микроконтроллер и анализируемое устройство. Данная связь является основной, и при ее отсутствии логический анализатор не сможет выполнить своих функций. В данном проекте анализируются пакеты интерфейса USB 2.0, поэтому необходимо организовать связь микроконтроллера с двумя проводами, предназначенных для передачи данных. Реализовывать это будем посредствам GPIO.

GPIO (General-Purpose Input/Output) - это универсальные входы/выходы, которые могут быть использованы для соединения с различными электронными компонентами и управления ими.

В общем случае, GPIO - это порты ввода/вывода на микроконтроллерах или микропроцессорах, которые позволяют им взаимодействовать с внешними устройствами. Каждый GPIO имеет два состояния - логический уровень "0" или "1". В зависимости от режима работы порта, он может либо считывать входные сигналы, либо генерировать выходные сигналы.

В современных микроконтроллерах GPIO-порты могут выполнять различные функции, например, они могут использоваться для работы с UART, SPI, I2C и другими интерфейсами связи. Также порты могут быть настроены на работу с различными типами сигналов, например, аналоговыми или цифровыми.

В данном случае нас интересуют GPIO-порты ESP32, которые настроены на работу с аналоговым сигналом.

Во-вторых, необходимо реализовать связь ESP32 и I2C коннектора. Для этого нам потребуется задействовать GPIO-порт ESP32, который мы будем использовать для работы с интерфейсом связи I2C.

В-третьих, I2C коннектор будет связан с LCD-дисплеем, взаимодействия будут происходить посредством I2C протокола.

2.2. Разработка принципиальной схемы логического анализатора

2.2.1. Обоснование выбора САПР для разработки принципиальной электрической схемы логического анализатора

Рассмотрим две основные САПР, которые позволяют разрабатывать принципиальные электрические схемы различных устройств:

KiCad (от англ. "Ki" - ключ, "CAD" - компьютерное проектирование) - это бесплатная и открытая система автоматизированного проектирования электронных схем и печатных плат. Она предоставляет пользователям полный набор инструментов для создания схем, размещения компонентов, маршрутизации проводников и создания производственных файлов для изготовления печатных плат. KiCad поставляется с библиотеками компонентов, моделями компонентов и футпринтами для более чем 750 000 электронных компонентов. KiCad также поддерживает многоязычную локализацию и доступен на нескольких языках. KiCad доступен для Windows, Linux и macOS.

KiCad имеет следующие основные функции:

- Создание схем
- Размещение компонентов
- Маршрутизация проводников
- Создание производственных файлов для изготовления печатных плат
- Просмотр 3D моделей компонентов
- Импорт/экспорт схем и печатных плат в различных форматах
- Создание пользовательских библиотек компонентов

KiCad предоставляет несколько основных компонентов для проектирования электронных схем и создания печатных плат. Эти компоненты включают в себя:

- Eeschema - для создания схематических диаграмм;
- CvPcb - для размещения компонентов на плате;

- Pcbnew - для проектирования печатной платы и проведения трассировки;
- Gerbview - для просмотра файлов герберов и документации.

KiCad поддерживает несколько форматов файлов для импорта и экспорта, включая SPICE, netlist, STEP, DXF, Gerber и др. KiCad является бесплатной и открытой системой автоматизации проектирования электронных схем, и позволяет пользователям создавать и разрабатывать электронные устройства без затрат на лицензирование коммерческого программного обеспечения.

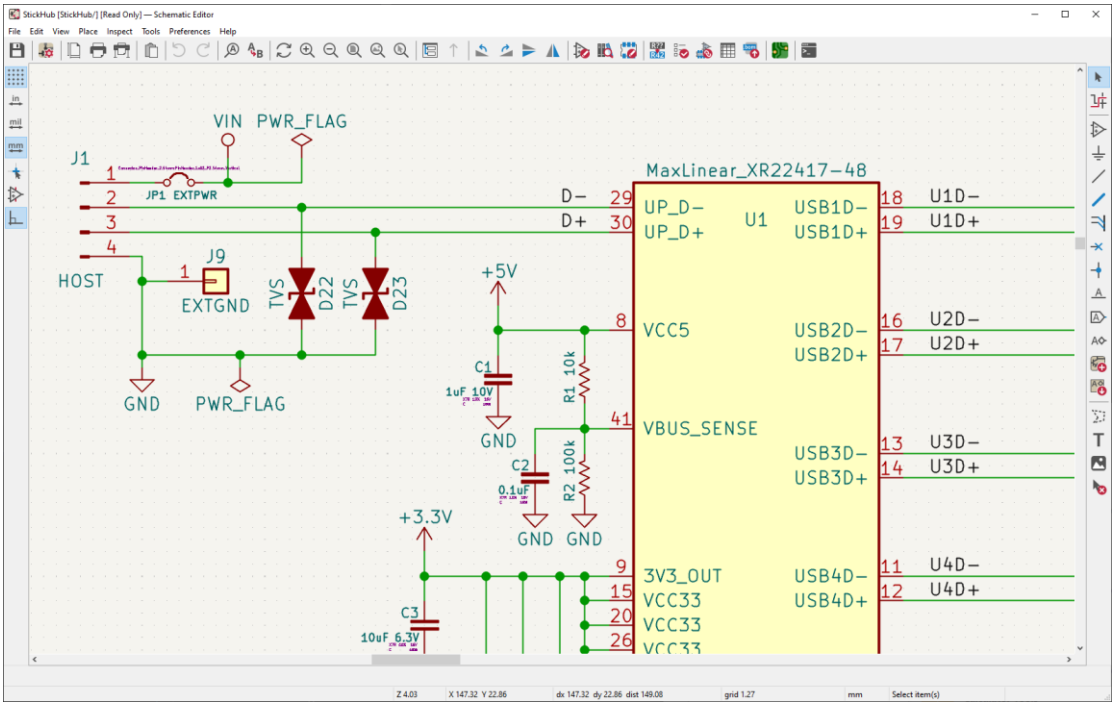


Рис. 2.2.1. Интерфейс САПР KiCad

Proteus - это программное обеспечение для проектирования электронных схем и имитации их работы. Оно позволяет проектировать, тестировать и отлаживать электронные схемы на виртуальном уровне, не требуя физической реализации на печатной плате.

Proteus состоит из двух основных модулей: ISIS (Intelligent Schematic Input System) и ARES (Advanced Routing and Editing Software). ISIS предназначен для создания схем, включая различные элементы, соединения и другие детали. ARES предназначен для размещения элементов на плате, настройки параметров печатных дорожек и т.д.

Одним из главных преимуществ Proteus является возможность имитирования работы электронных схем. Это позволяет выявлять ошибки и

уязвимости в проекте еще на этапе проектирования, что значительно снижает затраты на производство и отладку.

Кроме того, Proteus содержит библиотеку компонентов, которая включает в себя более 8 тысяч элементов, что делает его удобным инструментом для проектирования широкого спектра электронных устройств.

Программа также поддерживает экспорт файлов проектов в различные форматы, такие как Gerber, Excellon и другие, что облегчает процесс производства печатных плат.

Proteus доступен в двух версиях: Proteus Professional и Proteus Design Suite. Proteus Professional включает в себя дополнительные возможности, такие как моделирование систем автоматического управления, имитация протоколов передачи данных и другие, в то время как Proteus Design Suite является более базовой версией программного обеспечения.

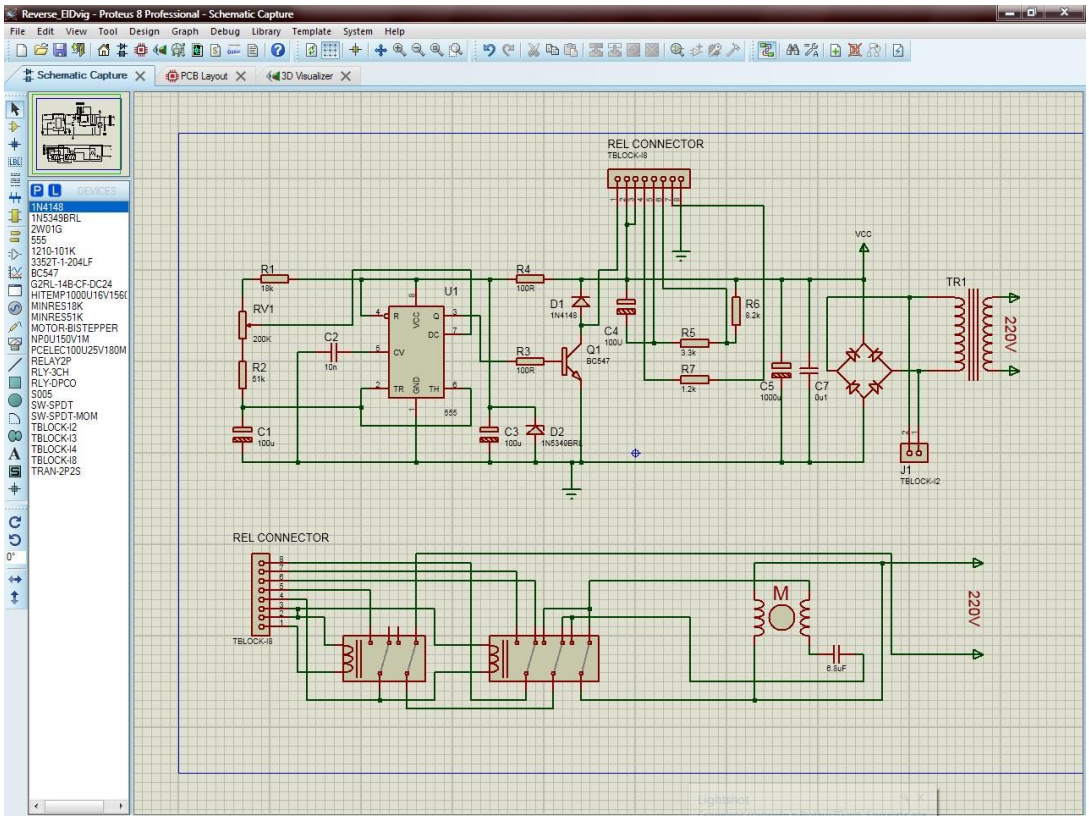


Рис. 2.2.2. Интерфейс САПР Proteus

Мы рассмотрели две мощные САПР для проектирования принципиальных электрических схем, каждая из них имеет свои преимущества и недостатки. Однако мы остановим свой выбор на САПР

KiCad, в виду ее широкой функциональности, кроссплатформенности и бесплатного распространения.

2.2.2. Описание используемых библиотечных элементов

Компонент ESP32-WROOM из библиотеки компонентов ESP32-footprints-Shem-Lib. ESP32-WROOM - это Wi-Fi и Bluetooth модуль на базе микроконтроллера ESP32, который разработала компания Espressif Systems. Он используется в различных проектах для добавления беспроводной связи и управления устройствами.

В KiCad доступна библиотека компонентов для ESP32-WROOM, которая содержит символы, модели и образцы печатных плат. В этой библиотеке представлены различные варианты ESP32-WROOM, в том числе с разным количеством контактов и конфигурацией выводов.

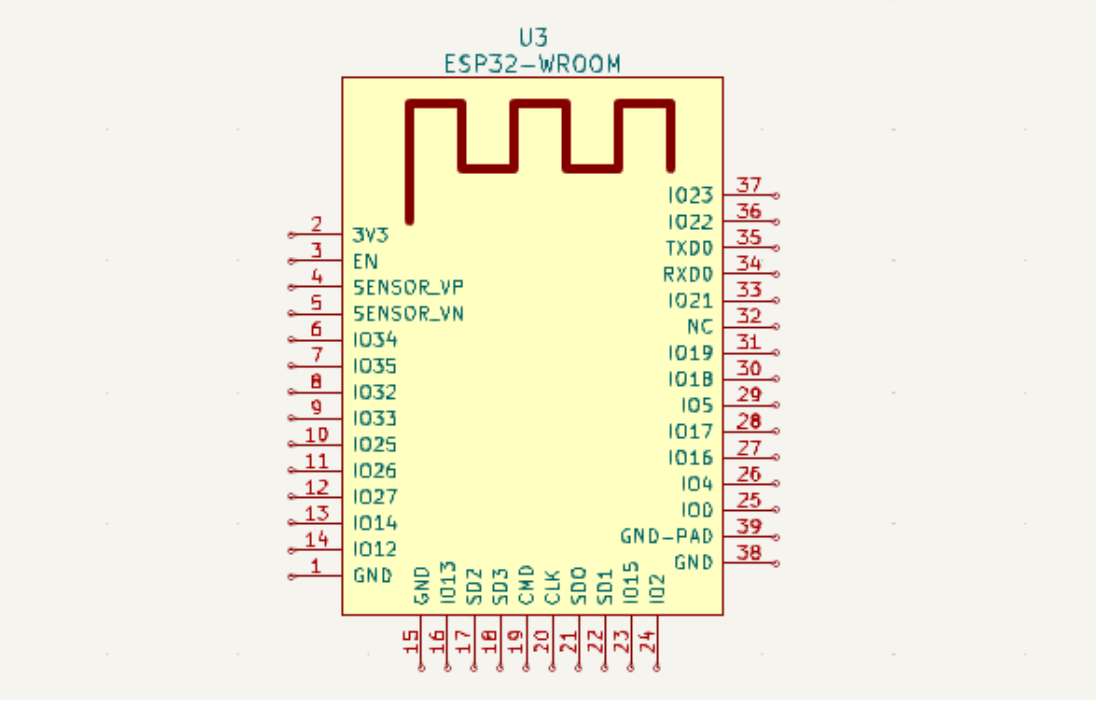


Рис. 2.2.3. Компонент ESP32-WROOM

В рамках данного курсового проекта, компонент ESP32-WROOM отлично подходит для того, чтобы использовать его при создании принципиальной электрической схемы логического анализатора, так как он

полностью соответствует отладочной плате ESP32, которая будет использоваться при дальнейшей сборке реального устройства.

Следующий компонент, который мы рассмотрим, будет компонент, который отобразит на принципиальной электронной схеме I2C коннектор.

Компонент PCF8574 из библиотеки Interface_Expansion KiCad представляет собой 8-битный расширитель портов I/O, который может быть использован для управления периферийными устройствами, такими как светодиоды, кнопки, дисплеи и т.д.

Данный компонент имеет следующие пины:

- SDA - сериальная линия данных, используемая для передачи данных между микроконтроллером и расширителем портов.
- SCL - линия тактирования, используемая для синхронизации передачи данных между микроконтроллером и расширителем портов.
- A0-A2 - адресные входы, используемые для выбора адреса расширителя портов в системе.
- P0-P7 - 8 портов ввода-вывода, которые можно настроить как входы или выходы, используя регистры управления.

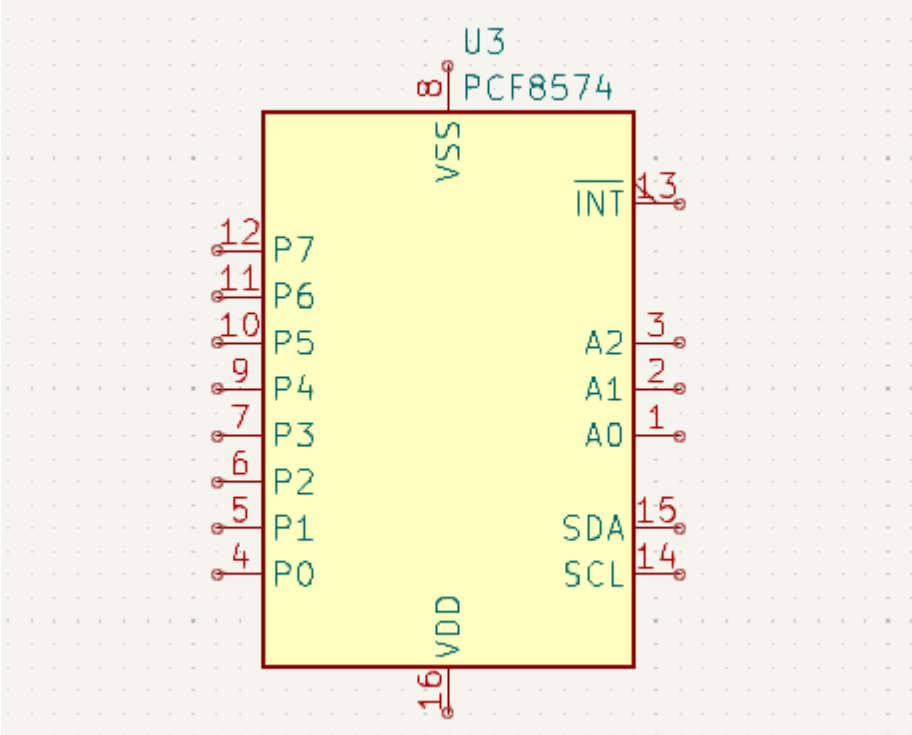


Рис. 2.2.4. Компонент PCF8574

Компонент WC1602A из библиотеки Display_Caracter будет использоваться для отображения в принципиальной электрической схеме LCD-дисплея.

Этот компонент представляет собой символьный ЖК-дисплей с 2 строками и 16 столбцами.

Компонент WC1602A имеет следующие основные характеристики:

- Размер экрана: 2 строки, 16 символов в каждой
- Интерфейс: параллельный
- Напряжение питания: 5 В
- Угол обзора: 6 часов
- Контрастность: регулируемая

Для настройки контрастности дисплея, можно подключить к нему переменный резистор, управляемый микроконтроллером. Для передачи информации на дисплей используется параллельный интерфейс, который требует от 6 до 11 пинов микроконтроллера.

Компонент WC1602A также поддерживает обратную связь посредством сигнала Busy (занятости), который показывает, когда дисплей готов принимать новые данные. Это позволяет контроллеру микроконтроллера ожидать, пока дисплей освободится, прежде чем отправлять новые данные.

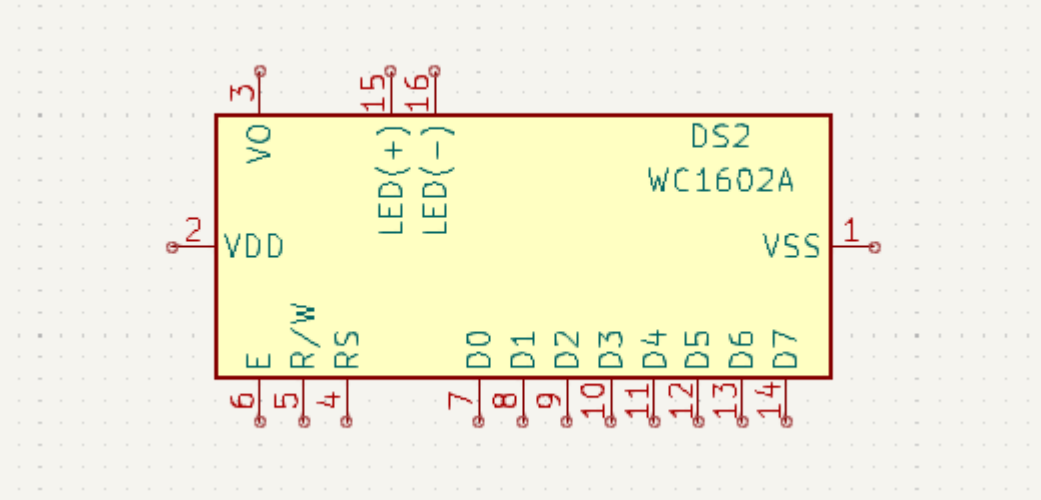


Рис. 2.2.5. Компонент WC1602A

3 Разработка алгоритма функционирования логического анализатора

3.1. Описание алгоритма функционирования логического анализатора

Как было рассмотрено ранее, каждый пакет интерфейса USB 2.0 оканчивается полем EOP. Поле EOP включает в себя 3 бита. В нашем проекте будем разрабатывать логический анализатор пакетов для режима передачи данных Low Speed.

В данном режиме скорость передачи может достигать до 1,5 Мбит/с. Это значит, что передача данных происходит на частоте 1,5 МГц. Таким образом, можем рассчитать, что биты данных передаются с периодичностью в 667 нс.

Объявим нашей задачей разработать логический анализатор пакетов USB, который смог бы предоставлять нам информацию о количестве переданных пакетов в единицу времени. Чтобы реализовать данную задумку, нам необходимо прибегнуть к специфике и устройству USB интерфейса, который был рассмотрен в предыдущих главах. Так как нас будет интересовать лишь количество пакетов, переданных в определенный промежуток времени, имеет смысл обратить внимание на поле EOP. Потому что данное поле описывается состоянием шины, которое соответствует состоянию SEO (линия D+ <0.8 В, линия D- <0.8 В), длящемуся два битовых интервала (около 1,3 мкс).

В случае, когда на линиях D+ и D- будет низкий уровень напряжения в течение двух битовых интервалов, будем считать, что передан один пакет. Количество таких пакетов, переданных за 1 секунду, будет определять скорость передачи данных, выраженных в величине – пакет в секунду. Данная величина будет динамической, она будет изменяться с течением времени.

3.2. Описание функций алгоритма

Написания алгоритма начинается с объявления необходимых констант и переменных, которые могут понадобится при работе программы.

```
#define MAX_V 3.3                       // Максимальный уровень напряжение на портах
#define MAX_DISCRET 4095                // Максимальное значение функции analogRead()
#define D1_PORT 4                       // Номер порта с линией D+
#define D2_PORT 2                       // Номер порта с линией D-
#define SPEED_UPDATE_INTERVALE 1000     // Интервал обновления скорости передачи пакетов
#define BIT_INTERVALE 0.000667          // Интервал чтения битов данных
```

Рис. 3.2.1. Константы и их описание

```
float D1_voltage = 0;                   // Значение уровня напряжения на линии D+
float D2_voltage = 0;                   // Значение уровня напряжения на линии D-
int count_SEO = 0;                      // Количество зафиксированных SEO состояний шины
int count_EOP = 0;                      // Количество EOP полей пакетов
bool check_double_SEO = false;          // Состояние шины SEO в течение двух битовых интервалов
unsigned long previous_speed_time = 0;  // Предыдущее время обновления скорости
unsigned long previous_packet_time = 0; // Предыдущее время чтения бита
```

Рис. 3.2.2. Переменные и их описание

Для того, чтобы преобразовать дискретные значения, считываемые с портов отладочной платы ESP32, в аналоговые, создадим функцию.

```
// Перевод дискретного значения функции analogRead() в аналоговое значение
float get_voltage(int port_state){
  return (MAX_V * port_state) / MAX_DISCRET;
}
```

Рис. 3.2.3. Функция преобразования дискретных величин в аналоговые

Создадим функцию для подсчета количества пакетов интерфейса USB.

```
// Фиксация поля конца пакета
void find_EOP() {
  count_EOP++;
  count_SEO = 0;
  check_double_SEO = false;
}
```

Рис. 3.2.4. Функция подсчета количества пакетов

Создадим функцию для определения состояния шины, которое соответствует состоянию SEO.

```
28  // Фиксация SEO состояния шины
29  void find_SEO() {
30    if (get_voltage(analogRead(D1_PORT)) < 0.8 &&
31        get_voltage(analogRead(D2_PORT)) < 0.8) {
32      if (count_SEO == 0) {
33        count_SEO++;
34        check_double_SEO = true;
35      }
36      else if (count_SEO == 1) {
37        find_EOP();
38      }
39    }
40    if (count_SEO == 1 && !check_double_SEO) {
41      count_SEO = 0;
42      check_double_SEO = false;
43    }
44  }
```

Рис. 3.2.5. Функция определения SEO состояния

Создадим функцию, которая будет выводить данные о скорости передачи пакетов.

```
46  // Вывод данных о скорости передачи пакетов
47  void speed(){
48    if (count_EOP >= 499) {
49      Serial.print(0);
50      Serial.println(" packets per second");
51    }
52    else {
53      Serial.print(count_EOP);
54      Serial.println(" packets per second");
55    }
56    count_EOP = 0;
57  }
```

Рис. 3.2.6. Функция отображения скорости передачи пакетов

Добавим в главный цикл программы отсчет временных интервалов, необходимых для считывания бит данных и вывода информации о скорости.

```
59  void setup() {
60    Serial.begin(115200);
61  }
62
63  void loop() {
64    unsigned long current_speed_time = millis();
65    unsigned long current_packet_time = millis();
66    if (current_speed_time - previous_speed_time >= SPEED_UPDATE_INTERVALE) {
67      previous_speed_time = current_speed_time;
68      speed();
69    }
70    if (current_packet_time - previous_packet_time >= BIT_INTERVALE) {
71      previous_packet_time = current_packet_time;
72      find_SEO();
73    }
74  }
```

Рис. 3.2.7. Функция setup() и главный цикл программы

3.3. Разработка корпуса логического анализатора

Корпус (или "кейс") - это внешняя оболочка электронного устройства, которая защищает его внутренние компоненты от повреждений, пыли, влаги, перегрева и других негативных воздействий окружающей среды. Кроме того, корпус также может служить для удобства использования устройства, обеспечивая удобный доступ к разъемам, кнопкам, экрану и т.д.

Корпусы могут быть различной формы, размера и материала изготовления в зависимости от типа устройства, его функциональности и требований к безопасности и надежности.

Корпус логического анализатора должен вместить в себя: отладочную плату ESP32, LCD-дисплей, I2C-коннектор и соединяющие данные устройства провода. Корпус должен иметь отверстия для разъемов USB 2.0. И посадочное место под LCD-дисплей. Корпус не должен иметь сложную геометрическую форму. Проанализировав размеры компонентов логического анализатора, пришли к выводу, что для размещения всех компонентов, понадобиться корпус со следующими характеристиками:

- Ширина – 60 мм;
- Длина – 100 мм;
- Высота – 30 мм.

3D-модель корпуса продемонстрирована на рисунке 3.3.1.

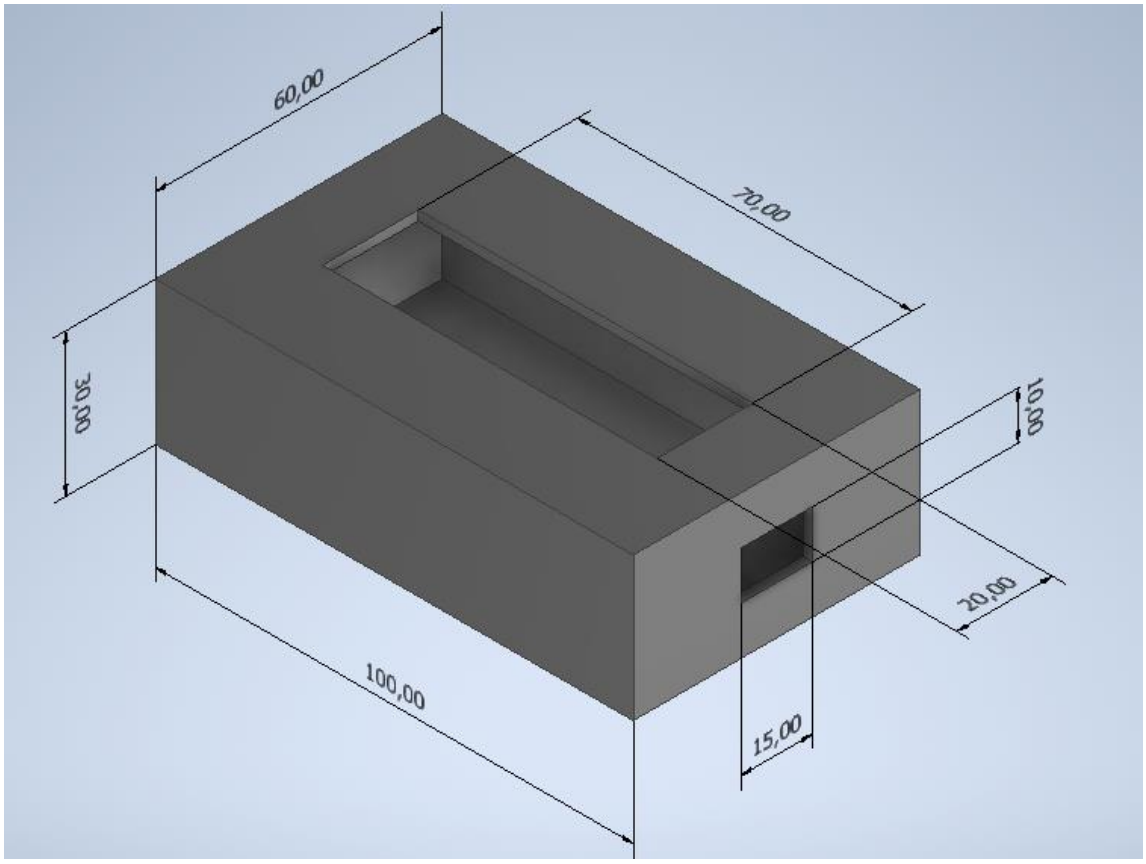
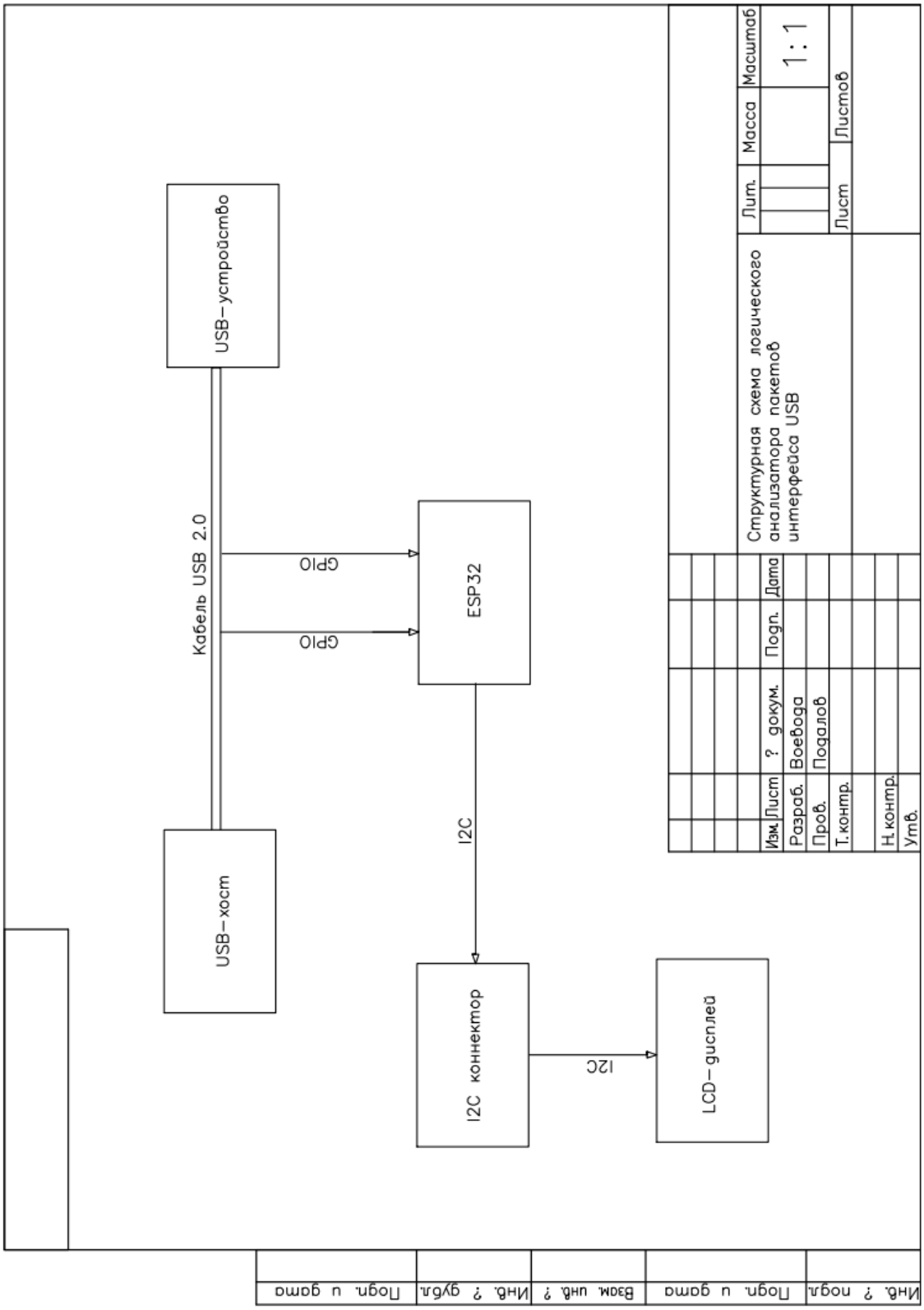


Рис. 3.3.1. 3D-модель корпуса логического анализатора

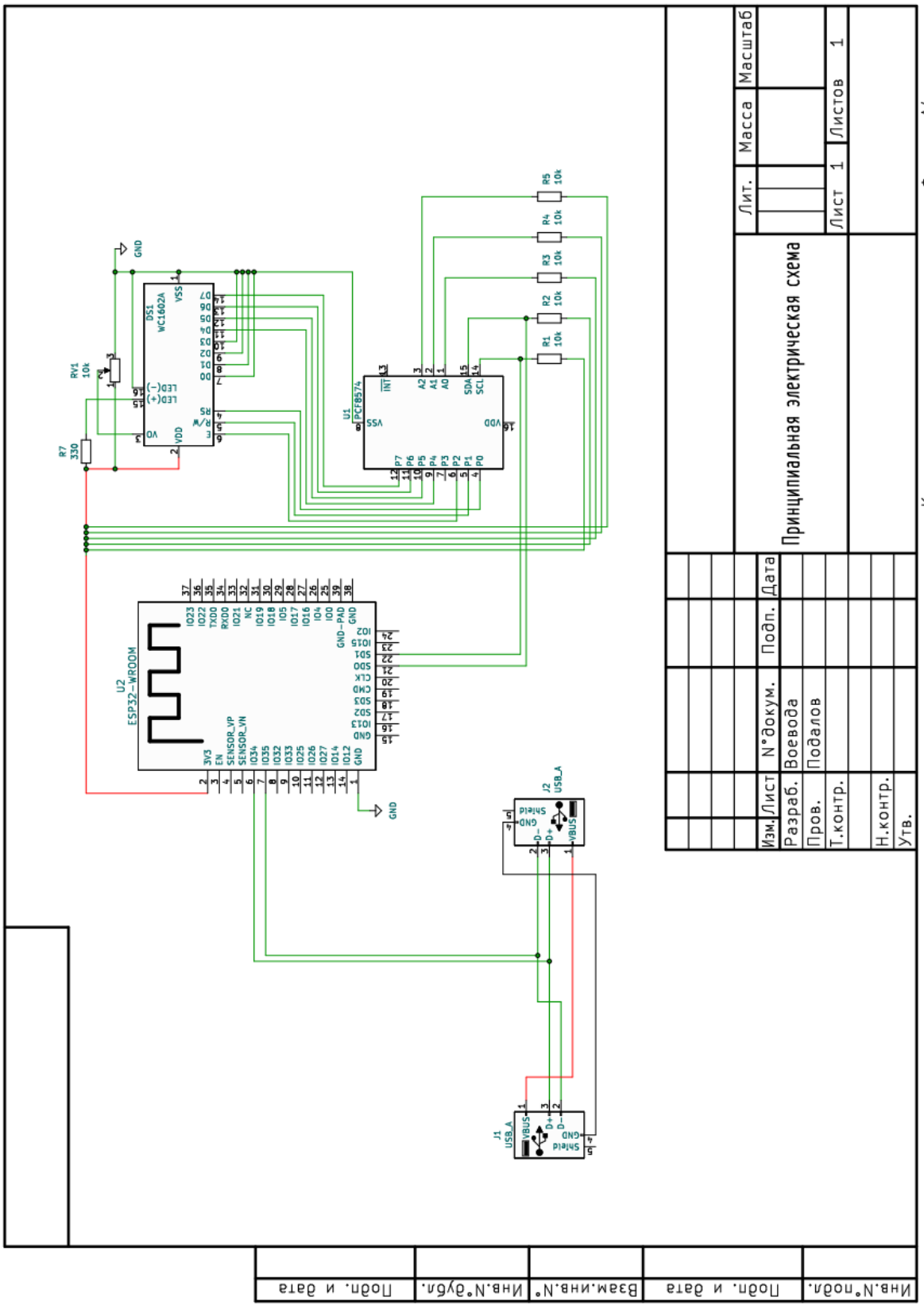
Приложение А

Структурная электрическая схема логического анализатора



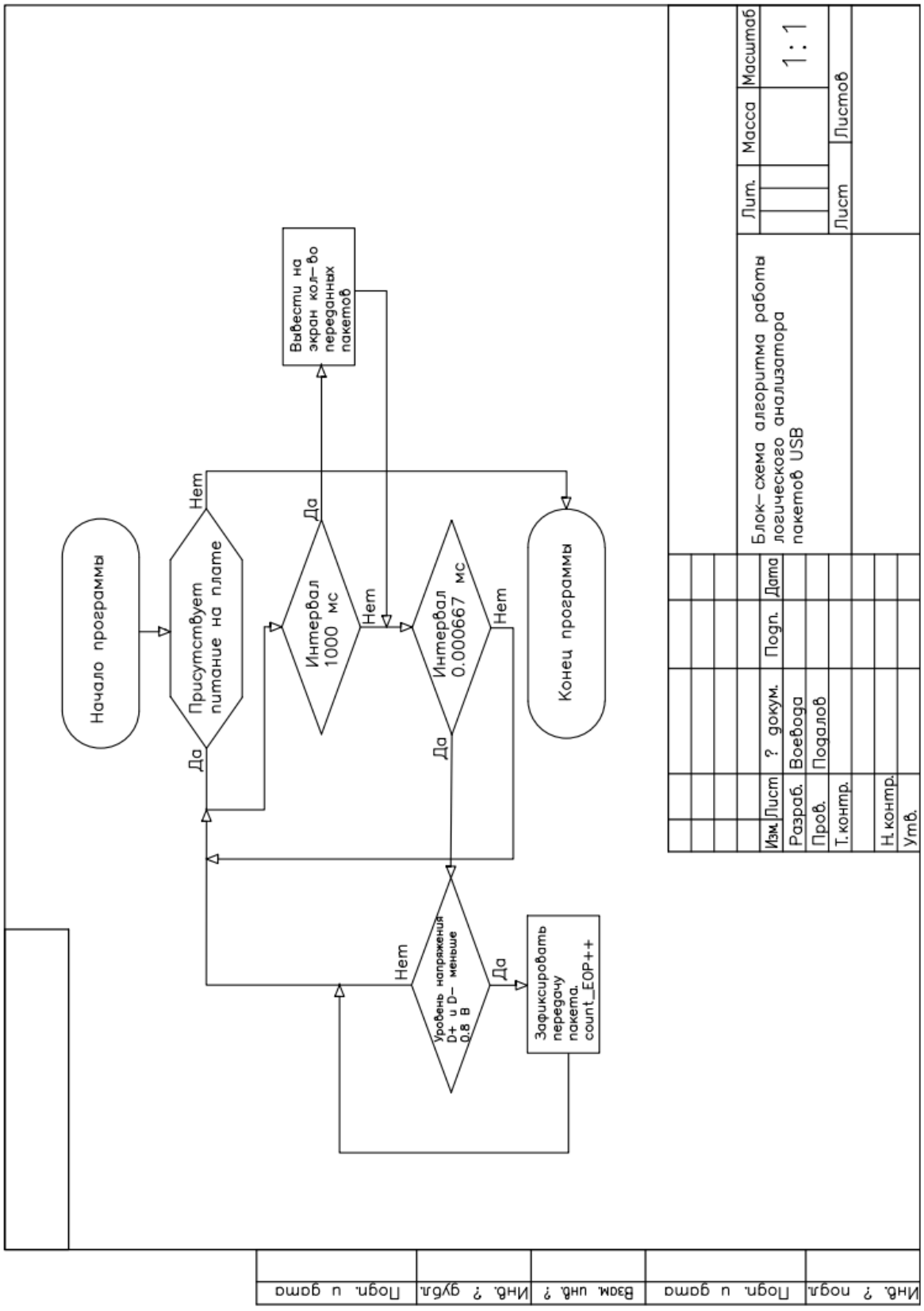
Приложение Б

Принципиальная электрическая схема логического анализатора



Приложение В

Блок-схема алгоритма программы логического анализатора



Приложение Г

Код алгоритма логического анализатора пакетов интерфейса USB

```
#define MAX_V 3.3
#define MAX_DISCRET 4095
#define D1_PORT 4
#define D2_PORT 2
#define SPEED_UPDATE_INTERVALE 1000
#define BIT_INTERVALE 0.000667

float D1_voltage = 0;
float D2_voltage = 0;
int count_SEO = 0;
int count_EOP = 0;
bool check_double_SEO = false;
unsigned long previous_speed_time = 0;
unsigned long previous_packet_time = 0;

float get_voltage(int port_state){
  return (MAX_V * port_state) / MAX_DISCRET;
}

void find_EOP() {
  count_EOP++;
  count_SEO = 0;
  check_double_SEO = false;
}

void find_SEO() {
  if (get_voltage(analogRead(D1_PORT)) < 0.8 &&
      get_voltage(analogRead(D2_PORT)) < 0.8) {
    if (count_SEO == 0) {
      count_SEO++;
      check_double_SEO = true;
    }
    else if (count_SEO == 1) {
      find_EOP();
    }
  }
  if (count_SEO == 1 && !check_double_SEO) {
    count_SEO = 0;
    check_double_SEO = false;
  }
}
```

```
void speed(){
  if (count_EOP >= 499) {
    Serial.print(0);
    Serial.println(" packets per second");
  }
  else {
    Serial.print(count_EOP);
    Serial.println(" packets per second");
  }
  count_EOP = 0;
}

void setup() {
  Serial.begin(115200);
}

void loop() {
  unsigned long current_speed_time = millis();
  unsigned long current_packet_time = millis();
  if (current_speed_time - previous_speed_time >= SPEED_UPDATE_INTERVALE) {
    previous_speed_time = current_speed_time;
    speed();
  }
  if (current_packet_time - previous_packet_time >= BIT_INTERVALE) {
    previous_packet_time = current_packet_time;
    find_SEO();
  }
}
```

Список использованных источников

1. USB [Электронный ресурс] // Свободная энциклопедия Википедия. – URL: https://ru.wikipedia.org/wiki/USB. – Дата доступа: 29.03.2023.
2. Интерфейс USB. Полный обзор и структура пакетов данных. [Электронный ресурс] // MicroTechnics. – URL: https://microtechnics.ru/osnovy-interfejsa-usb/?ysclid=lfhy6tv24a784689340. – Дата доступа: 29.03.2023.
3. Логический анализатор [Электронный ресурс] // Свободная энциклопедия Википедия. – URL: https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%82%D0%BE%D1%80. – Дата доступа: 29.03.2023.
4. Как выбрать логический анализатор [Электронный ресурс] // Измерительное и оптическое оборудование Суперайс. – URL: https://super-eyes.ru/articles/other/logicheskiy-analizator-kak-vybrat/?ysclid=lftam9h3p602157655. – Дата доступа: 29.03.2023.
5. I2C [Электронный ресурс] // Свободная энциклопедия Википедия. – URL: https://ru.wikipedia.org/wiki/I%C2%B2C. – Дата доступа: 03.04.2023.
6. Arduino [Электронный ресурс] // Свободная энциклопедия Википедия. – URL: https://ru.wikipedia.org/wiki/Arduino.– Дата доступа: 04.04.2023.