МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Университет ИТМО

дисциплина “Специальные вопросы программирования”
Курсовой проект

Реализация ПО для взаимодействия с устройствами в сети посредством протокола CC Over UDP multicast v3.

Работу выполнил:
Штеле Богдан(E4171)

Работу проверил:
Шарков И.А., к.т.н., ассистент ИВИТШ,
Университет ИТМО

Санкт-Петербург
2025

Оглавление

Теоретическая часть

Протокол Control Constants over UDP v3 возник по причине необходимости изменения IP адреса устройства. В протоколе описаны два вида запросов Read Request – для чтения параметров из устройства и Write Request – для записи каких-либо данных в устройство.

Структура запроса:

Header				DevId	Unix timestamp	Packet number	Param
0xC	0xC	0x3	0x0				
2 bytes				4 bytes	4 bytes	2 bytes	10 bytes

Header – состоит из двух байт, поделенных на 4 ниббла. Первые два это наименование протокола, третий отражает версию протокола, последний указывает тип запроса, 0x0 запрос на чтение.
DevId – это идентификатор устройства, для этих целей используется IP адрес.
Unix timestamp – здесь находятся метки времени, допускается начало отсчета времени с момента запуска устройства.
Packet number – хранит номер пакета, используется для отслеживания потерянных пакетов.
В поле Param содержится следующая структура.

Address	Data
2 bytes	8 bytes

Поле Address представляет собой адрес ячейки памяти куда будут записаны данные или откуда они будут считаны. Поле Data содержит данные для записи или чтения. При запросе на чтение поле Data должно быть пустым. При попытке записи данных по недопустимому адресу, может возникнуть ошибка.
В ответ на Read Request приходит UDP datagram. В поле Data будут содержаться данные, которые были получены из адреса. В четвертом ниббле Header будет содержаться значение 0x1, которое указывает на то, что это ответный пакет.
Для проведения тестов была разработана программа, имитирующая работу устройства. Были выбраны следующие адреса для записи и чтения:

Адрес в памяти	Функция адреса
0x0001	Адрес для записи данных
0x4001	Адрес для записи данных
0x0006	Адрес для чтения данных
0x0008	Адрес для чтения данных

С помощью адресов, представленных выше, можно проверить корректность работы разработанной программы.

Описание программных компонент

Основная программа состоит из двух файлов ControlConstantsOverUdp.cpp и ControlConstantsOverUdp.h. В .cpp файле описаны методы класса ControlConstants. В .h файле представлены объявления классов ControlConstants и Device, также методы класса, различные структуры для описания и формирования UDP-datagram.

Класс ControlConstants:

Метод	Параметры метода	Функция метода
get_ControlConstants	None	Функция возвращает указатель на объект, используется паттерн Singleton.
do_request	Тип отправляемого request и заполненная структура request_t	Проверяет тип сформированного запроса и отправляет через UDP.
make_write_request	Указатель на структуру, тип команды определяющий адрес для записи, помощью указателя передаются данные и размер этих данных не больше 8 байт, также передается адрес устройства.	Формирует запрос на запись данных в определенный адрес устройства.
make_read_request	Указатель на структуру, тип команды, адрес устройства.	Формирует запрос для чтения данных определенного адреса устройства.
is_working	None	Метод проверяет активен ли объект класса.
stop_socket_and_thread	None	Метод останавливает работу объекта класса, закрывает сокет и сворачивает поток на приеме.

Выше были описаны основные public методы класса, с помощью них пользователь программы может формировать необходимые request и отправлять их на выбранное устройство.

При первом вызове функции get_ControlConstants запускается поток на прием данных из сети и открывает сокет. Далее пользователь вызывает функции для формирования выбранного типа запроса и когда запрос сформирован, может быть вызвана функция do_request для отправки данных. Программа была разработана для работы на ОС Linux и Windows.

Класс Device используется для хранения данных о девайсах с которыми производится коммуникация. Поля класса содержат такую информацию как, номер последнего отправленного пакета и полученного пакета, адрес устройства, а также последний полученный пакет.

Пользовательский тип данных request_t и param_t описывают структуру запроса. С помощью них можно считывать запросы из принятого UDP пакета и формировать запросы. Все устройства хранятся в структуре данных map в классе ControlConstants при отправке пакета на новое устройство эта информация сохраняется в map_of_devices.

Проверка работоспособности ПО

Для проверки работоспособности был разработан скрипт имитирующий работу устройства на языке Pyhton. Также для удобства анализа будет использовано ПО Wireshark, которое позволит удобно отображать сетевые пакеты.

Во время эксперимента рекомендуется отключиться от сети т.к. другие пакеты могут мешать проведению. В начальный момент окно Wireshark выглядит как показано на рисунке 1.

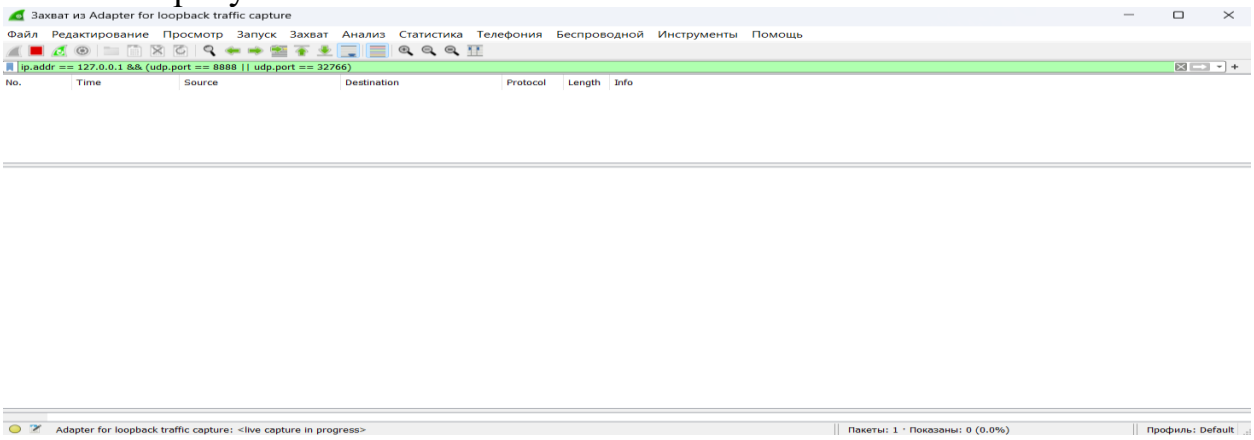

Рисунок 1 – Окно в начальный момент.

Также на рисунке один можно увидеть условия фильтра. Мы фильтруем трафик по адресу и портам. При отправке read request модно наблюдать два пакета сам request и ответ устройства.

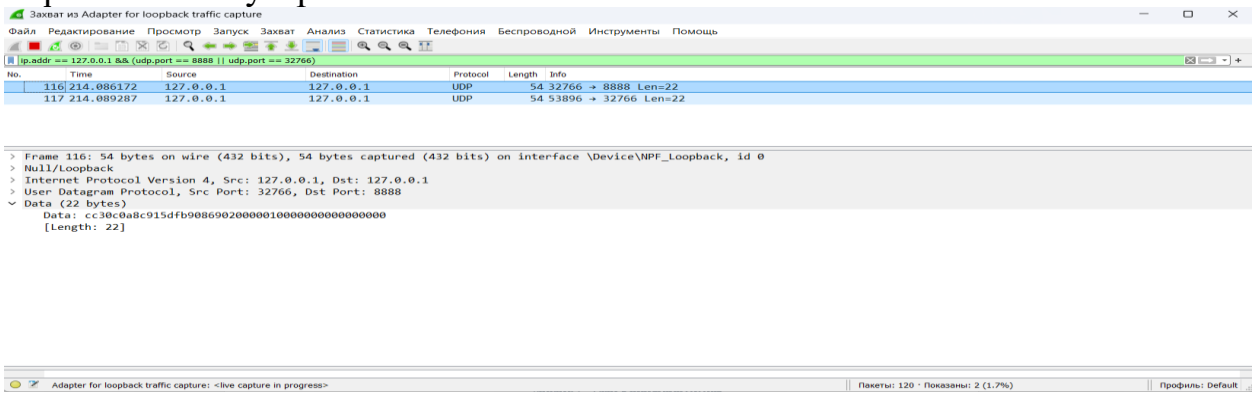

Рисунок 2 – Запрос на чтение.

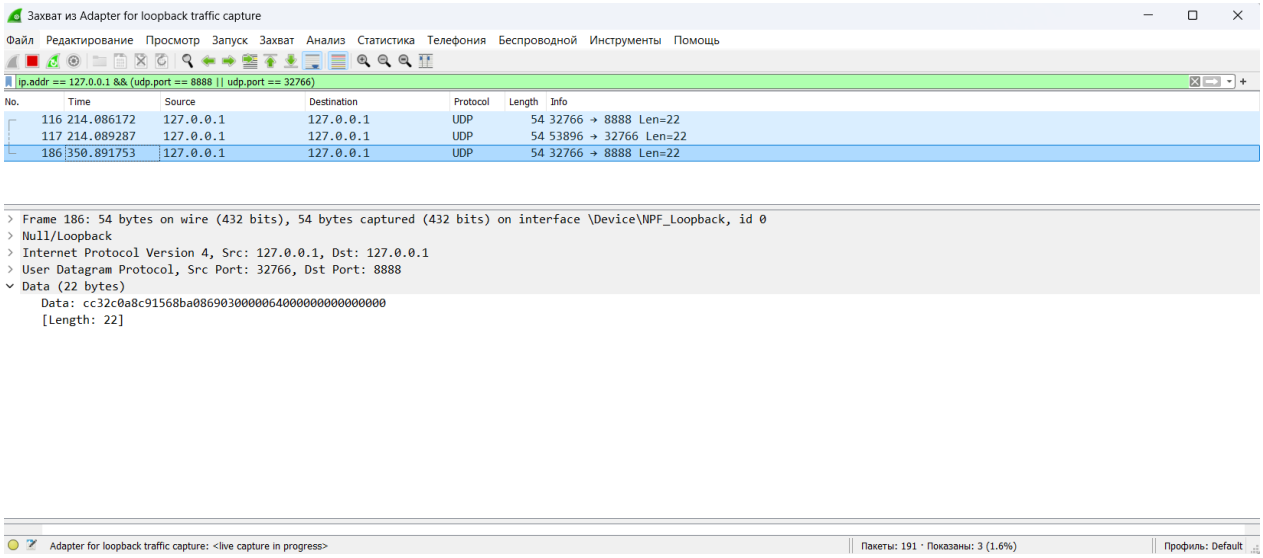

Рисунок 3 – Запрос на запись.

На рисунке 3 изображен запрос на запись, при запросе на запись устройство не отвечает. На рисунке 4 изображен вывод программы при запросе на чтение, можно сравнить данные которые программа хранит внутри себя и данные которые были получены с устройства. С помощью запроса на запись было отправлено число 3 во float типе данных, а при запросе на чтение были получены данные без изменений.

```
FFFFFFCC 30 FFFFFFC0 FFFFFFA8 FFFFFFC9 15 FFFFFFE2 FFFFFFBA 08 69 05 00 00 01 00 00 00 00 00 00 00 00 Data was
Address from net : 15C9A8C0
Data was recieved!
Success in recieveng data!
DATA IN MAIN data 3
DATA IN MAIN hex 40400000
DATA IN MAIN te1 40400000
DATA IN MAIN d 3
DATA IN MAIN te 0
```

Рисунок 4- Вывод программы

Таким образом можно сделать вывод, что программа не искажает данные при отправке и приеме.

Вывод

По итогам проведенной курсовой работы был разработан программный комплекс для обмена сообщениями внутри локальной сети посредством протокола ControlConstantsOverUdp v3. Также был разработан имитирующий работу устройства скрипт на языке программирования Python. Было проведено тестирование разработанной программы для выявления искажений в приемной или передающий части. Был создан и оформлен репозиторий на GitHub.