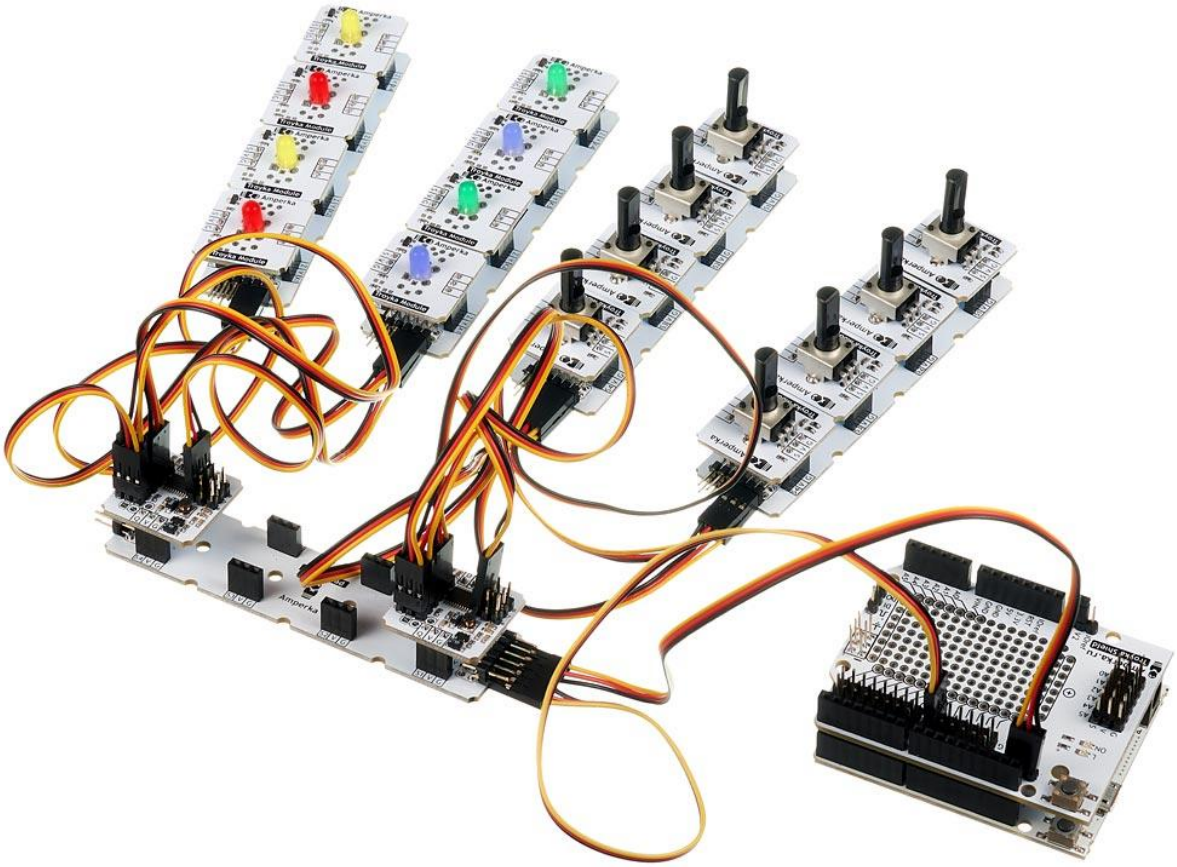
Расширитель GPIO-портов (Troyka-модуль)

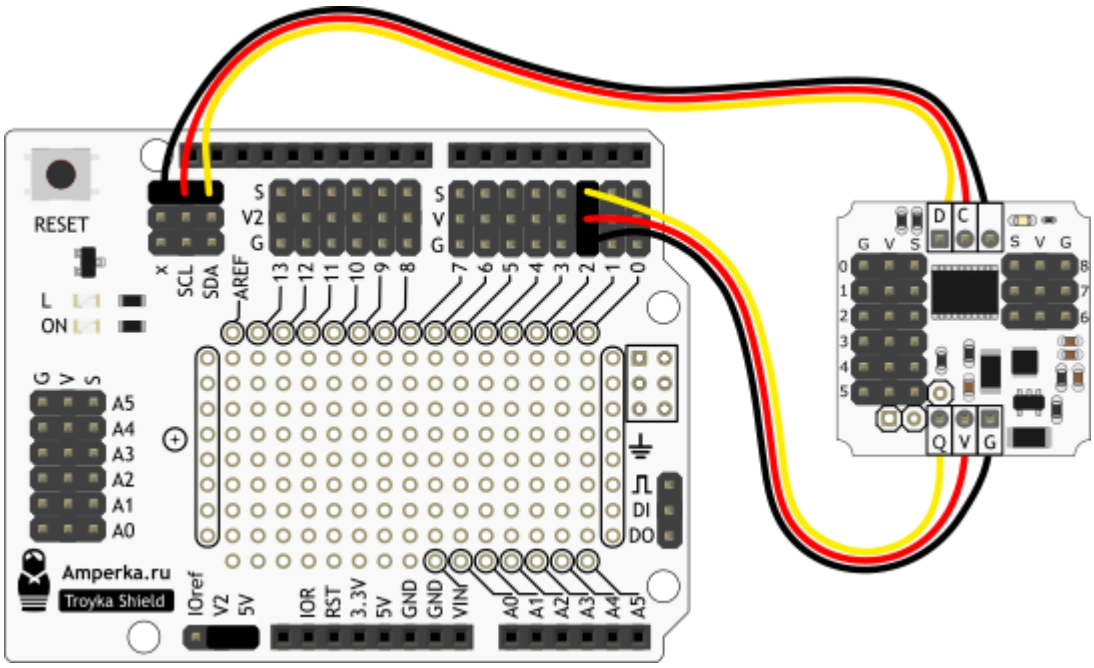


GPIO Expander (Troyka-модуль) решает проблему внезапно кончившихся пинов ввода/вывода.

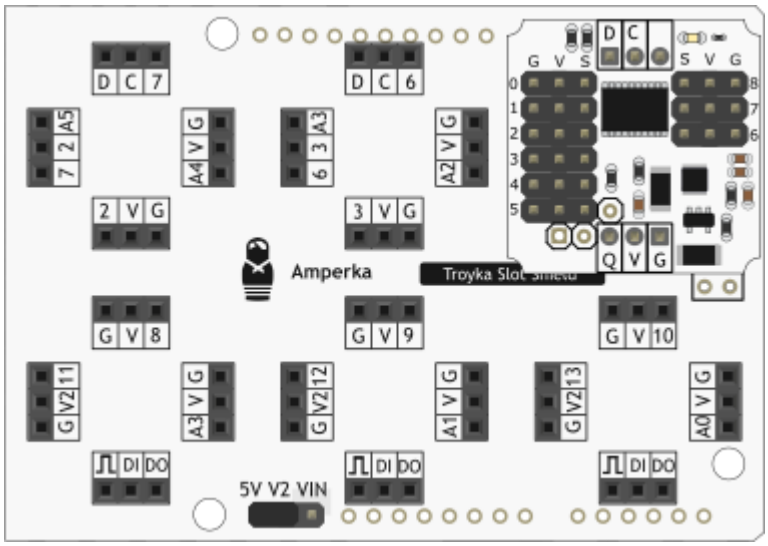
Модуль получает команды от управляющей платы по протоколу I²C/TWI и позволяет использовать любой из своих портов как цифровой или аналоговый вход или выход.

Подключение и настройка

GPIO расширитель портов общается с управляющей платой по протоколу I²C/TWI. При подключении к Arduino или Iskra JS удобно использовать Troyka Shield.



С [Troyka Slot Shield] можно сэкономить два трёхпроводных шлейфа.



Теперь к устройству можно подключить до девяти [Troyka-модулей].

Примеры работы

Рассмотрим несколько примеров работы модуля с управляющими платами Arduino. Работа с Iskra JS описана на странице библиотеки [@gpio-expander].

Маячок

В силу светодиодной робототехники мигнём светодиодом через GPIO Expander (Troyka-модуль).

Подключите GPIO расширитель к управляющей плате через Troyka Shield или Troyka Slot Shield.

Подключите светодиод «Пиранья» (Troyka-модуль) через стандартный 3-проводной шлейф к `8` пину GPIO Expander.

Схема для Troyka Shield

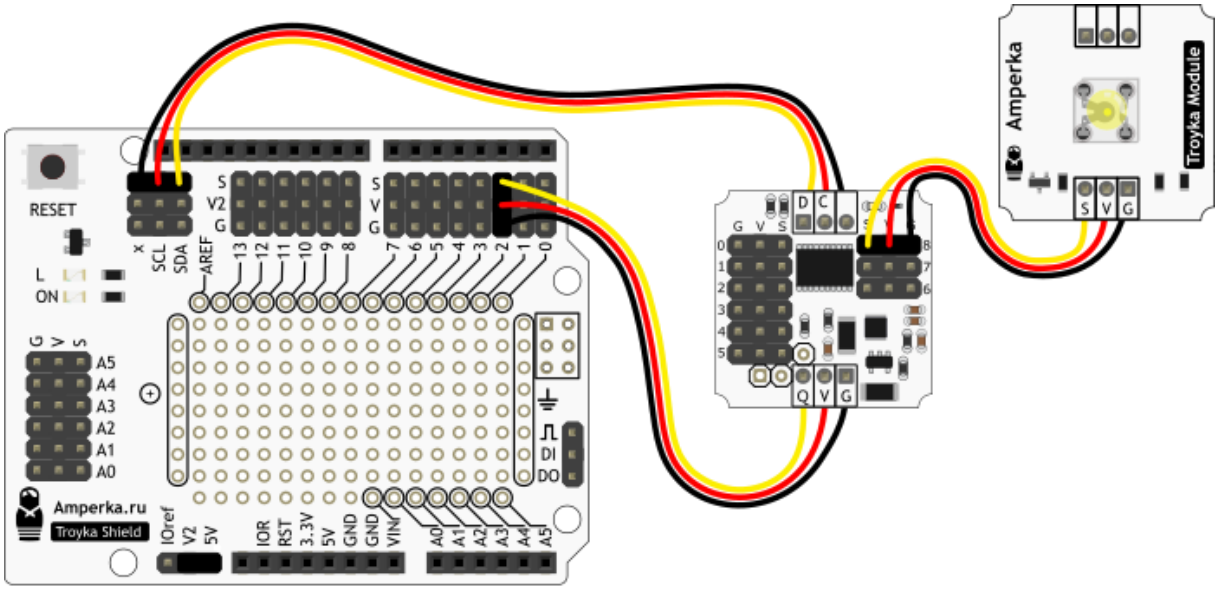
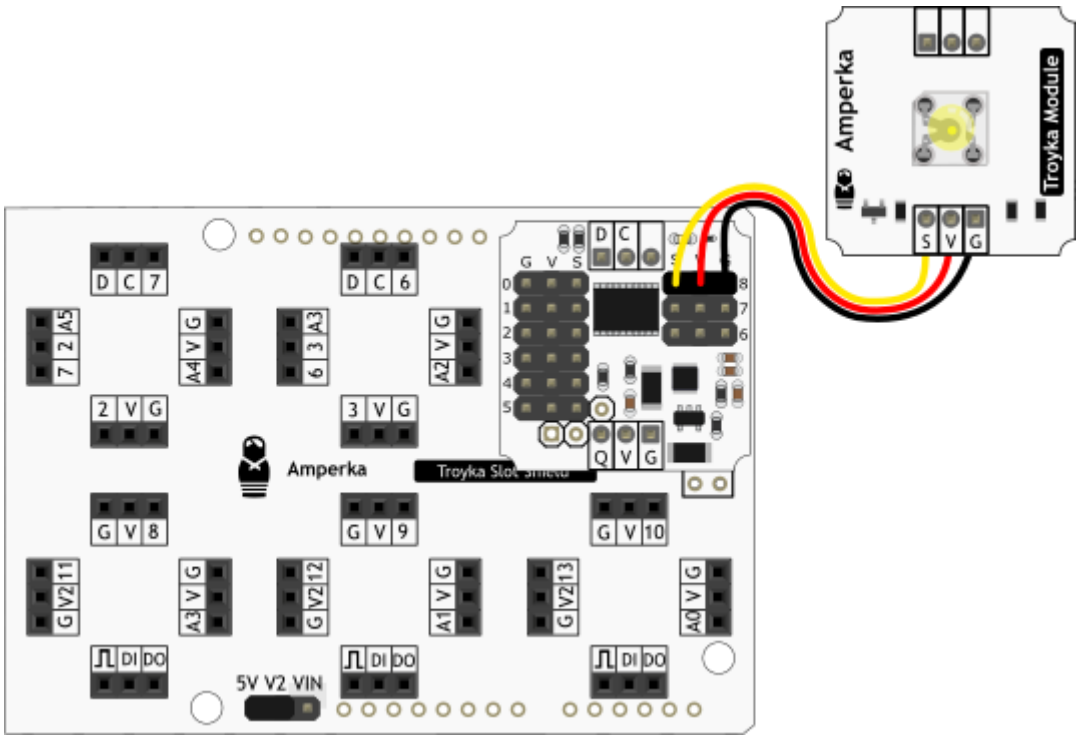


Схема для Troyka Slot Shield



Код для Arduino

Прошейте управляющую плату следующим скетчем.

troyka-gpio-expander_blink.ino

```
// библиотека для работы I²C
#include <Wire.h>
// библиотека для работы с модулем GPIO Expander (I²C IO)
#include <GpioExpander.h>
// создаём объект adio класса GpioExpander по адресу 42
GpioExpander adio(42);

void setup()
{
  // включаем I²C
  Wire.begin();
  // настраиваем пин 8 на модуле GPIO Expander в режим выхода
  adio.pinMode(8, OUTPUT);
}

void loop()
{
  // подаём на пин 8 модуля GPIO Expander «высокий сигнал»
  adio.digitalWrite(8, HIGH);
  // ждём 1 секунду
  delay(1000);
  // подаём на пин 8 модуля GPIO Expander «низкий сигнал»
  adio.digitalWrite(8, LOW);
  // ждём 1 секунду
  delay(1000);
}
```

Светодиод будет загораться и гаснуть раз в секунду.

Кнопочный выключатель

Добавим к предыдущему эксперименту кнопку и подключим её к расширителю портов стандартным трёхпроводным шлейфом к `1` пину.

Схема для Troyka Shield

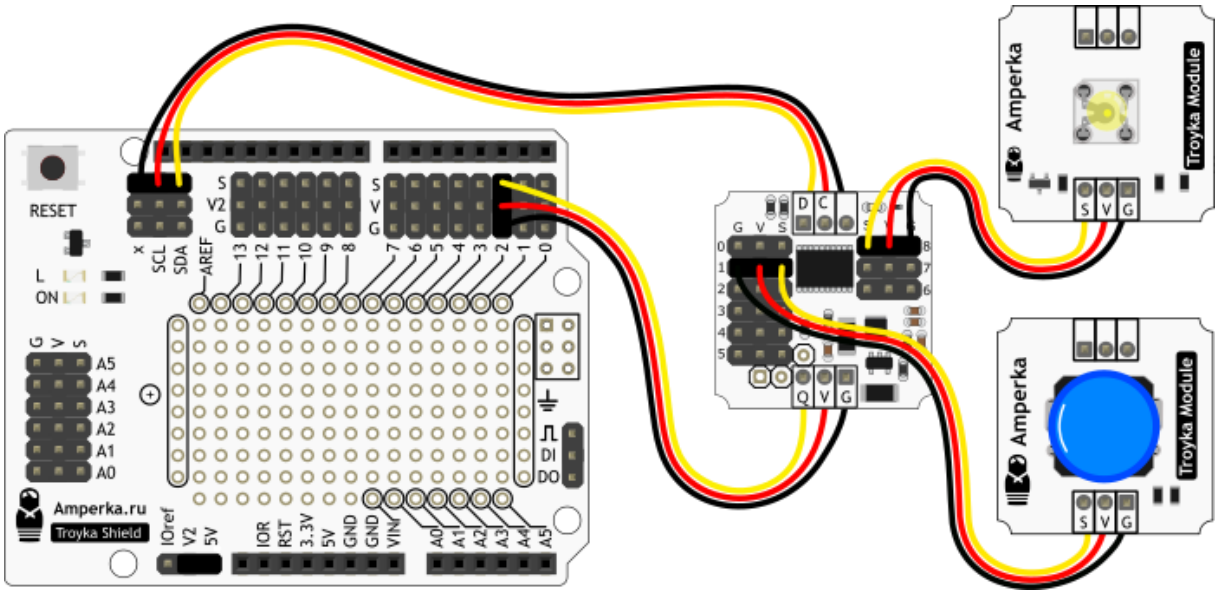
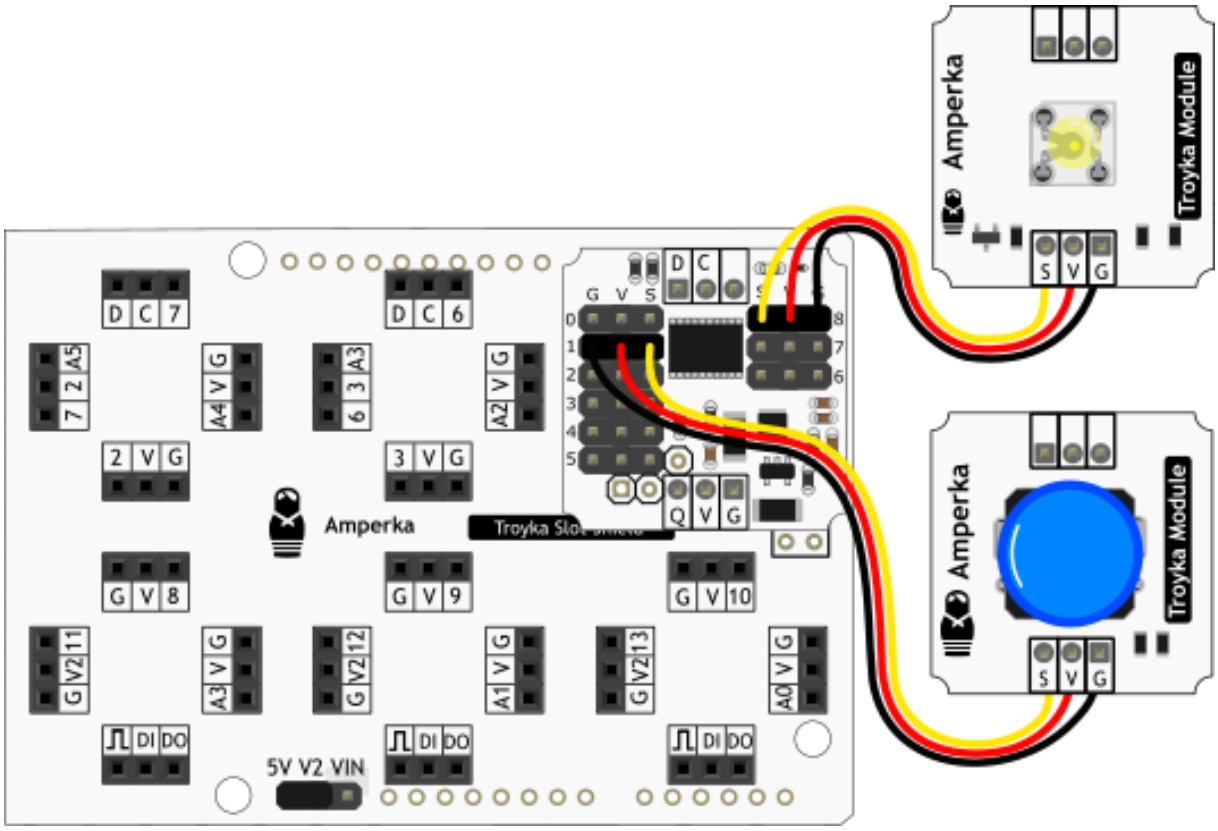


Схема для Troyka Slot Shield



Код для Arduino

Прошейте Arduino следующим скетчем.

troyka-gpio-expander_button_switch.ino

```
// библиотека для работы I²C
#include <Wire.h>
// библиотека для работы с модулем GPIO Expander (I²C IO)
#include <GpioExpander.h>
// создаём объект adio класса GpioExpander по адресу 42
GpioExpander adio(42);

void setup()
{
  // включаем I²C
  Wire.begin();
  // настраиваем пин 8 на модуле GPIO Expander в режим выхода
  adio.pinMode(8, OUTPUT);
  // настраиваем пин 8 на модуле GPIO Expander в режим входа
  adio.pinMode(1, INPUT);
}

void loop()
{
  // считываем состояние кнопки с 1 пина на модуле GPIO Expander
  if (!adio.digitalRead(1)) {
    // подаём на пин 8 модуля GPIO Expander «высокий сигнал»
    adio.digitalWrite(8, HIGH);
  } else {
    // подаём на пин 8 модуля GPIO Expander «низкий сигнал»
    adio.digitalWrite(8, LOW);
  }
```

```
}
```

Светодиод будет загораться при нажатии на кнопку.

Светильник с регулируемой яркостью

GPIO расширитель умеет считывать аналоговый сигнал с датчиков и выводить ШИМ. Усложним наш проект, заменив кнопку на потенциометр.

Схема для Troyka Shield

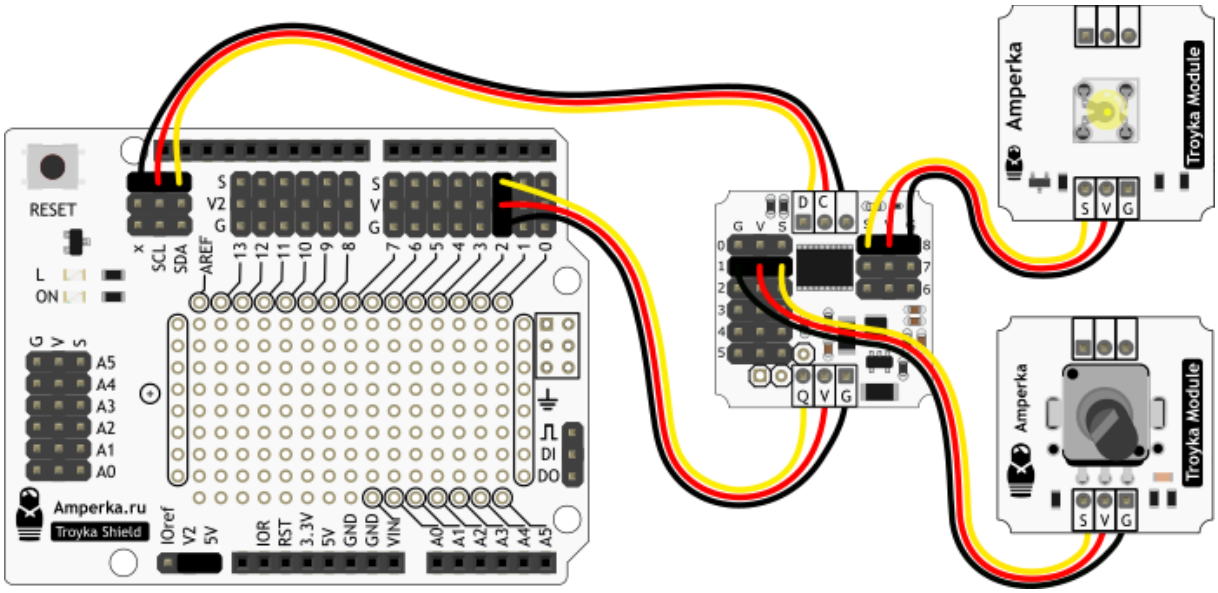
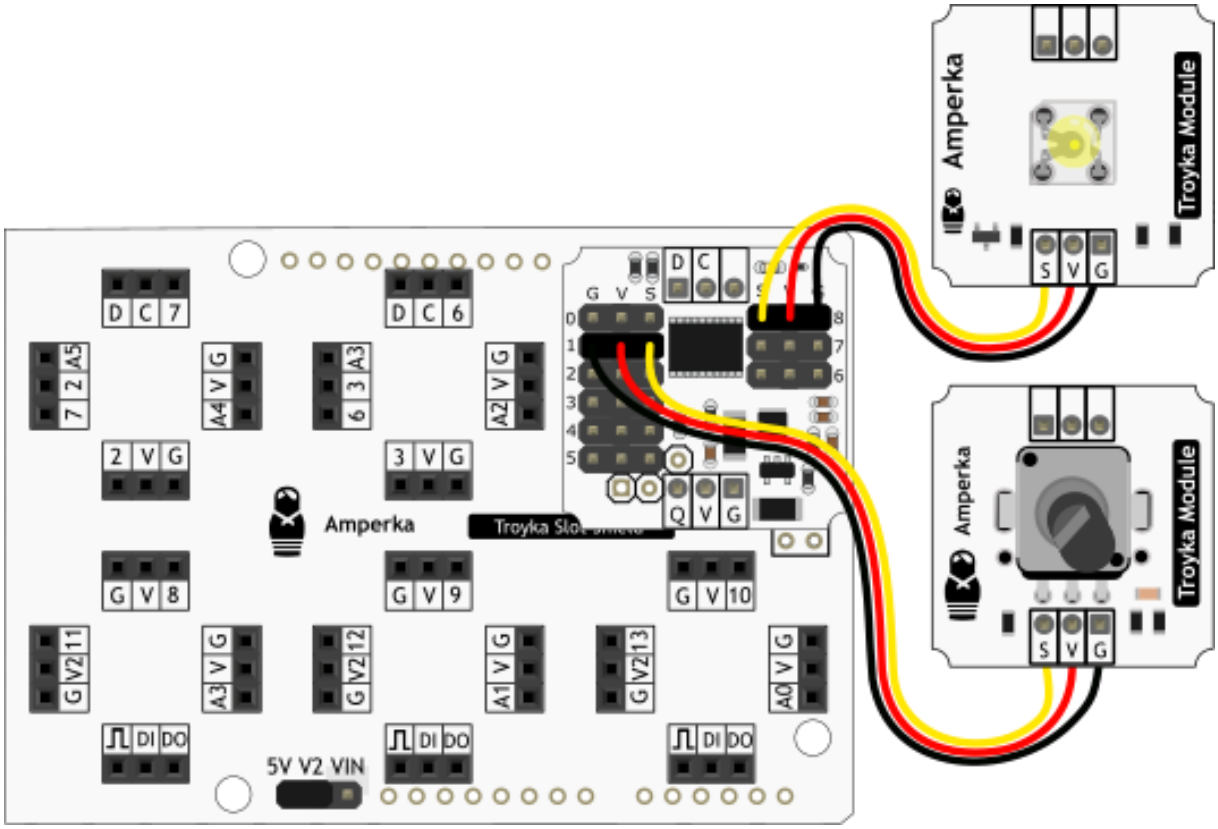


Схема для Troyka Slot Shield



Код для Arduino

Прошейте Arduino следующим скетчем.

troyka-gpio-expander_brightness_led.ino

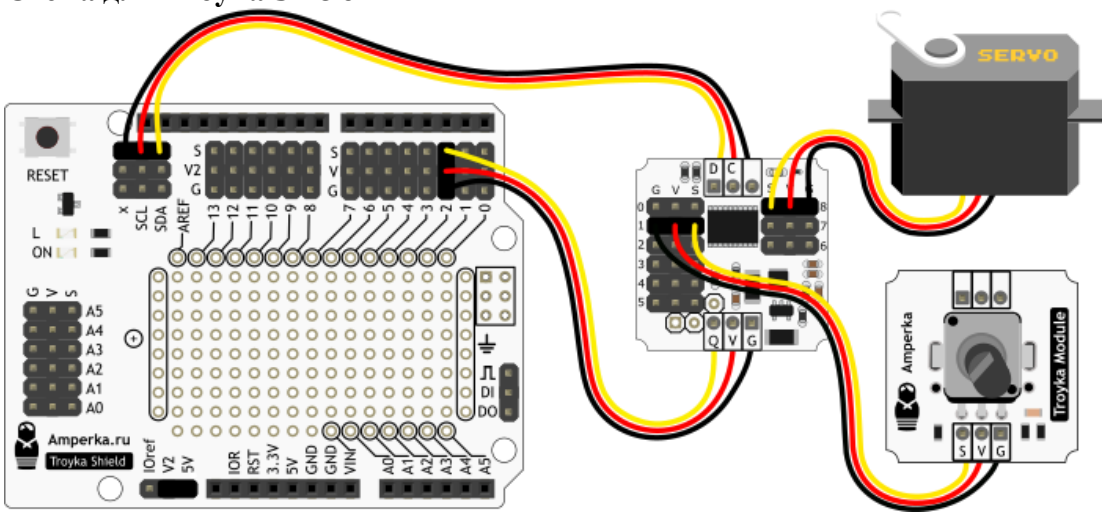
```
// библиотека для работы I²C
#include <Wire.h>
// библиотека для работы с модулем GPIO Expander (I²C IO)
#include <GpioExpander.h>
// создаём объект adio класса GpioExpander по адресу 42
GpioExpander adio(42);

void setup()
{
  // включаем I²C
  Wire.begin();
  // настраиваем пин 8 на модуле GPIO Expander в режим выхода
  adio.pinMode(8, OUTPUT);
  // настраиваем пин 8 на модуле GPIO Expander в режим входа
  adio.pinMode(1, INPUT);
}

void loop()
{
  // считываем аналоговое значение с 1 пина на модуле GPIO Expander
  int sensorValue = adio.analogRead(1);
  // преобразуем диапазон значений с потенциометра (0-4096)
  // в диапазон значений для ШИМ-сигнала (0-255)
  int brigtness = map(sensorValue, 0, 4096, 255, 0);
  // устанавливаем светодиоду яркость
  // в соответствии от положегния ручки потенциометра
  adio.analogWrite(8, brigtness);
  // ждём 100 мс
  delay(100);
}
```

Светодиод будет изменять яркость в зависимости от положения потенциометра.

Управляем сервоприводом

GPIO расширитель умеет управлять сервоприводами. Заменим светодиод на сервопривод и будем управлять им с помощью потенциометра.

Схема для Troyka Shield



Код для Arduino

troyka-gpio-expander_servo_pot.ino

```
// библиотека для работы I²C
#include <Wire.h>
// библиотека для работы с модулем GPIO Expander (I²C IO)
#include <GpioExpander.h>
// создаём объект adio класса GpioExpander по адресу 42
GpioExpander adio(42);
void setup()
{
  // включаем I²C
  Wire.begin();
  // настраиваем пин 8 на модуле GPIO Expander в режим выхода
  adio.pinMode(8, OUTPUT);
  // настраиваем пин 8 на модуле GPIO Expander2 в режим выхода
  adio.pinMode(1, INPUT);
  // устанавливаем частоту шим 50 Гц
  adio.pwmFreq(50);
}

void loop()
{
  // сохраним в переменную показания с потенциометра
  int pot = adio.analogRead(1);
  // сделаем пересчет из показаний потенциометра в крайние значения
работы сервоопривода
  int sweep = map(pot, 0, 4095, 1750, 7900);
  // подаём аналоговый 16-ти битный сигнал на пин сервопривода
  adio.analogWrite_16(8, sweep);
  // ждём 50 секунд
  delay(50);
}
```

Двойной маячок

По умолчанию модуль имеет адрес `42`, который можно изменить прямо в скетче вашей программы. Это позволяет подключить к плате до 126 устройств расширителей портов.

Смена адреса устройства

Для начала необходимо подключить только один GPIO Expander, адрес которого хотим изменить.

Схема для Troyka Shield

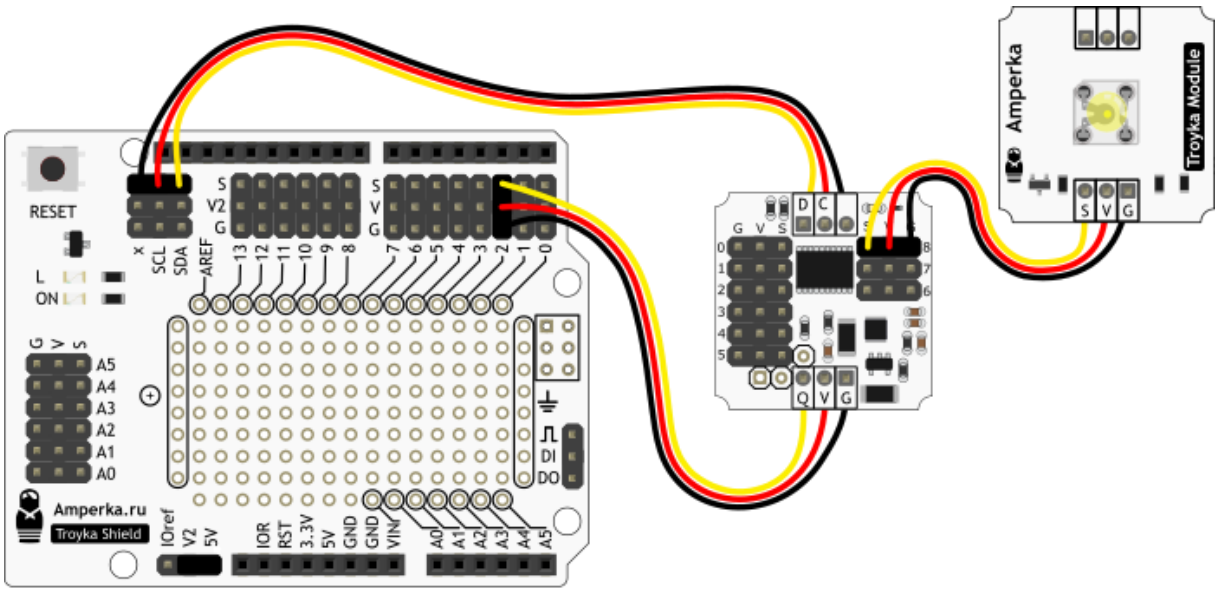
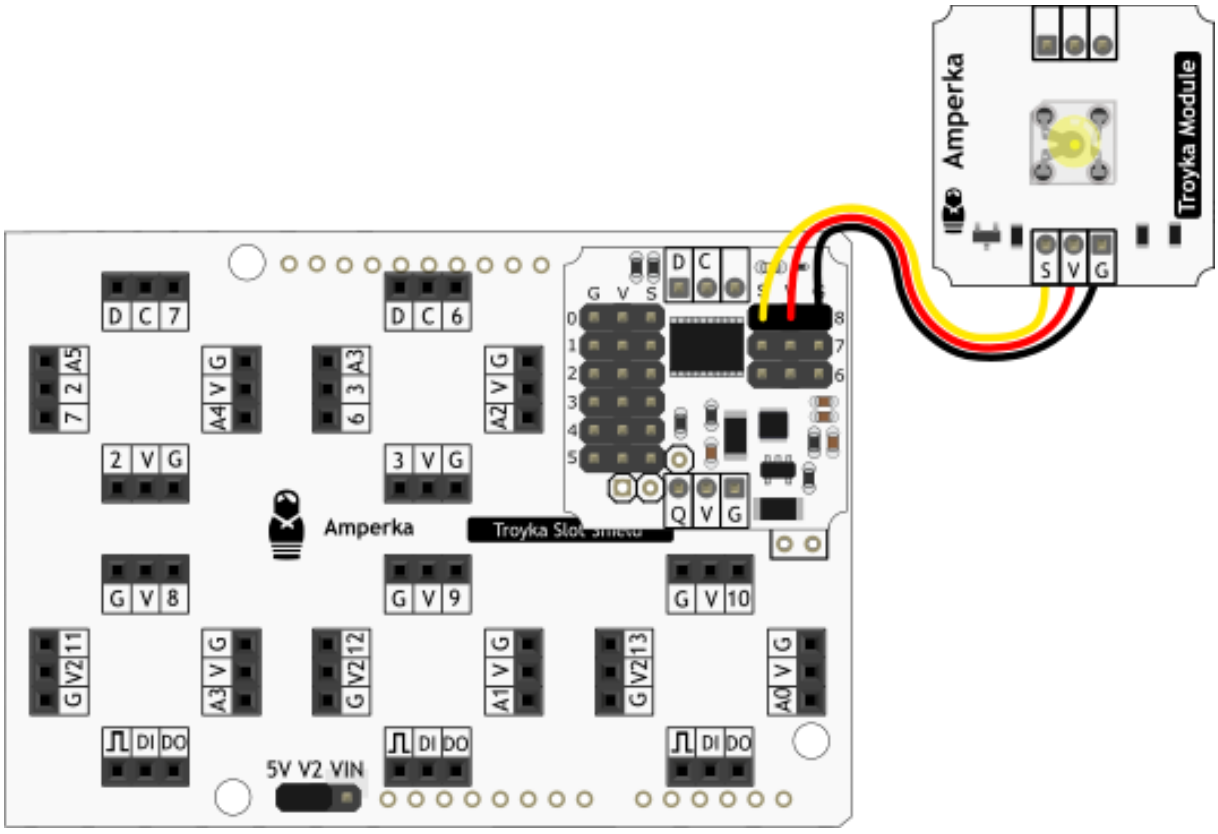


Схема для Troyka Slot Shield



Код для Arduino

Прошейте управляющую плату следующим скетчем.

troyka-gpio-expander_change_address.ino

```
// библиотека для работы I²C
#include <Wire.h>
// библиотека для работы с модулем GPIO Expander (I²C IO)
#include <GpioExpander.h>
// создаём объект adio класса GpioExpander по адресу 42
GpioExpander adio(42);
```

```
void setup()
{
  // включаем I²C
  Wire.begin();
  // меняем адрес модуля на «43»
  adio.changeAddr(43);
  delay(100);
  // сохраняем адрес во Flash-памяти контроллера на модуле GPIO
Expander
  adio.saveAddr();
}

void loop()
{
}
```

Адрес успешно сменён. Изменим пример с маячком, добавив второй модуль и подключим к нему второй [светодиод «Пиранья»](#).

Схема для Troyka Shield

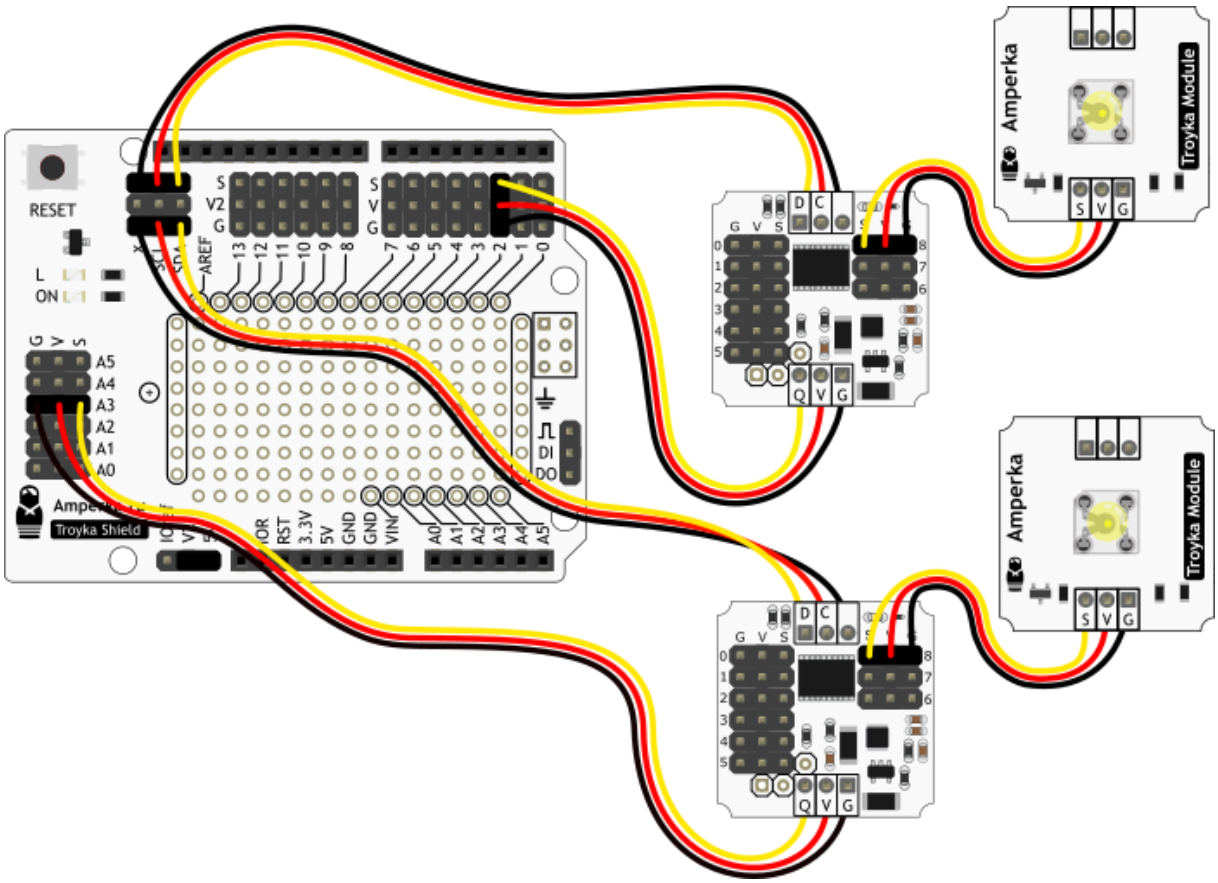
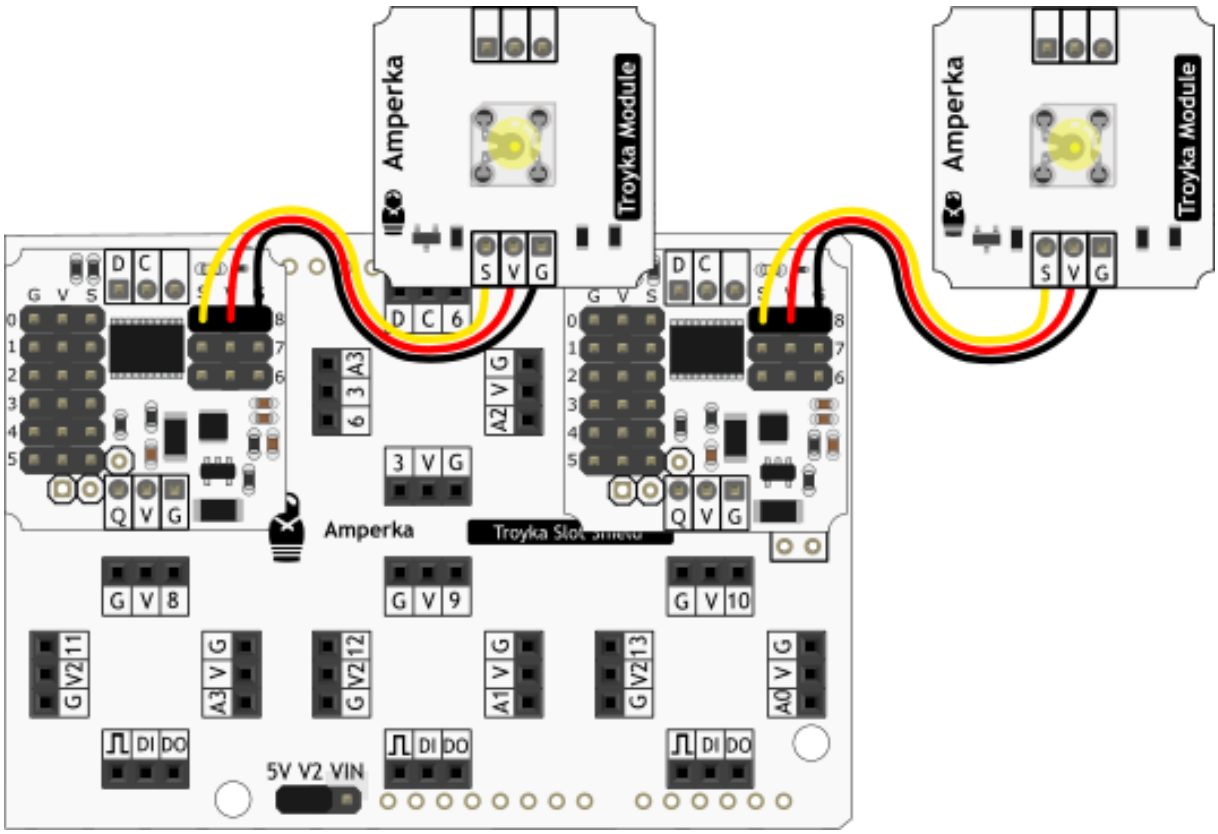


Схема для Troyka Slot Shield



Код для Arduino

Прошейте управляющую плату следующим скетчем.

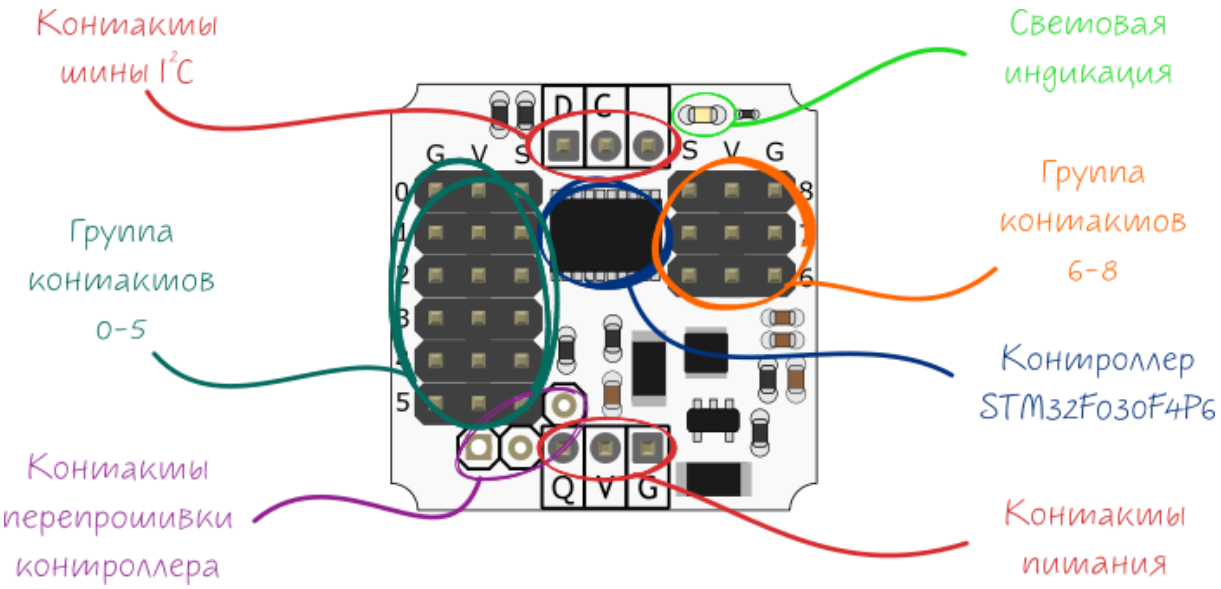
troyka-gpio-expander_double_leds.ino

```
// библиотека для работы I²C
#include <Wire.h>
// библиотека для работы с модулем GPIO Expander (I²C IO)
#include <GpioExpander.h>
// создаём объект adio класса GpioExpander по адресу 42
GpioExpander adio(42);
// создаём объект adio класса GpioExpander по адресу 43
GpioExpander adio2(43);
void setup()
{
  // включаем I²C
  Wire.begin();
  // настраиваем пин 8 на модуле GPIO Expander в режим выхода
  adio.pinMode(8, OUTPUT);
  // настраиваем пин 8 на модуле GPIO Expander2 в режим выхода
  adio2.pinMode(8, OUTPUT);
}

void loop()
{
  // подаём на пин 8 модуля GPIO Expander «высокий сигнал»
  adio.digitalWrite(8, HIGH);
  // ждём 1 секунду
  delay(1000);
  // подаём на пин 8 модуля GPIO Expander «низкий сигнал»
  adio.digitalWrite(8, LOW);
  // ждём 1 секунду
  delay(1000);
  // подаём на пин 8 модуля GPIO Expander2 «высокий сигнал»
```

```
  adio2.digitalWrite(8, HIGH);
  // ждём 1 секунду
  delay(1000);
  // подаём на пин 8 модуля GPIO Expander2 «низкий сигнал»
  adio2.digitalWrite(8, LOW);
  // ждём 1 секунду
  delay(1000);
}
```

Светодиоды подключенные к разным расширителям будут мигать по очереди.

Элементы платы



Микроконтроллер STM32F030F4P6

Мозгом модуля является мощный 32-разрядный микроконтроллер фирмы STMicroelectronics — STM32F030F4P6 с вычислительном ядром ARM Cortex® M0.

Группа GPIO 0-8

GPIO контакты модуля расчитаны на напряжение 3.3 В. Каждый GPIO может быть независимо настроен как цифровой вход или выход и имеет возможность подтяжки как к + так и к -. Для работы с аналоговыми модулями любой порт GPIO может работать в режиме АЦП или генерировать PWM сигнал.

Troyka контакты

Группа питания и перепрошивки

- Сигнальный пин (Q) — используется для перепрошивки контроллера STM32. При обычной работе с модулем не используется.
- Питание (V) — соедините с питанием микроконтроллера
- Земля (G) — соедините с землёй микроконтроллера

Шина I2C

- Сигнальный (D) — подключите к пину `SDA`микроконтроллера.
- Сигнальный (C) — подключите к пину `SCL`микроконтроллера.

Для подключения нескольких модулей предусмотрена возможность смены адресов каждого модуля с помощью функции adio.changeAddr(); параметром которой является новый адрес модуля. По умолчанию на адрес модуля имеет значение 42.

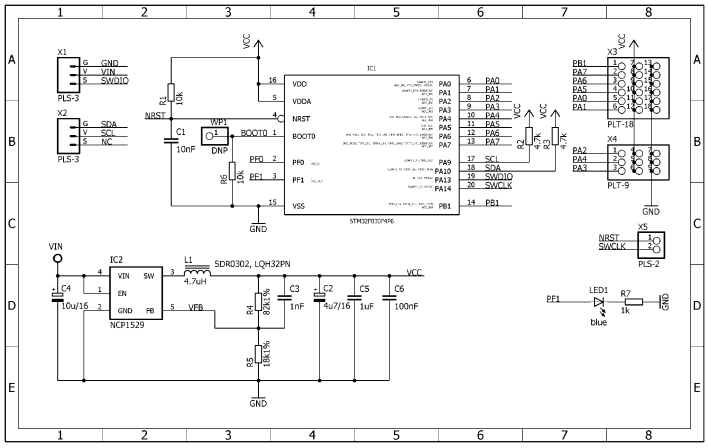
Светодиодная индикация

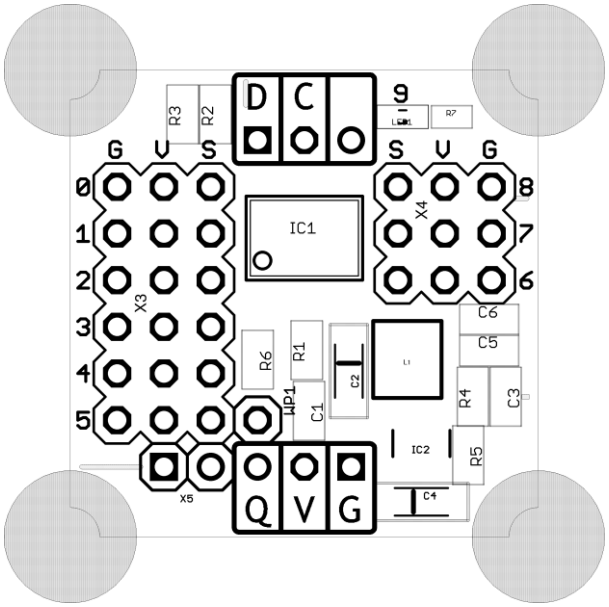
Во время работы светодиод индицирует обмен данными между модулем и управляющей платой.

Контакты перепрошивки контроллера

Используются для программирования контроллера STM32 через Serial wire debug (SWD).

Принципиальная и монтажная схемы





Характеристики

- Микроконтроллер: STM32F030F4P6

- Интерфейс: I²C (адрес по умолчанию: 0x2A)
- Напряжение питания модуля: 3,3–5 В
- Портов ввода-вывода общего назначения: 9
- Напряжение логических уровней: 3,3 В
- Портов с поддержкой ШИМ: 9
- Портов с АЦП: 9
- Габариты: 25,4×25,4 мм

Ресурсы

- Векторное изображение GPIO Expander
- Библиотека для Arduino
- Библиотека для Python (RPi 3)
- Описание библиотеки для Iskra JS