ГУАП

КАФЕДРА № 41

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

доц., канд. техн. наук		О.А. Кононов
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №13

ПРИЕМ-ПЕРЕДАЧА ДАННЫХ ПО UART

по курсу: ОСНОВЫ МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4711		Хасанов Б.Р.
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2020

Цель работы

Научиться использовать универсальный синхронный/асинхронный приемопередатчик, организовывать передачу и приём данных от stm32f4discovery.

1. Выполнение работы

1.1. Краткие теоретические сведения

Последовательный интерфейс использует одну сигнальную линию для передачи данных, по которой биты информации передаются друг за другом последовательно. При последовательной передаче сокращается количество сигнальных линий, что упрощает разводку проводников на печатной плате, уменьшает габариты устройства и позволяет делать более помехозащищенные интерфейсы. При последовательной передаче каждый информационный бит должен сопровождаться импульсом синхронизации — стробом. Если импульсы синхронизации передаются от одного устройства к другому по выделенной линии, то такой интерфейс называют синхронным, в этом случае генератор синхронизации располагается на стороне устройства инициирующего передачу. Если же приемник и передатчик содержат каждый свой генератор синхроимпульсов, работающий на одной частоте, то такой интерфейс называется асинхронным. Получается, что приемник информации сам вырабатывает синхроимпульсы.

Типичный представитель асинхронного последовательного интерфейса — UART (Universal Asynchronous Receiver-Transmitter — универсальный асинхронный приёмопередатчик).

При передаче по интерфейсу UART каждому байту данных предшествует СТАРТбит, сигнализирующий приемнику о начале посылки, за СТАРТ-битом следуют биты данных. Завершает посылку СТОП-бит, гарантирующий паузу между посылками. СТАРТбит следующего байта посылается в любой момент после СТОП-бита, то есть между передачами возможны паузы произвольной длительности. СТАРТ-бит, обеспечивает

простой механизм синхронизации приемника по сигналу от передатчика. Внутренний генератор синхроимпульсов приемника использует счетчик-делитель опорной частоты, обнуляемый в момент приема начала СТАРТ-бита. Этот счетчик генерирует внутренние стробы, по которым приемник фиксирует последующие принимаемые биты. Диаграмма передачи байта информации в общем случае представлена на рисунке 1.

Рисунок 1 – Диаграмма передачи байта информации

Микроконтроллер STM32F407VG содержит в общей сложности 6 приемопередатчиков. При этом 4 из них являются синхронно-асинхронными (USART1, USART2, USART3, USART6) и 2 только асинхронными (UART4, UART5).

Figure 61. USART Block Diagram[1]

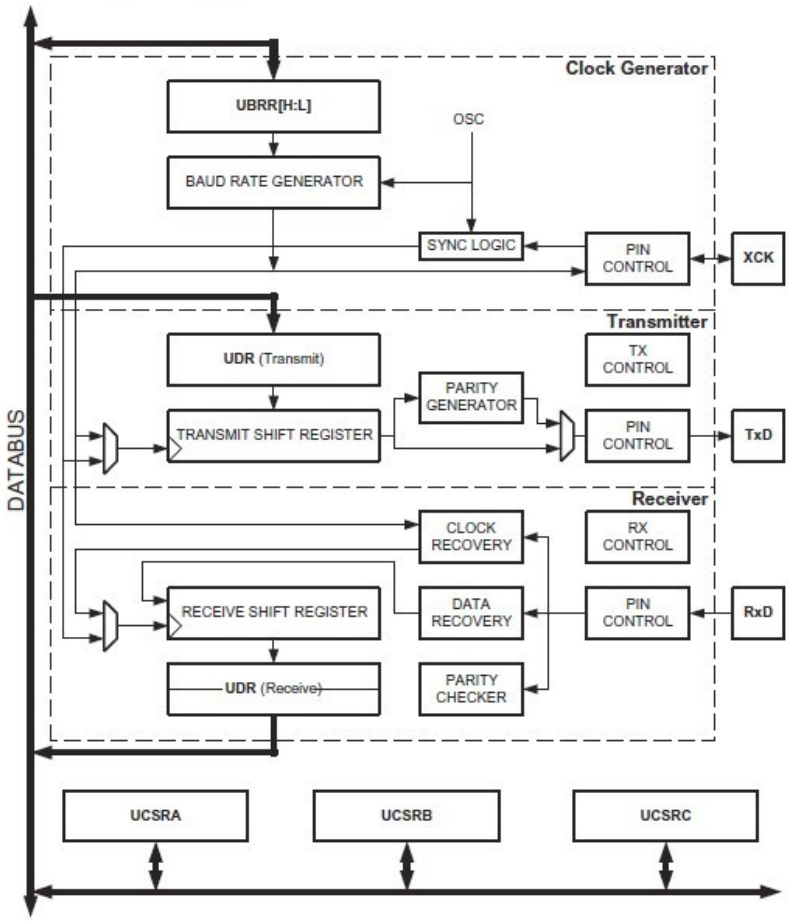

Рисунок 2 – Структурная схема USART

Как видно из структурной схемы, в USART используются три основных выхода RxD, TxD и XCK. RxD — ножка для приема информации; TxD — ножка для передачи информации; XCK — ножка синхронизации. При использовании асинхронного режима ножка XCK не подключается.

При коммутации двух устройств с помощью UART используют схему, представленную на рисунке 3.

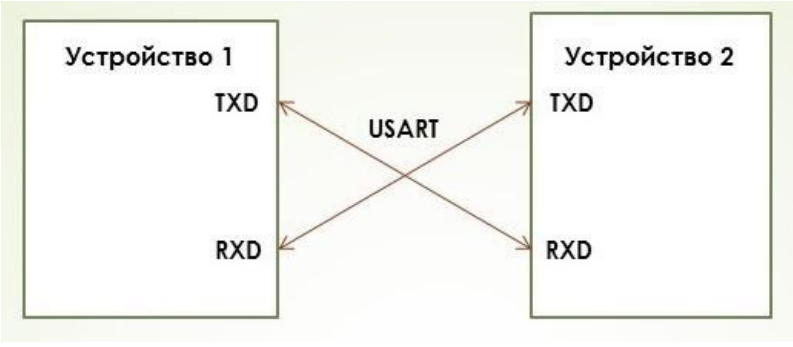

Рисунок 3 – Схема подключения двух устройств, использующих UART

В данной лабораторной работе удобно использовать сразу два модуля UART на одном микроконтроллере, имитируя подключение стороннего устройства. В таком случае можно одновременно отслеживать как процесс передачи данных, так и их приёма.

1.2. Создание проекта

Проект был создан с помощью программы STM32CubeMX, где

производилась настройка необходимых портов ввода/вывода, UART, GPIO, а также тактовой частоты процессора Пример настройки представлен на рисунках 4-7.

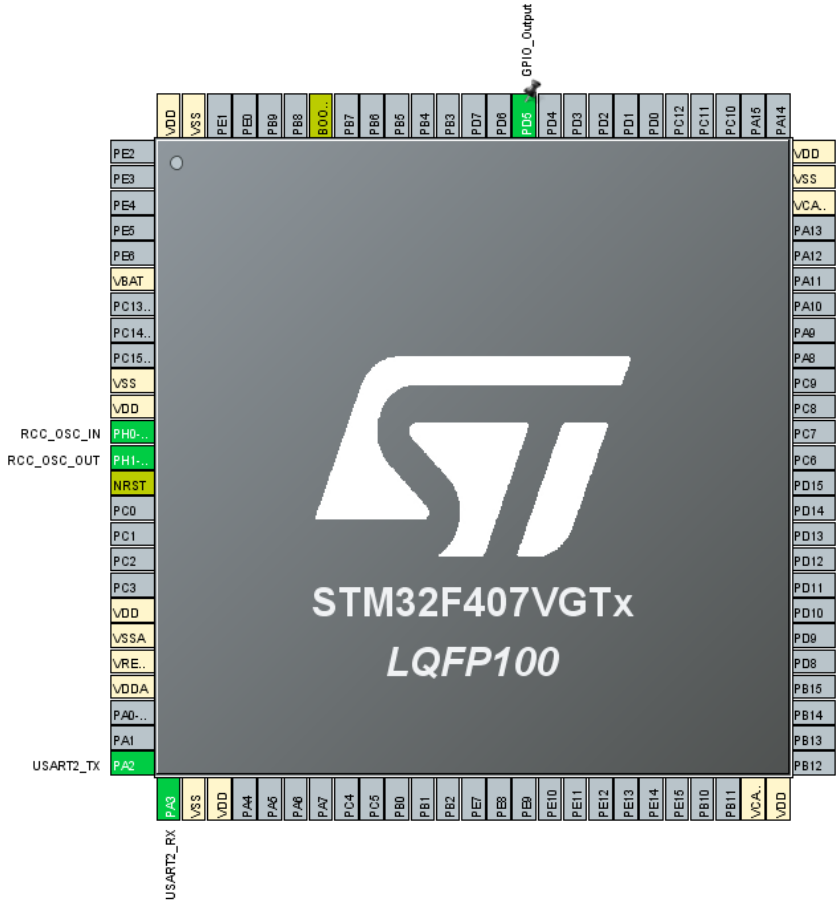

Рисунок 4 - Схема выводов STM32F407VGT после настройки проекта в STM32CubeMX

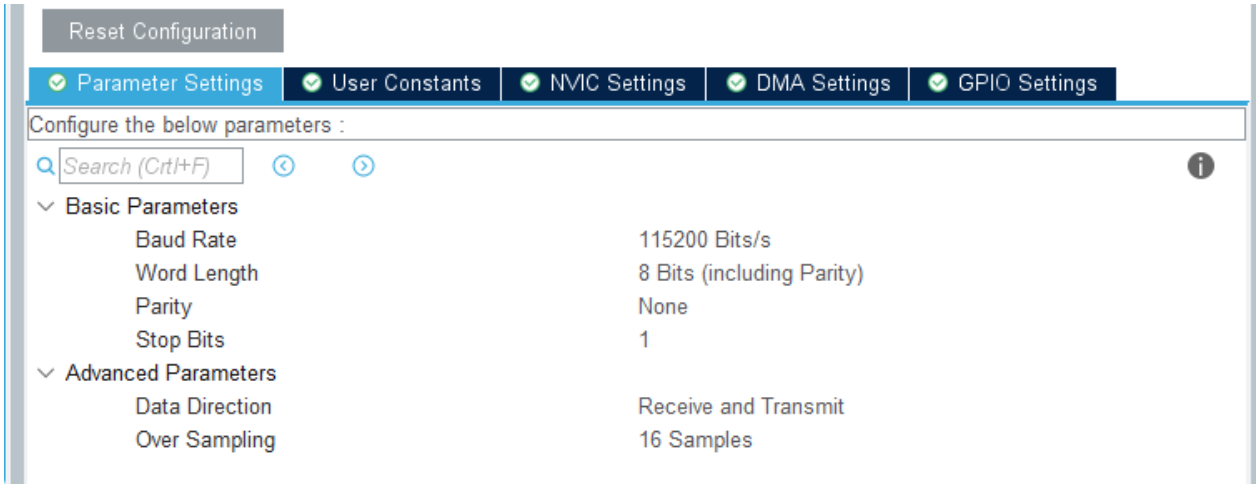

Рисунок 5 – Настройка USART в STM32CubeMX

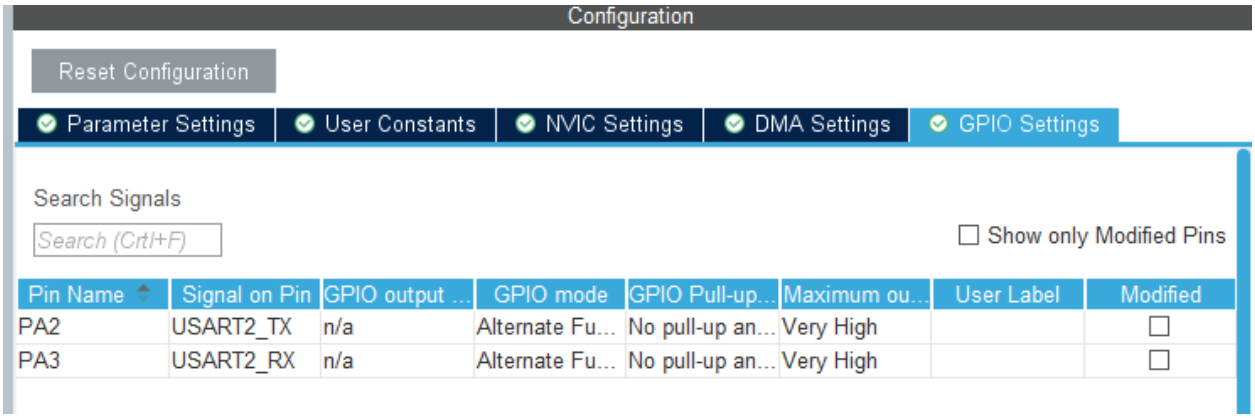

Рисунок 6 – Настройка GPIO в STM32CubeMX

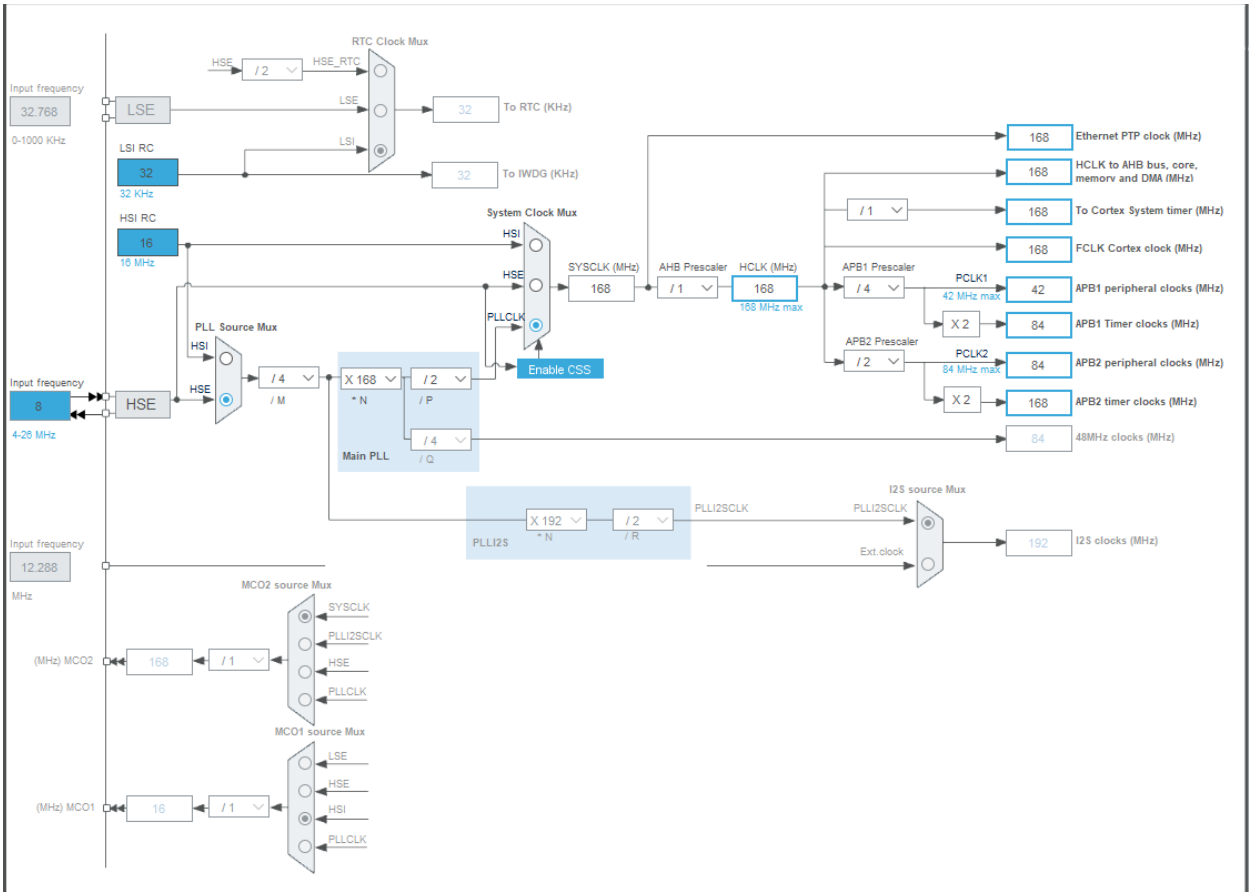

Рисунок 7 – Окно настройки тактирования в Cube

2.3 Код программы на С

Код программы main.c

```
#include "main.h"
UART_HandleTypeDef huart2;
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_USART2_UART_Init(void);
__STATIC_INLINE void DelayMicro(__IO uint32_t micros)
{
micros *=(SystemCoreClock / 1000000) / 5;
while (micros--);
}
int main(void)
{
 uint8_t str[1];
  HAL_Init();
  SystemClock_Config();
  MX_USART2_UART_Init();
```

```
  while (1)
  {
          uint8_t state = HAL_UART_GetState(&huart2);
              // Если передача не идёт
          if(  (state  !=  HAL_UART_STATE_BUSY_RX)  &&  (state  !=
HAL_UART_STATE_BUSY_TX_RX) ) {

         while(  HAL_UART_Transmit_IT(&huart2,  str,  1)  ==  HAL_BUSY  );
//передаём данные
    HAL_UART_Receive_IT (&huart2, str, 1); //принимаем данные
                                }
  }
}
```

3 Результат выполнения программы

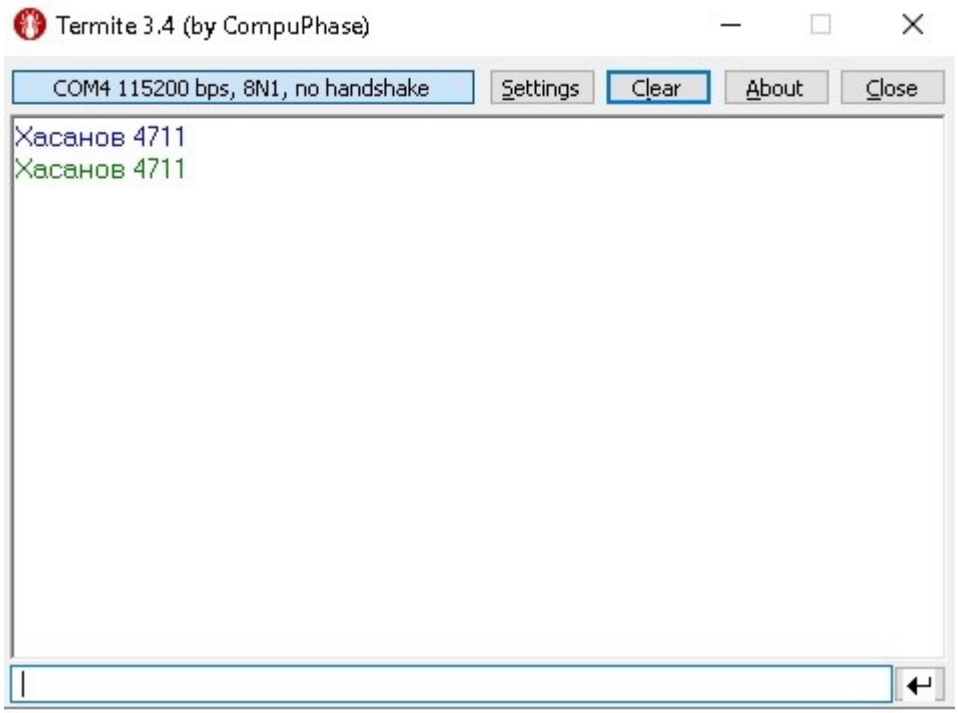

Рисунок 8– Результат отправки и приема данных

Вывод

В ходе данной лабораторной работы были исследованы принцип работы и настройка последовательного асинхронного приёмопередатчика UART, реализованы приём и передача данных по UART с помощью микроконтроллера STM32F407VG.