Обмен данными между компьютером и микроконтроллером *STM*32*F*100 по последовательному интерфейсу связи *RS*-232

В.А. Жмудь, И.В.Трубин, М.В.Трубин
ФГБОУ ВПО НГТУ, Новосибирск, Россия

Аннотация: **В современном мире большой объем занимают автоматизированные системы управления, но управление не возможно без связи между устройствами. В настоящее время применяется множество интерфейсов связи, но бесспорно одним из самых простых и популярных является до сих пор *RS*-232. В статье рассматриваются вопросы передачи данных в последовательном виде между микроконтроллером из серии *STM*32F100 и компьютером по интерфейсу *RS*-232. Приводится краткая историческая информация по данному интерфейсу. Приводится описание преобразователя USB/RS232 на базе популярной микросхемы *PL*2303. Подробно рассматривается способ подключения компьютеру. Приводится методика проверки преобразователя без использования других устройств связи. Рассматривается пример программного кода. Раскрываются возможности программы Terminal. Для улучшения понимания процесса передачи/приёма данных приводятся осциллограммы сигналов.**

Ключевые слова*: *RS*-232, *USART*, *STM*32, *STM*32*F*100, *Terminal*, *STM*32*VLDISCOVERY

ВВЕДЕНИЕ

В современном мире большой объем занимают автоматизированные системы управления, но управление не возможно без связи между устройствами. Невозможно представить себе систему управления без связи между управляющим устройством и устройством-исполнителем. Целью данной статьи является рассмотрение интерфейса связи между устройствами, а конкретно *RS*-232. В мире применяется множество интерфейсов связи, но бесспорно одним из самых простых и популярный является до сих пор *RS*-232. Данный интерфейс позволяет подключать устройства к большинству ПК, он прост в применении и широко распространен. Одним из ограничивающих факторов применения данного интерфейса является ограничение длины линии связи - не более 15 *м*, что объясняется не высокой помехозащищённостью.

1. ИНТЕРФЕЙС И СТАНДАРТ *RS*-232

RS-232 (*Recommended Standard* 232) - стандарт, описывающий интерфейс для последовательной двунаправленной передачи данных между терминалом (*DTE, Data Terminal Equipment*) и конечным устройством (*DCE, Data Circuit-Terminating Equipment*). *RS*-232 - проводной дуплексный (т.е. можно одновременно передавать и принимать информацию) интерфейс. Информация передается по проводам двоичным сигналом с двумя уровнями напряжения. Логическому «0» соответствует положительное напряжение (от +5 до +15 *В*), а логической «1» отрицательное (от −5 до −15 *В*).

В 1962 году *Electronics Industries Association* (*EIA*) разработало рекомендации для производителей оборудования, назвав их "Рекомендованный стандарт 232".
Интерфейс *RS*-232 был разработан максимально универсальным, что позволяло многим производителям легко переделать своё оборудование под этот стандарт. Для электрического согласования линий *RS*-232 и стандартной цифровой логики *UART* выпускается большая номенклатура микросхем драйверов. Для передачи "полезных" данных в одной посылке допускалось использовать от 5 до 8 бит. Было предусмотрено 16 сервисных сигналов, использование которых было не обязательно.
Допускалась работа как в синхронном, так и асинхронном режиме передачи данных.

В 1969 году *EIA* выпустила редакцию стандарта *RS*-232C, в котором был учтен семилетний опыт применения стандарта *RS*-232*A/B*. Окончательно был узаконен 25 штырьковый разъем *DB*25 и электрические характеристики сигнала. Эта редакция стала основным интерфейсом передачи данных по последовательным каналам связи на многие годы вперед [1].

Таблица 1

Назначение выводов 9 - контактного разъема

Контакт	Сигнал	Направление	Описание
1	*CD*	Вход	Обнаружена несущая
2	*RXD*	Вход	Принимаемые данные
3	*TXD*	Выход	Передаваемые данные
4	*DTR*	Выход	Хост готов
5	*GND*	–	Общий провод
6	*DSR*	Вход	Устройство готово
7	*RTS*	Выход	Хост готов к передаче
8	*CTS*	Вход	Устройство готово к приему
9	*RI*	Вход	Обнаружен вызов

Таблица 2

Назначение выводов 25 - контактного разъема

Контакт	Сигнал	Направление	Описание
1	*SHIELD*	–	Экран
2	*TXD*	Выход	Передаваемые данные
3	*RXD*	Вход	Принимаемые данные
4	*RTS*	Выход	Хост готов к передаче
5	*CTS*	Вход	Устройство готово к приему
6	*DSR*	Вход	Устройство готово
7	*GND*	–	Общий провод
8	*CD*	Вход	Обнаружена несущая
9	–	–	Резерв
10	–	–	Резерв
11	–	–	Не используется
12	*SCD*	Вход	Обнаружена несущая #2
13	*SCTS*	Вход	Устройство готово к приему #2
14	*STXD*	Выход	Передаваемые данные #2
15	*TRC*	Вход	Тактирование передатчика
16	*SRXD*	Вход	Принимаемые данные #2
17	*RCC*	Вход	Тактирование приемника
18	*LLOOP*	Выход	Локальная петля
19	*SRTS*	Выход	Хост готов к передаче #2
20	*DTR*	Выход	Хост готов
21	*RLOOP*	Выход	Внешняя петля
22	*RI*	Вход	Обнаружен вызов
23	*DRD*	Вход	Определена скорость данных
24	*TRCO*	Выход	Тактирование внешнего передатчика
25	*TEST*	Вход	Тестовый режим

2. ОПИСАНИЕ ПРЕОБРАЗОВАТЕЛЯ *USB/RS*-232 НА БАЗЕ МИКРОСХЕМЫ *PL*2303

*PL*2303 - микросхема преобразователя интерфейса *USB* в *RS*232. Фотография платы преобразователя *USB/RS*232 представлена на *Рис*. 1.

Рис. 1 Внешний платы преобразователя *USB/RS*232 на базе микросхемы *PL*2303

Как можно заметить этот модуль имеет два разъема: *USB* для подключения к персональному компьютеру и 4-х контактную штыревую линейку для связи с устройствами. Контакты штыревой линейки подписаны: «*Vcc*» - питание +5 В от шины *USB* персонального компьютера, «*RXD*» - вход приемника преобразователя, «*TXD*» - выход передатчика преобразователя и «*GND*» - нулевой провод источника питания. На плате имеется лишь одна микросхема преобразователя *PL*2303*HX* фирмы *Prolific Technology*. Все подобные преобразователи обычно строятся на таких микросхемах, которые значительно упрощают процедуру преобразования сигналов *USB* в сигналы *RS*232. При подключении модуля к компьютеру, и при отсутствии замыканий, на плате загорается красный светодиод, сигнализирующий о наличии питания по шине *USB*. Для нормальной работы устройства в системе необходимо установить драйвер *PL*2303*HX*, дистрибутив которого можно найти на сайте *Prolific Technology*. Установка драйвера обычно не вызывает трудностей. После установки, в системе должен появиться новый виртуальный *COM*-порт, наличие которого для операционных систем *Windows* можно проверить в диспетчере устройств. Подключение преобразователя к отладочной плате можно осуществить при помощи стандартных соединительных проводов.

3. ОПИСАНИЕ ПРОГРАММЫ *TERMINAL*

Для передачи данных от компьютера в микроконтроллер, и для приёма данных от микроконтроллера с последующим отображением результатов на экране монитора будем использовать программу *Terminal v*1.9 от автора Br@y++ [3]. Эта программа включает в себя удобный интерфейс по настройке параметров работы *COM*-порта, а также расширенные возможности по управлению поступающим потоком данных из коммуникационного порта. Внешний вид программы представлен на *Рис*. 2.

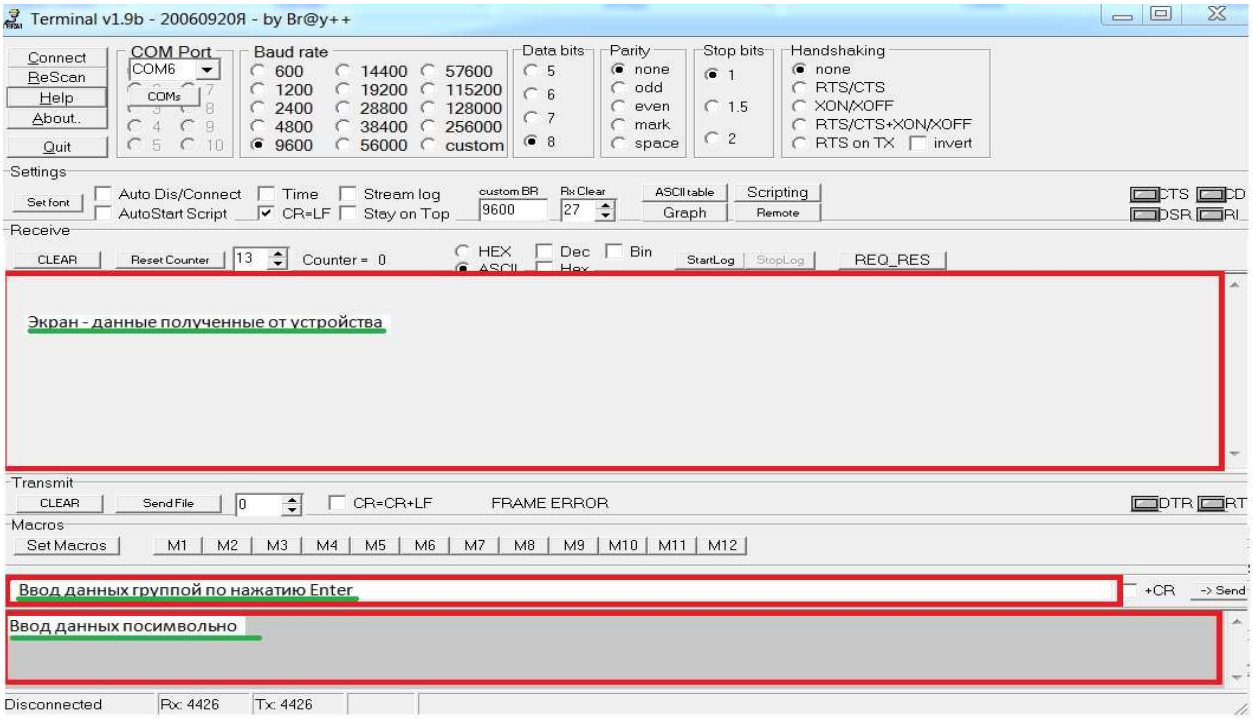


Рис. 2 Внешний вид программы *Terminal*

Рис. 3 Основные настройки программы *Terminal*

Программа *Terminal* содержит множество настроек. На *Рис*. 2 можно увидеть основные настройки для работы с программой:

- выбор COM-порта - при нажатие кнопки ReScan, программа найдет возможные порты для работы, после этого необходимый порт можно выбрать вручную;
- настройка скорости передачи данных - большинство устройств работают со скоростью 9600 бит/с; скорость обмена должна быть выставлена одинаковая на обоих устройствах, которые участвуют в обмене данными.

Количество информационных бит - по умолчанию 8 бит. Контроль четности - отсутствует. Количество стоповых бит - по умолчанию 1 бит. Управление служебными сигналами - отсутствует. Также в программе *Terminal* существует возможность во время работы вызвать таблицу *ASCII* кодов кнопкой *ASCII table*.

4. ПЕРЕДАЧА И ПОЛУЧЕНИЕ ДАННЫХ ЭХО-СИГНАЛА

Используя программу *Terminal* и преобразователь *PL*2303, передадим данные. Для того, чтобы передать и получить данные не задействовав дополнительные устройства замкнём контакты «*RXD*» и «*TXD*» используя джампер и подключим преобразователь к компьютеру как показано на *Рис*. 4.

Передадим код числа «1». Загружаем программу *Terminal*, в строке ввода данных набираем число «1». На экране вывода информации появляется «1» - см. *Рис*. 5.

Рис. 4 Подключение преобразователя *PL*2303 к компьютеру

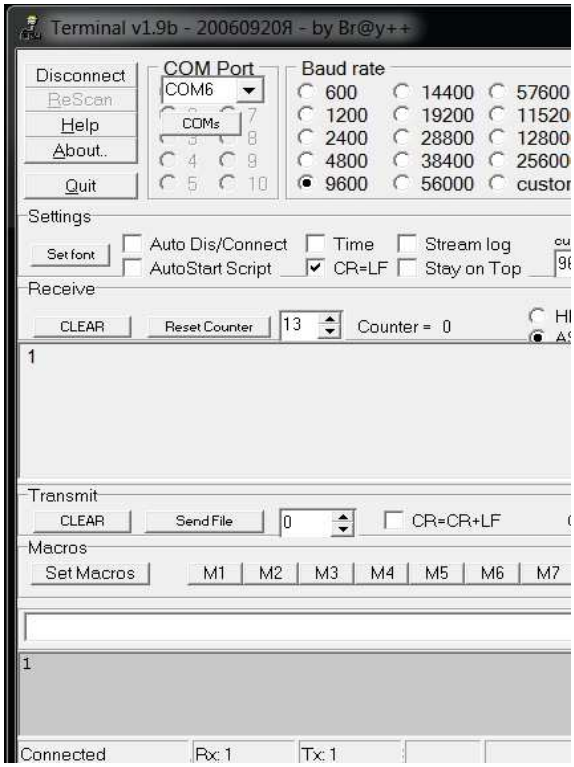


Рис. 5 Передача и получение данных используя программу *Terminal*

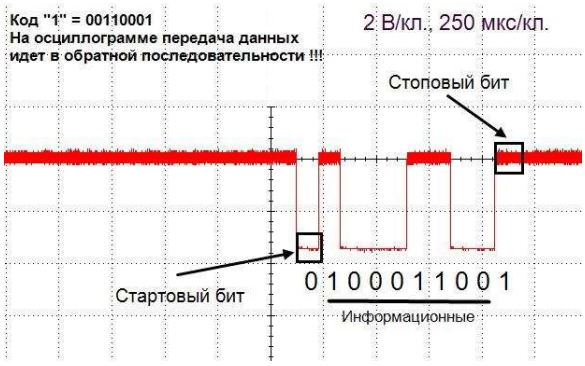


Рис. 6 Осциллограмма сигналов в устройстве

Подключим осциллограф в место соединения сигналов «*RXD*» - «*TXD*» и получим следующую осциллограмму - см. *Рис*. 6.

5. ОПИСАНИЕ STM32 USART

Обмен данными - одна из часто возникающих задач. Передача данных осуществляется между датчиком и устройством обработки данных, между управляющим блоком и исполнительным механизмом и т. д.

Существует множество способов передачи информации, но наиболее простым и надежным является передача данных через *USART*.

Рассмотрим *USART* модуль в микроконтроллере *STM*32. Универсальный асинхронный приёмопередатчик (УАПП, англ. *Universal Asynchronous Receiver-Transmitter, UART*) — узел вычислительных устройств, предназначенный для организации связи с другими цифровыми устройствами, в том числе по *RS*-232. Преобразует передаваемые данные в последовательный вид так, чтобы было возможно передать их по цифровой линии другому аналогичному устройству. Метод преобразования хорошо стандартизован и широко применялся в компьютерной технике. Представляет собой логическую схему, с одной стороны подключённую к шине вычислительного устройства, а с другой имеющую два или более выводов для внешнего соединения. *UART* может представлять собой отдельную микросхему (например, Intel I8251, I8250) или являться частью большой интегральной схемы (например, микроконтроллера) [4].

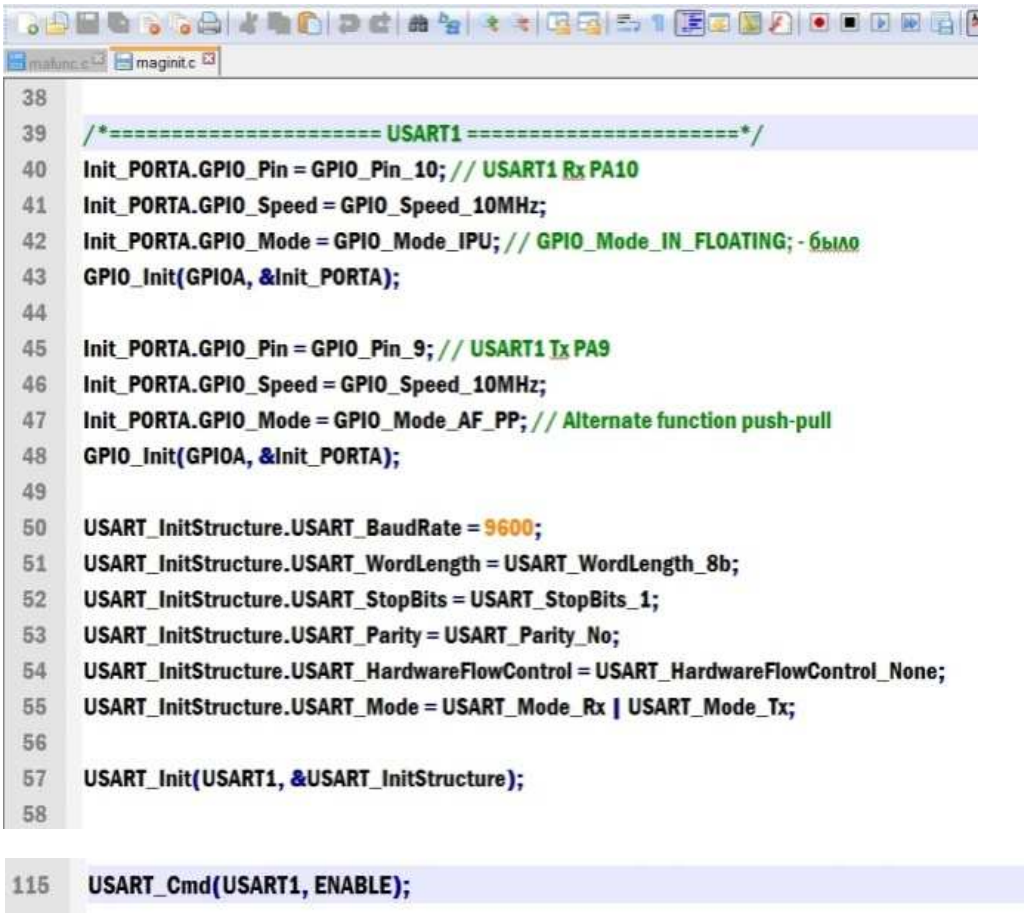
Для работы с модулем *USART* возьмем отладочную плату *STM*32*VLDISCOVERY* и проведем необходимо следующие настройки:

Вначале необходимо настроить параметры порта. Для облегчения работы будем использовать библиотеку производителя микроконтроллера *ST Microelectronics*. Создаём переменные на основе шаблонов, заполняем поля структур и вызываем функции, которые производят необходимую настройку "железа" - см. *Рис*. 7, 8.

```
1  //Объявляем переменные на основе шаблонов
2  GPIO_InitTypeDef  Init_PORTA;
3  USAR_InitTypeDef USART_InitStructure;
4  //Разрешаем подачу тактовых импульсов на подсистемы
5  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);
6  RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1, ENABLE);
```

Рис. 7. Объявление переменных

Используем простой пример части программы по приёму и передачи символа с кодом на единицу больше, см. *Рис*. 9.



```
38
39  /*====================== USART1 ======================*/
40  Init_PORTA.GPIO_Pin = GPIO_Pin_10; // USART1 Rx PA10
41  Init_PORTA.GPIO_Speed = GPIO_Speed_10MHz;
42  Init_PORTA.GPIO_Mode = GPIO_Mode_IPU; // GPIO_Mode_IN_FLOATING; - было
43  GPIO_Init(GPIOA, &Init_PORTA);
44
45  Init_PORTA.GPIO_Pin = GPIO_Pin_9; // USART1 Tx PA9
46  Init_PORTA.GPIO_Speed = GPIO_Speed_10MHz;
47  Init_PORTA.GPIO_Mode = GPIO_Mode_AF_PP; // Alternate function push-pull
48  GPIO_Init(GPIOA, &Init_PORTA);
49
50  USART_InitStructure.USART_BaudRate = 9600;
51  USART_InitStructure.USART_WordLength = USART_WordLength_8b;
52  USART_InitStructure.USART_StopBits = USART_StopBits_1;
53  USART_InitStructure.USART_Parity = USART_Parity_No;
54  USART_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
55  USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
56
57  USART_Init(USART1, &USART_InitStructure);
58

115 USART_Cmd(USART1, ENABLE);
```

Рис. 8 Настройка параметров порта

```
115 USART_Cmd(USART1, ENABLE);
```

```
53  /* ====================== MAIN ====================*/
54  int main(void)
55  {
56  static uint8_t ch;
57  #include "maginit.c"
58  //============== MAIN LOOP ====================
59  while (1) {
60  while (USART_GetFlagStatus(USART1, USART_FLAG_RXNE) == RESET) {};
61  ch = USART_ReceiveData(USART1); // Принимаем символ
62
63  while (USART_GetFlagStatus(USART1, USART_FLAG_TC) == RESET) {};
64  USART_SendData(USART1, ch+1); // Передаём символ с кодом +1
65  }
66  }
67
68  // КОНЕЦ
```

Рис. 9. Код программы получения и передачи символа с кодом на единицу больше

В данном примере в начале обращаемся к функции «*USART_GetFlagStatus*», которая проверяет состояния бита «*USART_FLAG_RXNE*» в «*USART*1». И если этот бит равен 0, т.е. нет данных от компьютера, то программа в этом месте зацикливается. Как только бит «*USART_FLAG_RXNE*» не будет равен 0, это означает что приняты данные от компьютера.

После этого переменной «ch» присваиваются полученные данные (байт). Далее обращаемся к функции «USART_GetFlagStatus», которая проверяет состояния бита «*USART_FLAG_TC*» в «*USART*1». И если этот бит равен 0, т.е. передатчик не готов передавать данные, то программа в этом месте зацикливается. Как только бит «*USART_FLAG_TC*» не будет равна 0 - программа передаст данные, которые хранятся в переменной «ch» предварительно добавив к ним число 1.

СХЕМА СОЕДИНЕНИЯ ПК-USB/RS-232-STM32.

Подключим отладочную плату *STM*32*VLDISCOVERY* к преобразователю *USB/RS*232 по следующей схеме (*Рис*. 10):

- *RX* платы к *TX* преобразователя;
- *TX* платы к *RX* преобразователя;
- *GND* платы к *GND* преобразователя.

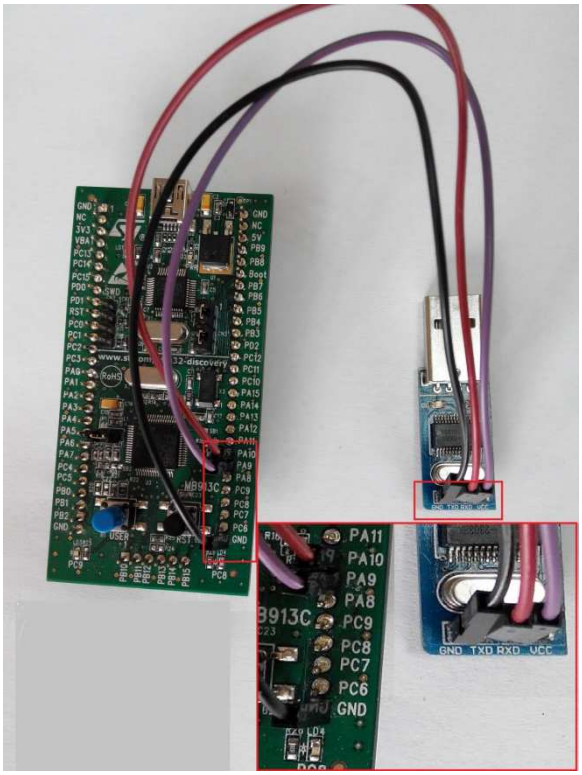


Рис. 10 Подключение *STM*32*VLDISCOVERY* к преобразователю *USB/RS*232

Запишем программу в микроконтроллер и запустим её. На компьютере запустим программу Terminal. В строке ввода данных набираем число «1». На экране вывода информации появляется «2» - см. *Рис*. 11.

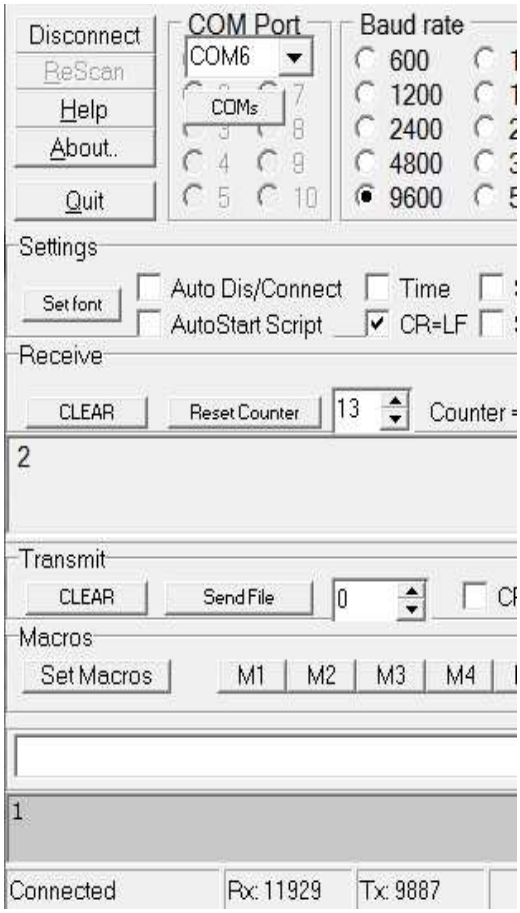


Рис. 11. Получение данных

Если подключиться параллельно к линиям *RX* и *TX*, то сможем наблюдать следующую осциллограмму, см. *Рис*.12.

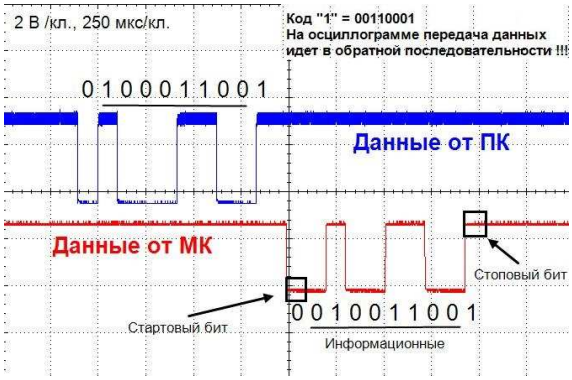


Рис. 12 Осциллограмма передачи/приёма данных ПК-STM32VLDISCOVERY

На данной осциллограмме хорошо видно некоторые особенности:

1. передача данных осуществляется в обратной последовательности, т.е. первым передаётся 0 бит данных, потом 1 и т.д.
2. вначале передачи идет стартовый бит - всегда «0»;
3. передача заканчивается стоповым битом - всегда «1».

ЗАКЛЮЧЕНИЕ

Работа выполнена при финансовой поддержке Минобрнауки России по государственному заданию №2014/138, тема проекта «Новые структуры, модели и алгоритмы для прорывных методов управления техническими системами на основе наукоемких результатов интеллектуальной деятельности».

ЛИТЕРАТУРА

[1] URL: http://www.gaw.ru/html.cgi/txt/interface/rs232/start.htm
[2] Terminal. Com port development tool. URL: https://sites.google.com/site/terminalbpp/ (дата обращения 5.02.14).
[3] Wikipedia URL: https://ru.wikipedia.org/wiki/Универсальный_асинхронный_приёмопередатчик

Вадим Аркадьевич Жмудь – заведующий кафедрой Автоматики НГТУ, профессор, доктор технических наук. Область научных интересов и компетенций – теория автоматического управления, электроника, лазерные системы, оптимизация, измерительная техника.
E-mail: oao_nips@bk.ru

Трубин Максим Витальевич - студент группы АА-46 кафедры Автоматики НГТУ
E-mail: tmv.kba@gmail.com

Трубин Игорь Витальевич - зам. дир. "КБ Автоматика"
E-mail: tiv.kba@gmail.com

Exchange of Data between the Computer and the Microcontroller STM32F100 by Serial Communication Interface RS-232

V.A. ZHMUD, I.V. TRUBIN, M.V. TRUBIN

Abstract: In today's world, automated control systems are widely used, but the control is impossible without communication between the devices. Currently set of communication interfaces are used, but undoubtedly RS-232 is still the easiest and most popular one. The paper deals with the serial data exchange between the microcontroller of a series of STM32F100 and computer via RS-232. The paper gives brief historical information on this interface. It proposes the description of the converter USB / RS232 based on the popular chip PL2303. It gives the way of connection to the computer in details. In addition, the paper gives the technique of checking of the transmitter without the use of other communication devices. Discuss of the example of the code is there also. Paper reveals the features of the program “Terminal”. The paper gives waveforms to improve the understanding of the process of data transmitting and receiving.

Keywords: RS-232, USART, STM32, STM32F100, Terminal, STM32VLDISCOVERY

REFERENCES

[1] URL: http://www.gaw.ru/html.cgi/txt/interface/rs232/start.htm
[2] Terminal. Com port development tool. URL: https://sites.google.com/site/terminalbpp/ (Data: 5.02.14).
[3] Wikipedia URL: https://ru.wikipedia.org/wiki/%D0%A3%D0%BD%D0%B8%D0%B2%D0%B5%D1%80%D1%81%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B0%D1%81%D0%B8%D0%BD%D1%85%D1%80%D0%BE%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BF%D1%80%D0%B8%D1%91%D0%BC%D0%BE%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%82%D1%87%D0%B8%D0%BA