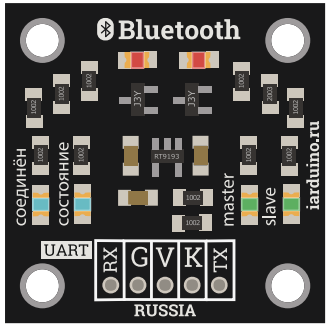
Bluetooth HC-05 (Trema-модуль v2.0)



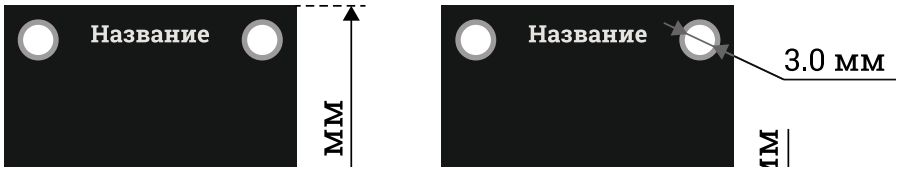
Общие сведения:

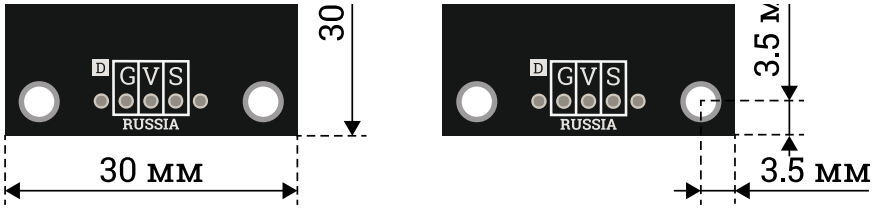
Trema-модуль Bluetooth HC-05 - это модуль беспроводной связи позволяющий передавать и принимать данные по радиоканалу на разрешённом ISM (Industry, Science and Medicine) диапазоне частот, от 2.4 ГГц до 2.5 ГГц, предназначенном для использования в индустриальных, научных и медицинских целях, используя метод AFH (Adaptive Frequency Hopping Feature) - адаптивной скачкообразной перестройки несущей частоты.

Спецификация:

- Напряжение питания: 3,3 ... 5 В
- Потребляемый ток при подключении: до 40 мА (поиск, сопряжение, подключение к другим Bluetooth устройствам)
- Потребляемый ток при передаче данных: до 8 мА
- Частотный диапазон: ISM 2,4 ... 2,48 ГГц
- Мощность передатчика: до +4 дБм
- Чувствительность приёмника: -80 дБм
- Дальность связи: 10 м
- Интерфейс: UART (с программируемой скоростью передачи данных)
- Максимальное напряжение на выводах TX и RX не должно превышать напряжение питания модуля.
- PIN-код по умолчанию: 1234 (у некоторых модулей 0000)
- Настройки UART по умолчанию: Скорость 38400 бит/сек, 8 бит данных, 1 стоп бит, без проверки чётности, с контролем данных.
- Поддерживаемые скорости UART: 9600,19200,38400,57600,115200,230400,460800 бит/сек.
- Рабочая температура: -25 ... +75 °C

Все модули линейки "Trema" выполнены в одном формате





Подключение:

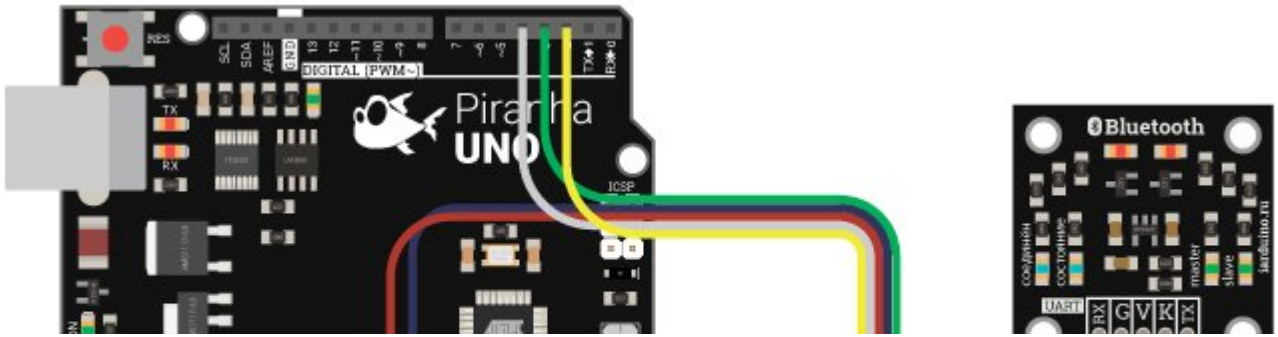
Trema Bluetooth модуль HC-05 подключается к Arduino по шине UART (можно использовать как аппаратную так и программную шину).

- Вывод модуля TX подключается к аппаратному (фиксированному) или программному (назначенному) выводу RX Arduino. Это линия шины UART для передачи данных от модуля к Arduino.
- Вывод модуля RX подключается к аппаратному (фиксированному) или программному (назначенному) выводу TX Arduino. Это линия шины UART для передачи данных в модуль от Arduino.
- Вывод модуля K подключается к любому выводу Arduino номер которого указывается в скетче. Это линия перевода модуля в режим AT-команд. Модуль в обычном режиме будет воспринимать AT-команды, только после того как на этот вывод кратковременно подать высокий уровень. Модуль перейдёт в режим AT-команд (на скорости 38400 бит/с и не будет соединяться с другими модулями) если на нём будет установлен высокий логический уровень при подаче питания или перезагрузке.

Модуль удобно подключать 3 способами, в зависимости от ситуации:

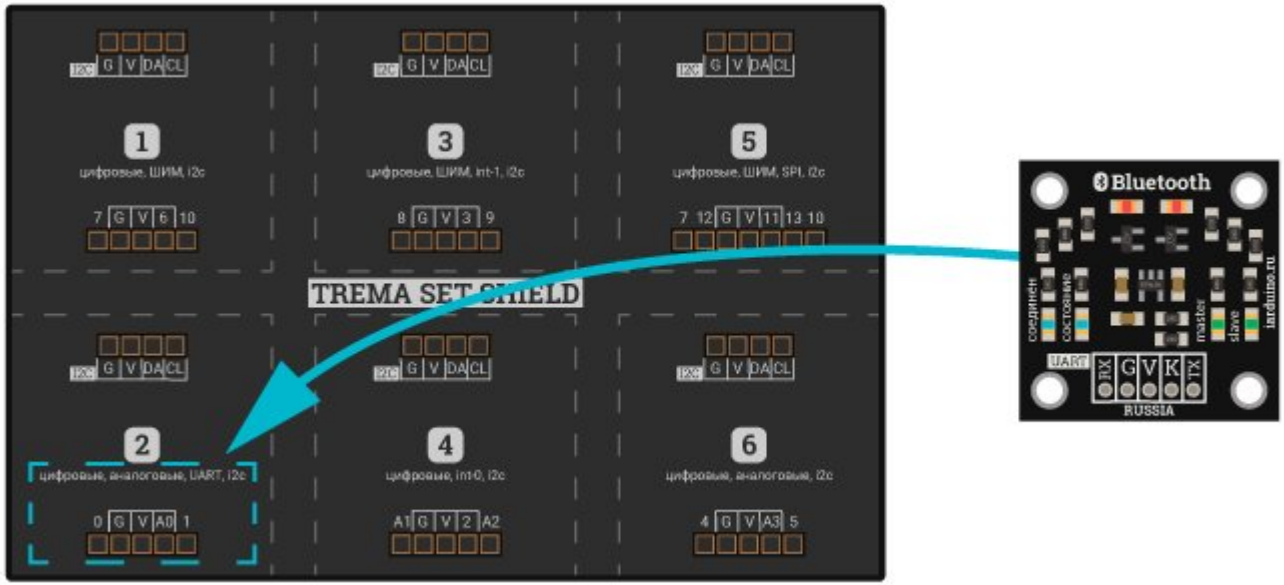
Способ - 1 : Используя проводной шлейф и Piranha UNO

Используя провода «Папа — Мама», подключаем напрямую к контроллеру Piranha UNO.



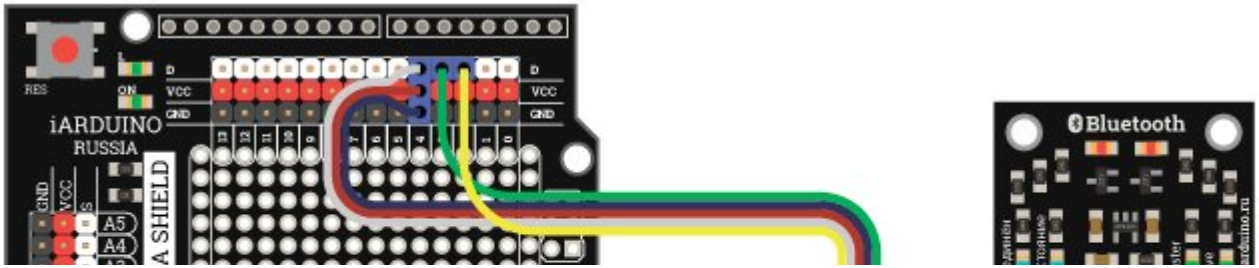
Способ - 2 : Используя Trema Set Shield

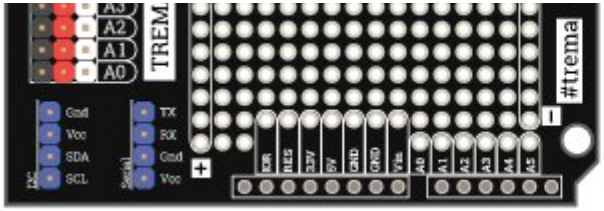
Модуль можно подключить к UART входу Trema Set Shield.

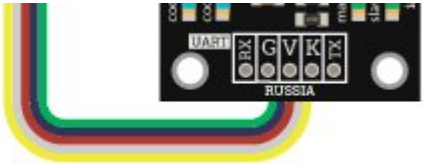


Способ - 3 : Используя проводной шлейф и Shield

Используя 2-х и 3-х проводные шлейфы, к Trema Shield, Trema-Power Shield, Motor Shield, Trema Shield NANO и тд.







Trema Bluetooth модуль HC-05 можно подключить и непосредственно к компьютеру (через адаптер USB-UART), передавая AT-команды из программ терминалов, о том как это сделать описано в разделе Wiki AT-команды Bluetooth.

Питание:

Входное напряжение питания 3,3 или 5 В постоянного тока, подаётся на выводы Vcc и GND модуля.

Подробнее о модуле:

В ISM диапазоне частот работают и радио модули nRF24L01+, но в отличии от этих модулей, которые работают на определённой частоте диапазона, Trema-модуль Bluetooth HC-05 используют метод AFH (Adaptive Frequency Hopping Feature) адаптивной скачкообразной перестройки несущей частоты (он меняет свою частоту 1600 раз в секунду). Несущая частота меняется псевдослучайным образом и заранее известна только паре «ведущий - ведомый», что обеспечивает не только устойчивость к помехам (занят канал? не беда, перейдём на другой) но и сохранение конфиденциальности передаваемых данных.

Преимуществом Bluetooth модулей перед другими модулями беспроводной передачи данных заключается в простоте работы (Вам не нужно знать протоколы, работать с регистрами, отслеживать сигналы и т.д.) и широкой распространённости данного типа передачи данных (Вы можете управлять Вашими устройствами, или получать их показания, практически с любого телефона, планшета, ноутбука).

Модуль не требует подключения антенны, т.к. она встроена (присутствует на ПП модуля).

В роли ведомого, Trema Bluetooth модуль HC-05 поддерживает метод автоматического сопряжения (создания пары) с другими Bluetooth устройствами выступающими в роли ведущих и инициировавших сопряжение.

Trema Bluetooth модуль HC-05 поддерживает автоматическое переподключение к последнему устройству (если информация о сопряжении с ним не была стёрта пользователем из списка пар). Если Вы установили подключение к другому Bluetooth устройству и связь с ним пропала

(отключилось питание, увеличилось расстояние и т.д.), то после устранения причины пропадания связи, устройства вновь будут готовы передавать/принимать данные, без Вашего вмешательства.

Trema Bluetooth модуль HC-05, в отличие от Bluetooth модулей HC-06, может работать не только в роли ведомого (slave), ожидая подключение, но и в роли ведущего (Master), инициируя поиск (обнаружение), сопряжение и подключение к другим Bluetooth модулям находящимся в радиусе действия. Модуль способен принимать и отправлять данные как в роли ведущего (master), так и в роли ведомого (slave).

Используя Trema Bluetooth модули HC-05 можно создавать связь между двумя Arduino, или между Arduino и другими устройствами, как ведущими (телефон, планшет, компьютер, ...), так и ведомыми (гарнитуры, клавиатуры, мышки, ...).

Специально для Trema Bluetooth модуль HC-05 нами разработана библиотека iarduino_Bluetooth_HC05 которая значительно упрощает процесс поиска любых Bluetooth устройств, создания ролей (master/slave) и сопряжения. Поиск и подключение к другим модулям Bluetooth осуществляется по их именам а не адресам. При использовании библиотеки для связи двух Trema Bluetooth модулей HC-05 можно передавать строки, значения и массивы любых типов, при этом библиотека осуществляет проверку доставки данных используя циклически избыточный код, так что Trema Bluetooth модуль HC-05 передавший данные получает подтверждение приёма, а Trema Bluetooth модуль HC-05 получивший данные может оперировать информацией о количестве элементов массива и его размере.

Подробнее про установку библиотеки читайте в нашей инструкции..

Примеры:

Trema Bluetooth модуль в роли ведомого (Slave) принимает данные:

Связь осуществляется между двумя Trema Bluetooth модулями HC-05 использующими библиотеку iarduino_Bluetooth_HC05. Скетч модуля исполняющего роль master приведён в следующем примере.

```
#include <SoftwareSerial.h>                                    // Подключаем библиотеку SoftwareSerial для об
#include <iarduino_Bluetooth_HC05.h>                           // Подключаем библиотеку iarduino_Bluetooth_HC
SoftwareSerial          softSerial(2,3);                       // Создаём объект softSerial указывая выводы R
iarduino_Bluetooth_HC05 hc05(4);                               // Создаём объект hc05 указывая любой вывод Ar
int                     myArray[3];                            // Объявляем массив в который будем получать д
```

```
                                                                    //
void setup(){                                                       //
                                        Serial.begin  (9600);       // Инициируем передачу данных по аппаратной ши
                                        Serial.print  ("begin: ");  // Выводим текст "begin: " в монитор последова
    if( hc05.begin(softSerial) )       {Serial.println("Ok");}      // Инициируем работу с Trema модулем hc05, ука
    else                               {Serial.println("Error");}   // Если работа с модулем не инициирована, то в
                                        Serial.print  ("create slave: "); // Выводим текст "create slave: " в монитор по
    if( hc05.createSlave("MyName","4567") ) {Serial.println("Ok");} // Создаем ведомую роль модулю, указывая его и
    else                               {Serial.println("Error");}   // Если роль не создалась - выводим сообщение
}                                                                   //
                                                                    //
void loop (){                                                       //
    if(hc05.available()){                                           // Если есть принятые данные, то ...
        hc05.read(myArray);                                         // Читаем полученные данные в ранее объявленны
                                        Serial.println(myArray[0]); // Выводим 0 элемент массива myArray в монитор
                                        Serial.println(myArray[1]); // Выводим 1 элемент массива myArray в монитор
                                        Serial.println(myArray[2]); // Выводим 2 элемент массива myArray в монитор
    }                                                               // Функции available() и read() работают тольк
}                                                                   // Если передача данных выполняется другим Blu
```

Инициализация модуля hc05.begin(); и создание ведомой роли hc05.createSlave(); может занять несколько секунд.
В этом примере модуль подключается через программный UART используя библиотеку SoftwareSerial, а при инициализации работы с модулем hc05.begin() указывается объект softSerial. Но модуль можно подключать и к аппаратному UART, тогда при инициализации работы с модулем hc05.begin() нужно указать Serial или Serial1, Serial2, Serial3, см. пример подключения к Arduino Mega.

При использовании Trema Bluetooth модуля HC-05 в качестве ведомого можно однократно вызвать функцию createSlave() с указанием имени и pin кода, после чего навсегда исключить эту функцию из кода. Тогда, при подаче питания, Trema Bluetooth модуля HC-05 будет стартовать в режиме ведомого и соединяться с первым ведущим который правильно укажет имя и pin ведомого.

Функция createSlave() объекта hc05 позволяет создать ведомую роль Trema Bluetooth модулю HC-05 при подключении к любым Bluetooth модулям, но функции available() и read() объекта hc05 работают только при организации связи между двумя Trema Bluetooth модулями HC-05

использующими библиотеку iarduino_Bluetooth_HC05! Если данные принимаются от другого Bluetooth модуля, то их нужно читать посимвольно из UART, тогда код loop будет выглядеть так:

```
...                                                                 // Если внешним Bluetooth устройством передаю
void loop(){                                                        // Данные принимаются посимвольным чтением из
    if(softSerial.available()){                                     // Если есть принятые данные, то ...
        String str;                                                 // Создаём строку str
        while(softSerial.available()){                              // Выполняем цикл пока есть что читать ...
            str1+=softSerial.read();                                // Читаем очередной принятый символ из UART в
            delay(5);                                               // Задержка на 5 мс на случай медленного приём
        }                                                           // Цикл завершён, значит читать больше нечего
        Serial.println(str);                                        // Выводим прочитанные данные одной строкой
    }                                                               //
}                                                                   //
```

Строки кода до функции loop остаются без изменений. Все данные отправленные внешним Bluetooth модулем принимаются как строки.

Trema Bluetooth модуль в роли ведущего (Master) передаёт данные:

Связь осуществляется между двумя Trema Bluetooth модулями HC-05 использующими библиотеку iarduino_Bluetooth_HC05. Скетч модуля исполняющего роль slave приведён в предыдущем примере.

```
#include <SoftwareSerial.h>                                         // Подключаем библиотеку SoftwareSerial для с
#include <iarduino_Bluetooth_HC05.h>                                // Подключаем библиотеку iarduino_Bluetooth_H
SoftwareSerial softSerial(2,3);                                     // Создаём объект softSerial указывая выводы
iarduino_Bluetooth_HC05 hc05(4);                                    // Создаём объект hc05 указывая любой вывод А
int myArray[3] = {123,456,789};                                     // Определяем массив с данными которые будем
void setup(){                                                       //
                                    Serial.begin  (9600);           // Инициируем передачу данных по аппаратной ш
                                    Serial.print  ("begin: ");      // Выводим текст "begin: " в монитор последов
    if( hc05.begin(softSerial) )   {Serial.println("Ok");}          // Инициируем работу с Trema модулем hc05, ук
```

```
    else                                    {Serial.println("Error");}          // Если работа с модулем не инициирована, то
                                             Serial.print  ("create master: "); // Выводим текст "create master: " в монитор
    if( hc05.createMaster("MyName","4567") ){Serial.println("Ok");}             // Создаем ведущую роль модулю, указывая имя
    else                                    {Serial.println("Error");}          // Если роль не создалась или не удалось подк
}                                                                              //
void loop (){                                                                  //
                                             Serial.print  ("send: ");         // Выводим текст "send: " в монитор последова
    if( hc05.send(myArray) )                {Serial.println("Ok");}            // Передаём данные массива myArray через Trem
    else                                    {Serial.println("Error");}          // Если данные не приняты ведомым bluetooth у
}                                                                              // Функция send() предназначена для отправки
```

Инициализация модуля hc05.begin(); и создание ведущей роли с подключением к ведомому hc05.createMaster(); может занять до минуты. В этом примере модуль подключается через программный UART используя библиотеку SoftwareSerial, а при инициализации работы с модулем hc05.begin() указывается объект softSerial. Но модуль можно подключать и к аппаратному UART, тогда при инициализации работы с модулем hc05.begin() нужно указать Serial или Serial1, Serial2, Serial3, см. пример подключения к Arduino Mega.

Функция createMaster() объекта hc05 позволяет создать ведущую роль Trema Bluetooth модулю HC-05 при подключении к любым Bluetooth модулям, но функция send() объекта hc05 работает только при организации связи между двумя Trema Bluetooth модулями HC-05 использующими библиотеку iarduino_Bluetooth_HC05! Так как функция send() добавляет 4 служебных байта к передаваемым данным (2 в начале и 2 в конце), которые приёмник будет считать за полученные данные. Если требуется отправить данные на другие Bluetooth модули, это лучше сделать отправкой строки по шине UART, тогда код loop будет выглядеть так:

```
...                                                                            // Если внешним Bluetooth устройством принимак
void loop(){                                                                   // Данные передаются строкой или символами чер
    softSerial.println(myArray[0]);                                            // Отправляем значение 0 элемента массива myAr
    softSerial.println(myArray[1]);                                            // Отправляем значение 1 элемента массива myAr
    softSerial.println(myArray[2]);                                            // Отправляем значение 2 элемента массива myAr
}                                                                              //
```

Строки кода до функции loop остаются без изменений. Все данные передаются символами (байтами) или строками. По этому элементы массива отправляются по отдельности.

Скетч поиска любых Bluetooth устройств в радиусе действия:

```
#include <SoftwareSerial.h>                                          // Подключаем библиотеку SoftwareSerial для об
#include <iarduino_Bluetooth_HC05.h>                                 // Подключаем библиотеку iarduino_Bluetooth_HC
SoftwareSerial softSerial(2,3);                                      // Создаём объект softSerial указывая выводы R
iarduino_Bluetooth_HC05 hc05(4);                                     // Создаём объект hc05 указывая любой вывод Ar
int myArray[3] = {123,456,789};                                      // Определяем массив с данными которые будем г
void setup(){                                                        //
                                     Serial.begin  (9600);           // Инициируем передачу данных по аппаратной ши
                                     Serial.print  ("begin: ");      // Выводим текст "begin: " в монитор последова
    if( hc05.begin(softSerial) )    {Serial.println("Ok");}          // Инициируем работу с Trema модулем hc05, ука
    else                             {Serial.println("Error");}      // Если работа с модулем не инициирована, то в
}                                                                    //
void loop (){                                                        //
    int i = hc05.find(10);                                           // Выполняем поиск устройств, не дольше 10 сек
    if(i){                           Serial.print  ("found ");
                                     Serial.print  (i);
                                     Serial.println(" devices:");
        for(int j=0; j<i; j++){      Serial.print  (  "Name: ");    Serial.print(hc05.findName[j]);
                                     Serial.print  (", Address: "); Serial.print(hc05.findAddr[j]);
                                     Serial.println(";");
        }
    }else{                           Serial.println("Device not found.");}
}
```

Инициализация модуля hc05.begin(); может занять несколько секунд.

В этом примере модуль подключается через программный UART используя библиотеку SoftwareSerial, а при инициализации работы с модулем hc05.begin() указывается объект softSerial. Но модуль можно подключать и к аппаратному UART, тогда при инициализации работы с

модулем hc05.begin() нужно указать Serial или Serial1, Serial2, Serial3, см. пример подключения к Arduino Mega.

Скетч поиска любых Bluetooth устройств при подключении модуля к аппаратной шине UART1 платы Arduino Mega:

```
#include <iarduino_Bluetooth_HC05.h>                                    // Подключаем библиотеку iarduino_Bluetooth_HC
iarduino_Bluetooth_HC05 hc05(2);                                        // Создаём объект hc05 указывая любой вывод Ar
int myArray[3] = {123,456,789};                                         // Определяем массив с данными которые будем г
                                                                        // Модуль подключается к аппаратной шине UART1
void setup(){                                                           //
                                    Serial.begin(9600);                 // Инициируем передачу данных по аппаратной ши
                                    Serial.print  ("begin: ");          // Выводим текст "begin: " в монитор последова
    if( hc05.begin(Serial1) )      {Serial.println("Ok");}              // Инициируем работу с Trema модулем hc05, ука
    else                           {Serial.println("Error");}           // Если работа с модулем не инициирована, то в
}                                                                       //
void loop (){                                                           //
    int i = hc05.find(10);                                              // Выполняем поиск устройств, не дольше 10 сек
    if(i){                          Serial.print("found ");
                                    Serial.print(i);
                                    Serial.println(" devices:");
        for(int j=0; j<i; j++){     Serial.print(  "Name: ");    Serial.print(hc05.findName[j]);
                                    Serial.print(", Address: "); Serial.print(hc05.findAddr[j]);
                                    Serial.println(";");

        }
    }else{                          Serial.println("Device not found.");}
    delay(1000);
}
```

Инициализация модуля hc05.begin(); может занять несколько секунд.

В этом примере модуль подключается к Arduino MEGA через аппаратный UART1, а при инициализации работы с модулем hc05.begin() указывается объект Serial1. Подключение дополнительных библиотек и инициализация шины UART1 не нужна ~~Serial1.begin(38400);~~. У платы

Arduino MEGA имеется 4 аппаратных шины UART с которыми работают классы: Serial, Serial1, Serial2, Serial3, любой из них можно использовать при инициализации модуля для работы с ним.

Описание основных функций библиотеки:

Подключение библиотеки:

```
#include <iarduino_Bluetooth_HC05.h> // Подключаем библиотеку для работы с модулем
iarduino_Bluetooth_HC05 hc05(4);     // Создаём объект hc05 указывая любой вывод Arduino который подключается к выводу K модул
```

Функция begin();

- Назначение: Инициализация работы с модулем по шине UART.
- Синтаксис: begin(ОБЪЕКТ_UART);
- Параметры:
  - ОБЪЕКТ - класс или объект который используется для работы с шиной UART к которой подключён модуль.
- Возвращаемые значения: bool - результат инициализации (true или false).
- Примечание:
  - Функция вызывается 1 раз в коде setup.
  - Используемый ОБЪЕКТ_UART не нуждается в предварительной инициализации (~~Serial.begin(СКОРОСТЬ);~~)
  - Можно указать объект программной шины UART или класс Serial, Serial1, Serial2, Serial3 и т.д. в зависимости от используемой платы Arduino. Если указать класс Serial, то при загрузке скетча потребуется отключать модуль от Arduino, а все выводы данных в монитор последовательного порта будут восприниматься модулем как команды.
- Пример:

```
hc05.begin(Serial1);                 // Инициируем работу модуля по аппаратной шине UART используя класс Serial1
```

- Пример:

```
#include <SoftwareSerial.h>          // Подключаем библиотеку SoftwareSerial для работы с программной шиной UART
```

```
SoftwareSerial softSerial(2,3);      // Создаём объект softSerial назначая выводы RX, TX (можно указывать любые выводы Arduinc
...
hc05.begin(softSerial);              // Инициируем работу модуля по программной шине UART используя объект softSerial
```

Функция createSlave();

- Назначение: Создания ведомого Bluetooth устройства (Slave) ожидающего подключение.
- Синтаксис: createSlave(ИМЯ , PIN-КОД);
- Параметры:
  - ИМЯ - строка содержащая имя назначаемое данному Bluetooth модулю.
  - PIN-КОД - строка содержащая код доступа к данному Bluetooth модулю.
- Возвращаемые значения: bool - результат создания ведомого устройства (true или false).
- Примечание:
  - ИМЯ не должно превышать 32 символа.
  - PIN-КОД не должен превышать 16 символов.
  - Ранее установленные соединения (если они были) будут разорваны.
  - Если функция вернула true, то её дальнейший вызов не обязателен (даже после отключения питания).
    Модуль будет соединяться с любыми ведущими (Master) Bluetooth устройствами, которые правильно введут Имя и PIN.
- Пример:

```
hc05.createSlave("MyName", "4567");  // Создание ведомого с именем "MyName" и PIN-кодом "4567"
```

Функция createMaster();

- Назначение: Создания ведущего Bluetooth устройства (Master) с подключением к ведомому устройству.
- Синтаксис: createMaster(ИМЯ , PIN-КОД);
- Параметры:
  - ИМЯ - строка содержащая имя ведомого Bluetooth устройства к которому требуется подключиться.
  - PIN-КОД - строка содержащая код доступа ведомого Bluetooth устройства к которому требуется подключиться.
- Возвращаемые значения: bool - результат создания ведущего устройства и подключения к ведомому (true или false).

- Примечание:
  - ИМЯ не должно превышать 32 символа.
  - PIN-КОД не должен превышать 16 символов.
  - Ранее установленные соединения (если они были) будут разорваны.
  - Ведомое устройство (Slave) должно находиться в радиусе действия и ожидать подключения.
  - Если функция вернула true, то её дальнейший вызов не обязателен (даже после отключения питания).
    Модуль будет автоматически соединяться с указанным ведомым (Slave) Bluetooth устройством, пока не будет вызвана функция библиотеки разрывающая данное соединение, как на ведущем (Master), так и на ведомом (Slave) Bluetooth устройстве.
- Пример:

```
hc05.createMaster("MyName", "4567");  // Создание ведущего с подключением к ведомому устройству "MyName" по PIN-коду "4567"
```

Функция find();

- Назначение: Поиск любых устройств Bluetooth в радиусе действия.
- Синтаксис: find(ВРЕМЯ , [ФЛАГ]);
- Параметры:
  - ВРЕМЯ - число от 1 до 48 определяющее максимальное время поиска Bluetooth устройств.
  - ФЛАГ - значение (true или false) разрешающее поиск по уровню сигнала.
- Возвращаемые значения: uint8_t - количество найденных Bluetooth устройств.
- Примечание:
  - Время поиска рассчитывается умножением числа ВРЕМЯ на 1,28 секунд.
  - Необязательный параметр ФЛАГ по умолчанию сброшен (поиск ведётся в стандартном режиме).
  - Функция прекращает поиск при обнаружении 5 устройств или по достижении указанного времени.
  - Ранее установленные соединения (если они были) будут разорваны.
  - Функция только находит Bluetooth устройства, но не сопрягает и не соединяет модуль с ними.
  - Имена и адреса найденных устройств доступны в строковых массивах findName и findAddr.
- Пример:

```

```

```
i=hc05.find(10);                        // Ищем Bluetooth устройства, но не более чем 10*1,28 секунд
if(i){                                  // Если найдено хоть одно Bluetooth устройство, то ...
    for(int j=0; j<i; j++){             // Проходим по всем найденным устройствам и выводим результат в монитор последовательног
        Serial.print(") Name: "); Serial.print(hc05.findName[j]); Serial.print(", Address: "); Serial.println(hc05.findAddr[j]
    }
}
```

Функция end();

- Назначение: Разрыв подключения к внешнему Bluetooth устройству.
- Синтаксис: end();
- Параметры: Нет.
- Возвращаемые значения: Нет.
- Примечание:
  - Функция работает вне зависимости от роли модуля и того было подключение или нет.
  - После выполнения функции, роль модуля сохраняется.
  - После выполнения функции, автоматическое переподключение к последнему Bluetooth устройству будет недоступно.
- Пример:

```
hc05.end();                             // Разорвать соединение с внешним Bluetooth устройством
```

Функция checkConnect();

- Назначение: Проверка подключения к внешнему Bluetooth устройству.
- Синтаксис: checkConnect();
- Параметры: Нет.
- Возвращаемые значения: bool - наличие соединения (true или false).
- Примечание: Функция работает вне зависимости от роли модуля.
- Пример:

```
```

```
bool f=hc05.checkConnect();          // Получаем результат наличия соединения с внешним Bluetooth устройством в переменную-фл
```

Функция send();

- Назначение: Отправка данных внешнему Trema Bluetooth модулю, так же использующему эту библиотеку.
- Синтаксис: send(ДАННЫЕ);
- Параметры:
  - ДАННЫЕ - строка, переменная или массив любого типа, которые требуется отправить.
- Возвращаемые значения: bool - подтверждение приёма данных внешним Bluetooth устройством (true или false).
- Примечание:
  - Функция работает только при организации связи между двумя Trema Bluetooth модулями использующими эту библиотеку.
  - Размер отправляемых данных не должен превышать 54 байта.
  - До передачи данных необходимо установить соединение функциями createMaster() или createSlave().
  - Отправлять данные модуль может вне зависимости от установленной роли (как Master, так и Slave).
  - Функция вернёт true только если внешнее Trema Bluetooth устройство использует данную библиотеку и оно получило отправленные данные полностью, и без ошибок (библиотека проверяет ошибки сверяя CRC16 на принимающей стороне.
  - Функция способна отправлять только те массивы, размер которых был явно указан при их объявлении.
  - Если данной функцией отправить данные на Bluetooth устройство не использующее эту библиотеку, то оно получит данные в виде строки с двумя дополнительными символами в начале и конце строки.
- Пример:

```
int    a     = 1234;               // Определяем переменную типа int (целочисленные значения)
long   b[2]  = {1234,5678};        // Определяем массив данных типа long (целочисленные длинные значения)
double c     = 12.34;              // Определяем переменную типа double (значения с плавающей точкой)
char   d[12] = "iArduino.ru"       // Определяем строку из 12 символов (11 значащих символов + символ конца строки)
...
    hc05.send(a);                  // Отправляем переменную «a» (без проверки доставки)
if( hc05.send(a) ){ ... }          // Отправляем переменную «a» (если данные доставлены, то выполнится код в теле оператора
if( hc05.send(b) ){ ... }          // Отправляем переменную «b» (если данные доставлены, то выполнится код в теле оператора
if( hc05.send(c) ){ ... }          // Отправляем переменную «c» (если данные доставлены, то выполнится код в теле оператора
```

```
if( hc05.send(d) ){ ... }            // Отправляем переменную «d» (если данные доставлены, то выполнится код в теле оператора
    hc05.send("Prosto stroka");      // Отправляем строку         (без проверки доставки)
```

Функция available();

- Назначение: Проверка наличия принятых данных от внешнего Trema Bluetooth модуля, так же использующего эту библиотеку.
- Синтаксис: available([&ЭЛЕМЕНТЫ [,& БАЙТЫ]);
- Параметры:
  - ЭЛЕМЕНТЫ - необязательная ссылка на переменную в которую вернётся количество элементов в принятом массиве. Если принята переменная, а не массив, то вернётся 0.
  - БАЙТЫ - необязательная ссылка на переменную в которую вернётся количество байт в принятом массиве или переменной.
- Возвращаемые значения: bool - наличие или отсутствие принятых данных доступных для чтения (true или false).
- Примечание:
  - Функция работает только при организации связи между двумя Trema Bluetooth модулями использующими эту библиотеку.
  - Если нет принятых данных, то функция вернёт false, а переменные на которые указывают ссылки останутся без изменений.
  - Если есть принятые данные доступные для чтения, то функция вернет true, а сами данные можно прочитать функцией read().
  - Если данные приняты с ошибкой (не совпал CRC16) то функция поведёт себя так, как будто нет принятых данных.
  - Принимать данные модуль может вне зависимости от установленной роли (как Master, так и Slave).
  - Функцию нельзя использовать для проверки получения данных от Bluetooth модулей не использующих эту библиотеку, так как в их данных не будет присутствовать CRC16.
- Пример:

```
int a;                               // Объявляем переменную в которой будет храниться количество элементов принятого массива
int b;                               // Объявляем переменную в которой будет храниться размер принятых данных в байтах
if( hc05.available()      ){ ... }   // Если есть принятые данные, то выполнится код в теле оператора if
if( hc05.available(&a)    ){ ... }   // Тоже самое, только в переменной «a» будет находиться число равное количеству элементо
if( hc05.available(&a,&b) ){ ... }   // Тоже самое, только в переменной «b» будет находиться число равное количеству байт в п
```

Функция read();

- Назначение: Чтение данных пришедших от внешнего Trema Bluetooth модуля, так же использующего эту библиотеку.
- Синтаксис: read(ПЕРЕМЕННАЯ);
- Параметры:
  - ПЕРЕМЕННАЯ - в которую будут записаны принятые данные (строки, числа, массивы).
- Возвращаемые значения: bool - подтверждение успешного чтения (true или false).
- Примечание:
  - Функция работает только при организации связи между двумя Trema Bluetooth модулями использующими эту библиотеку.
  - До приёма данных необходимо проверить их наличие функцией available().
  - Тип переменной в которую читаются данные должен совпадать с типом передаваемых данных.
    Чтение произойдёт даже если типы не совпадают, но принятые данные могут отличаться от переданных.
  - Принимать данные модуль может вне зависимости от установленной роли (как Master, так и Slave).
  - Если после функции available() и до функции read() выполнить любые другие функции библиотеки, то принятые данные будут потеряны.
  - Функцию нельзя использовать для получения данных от Bluetooth модулей не использующих эту библиотеку, так как перед данной функцией требуется обратиться к функции available().
- Пример:

```
int a;                                // Объявляем переменную типа int (целочисленные значения)
if( hc05.available() ){hc05.read(a);} // Если есть принятые данные, то читаем их в переменную «a»
```

- Пример:

```
long b[2];                            // Объявляем массив данных типа long (целочисленные длинные значения)
if( hc05.available() ){hc05.read(b);} // Если есть принятые данные, то читаем их в массив «b»
```

Функция runAT();

- Назначение: Выполнение AT-команд и вывод результата.

- Синтаксис: runAT(КОМАНДА [, ВРЕМЯ [, ВЫХОД]]);
- Параметры:
  - КОМАНДА - строка содержащая AT-команду которую требуется выполнить.
  - ВРЕМЯ - положительное целое число миллисекунд которое отводится на выполнение команды (по умолчанию 500 мс).
  - ВЫХОД - флаг (true или false) разрешающий досрочный выход при успешном выполнении команды (по умолчанию true).
- Возвращаемые значения: char* - указатель на строку с ответом модуля.
- Примечание:
  - Данная функция позволяет выполнять операции с модулем которые не вошли в функционал библиотеки.
  - Результат выполнения AT-команды можно анализировать как по строке ответа модуля, так и по значению переменной flgResult.
- Пример:

```
Serial.println(hc05.runAT("AT\r\n")); // Выполняем AT-команду "AT" - тест, результат выводим в монитор последовательного порта
```

- Пример:

```
hc05.runAT("AT\r\n");                // Выполняем AT-команду "AT" - тест
if( hc05.flgResult < 0 ){ ... }      // Если в ответе встретилась фраза "ERROR:(x)" - ошибка, где x это число
if( hc05.flgResult > 0 ){ ... }      // Если в ответе встретилась фраза "OK\r\n" - успешное выполнение команды
if( hc05.flgResult ==0 ){ ... }      // Если в ответе нет ни ошибки ни успеха - результат выполнения AT-команды не определён
```

Переменная flgResult:

Содержит результат выполнения последней AT-команды (тип int8_t), -1: Error(*), 0: Неизвестно, 1: OK.

Массив findName:

Содержит пять строк с именами (или адресами) найденных Bluetooth устройств (тип char*).

Массив findAddr:

Содержит пять строк с адресами найденных Bluetooth устройств (тип char*).

Библиотека работает только с Trema Bluetooth модулями HC-05.

Если требуется принять или передать данные Bluetooth устройству которое не является Trema-модулем, или не использует данную библиотеку, то соединение с этим устройством выполняется функциями createMaster() или createSlave(), а передача и приём данных осуществляется через стандартные функции объекта UART через который осуществляется связь с модулем, см. дополнение к первым двум примерам.

Применение:

- Создание связи между двумя Arduino.
- Создание связи между Arduino и другими ведущими Bluetooth устройствами: телефонами, планшетами, компьютерами и т.д.
- Создание связи между Arduino и другими ведомыми Bluetooth устройствами: гарнитурами, клавиатурами, мышками и т.д.
- Дистанционное управление роботами, устройствами, проектами и т.д.
- Дистанционное получение данных от датчиков, детекторов, сигнализаций и т.д.
- Создание Bluetooth ретрансляторов для увеличения дальности беспроводной связи.

Ссылки:

- Bluetooth HC-05 (Trema-модуль).
- Библиотека iarduino_Bluetooth_HC05.
- Wiki - Установка библиотек в Arduino IDE.
- Wiki - AT-команды Bluetooth.