Датчик температуры и влажности, FLASH-I2C (Trema-модуль)

Общие сведения:

Trema модуль - Датчик температуры и влажности, I2C-flash - является цифровым датчиком способным возвращать значение температуры окружающей среды в °C и значение относительной влажности воздуха в %.

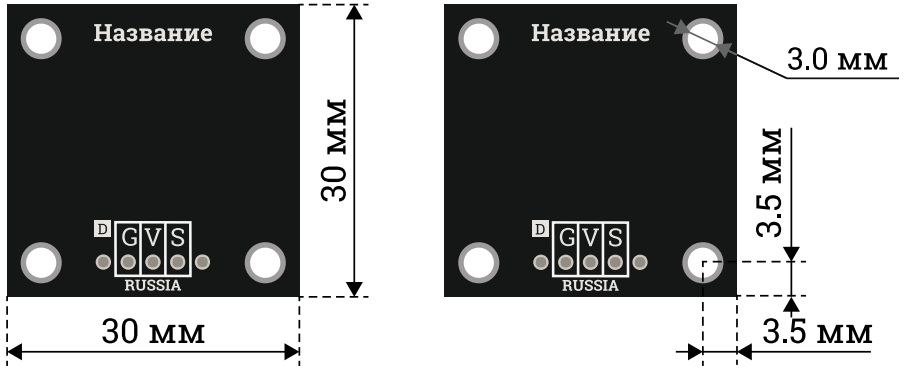
Модуль относится к серии «Flash», а значит к одной шине I2C можно подключить более 100 модулей, так как их адрес на шине I2C (по умолчанию 0x09), хранящийся в энергонезависимой памяти, можно менять программно.

Модуль можно использовать в большинстве проектов где требуется следить за метеоданными.

Спецификация:

- Напряжение питания: 3,3 В или 5 В (постоянного тока).
- Потребляемый ток: до 5 мА.
- Диапазон измерений температуры окружающей среды: от -40,0 до +125,0 °C.
- Диапазон измерений относительной влажности воздуха: от 0,0 до 100,0%.
- Интерфейс: I2C.
- Скорость шины I2C: 100 кбит/с.
- Адрес на шине I2C: устанавливается программно (по умолчанию 0x09).
- Уровень логической 1 на линиях шины I2C: 3,3 В (толерантны к 5 В).
- Рабочая температура: от -20 до +70 °C.
- Габариты: 30 x 30 мм.
- Вес: 4 г.

Все модули линейки "Trema" выполнены в одном формате



Подключение:

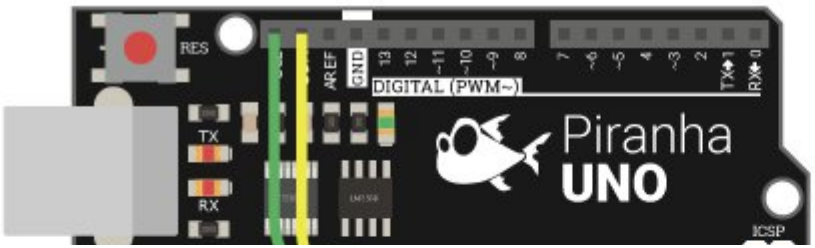
Модуль подключается к аппаратной или программной шине I2C Arduino.
Для удобства подключения, предлагаем воспользоваться TremaShield.

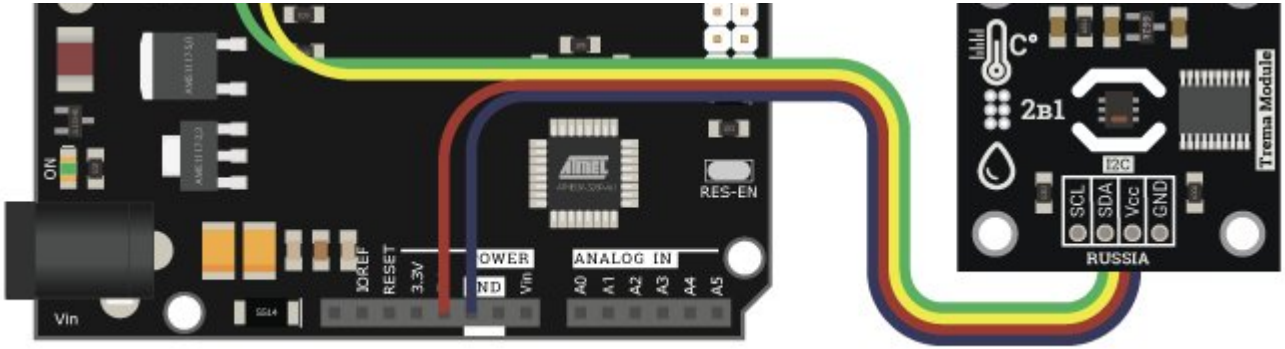
> Перед подключением модуля ознакомьтесь с разделом "Смена адреса модуля на шине I2C" в данной статье.

Модуль удобно подключать 3 способами, в зависимости от ситуации:

Способ - 1 : Используя проводной шлейф и Piranha UNO

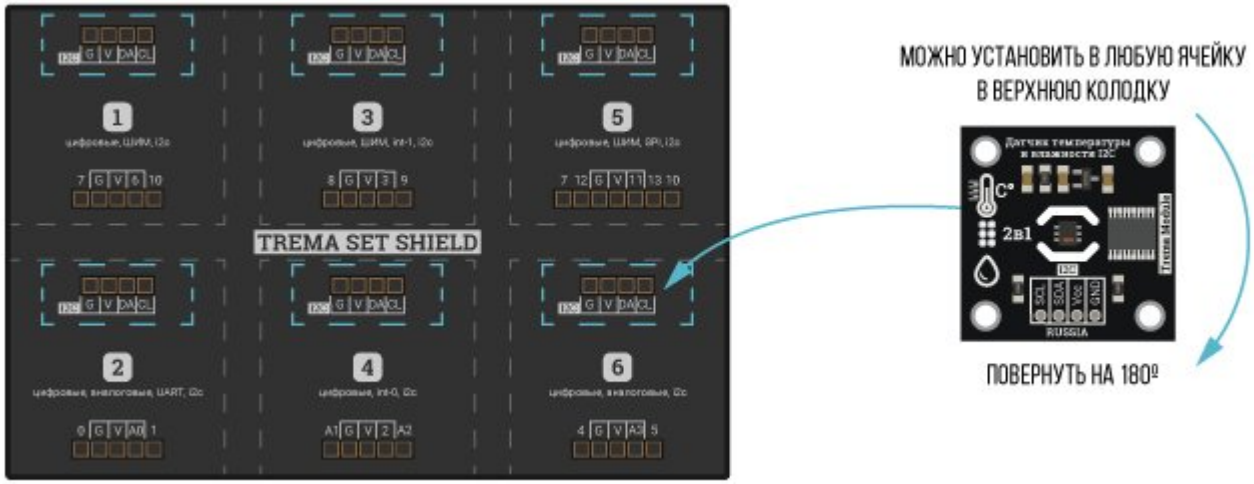
Используя провода «Папа — Мама», подключаем напрямую к контроллеру Piranha UNO.





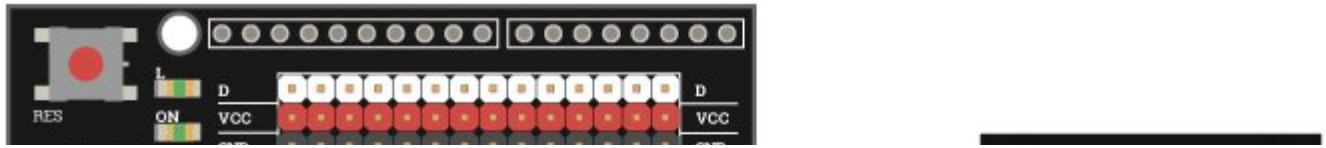
Способ - 2 : Используя Trema Set Shield

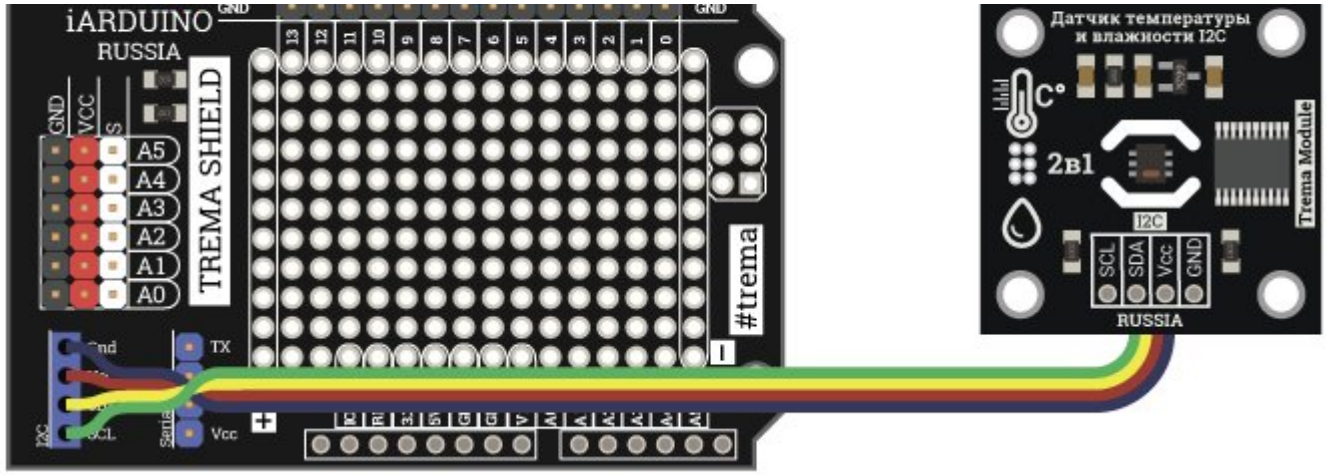
Модуль можно подключить к любому из I2C входов Trema Set Shield.



Способ - 3 : Используя проводной шлейф и Shield

Используя 4-х проводной шлейф, к Trema Shield, Trema-Power Shield, Motor Shield, Trema Shield NANO и тд.





Питание:

Входное напряжение питания модуля 3,3В или 5В постоянного тока (поддерживаются оба напряжения питания), подаётся на выводы Vcc и GND.

Подробнее о модуле:

Модуль построен на базе микроконтроллера STM32F030F4 и снабжен собственным стабилизатором напряжения. Модуль самостоятельно считывает значения с датчика температуры и влажности, обрабатывает их, и возвращает запрошенные результаты.

Модуль позволяет:

- Менять свой адрес на шине I2C.
- Получать относительную влажность воздуха.
- Получать температуру окружающей среды.
- Реагировать на изменение влажности с заданным порогом чувствительности.
- Реагировать на изменение температуры с заданным порогом чувствительности.
- Задать период обновления показаний влажности и температуры.

Специально для работы с Trema модулем - Датчик температуры и влажности, I2C-flash , нами разработана библиотека iarduino_I2C_SHT

которая позволяет реализовать все функции модуля.

Подробнее про установку библиотеки читайте в нашей инструкции.

Смена адреса модуля на шине I2C:

По умолчанию все модули FLASH-I2C имеют установленный адрес 0x09. Если вы планируете подключать более 1 модуля на шину I2C, необходимо изменить адреса модулей таким образом, чтобы каждый из них был уникальным. Более подробно о том, как изменить адрес, а также о многом другом, что касается работы FLASH-I2C модулей, вы можете прочесть в **этой статье**.

В первой строке скетча необходимо записать в переменную **newAddress** адрес, который будет присвоен модулю. После этого подключите модуль к контроллеру и загрузите скетч. **Адрес может быть от 0x07 до 0x7F**.

```
uint8_t newAddress = 0x09;                          //   Назначаемый модулю адрес (0x07 < адрес < 0x7F).
                                                    //
#include <Wire.h>                                   //   Подключаем библиотеку для работы с аппаратной шиной I2C.
#include <iarduino_I2C_SHT.h>                       //   Подключаем библиотеку для работы с датчиком температуры и влажности I2C
iarduino_I2C_SHT sht;                               //   Объявляем объект sht для работы с функциями и методами библиотеки iardu
                                                    //   Если при объявлении объекта указать адрес, например, sht(0xBB), то прим
void setup(){                                       //
    Serial.begin(9600);                             //
    if( sht.begin() ){                              //   Инициируем работу с датчиком температуры и влажности.
        Serial.print("Найден модуль с адр. 0x");     //
        Serial.println( sht.getAddress(), HEX );    //   Выводим текущий адрес модуля.
        if( sht.changeAddress(newAddress) ){        //   Меняем адрес модуля на newAddress.
            Serial.print("Адрес изменён на 0x");    //
            Serial.println(sht.getAddress(),HEX );//   Выводим текущий адрес модуля.
        }else{                                      //
            Serial.println("Адрес не изменён!");    //
        }                                           //
    }else{                                          //
```

```
        Serial.println("Модуль не найден!");      //
    }                                             //
}                                                 //
                                                  //
void loop(){}                                     //
```

Примеры:

В данном разделе раскрыты примеры получения данных от модуля по шине I2C с использованием библиотеки iarduino_I2C_SHT. Сама библиотека содержит больше примеров, доступных из меню Arduino IDE: Файл / Примеры / iarduino I2C SHT (датчик температуры и влажности).

Чтение всех значений модуля:

```
#include <Wire.h>                                 // * Подключаем библиотеку для работы с аппаратной шиной I2C.
#include <iarduino_I2C_SHT.h>                     //   Подключаем библиотеку для работы с датчиком температуры и влажности I2C
iarduino_I2C_SHT sht;                             //   Объявляем объект sht для работы с функциями и методами библиотеки iardu
                                                  //   Если при объявлении объекта указать адрес, например, sht(0xBB), то прим
void setup(){                                     //
    delay(500);                                   // * Ждём завершение переходных процессов связанных с подачей питания.
    Serial.begin(9600);                           //
    while(!Serial){;}                             // * Ждём завершения инициализации шины UART.
    sht.begin();                                  //   Инициируем работу с датчиком температуры и влажности.
}                                                 //
                                                  //
void loop(){                                      //
    Serial.print("Температура = ");                //
    Serial.print( sht.getTem() );                 //   Выводим текущую температуру, от -40.0 до +125 °C.
    Serial.print(" °C, влажность = ");            //
    Serial.print( sht.getHum() );                 //   Выводим текущую влажность воздуха, от 0 до 100%.
    Serial.print(" %.\r\n");                      //
    delay(500);                                   // * Задержка позволяет медленнее заполнять монитор последовательного порта.
```

```
}                                                   //
```

После загрузки данного примера, в мониторе последовательного порта будут появляться все параметры, которые может вернуть модуль: температура и влажность.

Чтение значений модуля при их изменении:

```
#include <Wire.h>                                   //   Подключаем библиотеку для работы с аппаратной шиной I2C.
#include <iarduino_I2C_SHT.h>                       //   Подключаем библиотеку для работы с датчиком температуры и влажности I2C
iarduino_I2C_SHT sht;                               //   Объявляем объект sht для работы с функциями и методами библиотеки iardu
                                                    //   Если при объявлении объекта указать адрес, например, sht(0xBB), то прим
void setup(){                                       //
    delay(500);                                     // * Ждём завершение переходных процессов связанных с подачей питания.
    Serial.begin(9600);                             //
    while(!Serial){;}                               // * Ждём завершения инициализации шины UART.
    sht.begin();                                    //   Инициируем работу с датчиком температуры и влажности.
    sht.setTemChange( 0.1 );                        //   Указываем фиксировать изменение температуры окружающей среды более чем
    sht.setHumChange( 1.0 );                        //   Указываем фиксировать изменение влажности воздуха более чем на 1%.
}                                                   //
                                                    //
void loop(){                                        //
    if( sht.getTemChanged() ){                      //   Если температура окружающей среды изменилась более чем на значение указ
        Serial.print("Температура = ");             //
        Serial.print( sht.getTem() );               //   Выводим текущую температуру окружающей среды, от -40 до +125°C.
        Serial.print(" °C.\r\n");                   //
    }                                               //
    if( sht.getHumChanged() ){                      //   Если влажность воздуха изменилась более чем на значение указанное в фун
        Serial.print("Влажность = ");               //
        Serial.print( sht.getHum() );               //   Выводим текущую влажность воздуха, от 0 до 100%.
        Serial.print(" %.\r\n");                    //
    }                                               //
```

```
}                                                    //
```

После загрузки данного примера, в мониторе последовательного порта будут появляться показания температуры и(или) влажности, если эти показания изменились.

Описание функций библиотеки:

В данном разделе описаны функции библиотеки iarduino_I2C_SHT для работы с Trema модулем - Датчик температуры и влажности, I2C-flash.

Данная библиотека может использовать как аппаратную, так и программную реализацию шины I2C. О том как выбрать тип шины I2C рассказано в статье Wiki - расширенные возможности библиотек iarduino для шины I2C.

Подключение библиотеки:

- Если адрес модуля известен (в примере используется адрес 0x09):

```
#include <iarduino_I2C_SHT.h>       // Подключаем библиотеку для работы с модулем.
iarduino_I2C_SHT sht(0x09);         // Создаём объект sht для работы с функциями и методами библиотеки iarduino_I2C_SHT, указы
```

- Если адрес модуля неизвестен (адрес будет найден автоматически):

```
#include <iarduino_I2C_SHT.h>       // Подключаем библиотеку для работы с модулем.
iarduino_I2C_SHT sht;               // Создаём объект sht для работы с функциями и методами библиотеки iarduino_I2C_SHT.
```

При создании объекта без указания адреса, на шине должен находиться только один модуль.

Функция begin();

- Назначение: Инициализация работы с модулем.
- Синтаксис: begin();

- Параметры: Нет.
- Возвращаемое значение: bool - результат инициализации (true или false).
- Примечание: По результату инициализации можно определить наличие модуля на шине.
- Пример:

```
if( sht.begin() ){ Serial.print( "Модуль найден и инициирован!" ); }
else             { Serial.print( "Модуль не найден на шине I2C" ); }
```

Функция reset();

- Назначение: Перезагрузка модуля.
- Синтаксис: reset();
- Параметры: Нет.
- Возвращаемое значение: bool - результат перезагрузки (true или false).
- Пример:

```
if( sht.reset() ){ Serial.print( "Модуль    перезагружен" ); }
else             { Serial.print( "Модуль не перезагружен" ); }
```

Функция changeAddress();

- Назначение: Смена адреса модуля на шине I2C.
- Синтаксис: changeAddress(АДРЕС);
- Параметры:
  - uint8_t АДРЕС - новый адрес модуля на шине I2C (целое число от 0x08 до 0x7E)
- Возвращаемое значение: bool - результат смены адреса (true или false).
- Примечание: Текущий адрес модуля можно узнать функцией getAddress().
- Пример:

```
if( sht.changeAddress(0x12) ){ Serial.print( "Адрес модуля изменён на 0x12" ); }
```

```
else                          { Serial.print( "Не удалось изменить адрес"    ); }
```

Функция getAddress();

- Назначение: Запрос текущего адреса модуля на шине I2C.
- Синтаксис: getAddress();
- Параметры: Нет.
- Возвращаемое значение: uint8_t АДРЕС - текущий адрес модуля на шине I2C (от 0x08 до 0x7E)
- Примечание: Функция может понадобиться если адрес модуля не указан при создании объекта, а обнаружен библиотекой.
- Пример:

```
Serial.print( "Адрес модуля на шине I2C = 0x" );
Serial.println( sht.getAddress(), HEX );
```

Функция getVersion();

- Назначение: Запрос версии прошивки модуля.
- Синтаксис: getVersion();
- Параметры: Нет
- Возвращаемое значение: uint8_t ВЕРСИЯ - номер версии прошивки от 0 до 255.
- Пример:

```
Serial.print( "Версия прошивки модуля " );
Serial.println( sht.getVersion(), HEX );
```

Функция getTem();

- Назначение: Запрос текущей температуры окружающей среды в °C.
- Синтаксис: getTem();
- Параметры: Нет.
- Возвращаемое значение: float - температура окружающей среды от -40,0 до +125.0 °C.

- Пример:

```
Serial.print( "Температура = " );
Serial.print( sht.getTem() );
Serial.println( "°C." );
```

Функция getHum();

- Назначение: Запрос текущей относительной влажности воздуха в процентах.
- Синтаксис: getHum();
- Параметры: Нет.
- Возвращаемое значение: float - относительная влажность воздуха от 0,0 до 100,0%.
- Примечание:
- Пример:

```
Serial.print( "Влажность = " );
Serial.print( sht.getHum() );
Serial.println( "%." );
```

Функция getTemChanged();

- Назначение: Запрос подтверждения изменения температуры.
- Синтаксис: getTemChanged();
- Параметр: Нет.
- Возвращаемое значение: bool - подтверждение изменения температуры (true или false).
- Примечание:
  - Функция возвращает положительный результат `true`, если с момента последнего положительного результата, температура изменилась на значение указанное функцией `setTemChange()`.

Функция setTemChange();

- Назначение: Установка порога чувствительности фиксации изменения температуры.
- Синтаксис: setTemChange(ПОРОГ);
- Параметры: float ПОРОГ - значение от 0,1 до 25,5 °C.
- Возвращаемое значение: bool - результат применения новых настроек (true или false).
- Примечание:
  - Данная функция определяет, как сильно должна измениться температура, чтоб функция `getTemChanged()` вернула `true` .
- Пример:

```
sht.setTemChange(2.0);                  // Указываем реагировать на изменения в 2°C.
void loop() {                           //
    if( sht.getTemChanged() ){          // Если температура изменилась, то ...
        Serial.print( "Температура = "); //
        Serial.print( sht.getTem()    ); // Выводим текущую температуру.
        Serial.print( " °C.\r\n"      ); //
    }                                   //
}                                       //
```

Функция getHumChanged();

- Назначение: Запрос подтверждения изменения влажности.
- Синтаксис: getHumChanged();
- Параметр: Нет.
- Возвращаемое значение: bool - подтверждение изменения влажности (true или false).
- Примечание:
  - Функция возвращает положительный результат `true` , если с момента последнего положительного результата, влажность изменилась на значение указанное функцией `setHumChange()` .

Функция setHumChange();

- Назначение: Установка порога чувствительности фиксации изменения влажности.
- Синтаксис: setHumChange(ПОРОГ);

- Параметры: float ПОРОГ - значение от 0,1 до 25,5 %.
- Возвращаемое значение: bool - результат применения новых настроек (true или false).
- Примечание:
  - Данная функция определяет, как сильно должна измениться влажность, чтоб функция `getHumChanged()` вернула `true`.
- Пример:

```
sht.setHumChange(5.0);                  // Указываем реагировать на изменения в 5%.
void loop() {                           //
    if( sht.getHumChanged() ){          // Если влажность изменилась, то ...
        Serial.print( "Влажность = " ); //
        Serial.print( sht.getHum()   ); // Выводим текущую влажность.
        Serial.print( " %.\r\n"      ); //
    }                                   //
}                                       //
```

Функция setPeriod();

- Назначение: Установка периода обновления данных модуля.
- Синтаксис: setPeriod(ВРЕМЯ);
- Параметр:
  - uint8_t ВРЕМЯ - значение от 0,2 до 25,5 секунд.
- Возвращаемое значение: bool - результат сохранения настроек (true или false).
- Примечание:
  - Период обновления данных модуля определяет, как часто модуль должен обновлять данные своих датчиков. Значение по умолчанию 0,5 секунд.
  - Увеличение времени между обновлениями данных может сократить энергопотребление модуля.
  - Не рекомендуется устанавливать период обновления данных ниже 0,5 секунд.
- Пример:

```
sht.setPeriod( 1.0 );
```

Ссылки:

- Библиотека iarduino_I2C_SHT.
- Расширенные возможности библиотек iarduino для шины I2C.
- Wiki - Установка библиотек в Arduino IDE.