МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»

Кафедра телекоммуникационных систем и вычислительных средств
(ТС и ВС)

Отчет по производственной практике
по дисциплине
SDR

по теме:
Введение в архитектуру SDR-устройств. Знакомство с библиотеками Soapy SDR, Libiio. Инициализация SDR-устройства. Работа с буфером: получение I/Q-отсчётов

Студент:
Группа ИА-331 *Д.В Шкляев*

Предподаватели:
Лектор *Калачиков А.А*
Практик *Ахпашев А.В*
Практик *Попович И.А*

Новосибирск 2025 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ В АРХИТЕКТУРУ SDR-УСТРОЙСТВ. ЗНАКОМСТВО С БИБЛИОТЕКАМИ SOAPY SDR, LIBIIO. ИНИЦИАЛИЗАЦИЯ SDR-УСТРОЙСТВА. РАБОТА С БУФЕРОМ: ПОЛУЧЕНИЕ I/Q-ОТСЧЁТОВ

Цель работы: Ознакомиться с архитектурой SDR-устройства. Изучить библиотеки Soapy SDR, Libiio. Научиться инициализировать SDR-устройство и работать с буфером для получения I/Q-отсчётов.

Краткие теоретические сведения

SoapySDR - это универсальная библиотека с открытым исходным кодом, предназначенная для взаимодействия с различными программно-определяемыми радиоустройствами (SDR). Она предоставляет единый интерфейс для работы с множеством SDR-устройств от разных производителей, упрощая процесс разработки приложений, использующих SDR.

Передача сэмплов

Передача данных (IQ-сэмплов) между Adalm Pluto и хост-компьютером осуществляется посредством USB 2.0. Важно отметить, что в случае с SDR, данные необходимо передавать непрерывно в обе стороны (с хост-компьютера на SDR и обратно) одновременно. Хоть и теоретическая пропускная способность USB 2.0 равна 480 Mb/s, работа в полудуплексном режиме с передачей данных в обе стороны одновременно (с точки зрения пользователя) разительно снижается. Целевое значение (из опыта) частоты дискретизации желательно задавать в пределах 6 Msps.

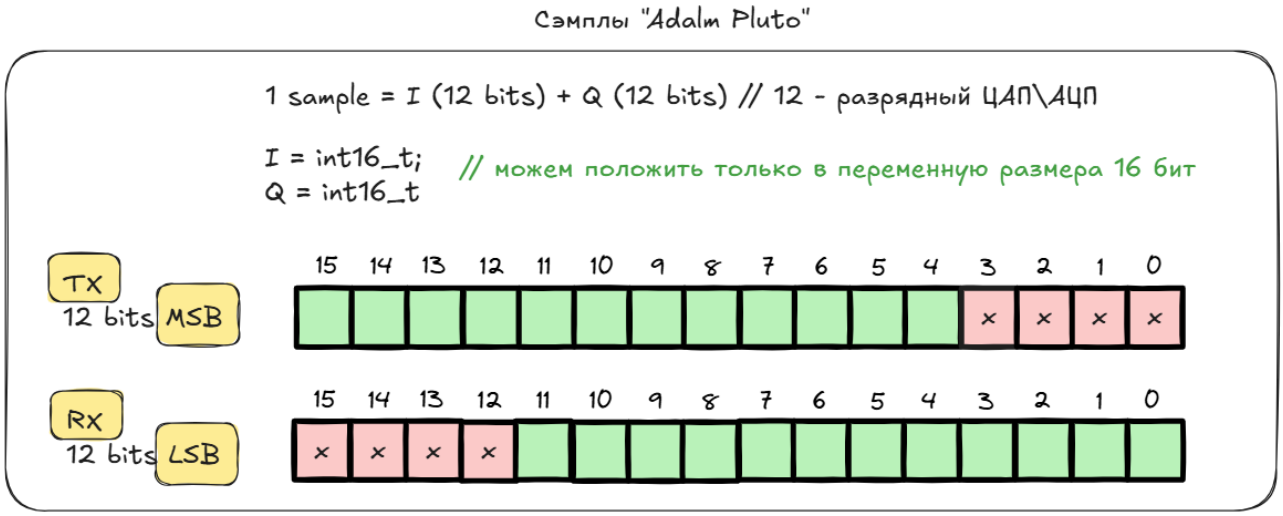

Рисунок 1 — Структура сэмпла

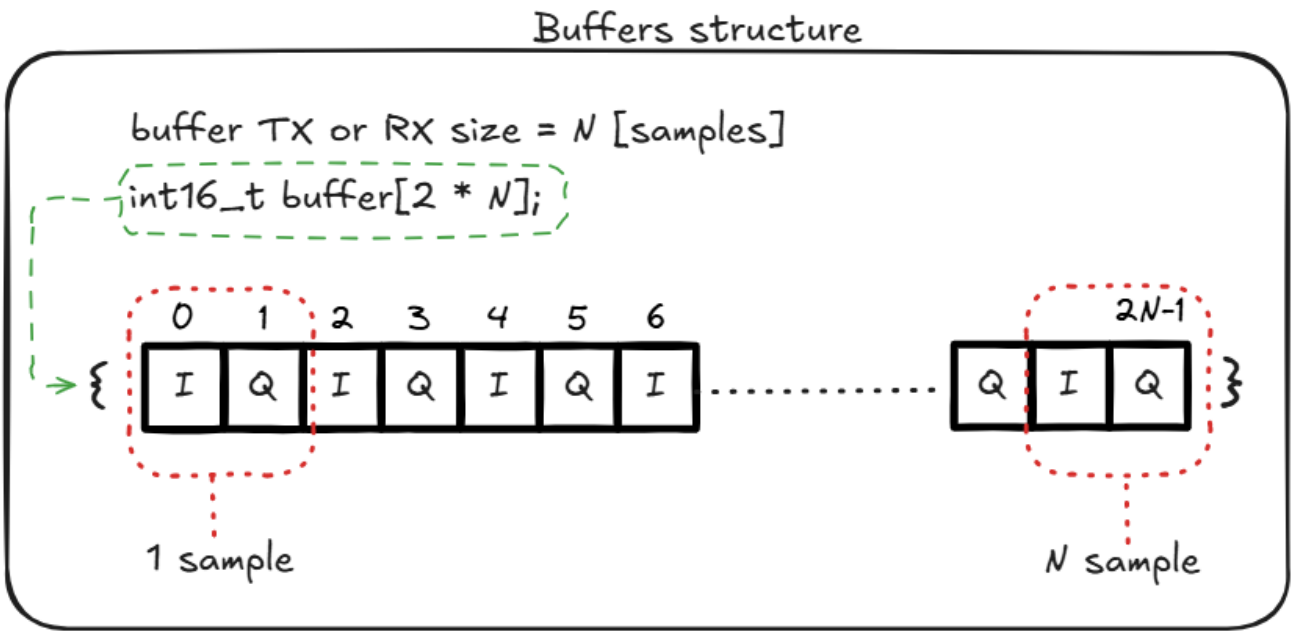

Рисунок 2 — Структура буфера IQ-сэмплов

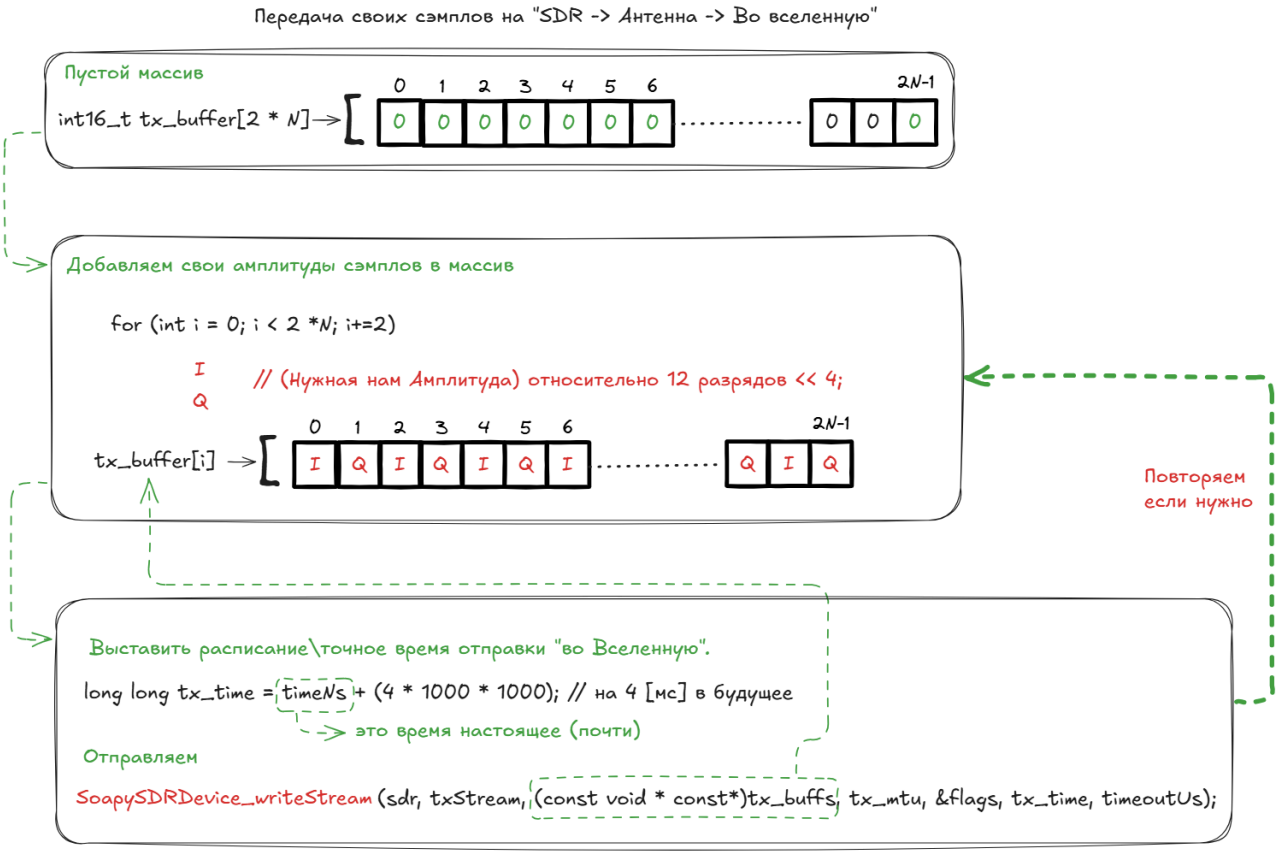

Рисунок 3 — Генерация и запись IQ-сэмплов

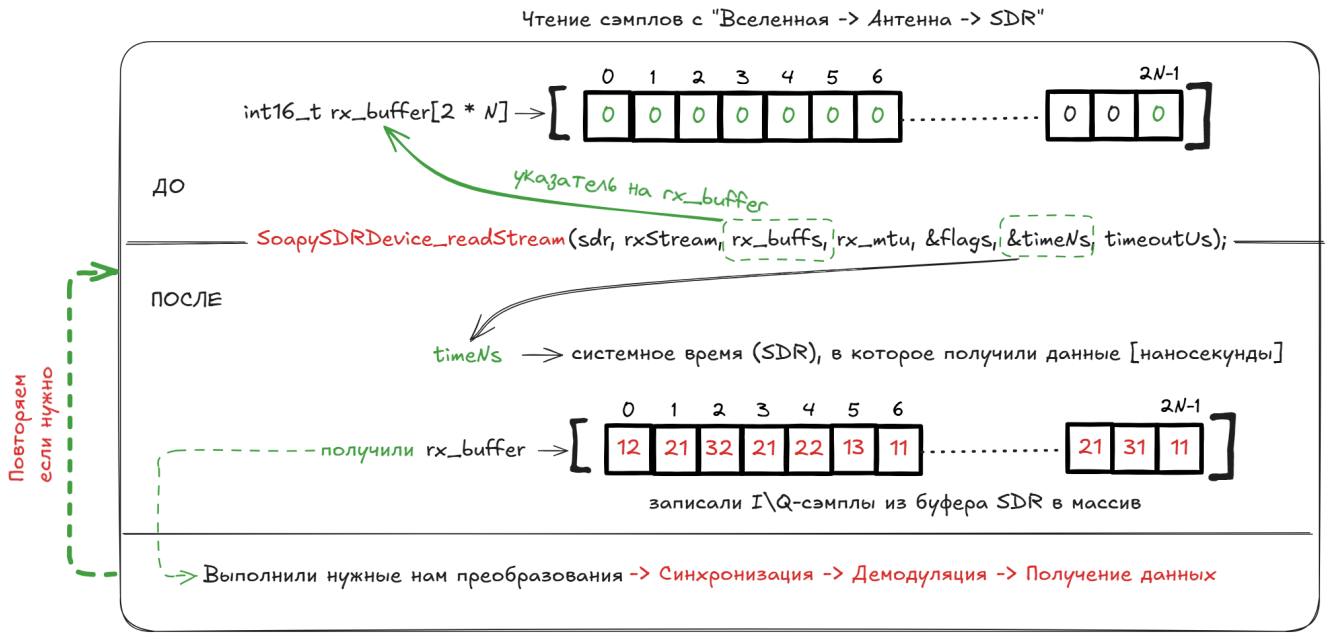

Рисунок 4 — Чтение IQ-сэмплов

Timestamp

Временные метки (timestamp) привязаны к каждому запросу данных с буфера ПЛИС, что, в свою очередь, позволяет синхронно получать передавать данные в потоках RX/TX. Более того, из-за проблем с пропускной способностью USB 2.0 возникает проблема увеличения частоты дескритизации, при больших значениях которой, USB 2.0 не может обеспечить полноценную передачу и прием (одновременных) сэмплов из Adalm Pluto на хост-компьютер.

Ход работы

Установка необходимых библиотек

SoapySDR:

```
sudo apt-get install python3-pip python3-setuptools
sudo apt-get install cmake g++ libpython3-dev python3-numpy swig
 ↪  python3-matplotlib

git clone --branch soapy-sdr-0.8.1
 ↪  https://github.com/TelecomDep/SoapySDR.git

cd SoapySDR
mkdir build && cd build

cmake ../
```

```
make -j`nproc` # nproc - количество потоков, например make -j16
sudo make install
sudo ldconfig
```

LibIIO:

```
sudo apt-get install libxml2 libxml2-dev bison flex libcdk5-dev
 ↪  cmake
sudo apt-get install libusb-1.0-0-dev libaio-dev pkg-config
sudo apt install libavahi-common-dev libavahi-client-dev


git clone --branch v0.24 https://github.com/TelecomDep/libiio.git

cd libiio
mkdir build && cd build
cmake ../
make -j`nproc`
sudo make install
```

LibAD9361:

```
git clone --branch v0.3
 ↪  https://github.com/TelecomDep/libad9361-iio.git
cd libad9361-iio

mkdir build && cd build

cmake ../

make -j`nproc`
sudo make install
sudo ldconfig
```

SoapyPlutoSDR:

```
git clone --branch sdr_gadget_timestamping
 ↪  https://github.com/TelecomDep/SoapyPlutoSDR.git
```

```
cd SoapyPlutoSDR

mkdir build && cd build

cmake ../

make -j`nproc`
sudo make install
sudo ldconfig
```

Настройка параметров SDR

Используемые библиотеки:

```
#include <SoapySDR/Device.h>   // Инициализация устройства
#include <SoapySDR/Formats.h>  // Типы данных, используемых для
 ↪  записи сэмплов
#include <cstdio>
#include <cstdlib>
#include <cstdint>
#include <complex>
```

Инициализация устройства:

```
SoapySDRKwargs args = {};
SoapySDRKwargs_set(&args, "driver", "plutosdr");        // Говорим
 ↪  какой Тип устройства
if (1) {
    SoapySDRKwargs_set(&args, "uri", "usb:");           // Способ
     ↪  обмена сэмплами (USB)
} else {
    SoapySDRKwargs_set(&args, "uri", "ip:192.168.2.1"); // Или по
     ↪  IP-адресу
}
SoapySDRKwargs_set(&args, "direct", "1");               //
SoapySDRKwargs_set(&args, "timestamp_every", "1920");   // Размер
 ↪  буфера + временные метки
SoapySDRKwargs_set(&args, "loopback", "0");             //
 ↪  Используем антенны или нет
SoapySDRDevice *sdr = SoapySDRDevice_make(&args);       //
 ↪  Инициализация
```

```
SoapySDRKwargs_clear(&args);
```

Настройка параметров устройств TX/RX:

```
int sample_rate = 1e6;
int carrier_freq = 800e6
// Параметры RX части
SoapySDRDevice_setSampleRate(sdr, SOAPY_SDR_RX, 0, sample_rate);
SoapySDRDevice_setFrequency(sdr, SOAPY_SDR_RX, 0, carrier_freq ,
↪  NULL);

// Параметры TX части
SoapySDRDevice_setSampleRate(sdr, SOAPY_SDR_TX, 0, sample_rate);
SoapySDRDevice_setFrequency(sdr, SOAPY_SDR_TX, 0, carrier_freq ,
↪  NULL);

// Инициализация количества каналов RX/TX (в AdalmPluto он один,
↪  нулевой)
size_t channels[] = {0}
// Настройки усилителей на RX/TX
SoapySDRDevice_setGain(sdr, SOAPY_SDR_RX, channels, 10.0); //
↪  Чувствительность приемника
SoapySDRDevice_setGain(sdr, SOAPY_SDR_TX, channels, -90.0);//
↪  Усиление передатчика
```

Инициализация потоков (stream) для передачи и примема сэмплов:

```
size_t channel_count = sizeof(channels) / sizeof(channels[0]);
// Формирование потоков для передачи и приема сэмплов
SoapySDRStream *rxStream = SoapySDRDevice_setupStream(sdr,
↪  SOAPY_SDR_RX, SOAPY_SDR_CS16, channels, channel_count, NULL);
SoapySDRStream *txStream = SoapySDRDevice_setupStream(sdr,
↪  SOAPY_SDR_TX, SOAPY_SDR_CS16, channels, channel_count, NULL);

SoapySDRDevice_activateStream(sdr, rxStream, 0, 0, 0); //start
↪  streaming
SoapySDRDevice_activateStream(sdr, txStream, 0, 0, 0); //start
↪  streaming
```

Подготовка массива для передачи (TX buffer):

```
//заполнение tx_buff значениями сэмплов первые 16 бит - I, вторые 16
 ↪  бит - Q.
for (int i = 2; i < 2 * tx_mtu; i+=2)
{
  // ЗДЕСЬ БУДУТ ВАШИ СЭМПЛЫ
  tx_buff[i] = 1500 << 4;   // I
  tx_buff[i+1] = 1500 << 4; // Q
}

for(size_t i = 0; i < 2; i++)
{
  tx_buff[0 + i] = 0xffff;
  // 8 x timestamp words
  tx_buff[10 + i] = 0xffff;
}

last_time = timeNs;

// Переменная для времени отправки сэмплов относительно текущего
 ↪  приема
long long tx_time = timeNs + (4 * 1000 * 1000); // на 4 [мс] в
 ↪  будущее

// Добавляем время, когда нужно передать блок tx_buff, через tx_time
 ↪  -наносекунд
for(size_t i = 0; i < 8; i++)
{
  uint8_t tx_time_byte = (tx_time >> (i * 8)) & 0xff;
  tx_buff[2 + i] = tx_time_byte << 4;
}

// Здесь отправляем наш tx_buff массив
void *tx_buffs[] = {tx_buff};
if( (buffers_read == 2) ){
  printf("buffers_read: %d\\n", buffers_read);
  flags = SOAPY_SDR_HAS_TIME;
  int st = SoapySDRDevice_writeStream(sdr, txStream, (const void *
   ↪  const*)tx_buffs, tx_mtu, &flags, tx_time, timeoutUs);
```

```
  if ((size_t)st != tx_mtu)
  {
    printf("TX Failed: %i\\n", st);
  }
}
```

Полученные отчеты из функции `SoapySDRDevice_readStream` записываются в файл `symbols.pcm` в бинарном формате записи

Отчеты считываются из файла с помощью Python (matplotlib и numpy):

```
import matplotlib.pyplot as plt
import numpy as np

rx = np.fromfile(f"/home/plutoSDR/sdr/pluto/dev/rx.pcm",
↪  dtype=np.int16)

samples = []

for x in range(0, len(rx), 2):
    samples.append(rx[x] + 1j * rx[x+1])

ampl = np.abs(samples)
phase = np.angle(samples)
time = np.arange(len(samples))

# plot
plt.subplot(3,1,1)
plt.legend
plt.plot(time, ampl)
plt.title("Amplitude")
plt.grid(True)

plt.subplot(3,1,2)
plt.legend
plt.plot(time, phase)
plt.title("Phase")
plt.grid(True)
```

```
plt.subplot(3,1,3)
plt.legend
plt.plot(time, rx[0::2])
plt.plot(time, rx[1::2])
plt.title("I - blue, Q - orange")
plt.grid(True)

plt.show()
```

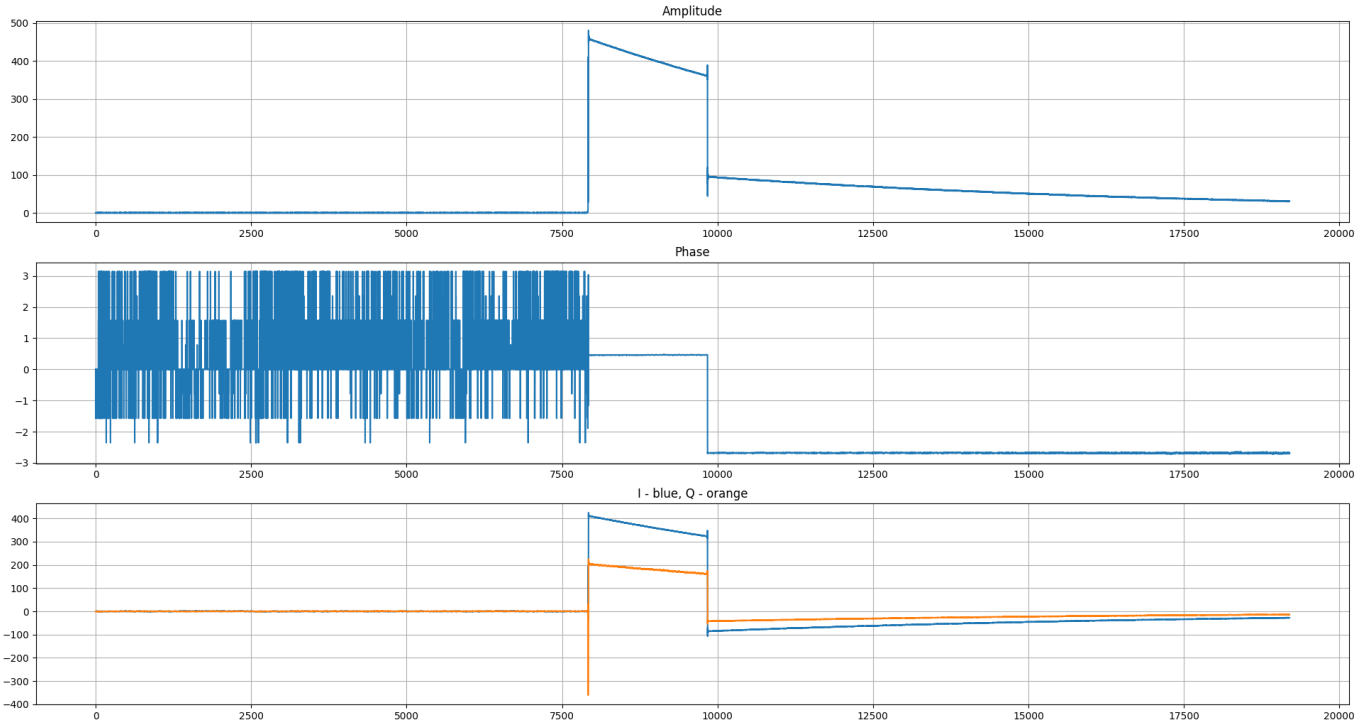

Рисунок 5 — Графики принятых I/Q-сэмплов

На графике можно наблюдать прямоугольный сигнал

Вывод

В данной работе я ознакомился с архитектурой SDR-устройств на примере Adalm Pluto. Изучил библиотеки SoapySDR и Libiio, научился инициализировать SDR-устройство и работать с буфером для получения I/Q-отсчётов. В процессе выполнения работы я установил необходимые библиотеки, настроил параметры устройства, инициализировал потоки для передачи и приёма сэмплов, а также реализовал передачу и приём IQ-сэмплов. В результате работы были получены графики принятых I/Q-сэмплов, что подтвердило успешность выполненных действий.