Федеральное государственное автономное образовательное
учреждение высшего образования
Национальный исследовательский университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Домашнее задание №1

по дисциплине «Системы ввода-вывода»

Вариант: **2**

Выполнили:

Барсуков Максим Андреевич (1.3)

Болорболд Аригуун (1.1)

Тернавский Константин Евгеньевич (1.1)

Преподаватель:

Быковский Сергей Вячеславович

г. Санкт-Петербург, 2025 г.

Оглавление

Оглавление.....1
Текст задания.....2
Реализация.....3
    Этап 1. Проектирование портов ввода/вывода.....3
    Этап 2. Проектирование протокола передачи данных.....5
    Этап 3. Описание сценариев использования и протокола транспортного уровня.....6
Вывод.....7

Текст задания

Спроектировать интерфейс передачи данных, удовлетворяющий следующим требованиям:

- Количество линий: 4
- Асинхронный
- Дуплексный

Реализация

Этап 1. Проектирование портов ввода/вывода

Распиновка разъема:

- In 1
- In 2
- Out 1
- Out 2
- Питание
- Земля

Земля необходима для корректного отсчета уровней сигнала. Питание - для запитки подключаемого устройства. In 1, In 2 Используются для получения данных; Out 1, Out 2 используются для отправки данных. Используется кросс-подключение Out -> In. То есть пин OutN в разъеме мастера соответствует пину InN в разъеме slave.

Примерный вид разъема в разрезе для случая для выбранного примера:

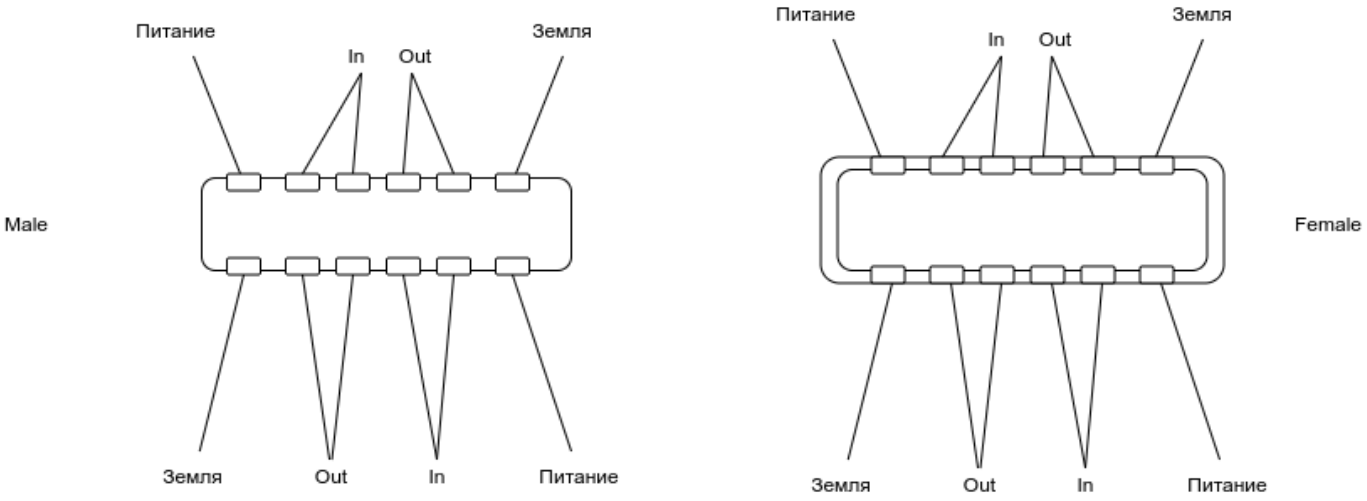

Рисунок 1. Разъем в разрезе

Прообразом нашего разъема является Lightning:

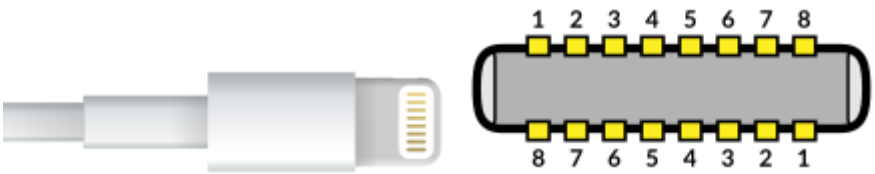

Рисунок 2. Разъем Lightning

В отличие от USB Type-C он обладает большей физической надежностью за счет отсутствия лишнего элемента в виде язычка в Female коннекторе. Пины располагаются на внешней стороне Male коннектора, а не на внутренней, как у USB Type-C.

На физическом уровне используется подключение (топология) вида **Один-к-одному** (point-to-point). Это классический способ подключение периферийных устройств в современном мире (вдохновлялись USB). В отличие от USB, инициировать передачу данных может и Slave. Инициация передачи данных обозначается с помощью специальной комбинации сигналов – стартовой последовательности (Start на рисунке):

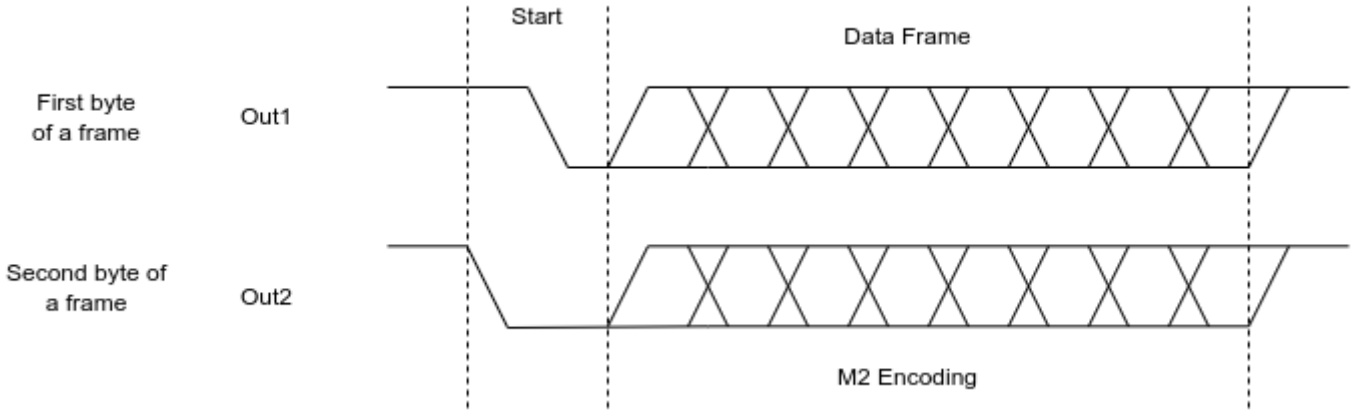

Рисунок 3. Формат передачи данных по проводу

Передаваемые данные кодируются с помощью M2 кода. Это позволяет сделать протокол самосинхронизирующимся. А значит, возможно передача длинных последовательностей битов без необходимости разбиения на кадры. Длина кадра определяется на логическом уровне. Специальные сигналы для обозначения конца кадра НЕ используются.

На логическом уровне топология представляет собой **дерево**. Корнем дерева является Master. Он управляет подключаемыми устройствами. Slave может связываться только с Master и НЕ может связывать с другими Slave-ами:

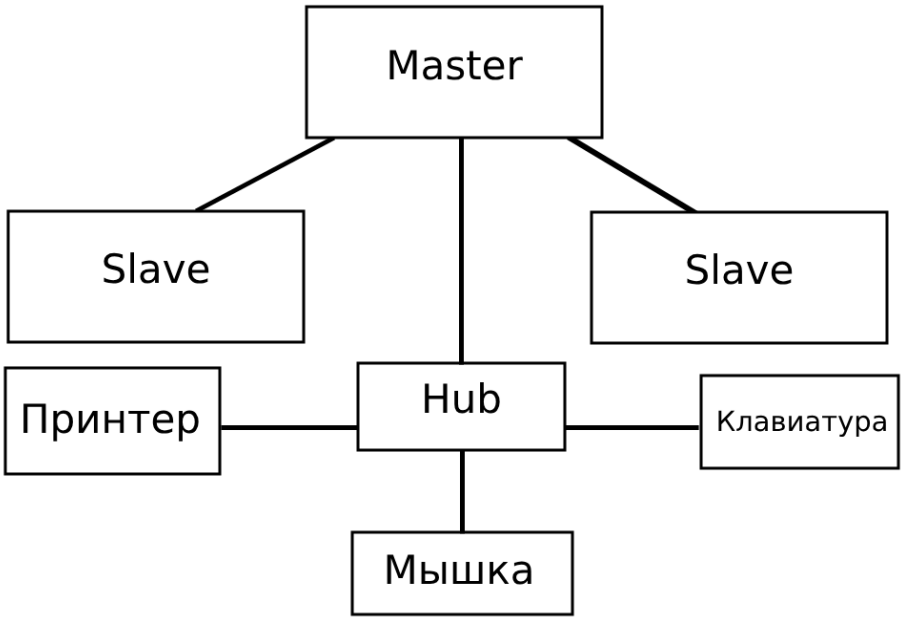

Рисунок 4. Пример логической топологии

Этап 2. Проектирование протокола передачи данных

Формат пакетов канального уровня:

Рисунок 5. Формат пакетов канального уровня

Используются два вида пакетов на канальном уровне. Для выбора Long Frame используется альтернативная стартовая последовательность на физическом уровне: первой из высокого напряжения в низкое уходит линия Out 1, а затем уже Out 2 (в отличие от базовой стартовой последовательности где первой к нулю подтягивается Out 2).

Оба вида кадров имеют одинаковую структуру и отличаются только длиной.

Поля кадра:

- Data - полезные данные. Получаются с более высокого уровня
- Checksum – контрольная сумма.

По проводам байты передаются следующим образом: четные биты байта передаются по второму каналу. Нечетные биты байта передаются по первому каналу. Передача битов байта осуществляется параллельно (см. Рисунок 3).

Рассчитаем полезную скорость передачи данных, с учетом скорости линии 1 Мбит/с и максимальной загрузки пакета, с полезной нагрузкой в 32768 байт.

- Коэффициент полезных данных:
  - Для короткого фрейма: 32/40 = 0,8
  - Для длинного фрейма: 32768/32832 ≈ 0,998
- Скорость передачи полезных данных по одной линии:
  1 Мбит/с × 32768/32832 = 0,998 Мбит/с
- Общая пропускная способность интерфейса (по двум линиям):
  0,998 Мбит/с × 2 = **1,996 Мбит/с**.

Этап 3. Описание сценариев использования и протокола транспортного уровня

Сценарии использования и прикладные области:

Интерфейс может быть полезен в системах, где требуется асинхронная дуплексная связь, высокая надежность передачи и поддержка гибридного трафика (короткие команды + длинные данные) на короткие расстояния. Особенности интерфейса — самосинхронизация через M2-код, устойчивый Lightning-подобный-разъем, возможность инициации передачи Slave-устройствами и логическая топология «дерево».

Данный интерфейс оптимизирован для работы с блочными устройствами (HDD, SSD, RAID-массивы) и передачи крупных массивов данных (например, файлы, образы дисков, потоковое видео). Вот почему:

1. **Поддержка длинных кадров** (4 КБ)
    - Для блочных устройств:
      Современные накопители работают с блоками данных размером 4 КБ, 8 КБ и более. Длинные кадры 32768 бит (4 КБ) позволяют передавать сектор размером 4 КБ за один пакет, что:
        - Снижает накладные расходы: вместо нескольких отдельных пакетов — один.
        - Уменьшает количество прерываний: контроллер обрабатывает меньше заголовков.
    - Пример:
      Запись файла размером 1 ГБ требует всего 262144 кадра (1 ГБ / 4 КБ = 262144), что ускоряет передачу и снижает нагрузку на ЦП – то есть быстрее, чем передавать по одному байту :-).
2. **Применение в блочных устройствах**
    - Внешние RAID-массивы:
        - Одновременная запись/чтение по дуплексным линиям (Out1→In1 и Out2→In2).
        - Контрольная сумма (Checksum) гарантирует целостность данных при восстановлении массивов.
    - Промышленные СХД (системы хранения данных):
        - Передача несжатых логов размером в гигабайты с минимальными накладными расходами.
    - Системы видеонаблюдения:
        - Потоковая передача RAW-видео с камер.

    - Асинхронный дуплекс: запись + чтение одновременно что подходит для баз данных, где происходит большое количество операций записи и чтения одновременно.
    - Логическая топология «дерево» упрощает управление множеством блочных устройств с одного компьютера.
3. **Параллельная передача и M2-код**
    - Четные/нечетные биты по двум линиям:
        - Байт разделяется на четные и нечетные биты, передаваемые параллельно по Out1/In1 и Out2/In2. Это удваивает полезную скорость и позволяет передавать большие данные без буферных задержек.
    - Самосинхронизация M2-кодом. Исключает необходимость отдельного тактового сигнала. Это критично для:
        - Используемых нами длинных фреймов, которые приводят к снижению накладных расходов при передаче.
        - Устойчивости к джиттеру и помехам.

Ключевые выгоды:
Для SSD: Меньше прерываний → выше скорость записи.
Для NAS: Работа с десятками Slave-устройств (топология «дерево»).
Для резервных копий: CRC-проверка исключает ошибки в архивах.

Интерфейс хорошо подходит для задач, где размер данных превышает 1 Мб, а целостность и скорость критичны:

- Резервное копирование/восстановление.
- Работа с виртуальными машинами (образы дисков).
- Потоковая передача несжатых данных (RAW-фото, видео).
- Системы, где Slave-устройства (например, NAS) должны инициировать передачу без задержек.

Протокол на транспортном уровне:

Header
version (8 bit)
Reserved (8 bit)
Parcel type (8 bit)
Slave Address (8bit)

Parcel
Resend Number (8 bit)
Sequence Number (32 bit)
Length in bytes (32 bit)
Payload
Check Sum (32 bit)

Рисунок 6. Формат пакетов транспортного уровня

Рассмотрим формат пакетов транспортного уровня:

Используется два основных пакета: Header и Parcel.

Header указывает версию протокола транспортного уровня (version). Она нужна для поддержки обратной совместимости. Затем указывает тип посылки (Parcel type: Payload, ACK, или другой). Тип посылки нужен чтобы отличать посылки с полезными данным от посылок служебного характера. Slave Address указывает, с каким Slave-ом на данный момент происходит общение. При отправке Master → Slave поле обозначает, какому Slave-у необходимо доставить посылку. При передаче Slave → Master – от какого Slave-а пришла посылка.

Parcel содержит служебные поля:

- Resend Number – позволяет отличать ACK для оригинального пакета и для повторных попыток передать пакет
- Sequence Number – вечно-возрастающее положительное целое число. Используется для упорядочивания пакетов (передача in-order a la TCP)
- Length in bytes – Длина в байтах позволяет передавать практически произвольные объемы данных
- Payload – собственно полезная нагрузка
- Checksum – контрольная сумма. Позволяет удостовериться, что данные были получены без искажений

Организация обмена данными между Master и Slave:

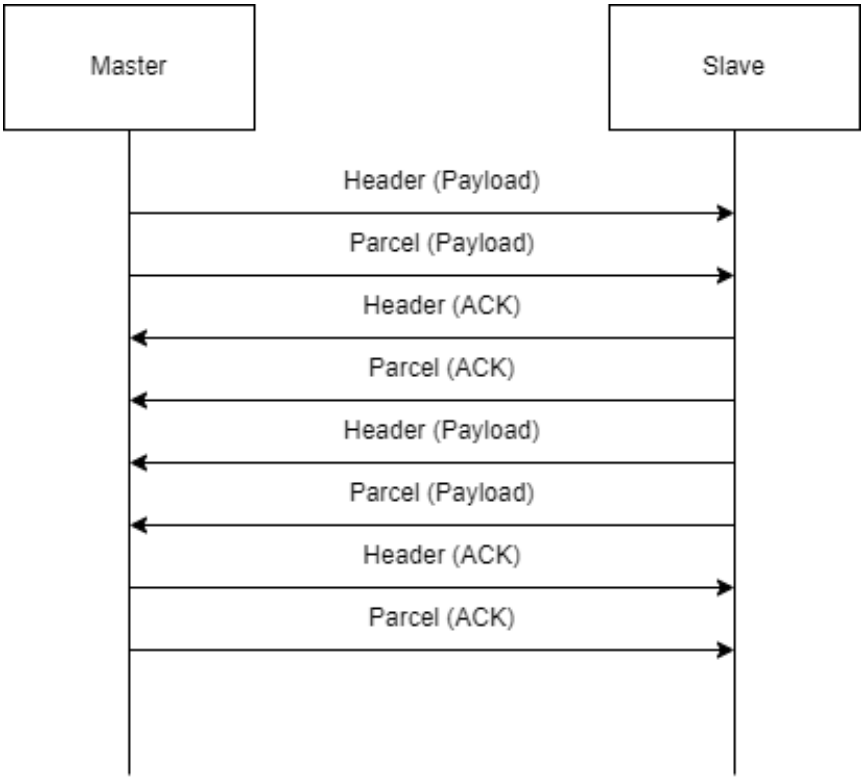

Рисунок 7. Организация обмена данными между Master и Slave

P.S. Коллизии невозможна в нашем варианте (канал у нас полнодуплексный).

Вывод

В ходе выполнения домашнего задания мы спроектировали собственный асинхронный дуплексный интерфейс ввода-вывода с 4 линиями связи, описали распиновку его разъема, топологии и способы подключения с помощью него устройств, разработали формат пакета передаваемых данных канального уровня, изобразили протокол передачи одного байта, рассчитали эффективную пропускную способность интерфейса, определили сценарии и прикладные области использования интерфейса, а также описали протокол транспортного уровня.