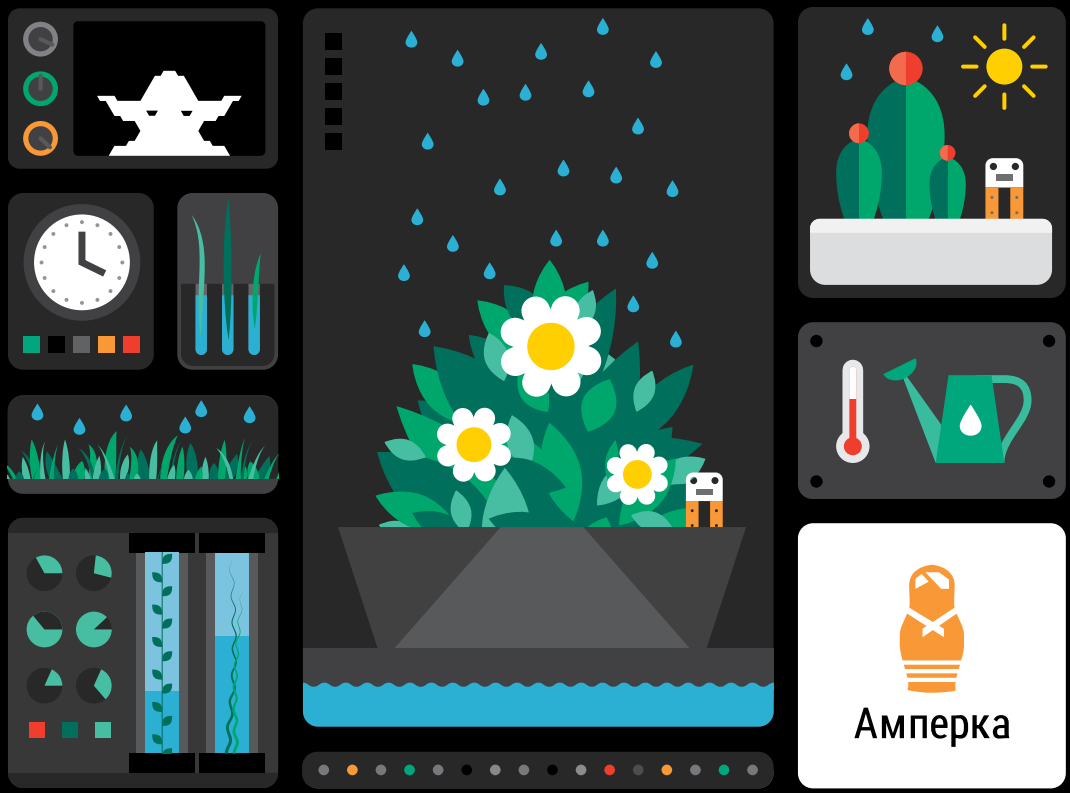
АВТОПОЛИВ



Амперка

НЕ ДАЙ РАСТЕНИЯМ ЗАСОХНУТЬ

WATER.AMPERKA.RU

СОДЕРЖАНИЕ

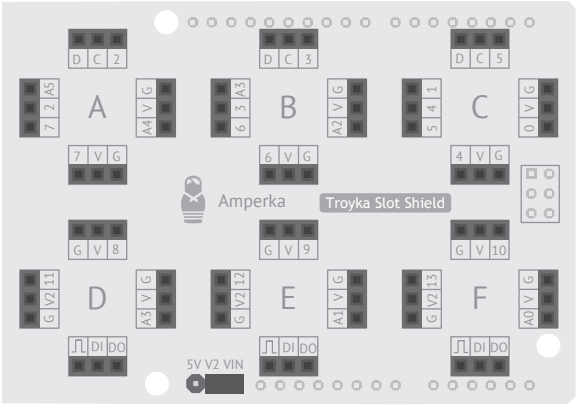
В эпизоде этом бороться будем против тёмной стороны — лени человеческой. Чтобы мир в Галактике сохранить и жизнь поддерживать. Водой управлять научимся и от хищников злых растения защитим.

Приготовь любое растение в горшке из твоей домашней коллекции. Хорошо, если оно очень любит воду. Найди пару ёмкостей для экспериментов с поливом. Подойдут кастрюля и большой стакан.

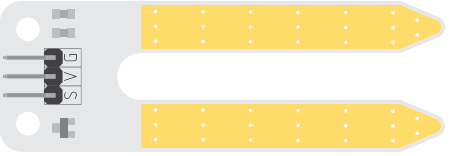
Приступай к первым проектам. Рекомендуем проходить их последовательно!


iskra

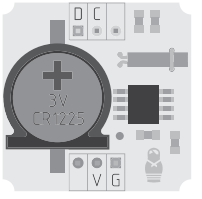
ЧТО В НАБОРЕ



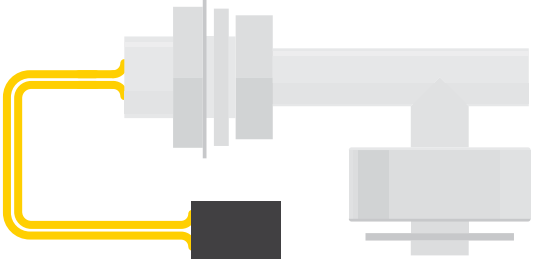
Troyka Slot Shield
Позволяет обойтись без лишних проводов



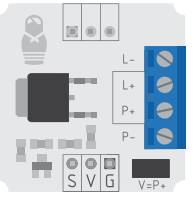
Датчик влажности почвы
Собственной персоной



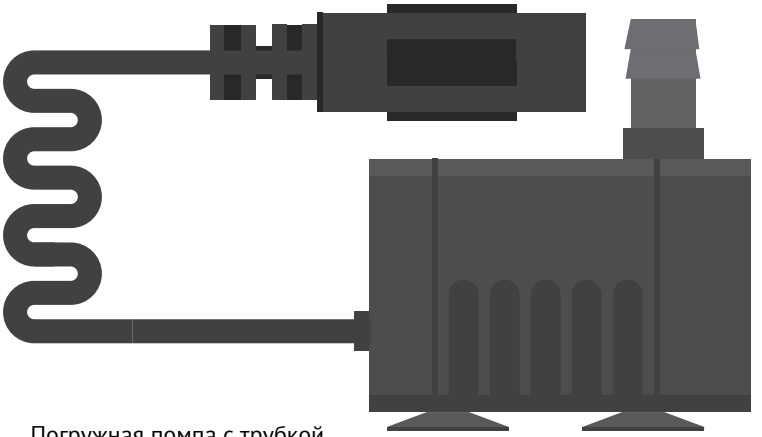
Часы реального времени
Отсчитывают мировое время



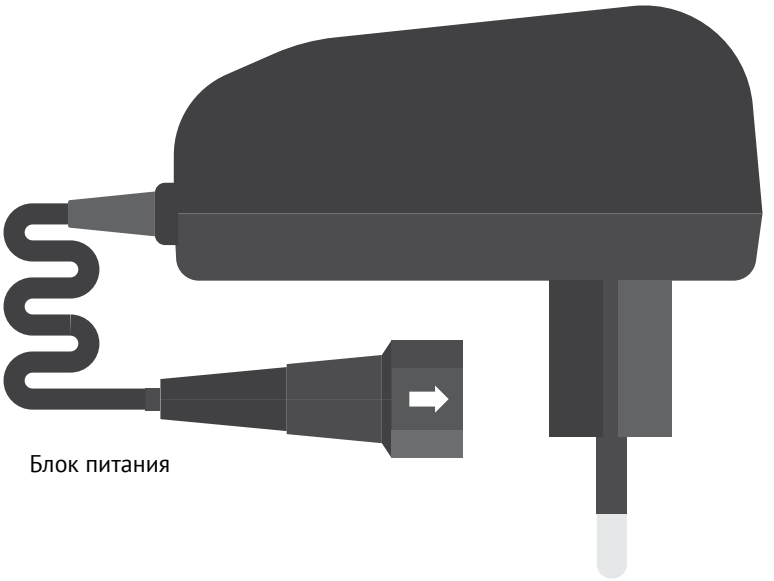
Датчик уровня воды
Замыкает контакты при поднятом поплавке



Силовой ключ
Управляет мощной нагрузкой

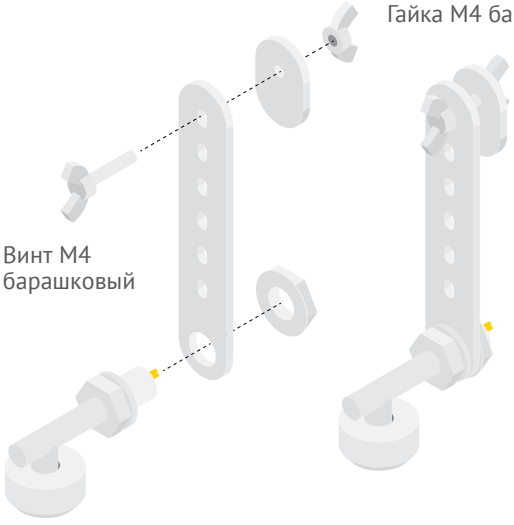


Погружная помпа с трубкой
Перекачивает воду из ёмкости,
в которой находится

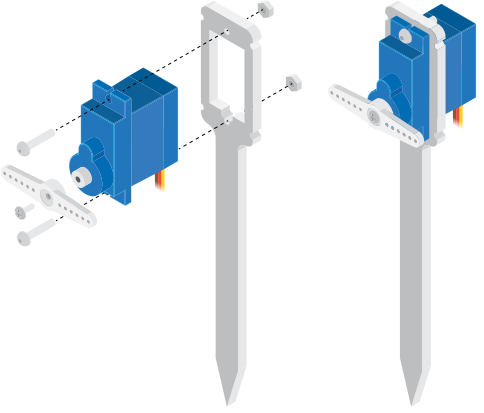


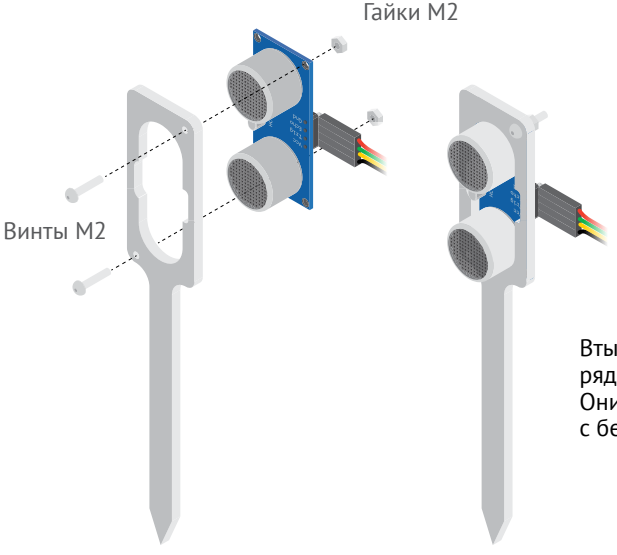
Блок питания

Колышки
для дальномера
и сервопривода



Собери колышки
и кронштейн. Они потребуются
в экспериментах. Колышки
втыкай в землю рядом
с растением. Осторожно,
не повреди корни!

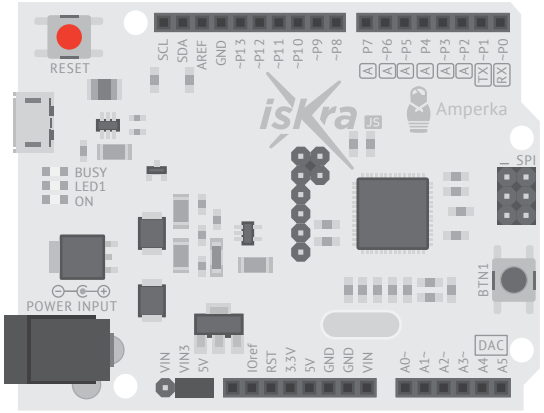




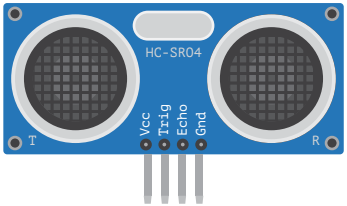
Втыкай колышки в землю
рядом со своим растением.
Они отлично сочетаются
с белыми Troyka-модулями.

ЕЩЁ ПОНАДОБИТСЯ

Для проектов тебе потребуются платы и модули из базового набора «Йодо».



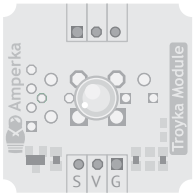
Iskra JS
Мозг твоего устройства



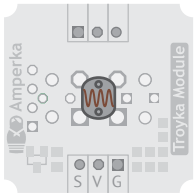
Ультразвуковой дальномер
Измеряет расстояние

Кабель
micro-USB

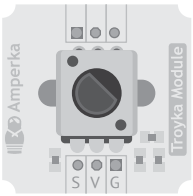
Сервопривод
Поворачивается
к заданному углу



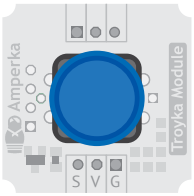
Светодиод
Светит и мигает



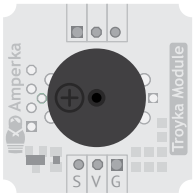
Датчик освещённости
Измеряет яркость
света



Потенциометр
Сообщает
о повороте ручки



Кнопка
Сообщает о нажатии



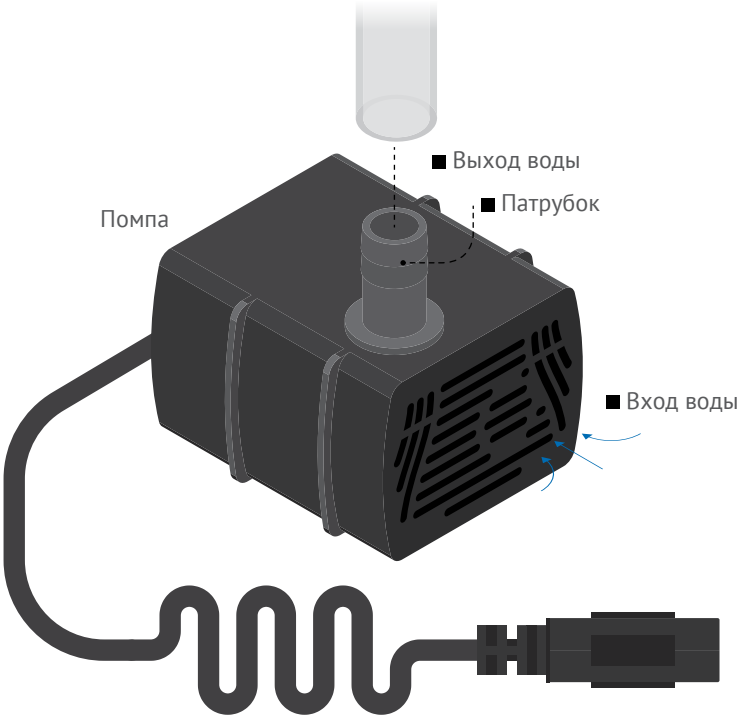
Зуммер
Пищит, издаёт
звуки

№ 1

ФОНТАН

Воду перемещать силой электроники научимся. Домашний фонтан из помпы сделаем.

1 Подсоедини шланг к патрубку помпы.

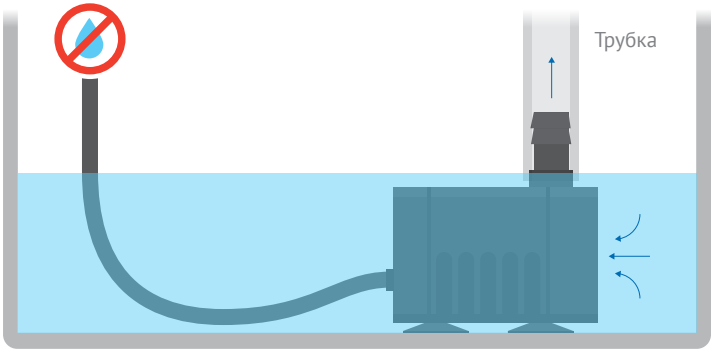


ВНИМАНИЕ!
В экспериментах будет много работы с водой. Нужно быть очень внимательным, чтобы не залить электронику.
На всякий случай подготовь заранее тряпку или салфетки.

Кабель питания
Напряжение 5...12 вольт

2 Возьми на кухне кастрюлю побольше. Прикрепи помпу ко дну.

3 Набери воды так, чтобы уровень был едва выше помпы.

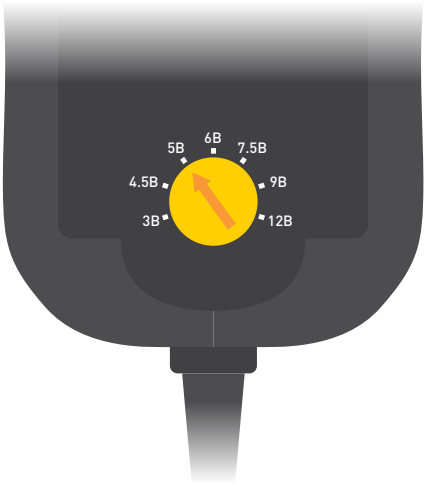


Ёмкость с водой

Помпа

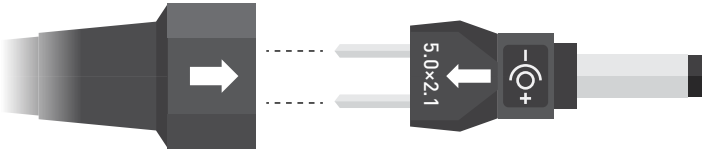
ОСТОРОЖНЕЕ!
Следи, чтобы на разъём питания не попала вода.

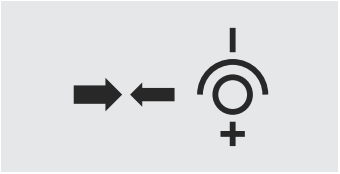
4 Возьми блок питания. Переведи переключатель напряжения в положение 5 вольт.



Воспользуйся ключом из комплекта, чтобы изменить положение переключателя.

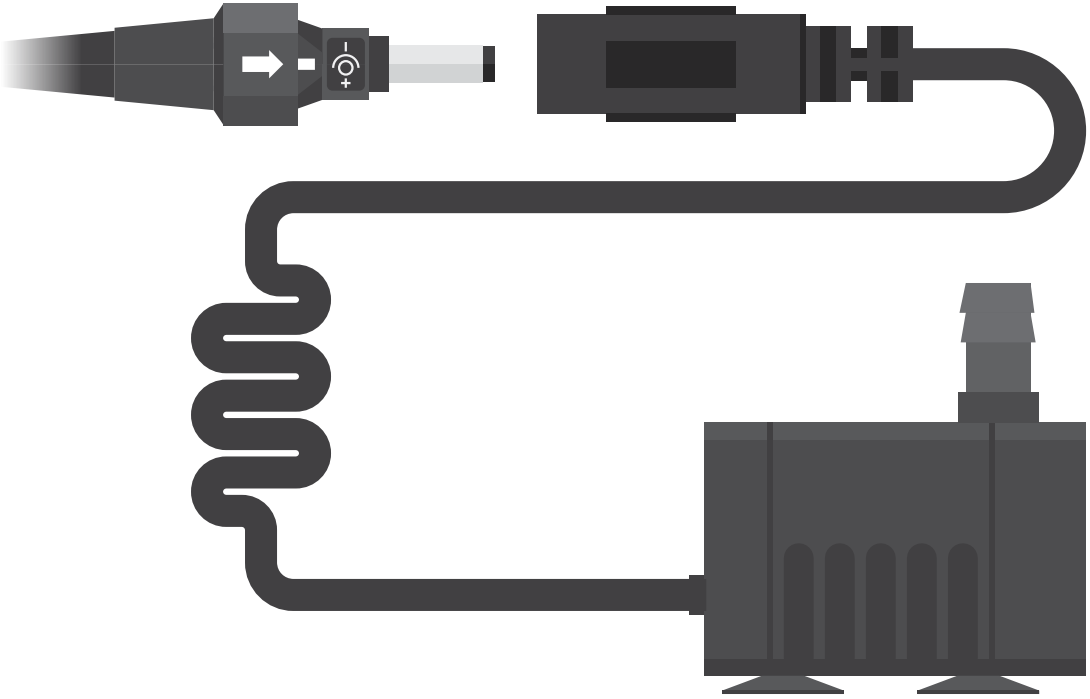
5 Возьми переходник с маркировкой 5.0×2.1. Вставь строго как на рисунке. Полярность важна!





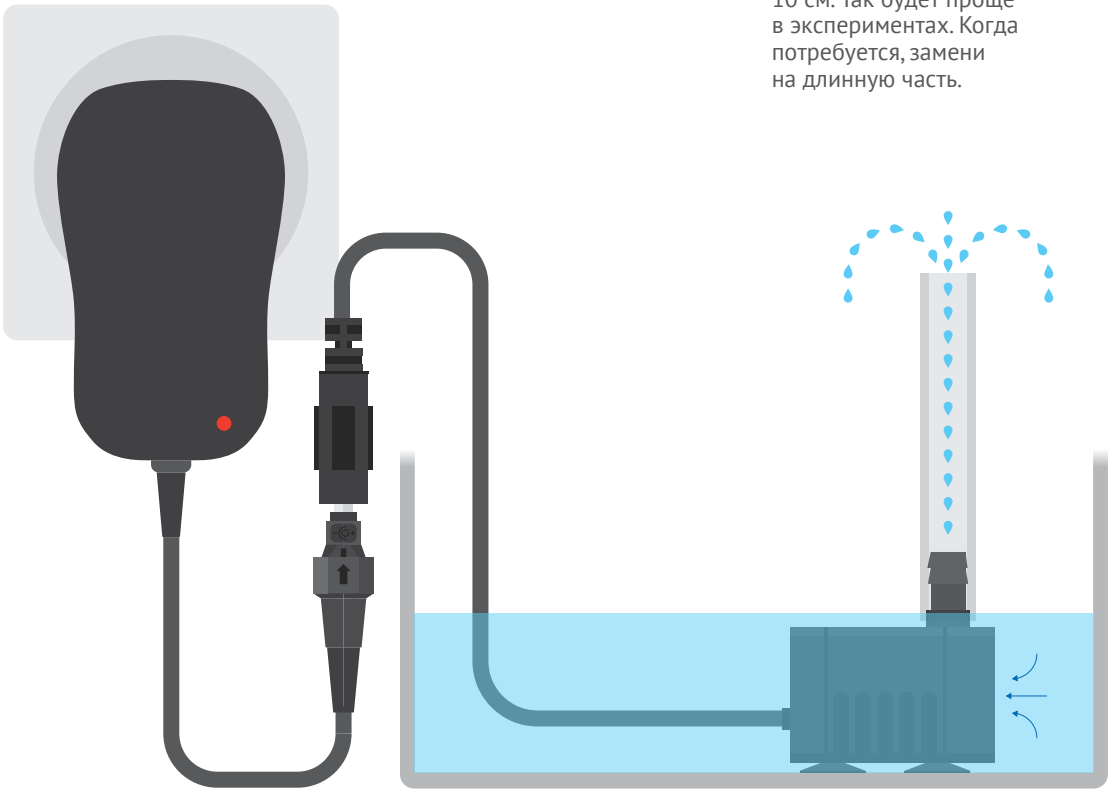
Пиктограмма говорит, что внешний цилиндр будет с полярностью «минус», а внутренний — «плюс».

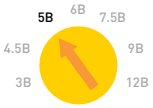
6 Включи блок питания в розетку.



7 Соедини кабели помпы и блока питания. Ты увидишь небольшой фонтан!

Отрежь от длинного шланга небольшой кусочек длиной 10 см. Так будет проще в экспериментах. Когда потребуется, замени на длинную часть.



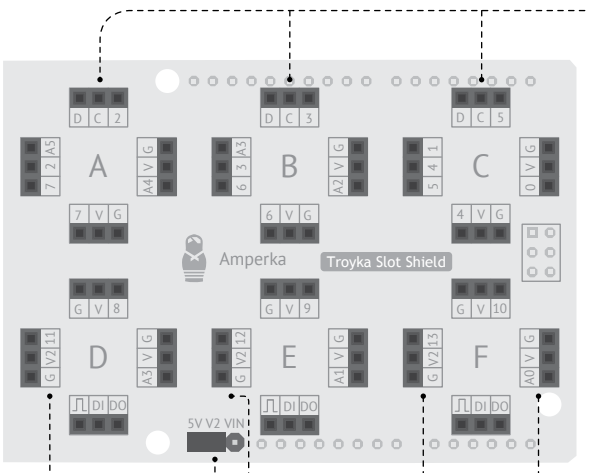


ВАЖНО!
Обязательно в конце эксперимента установи переключатель в положение 5 вольт.

ЗАДАНИЕ
Переведи переключатель напряжения на блоке питания в положение 6 вольт, 7,5 вольт и так далее. Высота фонтана будет увеличиваться.

TROYKA SLOT SHIELD

Troyka Slot Shield — это плата расширения для быстрой сборки компактных устройств из Troyka-модулей без проводов и паяльника.



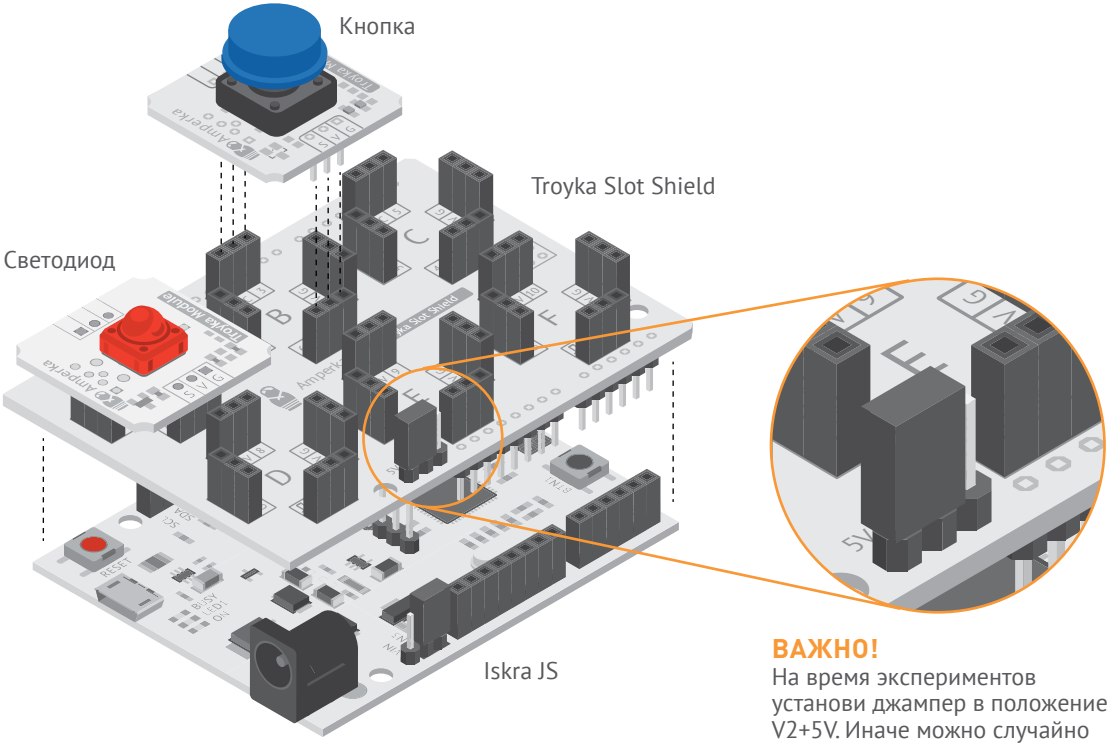
В ПЛАТУ МОЖНО ПОДКЛЮЧИТЬ:

- шесть Troyka-модулей;
- три модуля, работающих по протоколу I²C.

Подключение таких Troyka-модулей требует два трёхконтактных разъёма. Один разъём используется для подачи напряжения. Другой — для коммуникации с управляющей платой через пины SCL и SDA.

КАК ПОДКЛЮЧАТЬ

Вставь Slot Shield в пины Iskra JS сверху. Ты получишь единое устройство.



ВАЖНО!

На время экспериментов установи джампер в положение V2+5V. Иначе можно случайно сжечь модули высоким напряжением Vin.

НАПРЯЖЕНИЕ ПИТАНИЯ

В трёх слотах используется альтернативная линия питания V2, напряжение на которой можно выбирать джампером:

- V2+5V — на V2 будет 5 вольт вне зависимости от рабочего напряжения управляющей платы;
- V2+Vin — на V2 будет напряжение порта Vin управляющей платы.

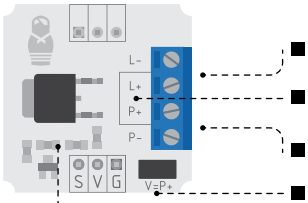
№ 2

ЖИВОЙ ФОНТАН

Фонтан силе своей подчиним. Через силовой ключ подсоединим и кнопку управления добавим.

Ток, потребляемый помпой, слишком большой, чтобы питать её напрямую от Iskra JS. Для управления большими нагрузками применяют силовые ключи.

Силовой ключ



- К контактам L+ и L– подключают нагрузку.
- Контакты L+ и P+ всегда замкнуты.
- Контакты P+ и P– служат для подключения питания.
- Джампер объединяет линию V и P+.
- При подаче напряжения на пин S контакты L– и P– замыкаются и загорается светодиод.

1 Вставь в клеммы L+ и L– провод для питания помпы и затяни винты.

Убедись, что джампер на месте

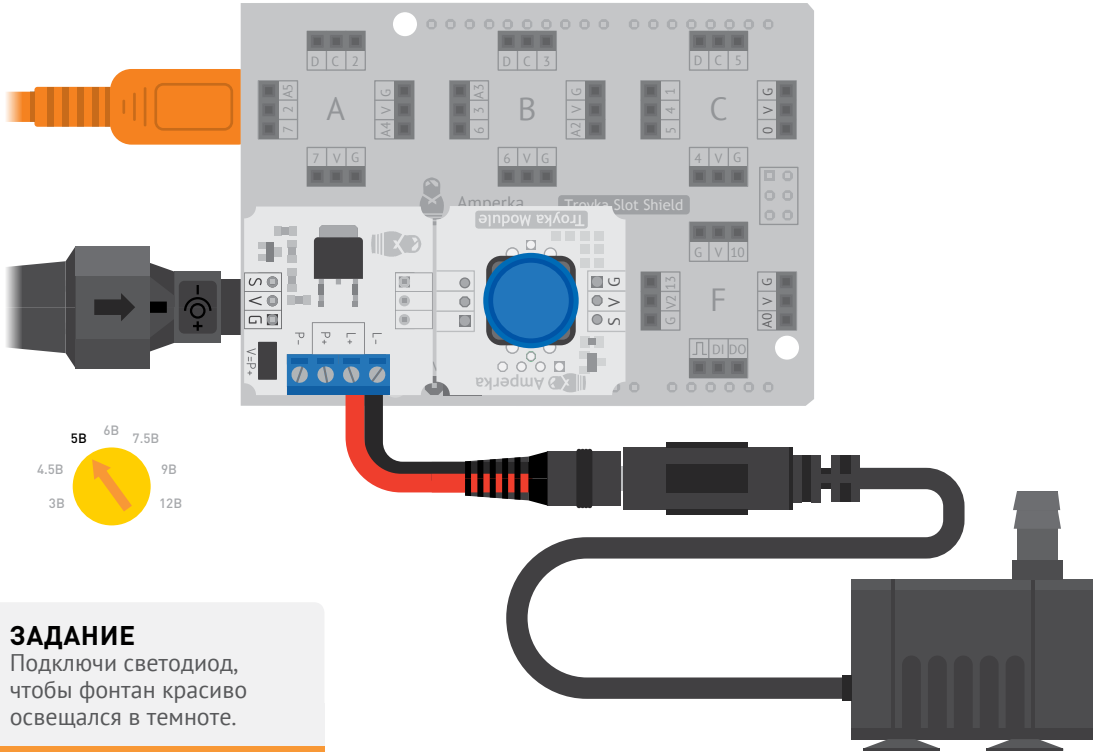
2 Открой Espruino IDE и напиши код.

```
var button = require('@amperka/button').connect(A1);
var pump = require('@amperka/power-control')
  .connect(P11);

button.on('press', function() {
  pump.toggle();
});
```

1 Подключаем библиотеку `'@amperka/power-control'` для управления силовыми ключами.

2 Нажатием на кнопку вызываем функцию `toggle()` для переключения ключа.

3 Загрузи код в Iskra JS и собери устройство. Фонтан будет запускаться и отключаться по нажатию кнопки.



ЗАДАНИЕ

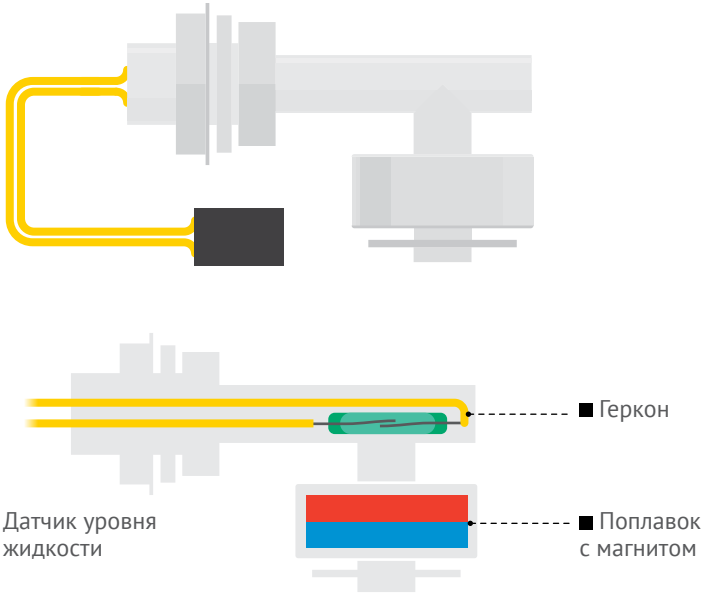
Подключи светодиод, чтобы фонтан красиво освещался в темноте.

№ 3

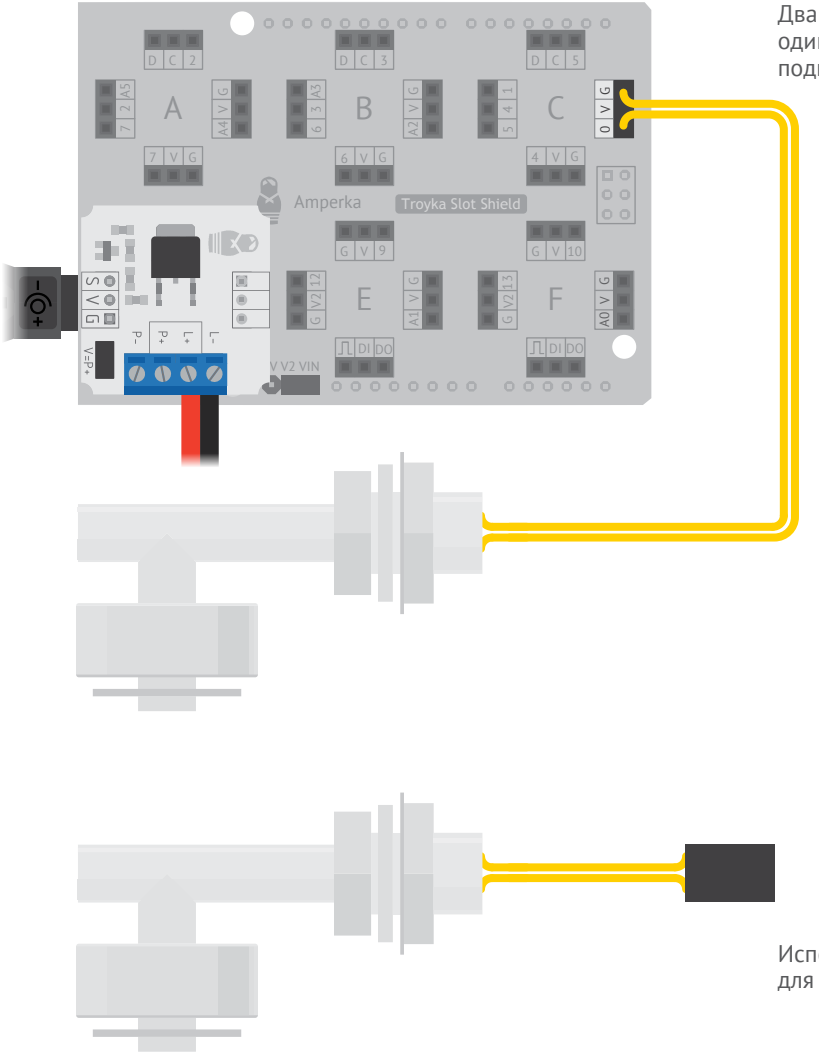
ДРЕНАЖ

Перегреется помпа и из строя выйдет, если без воды работать будет долго. Датчик уровня установим, чтобы силы её сберечь.

В поплавке на датчике установлен магнит. Он действует на *геркон*, а тот замыкает два жёлтых провода.



Геркон (*герметизированный контакт*) — кнопка из двух ферромагнитных контактов, притягивающихся друг к другу в магнитном поле.



Два провода совершенно одинаковы, поэтому полярность подключения не важна.

Используй переходники для соединения проводов.

```
var pump = require('@amperka/power-control').connect(P11);
pump.turnOn();

var level = require('@amperka/water-level').connect(P0);

level.on('down', function () {
  print('Water level is low');
  pump.turnOff();
});
```

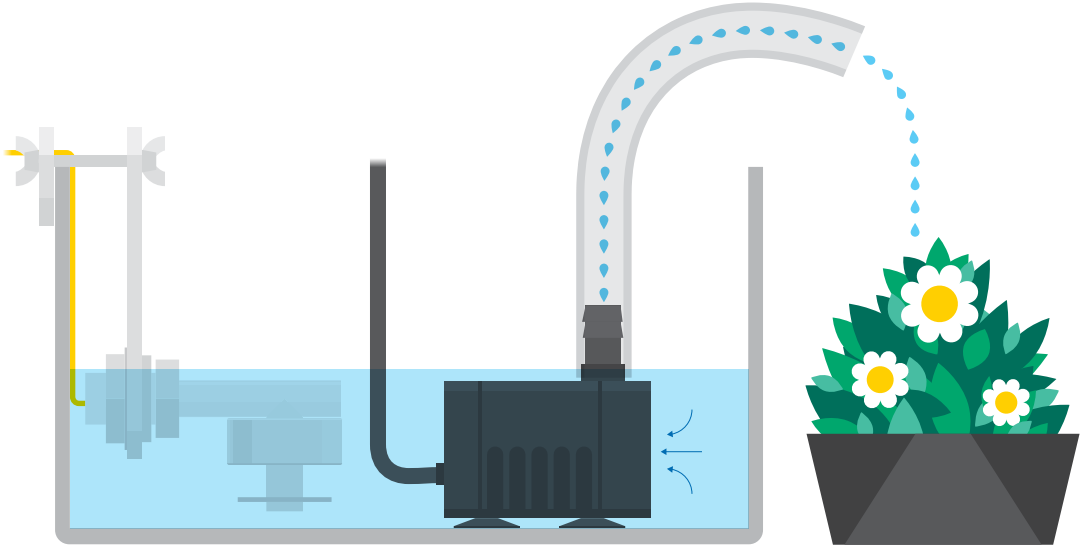
1 Сразу при подаче питания запускаем помпу.

2 Подключаем библиотеку датчика уровня `'@amperka/water-level'`.

3 Если уровень воды низок, вызываем событие `'down'`, печатаем в консоль сообщение и отключаем насос.

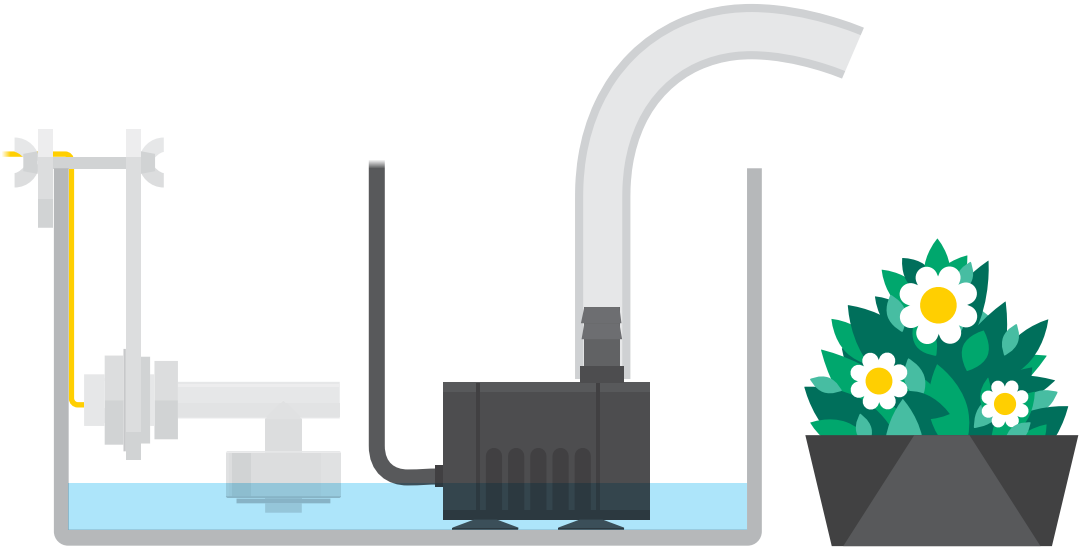
Смотри справочник по библиотекам в конце книги.

1 Погрузи датчик уровня в воду, чтобы поплавок всплыл и запусти код.



2 Когда вода закончится, фонтан отключится, чтобы избежать перегрева помпы.

ВАЖНО!
Эксперименты разумно проводить без живых растений. Так не погубишь свои цветы большим количеством воды.



Заходи на сайт water.amperka.ru за дополнительной информацией и помощью.

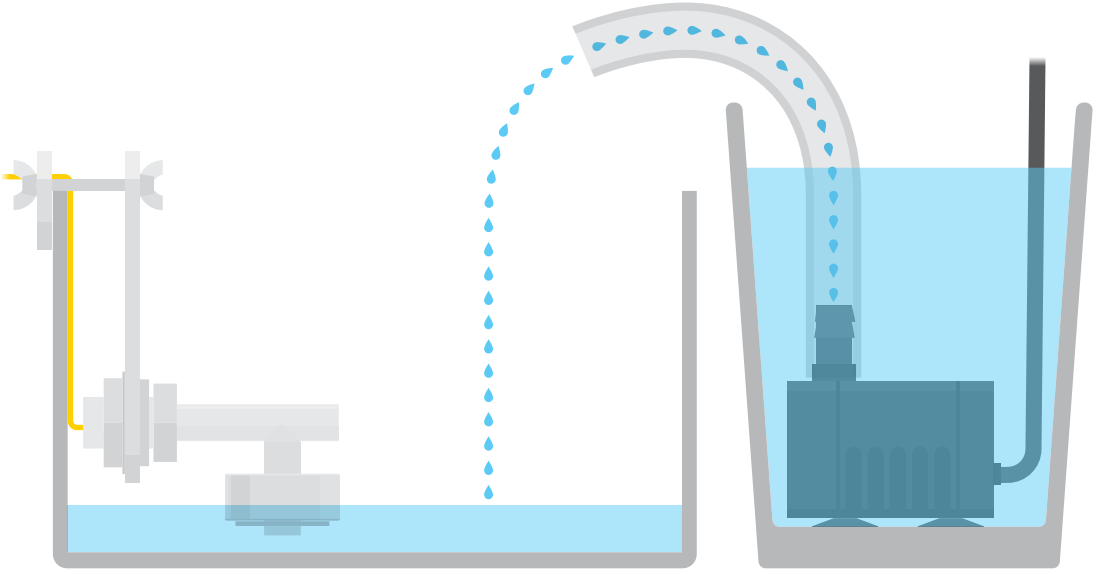
ЗАДАНИЕ
Добавь к проекту зуммер (пьезопищалку). Пусть звуковой сигнал напоминает, когда стоит подлить воды.

№ 4

АВТОПОПОЛНЕНИЕ

Тонкой настройкой датчика уровня займёмся. Автопополнение воды практикой нам будет.

Пусть помпа включается, когда уровень воды в ёмкости ниже датчика уровня.



1. Возьми стакан с водой и помести в него помпу.

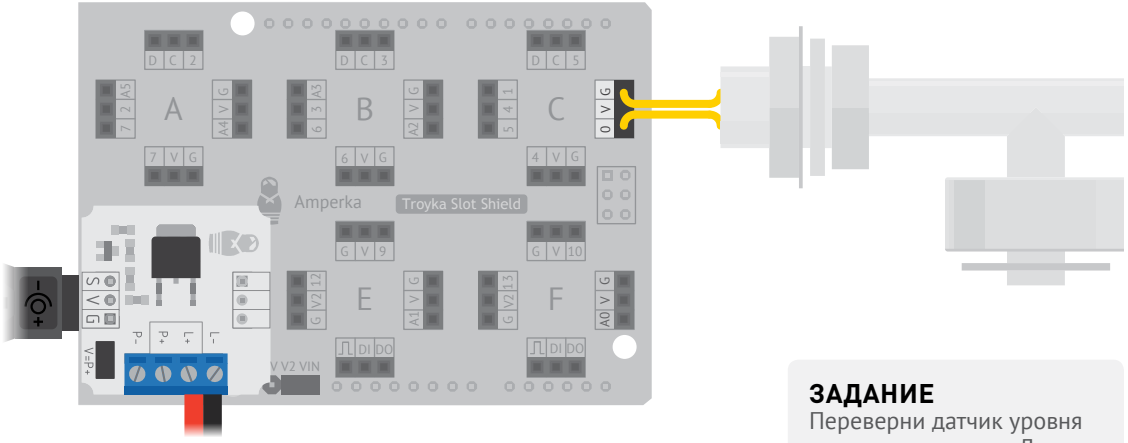
```
var pump = require('@amperka/power-control')
  .connect(P11);

var level = require('@amperka/water-level')
  .connect(P0, {debounce: 0.5});

level.on('up', function () {
  pump.turnOff();
});

level.on('down', function () {
  pump.turnOn();
});
```

1 Добавляем параметр `debounce: 0.5`. Это означает, что датчик вызовет события `'up'` или `'down'` через 0,5 секунды после изменения уровня. Так избавимся от ложных срабатываний при сильных «волнах» на поверхности воды.

2 При срабатывании события `'up'` выключаем помпу.

3 При срабатывании события `'down'` включаем помпу.

2 Собери схему и загрузи код.



3 Отчерпни воду из контейнера. Помпа включится, пытаясь восполнить объём. Как только уровень воды поднимется — помпа отключится.

ЗАДАНИЕ

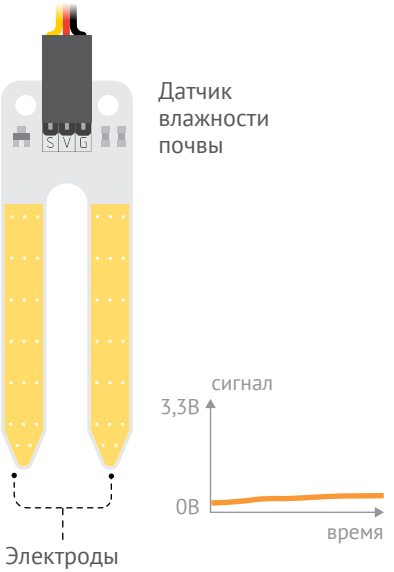
Переверни датчик уровня вверх тормашками. Логика программы изменится. Добавь параметр `mounted: 'onTop'` к датчику уровня, чтобы всё встало на свои места.

№ 5

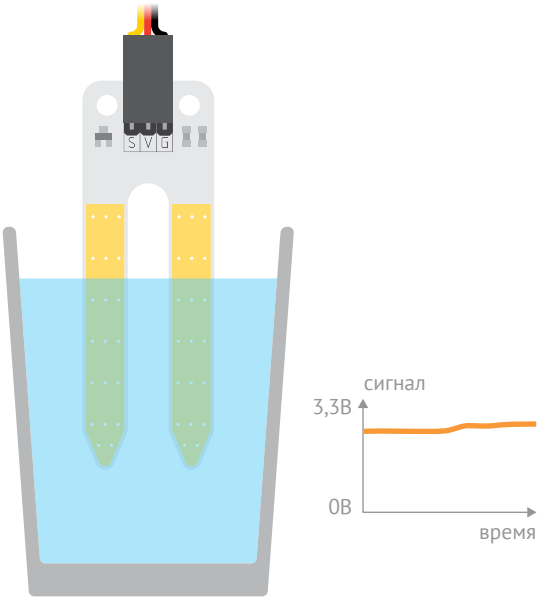
КАЛИБРОВКА

Азы датчика влажности почвы познаем и рабочий диапазон на Iskra JS увидим.

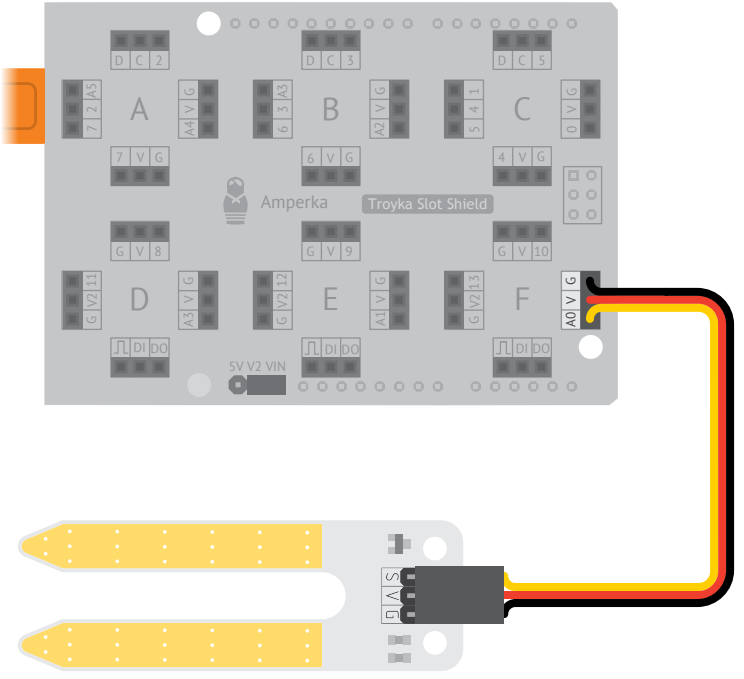
Датчик влажности почвы — аналоговый датчик. Между двумя электродами создаётся небольшое напряжение. Если почва сухая, её сопротивление велико, и ток будет небольшим. Если земля влажная, сопротивление становится меньше, а ток — чуть больше. По аналоговому сигналу можно судить о степени влажности почвы.



Сопротивление воздуха велико, поэтому аналоговое значение напряжения будет небольшим.



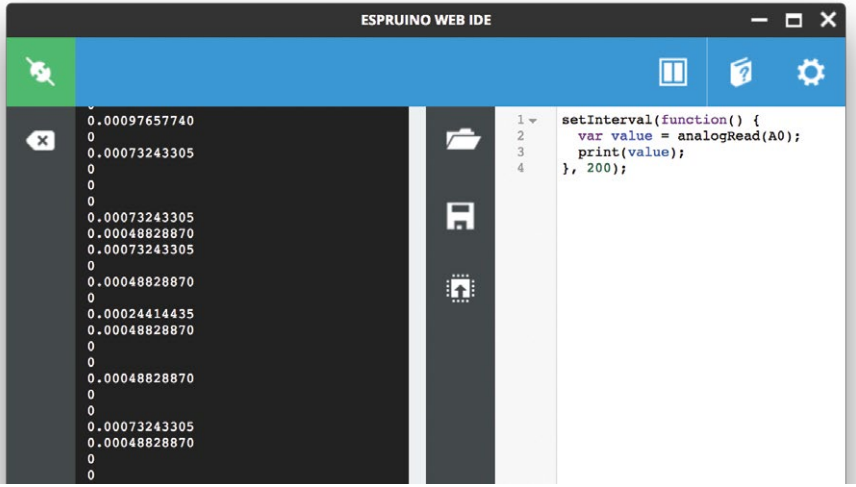
Сопротивление воды меньше, поэтому аналоговое значение напряжения вырастет.



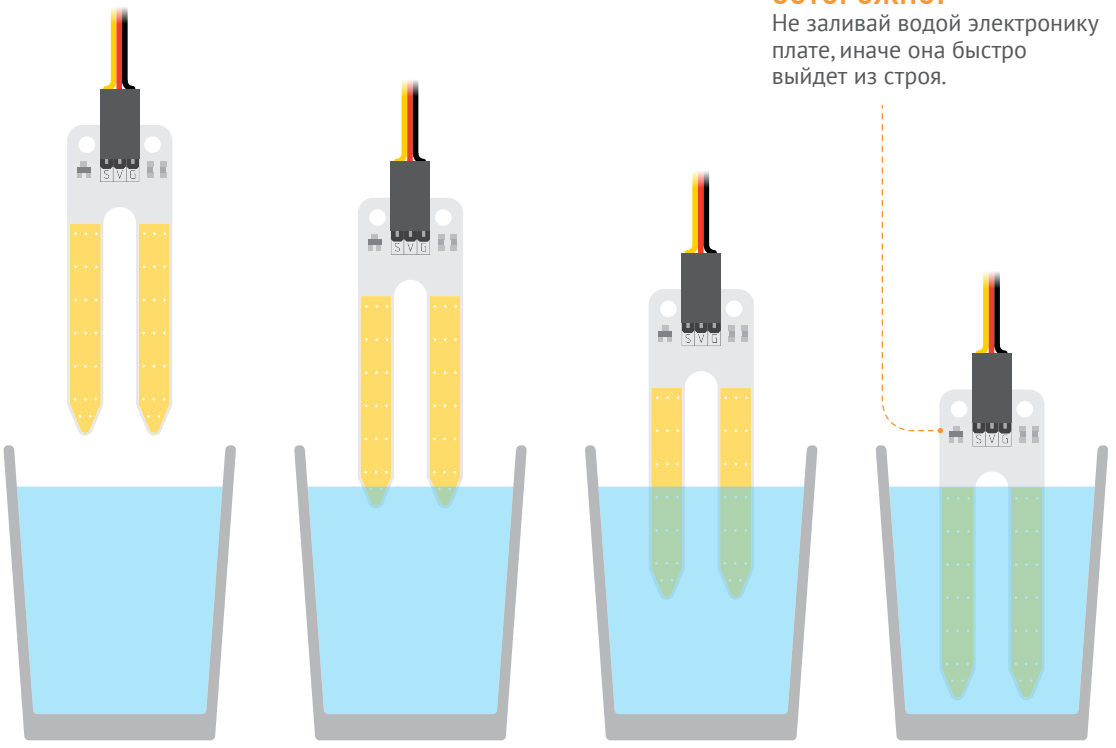
1 Загрузи в Iskra JS простой код: каждые 200 миллисекунд печатать значение аналогового сигнала датчика влажности почвы.

```
setInterval(function() {
  var value = analogRead(A0);
  print(value);
}, 200);
```

2 Посмотри, какие значения выдаёт датчик в воздухе. Запиши их.



3 Набери воды в стакан. Запиши значения для разных уровней погружения датчика в воду.



ОСТОРОЖНО!
Не заливай водой электронику плате, иначе она быстро выйдет из строя.

4 Опусти датчик целиком в воду. Возьми на кухне соль. Постепенно добавляй её в стакан и следи за изменениями значений.

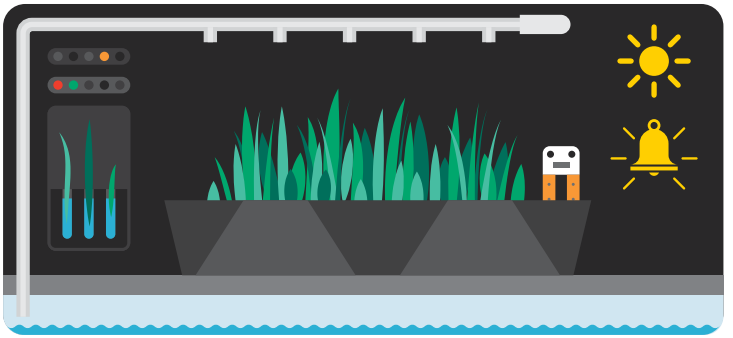
5 Попробуй добавить к воде другие примеси с кухни. Сравни результаты.

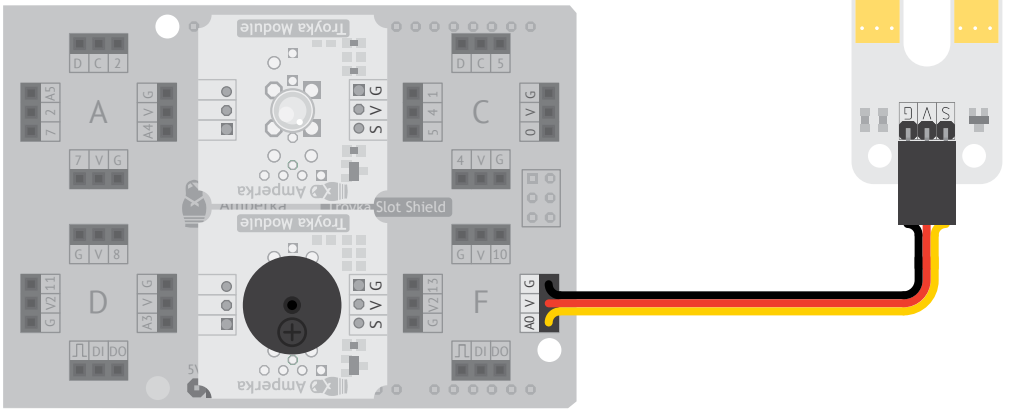
ЗАДАНИЕ
Воткни датчик влажности в горшок, который давно не поливали. Плавно начни добавлять воды и следи за значениями аналогового сигнала. Это пригодится тебе в следующих экспериментах.

№ 6

НАПОМИНАЛЬНИК

В помощь джедаю звуковое уведомление сделаем. Поливать растения напоминать будет, когда почва суха.





Воспользуемся библиотекой гистерезиса. Она знакома по эксперименту 22 из набора «Йодо».

```
var hyst = require('@amperka/hysteresis')
  .create({high: 0.5, highLag: 2, low: 0.4, lowLag: 2});

var led = require('@amperka/led').connect(A2);
var buzzer = require('@amperka/buzzer').connect(A1);

setInterval(function() {
  hyst.push(analogRead(A0));
}, 200);

hyst.on('low', function() {
  buzzer.beep(1, 0.5);
  led.blink(1, 0.5);
});

hyst.on('high', function() {
  buzzer.turnOff();
  led.turnOff();
});
```

1 Если значение влажности почвы меньше `0,4`, вызываем событие `'low'`. Как только влажность станет достаточной (польём цветок), вызываем событие `'high'`.

2 Каждые 200 миллисекунд обновляем входные значения фильтра.

Загрузи код. Проведи эксперимент с растением, которое давно не поливали. Если таких не окажется, погрузи датчик в стакан с водой.

ЗАДАНИЕ

Добавь к сигнализации датчик освещённости, чтобы писк и мигающий свет не разбудили тебя ночью.

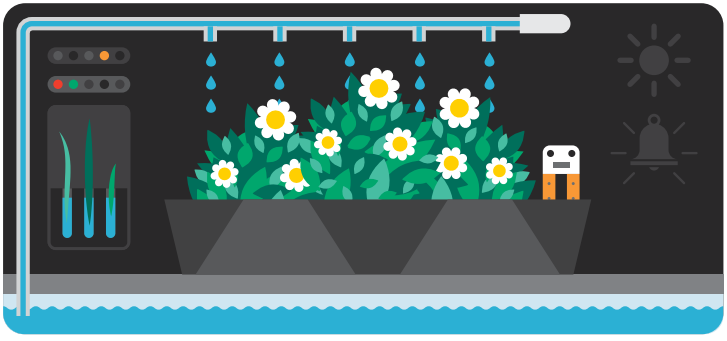
ВНИМАНИЕ!

Значения `high` и `low` гистерезиса следует подбирать самостоятельно. Они сильно зависят от химического состава почвы и воды, износа металлических площадок датчика.

№ 7

АВТОПОЛИВ

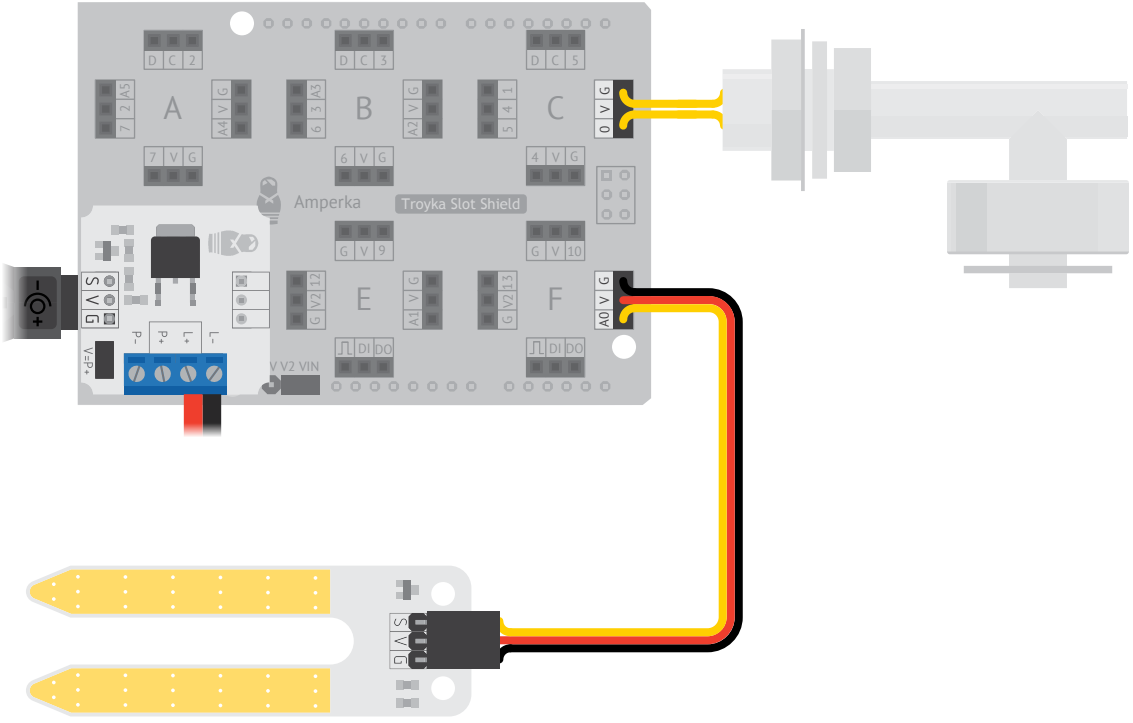
Автополив создадим. Если почва суха, пусть помпа включится и растение польёт. Датчик уровня помпе выйти из строя не даст.



```
var hyst = require('@amperka/hysteresis')
  .create({high: 0.5, highLag: 2, low: 0.4, lowLag: 2});

var pump = require('@amperka/power-control')
  .connect(P11);
var level = require('@amperka/water-level').connect(P0);

setInterval(function() {
  hyst.push(analogRead(A0));
}, 200);

hyst.on('low', function() {
  pump.turnOn();
});

hyst.on('high', function() {
  pump.turnOff();
});

level.on('down', function () {
  pump.turnOff();
});
```

1 Включаем подачу воды, если почва сухая. Отключаем, когда воды в земле достаточно.

2 Если во время полива закончилась вода, отключаем помпу во избежание перегрева.



Используя такое устройство ежедневно, нет необходимости обновлять значения гистерезиса так часто. Один раз в час (и даже реже) будет достаточно (потребуется доработать код).

ЗАДАНИЕ

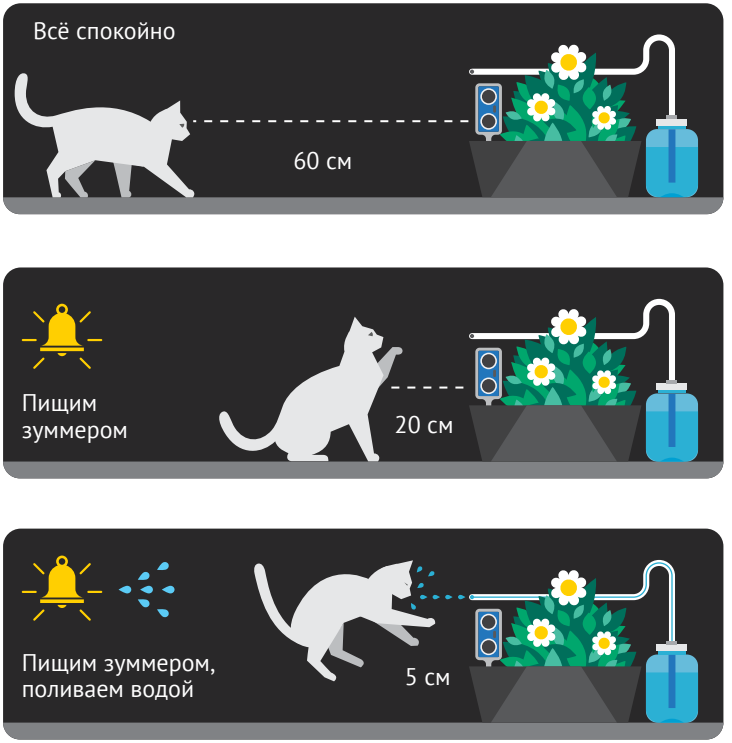
Добавь светодиод и зуммер, чтобы не забывать подливать воды в ёмкость с помпой.

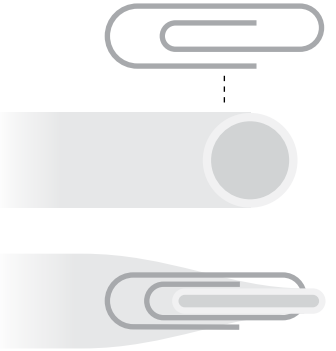
№ 8

ПУШИСТАЯ ОБОРОНА

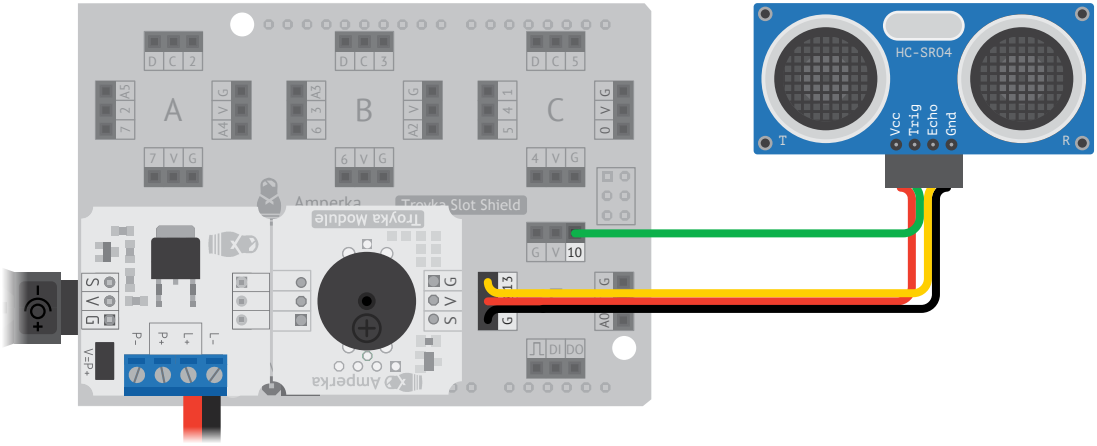
Домашних животных отучим растительность уничтожать.
Демотиватор для них сделаем.

Датчиком непрошеных гостей будет ультразвуковой дальномер.
Возьми за основу эксперимент «Парктроник» из набора «Йодо».





Зажми кончик шланга скрепкой, чтобы тонкая струйка стреляла дальше.




```
var sonic = require('@amperka/ultrasonic')
  .connect({trigPin: P10, echoPin: P13});

var pump = require('@amperka/power-control')
  .connect(P11);
var buzzer = require('@amperka/buzzer')
  .connect(A1)
  .frequency(440);

setInterval(function() {
  sonic.ping(function(err, value) {
    if (err) return;

    if (value < 20) {
      buzzer.beep(0.5, 0.2);
    } else if (value < 50) {
      buzzer.beep(1, 0.5);
    } else {
      buzzer.turnOff();
    }

    if (value < 5) {
      pump.turnOn();
    } else {
      pump.turnOff();
    }
  }, 'cm');
}, 200);
```

1 Предупреждаем злоумышленника звуковыми сигналами.

2 Если пушистый бандит подошёл к растению ближе чем на 5 сантиметров — включаем водяную пушку.

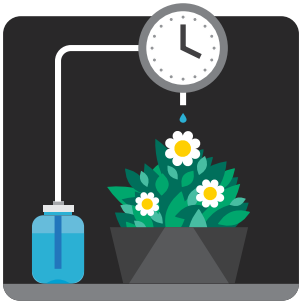
ЗАДАНИЕ

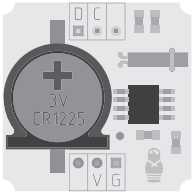
Добавь в проект потенциометр, чтобы быстро настраивать безопасное расстояние до цветка.

№ 9

ТАЙМЕР

Галактическое время чувствовать научимся, чтобы растения по расписанию поливать.





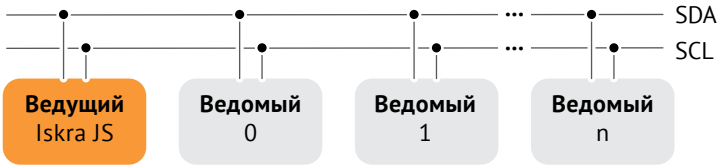
Troyka-модуль
часы реального времени

РЕЗЕРВНОЕ ПИТАНИЕ

Модуль не считает время, пока отключён от питания. Чтобы быть пунктуальным, модулю необходима батарейка CR1225.

ИНТЕРФЕЙС I²C

Модуль часов реального времени работает через специальный интерфейс — I²C (*Inter-Integrated Circuit*, «межмикросхемный» интерфейс). Этот интерфейс работает на двух линиях: передачи данных и тактирования. В отличие от интерфейса SPI (он встречался во втором эпизоде «Йодо. Интернет вещей»), полезная информация передаётся по одному проводу. Сначала ведущее устройство её запрашивает, а затем ждёт ответ от ведомых. Линия тактирования задаёт скорость передачи данных.



ОБОЗНАЧЕНИЕ ЛИНИЙ

Линия данных SDA (от data) обозначена символом D на Troyka-модулях.

Линия тактирования SCL (от clock) обозначена символом C.

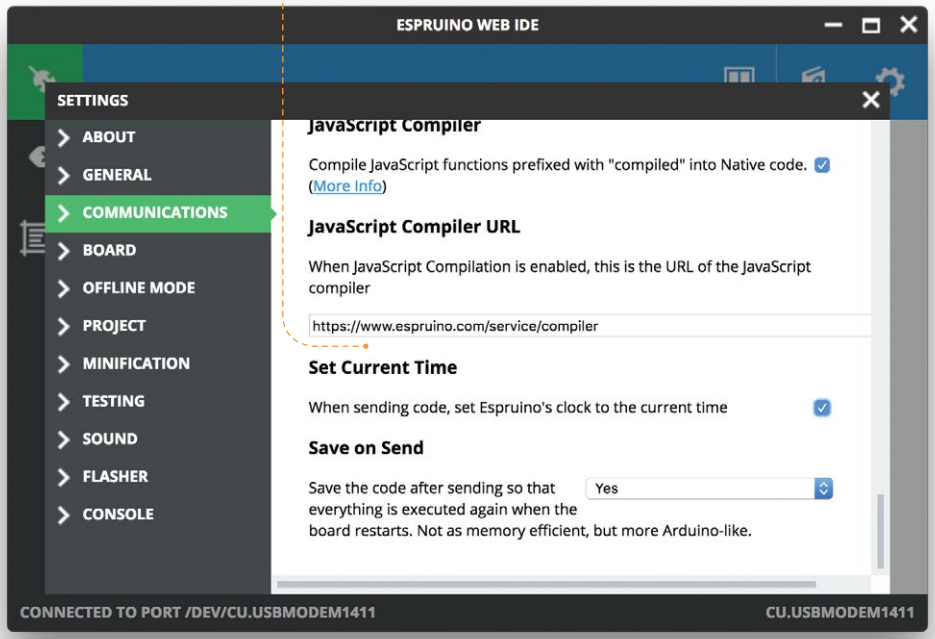
Ещё одной особенностью I²C является адресность. Каждое ведомое устройство должно иметь своё уникальное число, по которому к нему можно обратиться. Например, часы реального времени имеют адрес 104. Благодаря адресности в интерфейсе I²C нет линии CS выбора ведомого устройства.

Адрес модуля уже учтён в библиотеке, поэтому его можно не запоминать. Но это пригодится в более сложных проектах с новыми модулями.

Если прошивка твоей Iskra JS старше 1.93, будет полезно активировать функцию *Set Current Time* в настройках Espruino IDE.

1. Открой окно настроек ⚙ в Espruino WEB IDE.

2. Перейди в раздел Communications и поставь галочку напротив пункта Set Current Time.



```
var rtc = require('@amperka/rtc').connect();
rtc.setTime();
var pump = require('@amperka/power-control').connect(P11);

var lastWatering = -1;

setInterval(function () {
  var date = rtc.getTime();
  print(date.toString());
  if (lastWatering === date.getDate()) return;

  if (date.getHours() === 13) {
    lastWatering = date.getDate();
    pump.pulse(3);
  }
}, 5000);
```

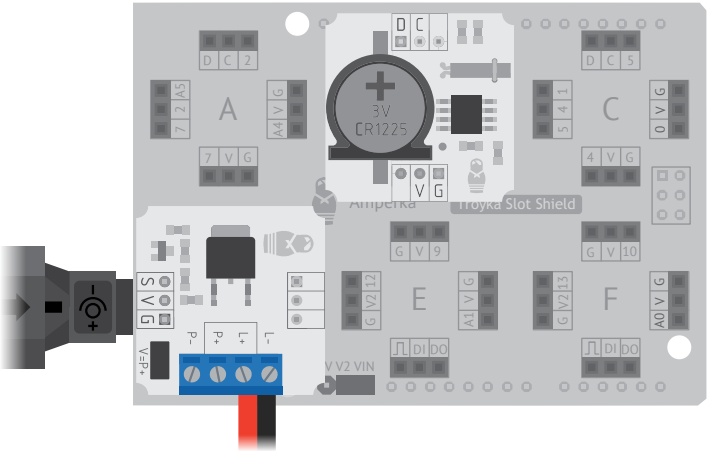
1 Подключаем библиотеку `'@amperka/rtc'` и устанавливаем актуальное время функцией `setTime()`. Пины подключения не указываем, так как модуль общается с Iskra JS по интерфейсу I²C.

2 В переменной `lastWatering` будем хранить дату последнего полива.

3 Каждые 5 секунд обновляем время и выводим на печать в виде строки.

4 Если на часах 13, поливаем цветы в течение трёх секунд и запоминаем сегодняшнюю дату.

3 Укажи в коде время, близкое к твоему. Так не придётся долго ждать результата эксперимента.



К сожалению, модуль не умеет работать с часовыми поясами. Исправим это самостоятельно.

Внимание! Если прошивка твоей Iskra JS старше 1.93, значит, она уже умеет работать с часовыми поясами! Просто добавь самой первой строкой в коде: `E.setTimeZone(+3);`

ВАЖНО!
Обновить прошивку поможет раздел вики js.amperka.ru → «Среда программирования» → «Обновление прошивки».

4 Добавь учёт временно́й зоны в код вместо установки времени модуля.

```
1  var timeZone = +3 * 60 * 60;
2  var current = Date().valueOf() / 1000 + timeZone;
3  rtc.setTime(current);
```

1 Переводим разницу в часах между Москвой и Гринвичем в секунды.

2 Узнаём текущее время Iskra JS в секундах и прибавляем разницу часовых поясов.

3 Устанавливаем на модуле время с учётом зоны.

5 Запусти программу и увидишь, что теперь время соответствует твоему часовому поясу.

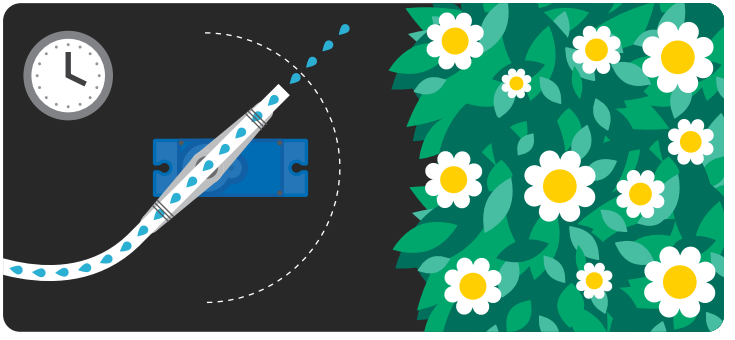
Справочник в конце этой брошюры поможет в работе с часами реального времени.

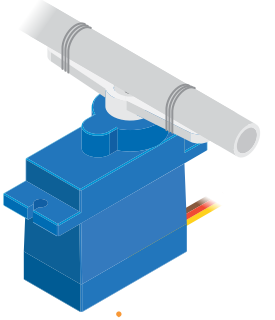
ЗАДАНИЕ
Сделай себе будильник. Пусть он звенит по будням в 8:30.

№ 10

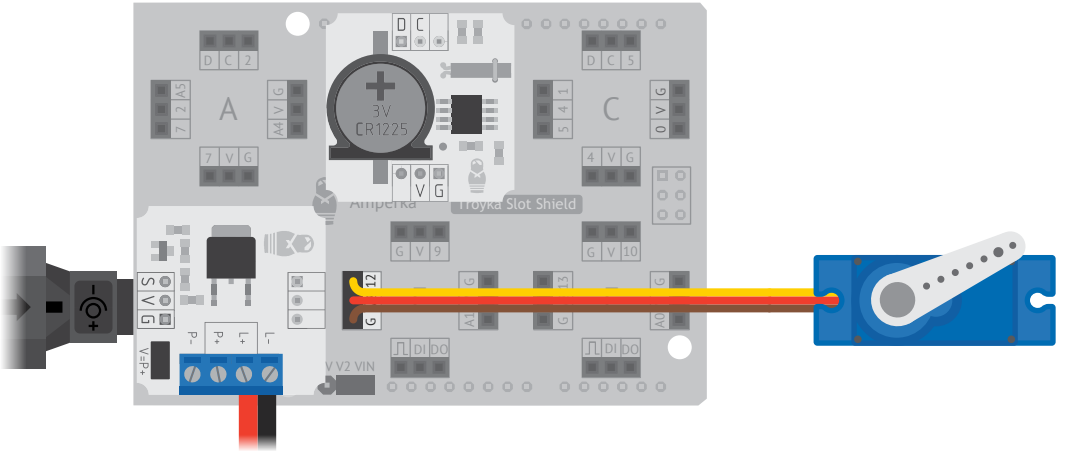
ДОЖДЕВАТОР

О растениях на даче забывать не стоит. Дождеватор для газона сделаем. Сервопривод большую площадь охватить поможет.





1 Закрепи конец шланга на качалке сервопривода. Приспособь для этого скрепку или канцелярскую резинку.



2 Загрузи код в Iskra JS и собери схему. Для начала потренируйся без воды.

```
var rtc = require('@amperka/rtc').connect();
rtc.setTime();

var pump = require('@amperka/power-control')
  .connect(P11);

var servo = require('@amperka/servo').connect(P12);
var swing = require('@amperka/animation').create({
  from: 30,
  to: 150,
  duration: 5,
  updateInterval: 0.02
}).queue({
  from: 150,
  to: 30,
  duration: 5
});

swing.on('update', function(val) {
  servo.write(val);
});

var lastWatering = -1;

setInterval(function () {
  var date = rtc.getTime();
  if (lastWatering === date.getDate()) return;

  if (date.getHours() === 6) {
    lastWatering = date.getDate();
    swing.play();
    pump.pulse(10);
  }
}, 1000);
```

1 Подключаем библиотеку `'@amperka/animation'` и задаём 2 последовательности по 5 секунд: с 30 градусов до 150 и в обратную сторону.

2 Обновляем угол сервопривода на каждом «кадре» анимации.

3 Каждый день в 6 утра запускаем дождеватор на 10 секунд.

Для экспериментов удобно указывать свой текущий час, чтобы программа сразу же запускала полив.

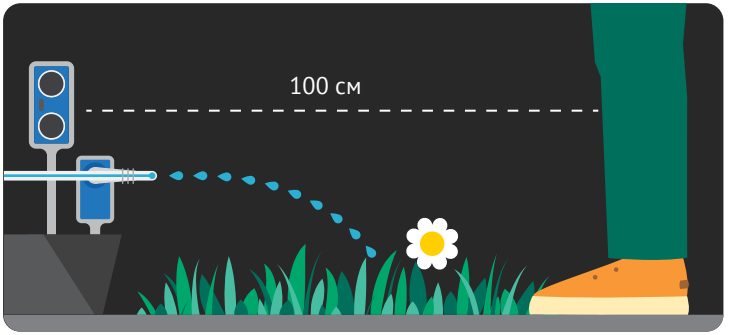
ЗАДАНИЕ

Добавь датчик влажности. Пусть он не даст затопить газон, если ночью был дождь.

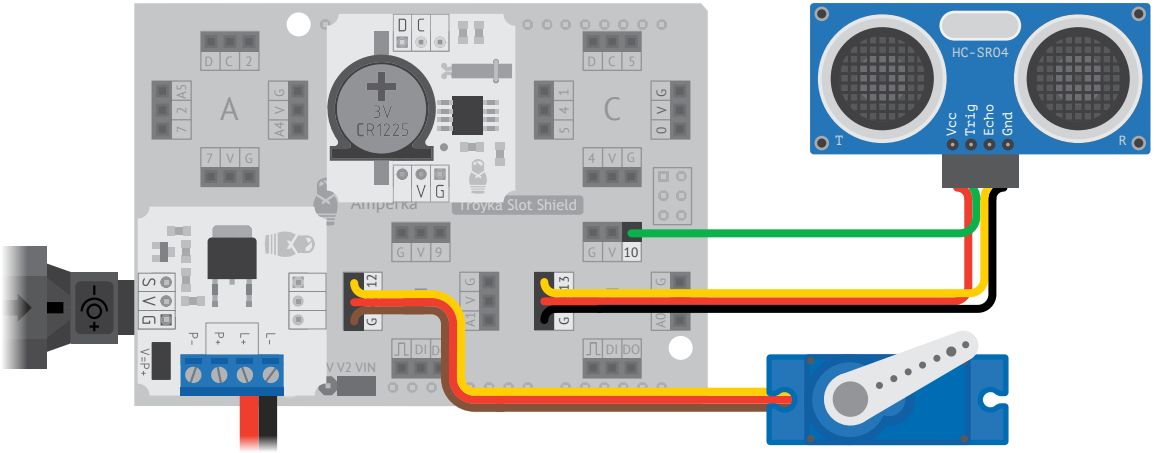
№ 11

ДОЖДЕВАТОР 3000

Добавим дождевателю ультразвуковой дальномер, чтобы он ненароком не облил водой проходящих мимо людей.



Если ультразвуковой датчик засечёт препятствие, полив встанет на паузу. Когда препятствие пропадёт — продолжится.



```
var rtc = require('@amperka/rtc').connect();
var pump = require('@amperka/power-control')
  .connect(P11);
var servo = require('@amperka/servo').connect(P12);
var sonic = require('@amperka/ultrasonic')
  .connect({trigPin: P10, echoPin: P13});

var angle = 90;
var delta = 1;
var lastWatering = -1;
var portions = 50;
var wateringIntervals = portions;

setInterval(function () {
  var date = rtc.getTime();
  if (lastWatering === date.getDate()) return;

  if (date.getHours() === 6) {
    wateringIntervals = 0;
    lastWatering = date.getDate();
  }
}, 1000);

setInterval(function() {
  sonic.ping(function(err, value) {
    if (value < 100 || wateringIntervals > portions) {
      pump.turnOff();
    } else {
      wateringIntervals = wateringIntervals + 1;
      servo.write(angle);
      pump.turnOn();
      if (angle === 150 || angle === 30) {
        delta = -delta;
      }
      angle = angle + delta;
    }
  }, 'cm');
}, 200);
```

1 В переменной `portions` храним количество «порций» (требуемый объём) ежедневного полива.

2 Каждый день в 6 утра запускаем полив газона, сбросив в ноль выполненный объём.

3 Если дальномер засёк предмет в зоне полива (менее 100 см) или весь объём полива выполнен, отключаем помпу...

4 ...иначе прибавляем «порцию» к выполненному объёму, включаем помпу и поворачиваем качельку сервопривода.

5 Если сервопривод дошёл до крайнего положения, меняем направление приращения угла.

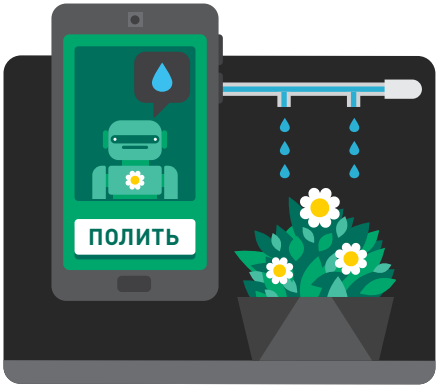
1 Запусти проект. Отладь все элементы и будь готов к дачному сезону!

ЗАДАНИЕ

Добавь потенциометр, чтобы быстро регулировать объём полива.

ИДЕИ ПРОЕКТОВ

На достигнутом не останавливайся, свои проекты создавай и мудрость джедая постигай.

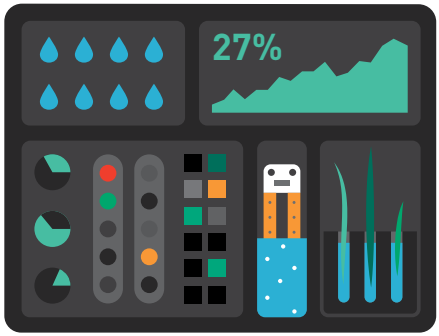


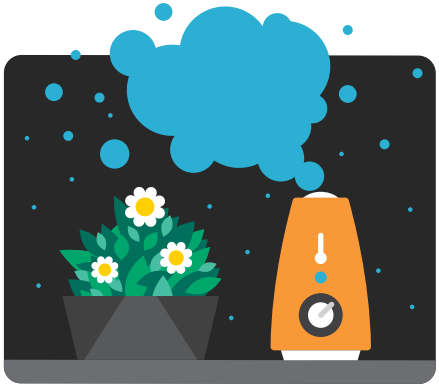
ТЕЛЕГРАМ-БОТ И IFTTT

Автоматизируй полив и общайся со своим садом с помощью Телеграм-бота. Установи программные триггеры с сервисом IFTTT. Всему этому ты научишься с эпизодом 2 — «Йодо Интернет вещей».

АВТОПОЛИВ 2.0

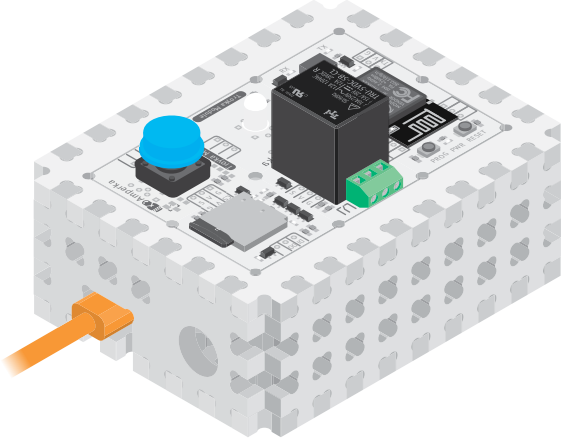
Добавь датчик влажности воздуха для качественного ухода за растениями. Отслеживай параметры на сайте dweet.io. Для этого потребуется модуль Wi-Fi или Ethernet Shield.





УВЛАЖНИТЕЛЬ ВОЗДУХА
Приспособь бытовой увлажнитель воздуха. Управляй им через модули Nano Switch. Используй часы реального времени для работы по расписанию.

СТРУКТОР
Устройства на Troyka Slot Shield можно удобно оформлять в корпус из #Структора с помощью набора плашек Slot Box (#Структор). Их можно найти на сайте amperka.ru.



ВСЕ ЭТИ ДАТЧИКИ И ПРИБОРЫ ТЫ СМОЖЕШЬ НАЙТИ НА AMPERKA.RU. ОСТАВАЙСЯ С НАМИ — ТЕБЯ ЖДЁТ ЕЩЁ БОЛЬШЕ КРУТЫХ ПРОЕКТОВ!

СПРАВОЧНИК

ЧАСЫ РЕАЛЬНОГО ВРЕМЕНИ

```
var rtc = require('@amperka/rtc').connect();
var time = '2016-03-28T13:45:43';
```

`rtc.setTime(time)` — устанавливает время `time` на часах.

Параметр `time` может быть:
- числом, значение которого равно количеству секунд с 1 января 1970 года (unixtime);
- строкой в формате ISO, например `'2016-03-28T13:45:43'`;
- объектом `Date`;
- объектом со свойствами:
  - `year` — четырёхзначный год;
  - `month` — номер месяца от 1 до 12;
  - `day` — дата от 1 до 31;
  - `hour` — час от 0 до 23;
  - `minute` — минута от 0 до 59;
  - `second` — секунда от 0 до 59.

`rtc.getTime(unit);` — возвращает текущее время на часах. Если параметр `unit` не передан, возвращается объект `Date`. Другие доступные варианты:
- `'unixtime'` — количество секунд с 1 января 1970 года (unixtime);
- `'iso'` — строка в формате ISO.

Подробнее на wiki.amperka.ru/js:rtc.

ДАТЧИК УРОВНЯ ВОДЫ

```
var water = require('@amperka/water-level').connect(P2, {debounce: 2});
```

`water.read();` — прочитать текущее положение поплавка. Возвращается `'up'` или `'down'`.

```
water.on('up', function () {
  print('high');
});
```
— напечатать в консоли `'high'`, если поплавок всплыл.

```
water.on('down', function () {
  print('low');
});
```
— напечатать в консоли `'low'`, если поплавок опустился.

```
var water = require('@amperka/water-level').connect(P2, {mounted: 'onTop'});
```
— работать с датчиком «вверх тормашками».

Подробнее на wiki.amperka.ru/js:water-level.

СИЛОВОЙ КЛЮЧ

```
var mosfet = require('@amperka/power-control').connect(P11);
mosfet.turnOn(); — включить нагрузку.
mosfet.power(0.6);  — убавить мощность до 60%.
mosfet.turnOff(); — выключить нагрузку.
mosfet.blink(0.9, 0.1); — включать на 0,9 секунды и выключать на 0,1 секунды.
```

Библиотека подходит и для Troyka-модуля реле за исключением функции `power()` — реле не способно плавно регулировать коммутируемую мощность.

Подробнее на wiki.amperka.ru/js:power-control.

АНИМАЦИЯ

Анимация — последовательность переходов, каждый из которых задаёт начальное и конечное положения и длительность перехода.

```
var myAnim = require('@amperka/animation').create({
  from: 30,
  to: 120,
  duration: 4,
  updateInterval: 0.02
});
```
— анимация от 30 до 120 продолжительностью 4 секунды с обновлением каждые 20 мс.

```
myAnim.queue({
  to: 90,
  duration: 1
});
```
— второй переход: от 120 до 90 продолжительностью 1 секунду.

Во время перехода анимация генерирует событие `'update'`.
Оно возвращает промежуточное значение для каждого кадра.

```
myAnim.on('update', function(val) {
  myServo.write(val);
});
```
— поворачивать серво на угол `val` на каждом кадре анимации.

`myAnim.play();` — запустить анимацию.

`myAnim.stop();` — остановить анимацию.

`myAnim.reverse();` — воспроизвести анимацию в обратную сторону.

Подробнее на wiki.amperka.ru/js:animation.

ГИСТЕРЕЗИС

Модуль преобразует нестабильный входной аналоговый сигнал в выходной стабильный цифровой сигнал, сравнивая вход с заданными пороговыми значениями.

```
var hyst = require('@amperka/hysteresis').create({
    high: 0.9,
    highLag: 1,
    low: 0.4,
    lowLag: 2
});
```
— вызвать событие `'high'`, если нестабильный сигнал станет больше 0,9 и продержится минимум 1 секунду. Вызвать событие `'low'`, если нестабильный сигнал станет меньше 0,4 и продержится минимум 2 секунды.

`hyst.push(analogRead(A0));` — обновить текущее значение сигнала для анализа.

```
hyst.on('low', function() {
    led.blink(1, 0.5);
});
```
— мигать светодиодом, если произошло событие `'low'`.

```
hyst.on('high', function() {
    led.turnOff();
});
```
— выключить светодиод, если произошло событие `'high'`.

wiki.amperka.ru/js:hysteresis.

Амперка

Мы в компании Амперка надеемся, что наш набор понравился.

Если у тебя есть вопросы, на них ответят на форуме: forum.amperka.ru

Обращайся к видеоканалу за порцией вдохновения: youtube.com/AmperkaRU

Ищи подробные руководства и инструкции на wiki.amperka.ru

Заходи в магазин amperka.ru за новыми наборами

Электронная версия книги: water.amperka.ru

vk.com/amperkaru

facebook.com/amperka.ru

instagram.com/amperkaru

twitter.com/amperka

Дизайн: студия «Кластер»
clusterstudio.ru

ЭЛЕКТРОНИКА НА СТОРОНЕ ДОБРА