

РУКОВОДСТВО ПО МЕЖПЛАНЕТНОМУ ПОЛЁТУ ІІ КОЛОНІЗАЦІІІ

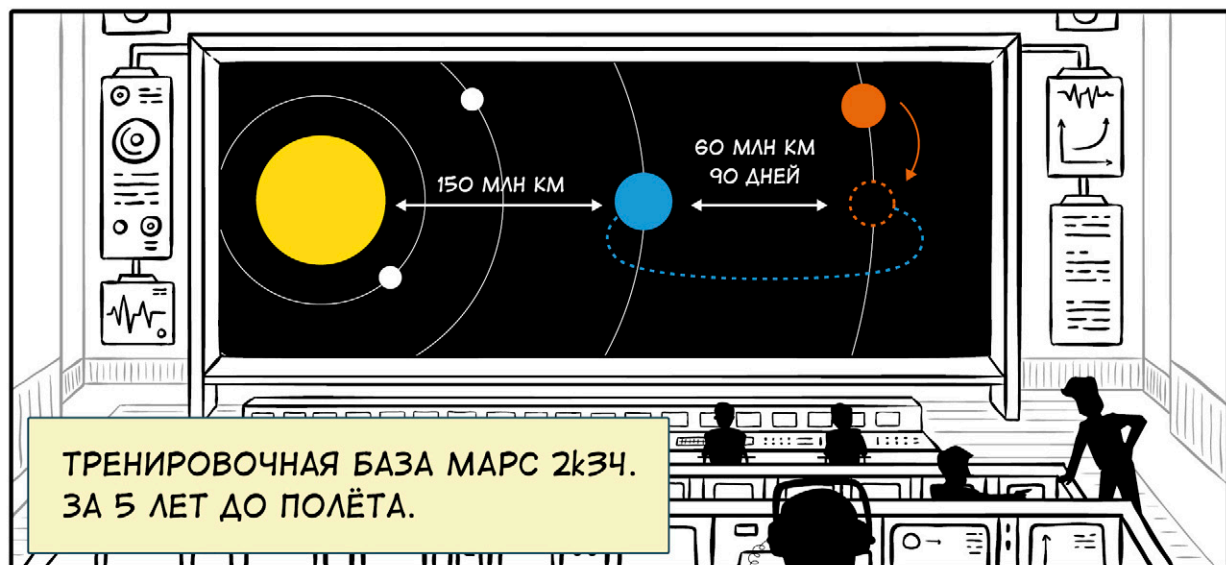


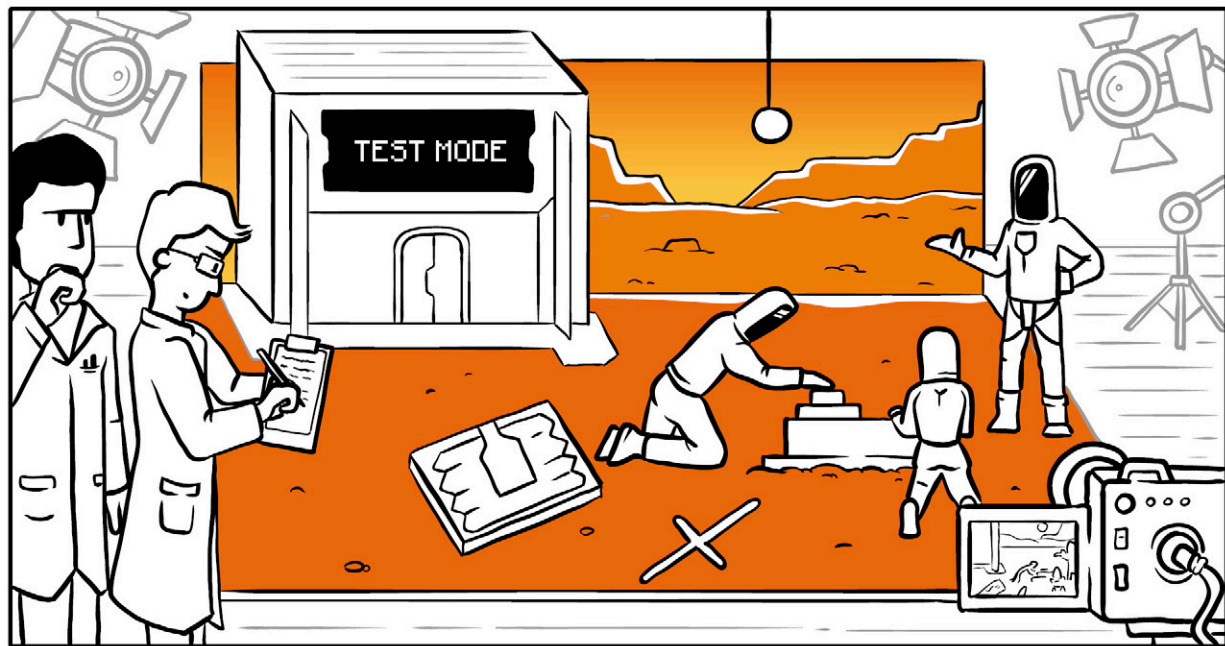
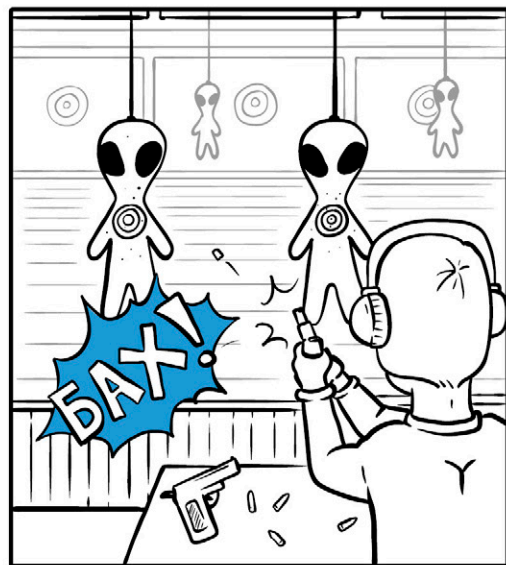
Планета XOD

СОДЕРЖАНИЕ

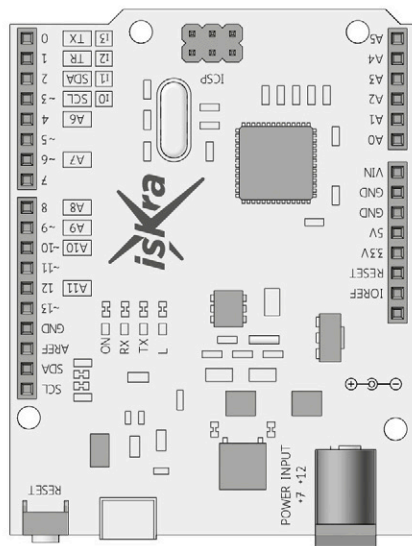
00.2	Что в наборе _ _ _ _ _	06
00.3	Установка и запуск XOD _ _ _ _ _	12
01	Добро пожаловать на борт _ _ _ _ _	16
02	Ноды, пины, инспектор _ _ _ _ _	20

03	Загрузка в Arduino _ _ _ _ _	24
04	Шим и значения пинов _ _ _ _ _	28
05	Арифметика _ _ _ _ _	32
06	Сервопривод _ _ _ _ _	34
07	Датчик света _ _ _ _ _	42
08	Кнопки и flip-flop _ _ _ _ _	48
09	Логические ноды _ _ _ _ _	52
10	Тип данных: пульс _ _ _ _ _	58
11	Debug _ _ _ _ _	70
12	Поиск максимума _ _ _ _ _	74
13	Дисплей _ _ _ _ _	78
14	Вывод переменных _ _ _ _ _	82
15	Нода if-else _ _ _ _ _	86
16	Создание собственных нод _ _ _ _ _	96
17	Нода оповещения _ _ _ _ _	98
18	Последние штрихи _ _ _ _ _	102
19	Новое начало _ _ _ _ _	110

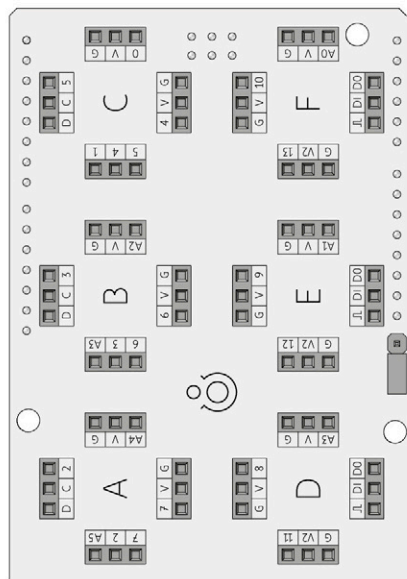




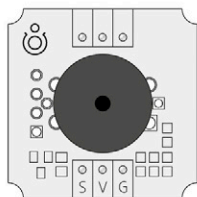
ЧТО В НАБОРЕ



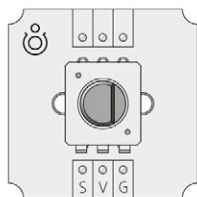
Микроконтроллер (Iskra Neo)



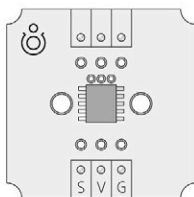
Плата расширения (Troyka Slot Shield)



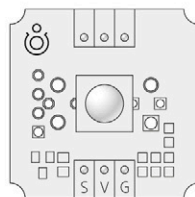
Зуммер



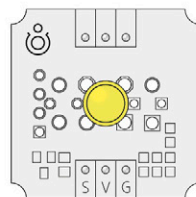
Потенциометр



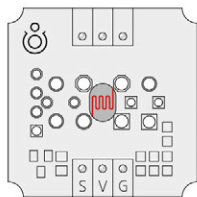
Термометр



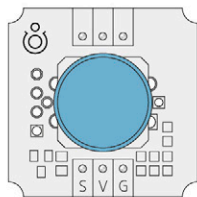
Сверхяркий
светодиод



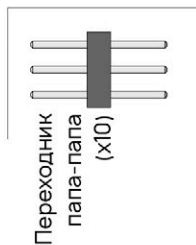
Светодиод (x2)



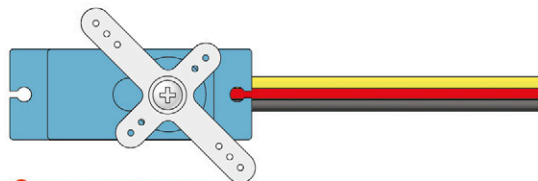
Датчик
освещенности



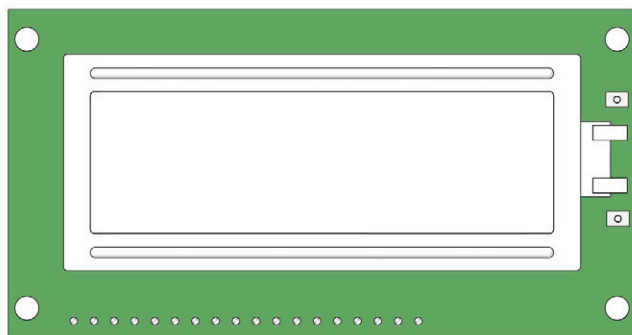
Кнопка
(x2)



Переходник
папа-папа
(x10)

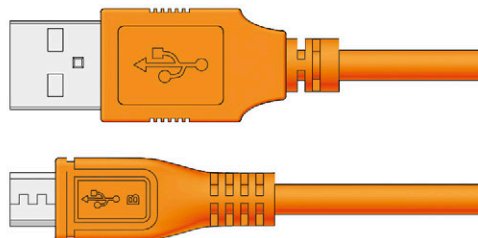


Сервопривод



Дисплей

Micro-USB

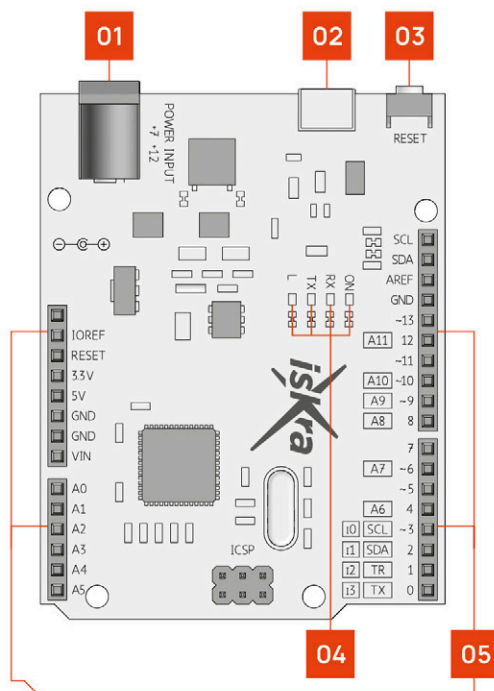


ПЛАТА ISKRA NEO

Iskra Neo — это маленький компьютер, мозг твоего устройства, который может измерять и подавать напряжение. Этого хватает для взаимодействия с окружающим миром: к нему можно подключить разнообразные датчики и приборы. Они будут передавать данные контроллеру, получать от него команды и выполнять их.

Чтобы Iskra понимала, что сообщают датчики и какие команды нужно отдать другим устройствам, для неё нужно написать программу. Это можно сделать разными способами. В этом наборе мы будем использовать визуальный язык программирования XOD.

Iskra Neo совместима с популярным семейством плат Arduino. Поэтому далее в этой книге мы будем называть её просто **Arduino**.



01

Разъём питания

02

Разъём USB

для загрузки кода и питания платы

03

Кнопка перезагрузки (Reset)

04

Светодиоды:

L — светодиод, напрямую подключенный к пину 13. **TX** и **RX** — мигает, когда плата взаимодействует с компьютером

05

Пины

Пины A0—A5 и 0—13

используются для подключения всевозможных внешних устройств: сенсоров, сервоприводов, светодиодов, кнопок и прочих. Они называются GPIO (General Purpose Input and Output — пины ввода и вывода общего назначения), или просто «порты». Они могут выдавать напряжение или получать и измерять его.

Пины A0—A5 и A6—A11

размещены на правой стороне платы, могут измерять точное напряжение в промежутке от 0,0 до 5,0 вольт. Они используются для подключения аналоговых сенсоров: потенциометров, датчиков света и температуры.

Пины 0, 1, 2, 4, 7, 8, 12

могут передавать стабильный сигнал в 5 вольт. А могут не передавать. То есть устройства, которые к ним подключены, имеют только два режима: включено и выключено.

Пины 3, 5, 6, 9, 10, 11, 13

отмечены символом ~ (тильда) на плате. Они могут имитировать аналоговый сигнал. Они переключают напряжение между 0 и 5 вольтами тысячи раз в секунду и, таким образом, регулируют яркость светодиодов, скорость моторов и т. д.

Например

Пин, рядом с которым на плате написано  12, на приём может получать точные значения между 0 и 5 вольтами, а передавать может только либо 0, либо 5 вольт.

Пин  10 отличается от предыдущего тем, что и на выход может подавать любое промежуточное значение. Например, светодиоду это позволяет светить вполсилы. А вот пин  7 может и принимать, и передавать либо 0, либо 5 вольт.

SCL и SDA

Из этих двух пинов состоит шина I²C. Это интерфейс, через который можно подключить много устройств одновременно. В нашем проекте мы будем использовать его для подключения жидкокристаллического дисплея.

GND

Земля — точка с напряжением 0 вольт.

5V и 3.3V

Пины питания с напряжением 5 и 3.3 вольта соответственно.

VIN

Пин подачи напряжения от источника питания от 5 до 12 вольт.

TROYKA SLOT SHIELD

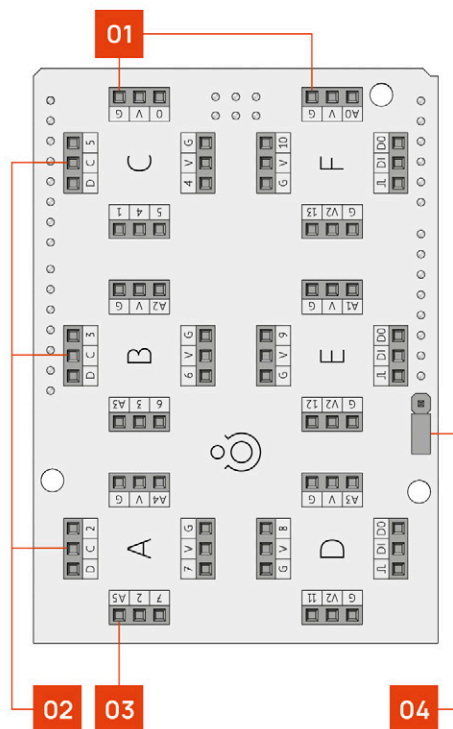
Troyka Slot Shield — плата расширения для Arduino, которая позволяет создать устройство с помощью Тройка-модулей без использования проводов. На ней есть 6 слотов: А, В, С, D, E и F.

Каждый слот состоит из 4 групп пинов. Они имеют 3 линии напряжения: S — сигнал (данные), V — питание и G — земля. Группы пинов скомпонованы так, чтобы к ним можно было подключить Тройка-модули с разными интерфейсами.

Амперка Вики

Подробнее об интерфейсах подключения рассказываем на «Амперка Вики». Это наш справочник, в котором можно узнать ответы на многие вопросы, которые могут возникнуть в процессе.

Её адрес: wiki.amperka.ru. Там можно почитать об Arduino или Troyka Slot Shield и Тройка-модулях.



01

G — пины земли

02

Пины для I²C интерфейса. Используется для работы с модулем ЖК-дисплея

03

Номера пинов аналогичны пинам Arduino

04

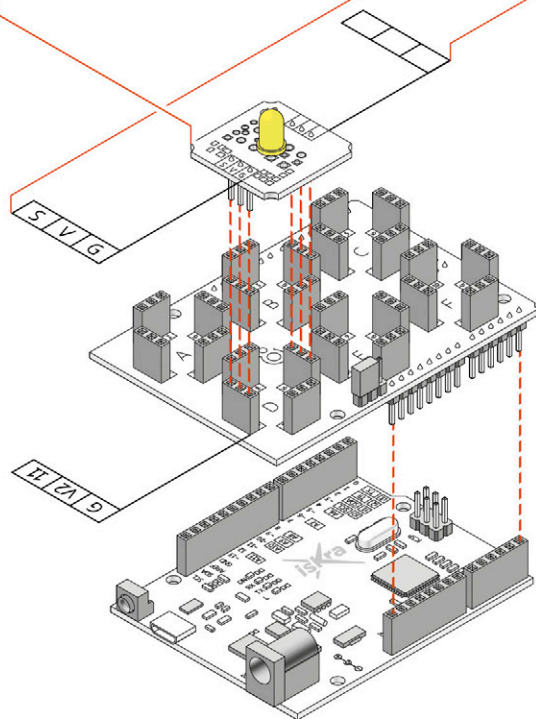
Джампер источника питания. Держи перемычку на пинах 5V и V2, как на картинке

Тройка-модули

Устройства, спроектированные специально для Slot Shield. На их платах уже распаяна вся необходимая микроэлектроника, и они готовы к использованию.

Это «ноги» для подключения. S — сигнал, V — питание и G — земля. S подключается к пину с номером, например A0 или 11.

Если ноги не подписаны, значит, они пустые и используются как крепление.



Соединяем всё вместе

Соедини Troyka Slot Shield и Arduino, как показано на рисунке. Между платой и шилдом не должно остаться лишних ног и пустых пинов.

Подключи USB-кабель к Arduino. На плате должен загореться светодиод ON.

Подключи к Slot Shield модуль со светодиодом. Используй для него пин 11.

После подключения светодиода ничего не произошло. Потому что мы ещё не рассказали Arduino, что с ним делать. Этим и займёмся — переворачивай страницу.

УСТАНОВКА И ЗАПУСК XOD



Для программирования Arduino мы будем использовать XOD IDE¹

Это программа для создания кода с помощью визуальных блоков. Код писать не нужно — достаточно добавить все элементы и правильно их соединить. Поэтому XOD существенно проще освоить, чем другие языки программирования.

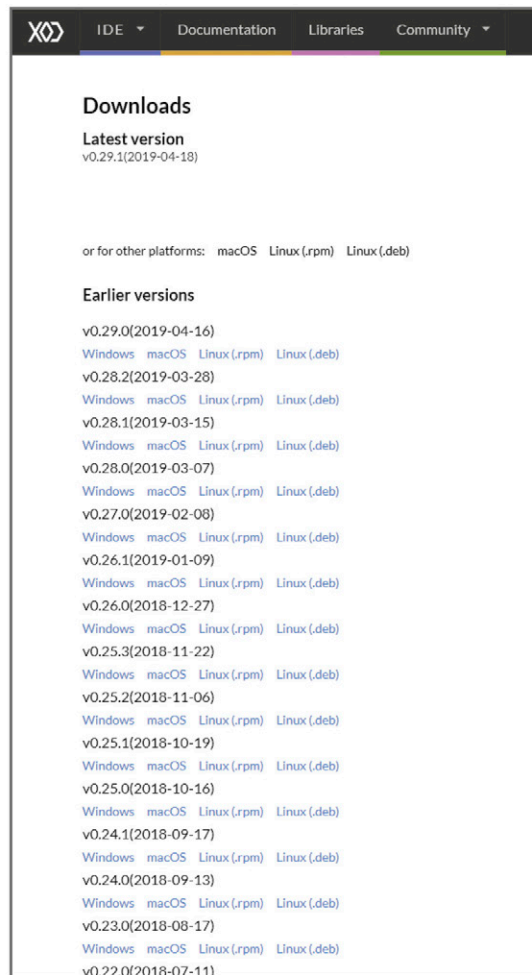
¹ IDE — интегрированная среда разработки. Это программа для написания кода, в которой есть инструменты, упрощающие и ускоряющие процесс (от англ. Integrated Development Environment).

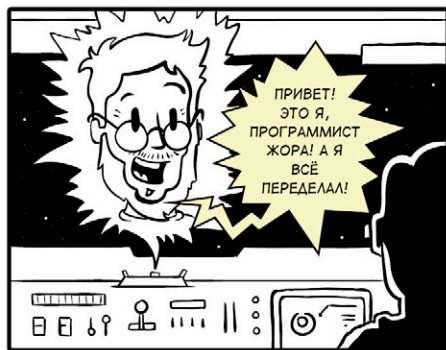
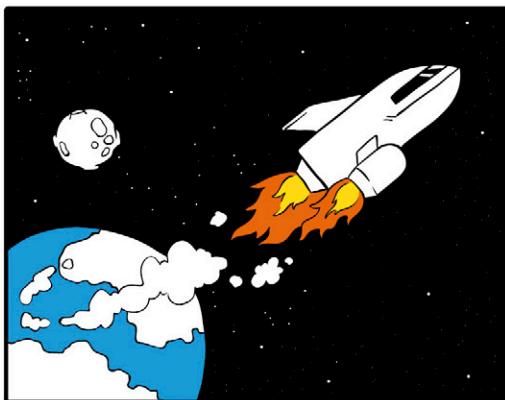
Установим XOD IDE на компьютер

- Открой xod.io/downloads в браузере.
- Скачай файл XOD IDE для своей операционной системы.
- Установи программу:
 - Для Windows:

Открой файл, нажми «Установить» в окне установки и следуй инструкциям на экране. В конце нажми «Завершить».
 - Для MacOS:

Открой файл, перетащи приложение из окна .dmg в папку «Приложения» для установки.
- Наконец, запусти XOD IDE.





3, 2, 1, ПОЕХАЛИ!

Мы хотим не только наступить нашей человеческой ногой на поверхность другой планеты впервые в истории человечества, но и развернуть жизнеспособную колонию. Марс — «младший брат» Земли. Когда-то там существовала жидкая вода и, возможно, даже жизнь. Но теперь это не самое гостеприимное место во Вселенной: большие перепады температур, почти полное отсутствие атмосферы и воды, пылевые бури, затмевающие солнечный свет.

Чтобы справиться со всем этим, нам потребуются надёжные технические средства и слаженная командная работа.

Наш программист, как всегда, «вовремя» обновил программу управления колонией: в последний момент перед вылетом. Благо, у нас есть время, чтобы ознакомиться с ней — на нашем корабле самый современный двигатель, но и с ним мы будем лететь целых три месяца.

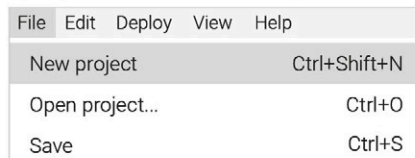


ДОБРО ПОЖАЛОВАТЬ НА БОРТ

**Добро пожаловать в XOD, капитан!
Здесь мы используем не текстовый
код, а визуальные объекты**

01

Нажми сверху в главном меню программы **File** → **New Project**.

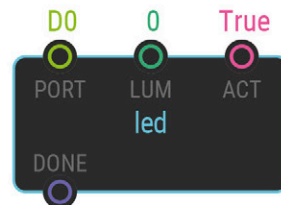


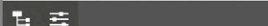
02

Дважды щёлкни на пустое поле и в появившемся окне набери `Led`.

03

В выпадающем списке кликни по строке `xod/common-hardware/Led`, и в поле должна появиться такая штука:





Project Browser



My Project

main

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

xod/debug

xod/gpio

xod/i2c

Inspector

main

Description

main x

led

xod/common-hardware/led

Drives a generic single color dimmable **LED**. The conversion between luminance and **LED** power is biased so that change in brightness on

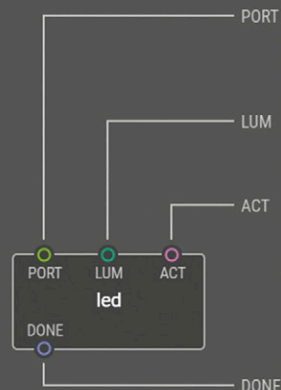
xod/core/fade

Lineary animates an internal value toward target value `TARG` with a rate `RATE`. Use the node to smoothen **LED** switching, motor starting,

02

led**xod/common-hardware/led**

Drives a generic single color dimmat luminance and LED power is biased : range 0...1 is perceived more or less errors: — Invalid port

**ВНИМАНИЕ!**

XOD IDE постоянно
улучшается и меняется.
Не удивляйся, если
иллюстрации в книге
отличаются от интерфейса
на компьютере.

Патч-борд

Большое серое поле — это патч-борд, место для нашей программы — патча¹. В этот патч ты добавил один интересный объект — ноду² `led`, прямо-угольник с цветными кружками. Используя ноды, ты сможешь создавать сложные программы.

Зажми мышкой ноду и перетащи её, куда больше нравится. Положение нод на патч-борде никак не влияет на работу программы, но позволяет сделать её удобнее для восприятия.

Браузер проектов

Слева находится список патчей, сгруппированных по проектам или именам библиотек. Этот список называется `Project Browser`. Сейчас первым элементом в списке проект `My Project`, а под ним название нашего первого патча — `Main`.

Проекты могут состоять из множества патчей. Чтобы не терять результаты, проект надо сохранять. Заодно придумаем название. Нажми в главном меню **File → Save as...** (Файл → Сохранить как...) и в открывшемся окне впиши название своей колонии, например `Phoenix`³. Затем нажми **Save**.

1 Патч — программа (от англ. patch).

2 Нода — узел (от англ. node).

3 Phoenix — область на Марсе.

01

Поиск

Это младший брат Google, его можно вызвать двойным щелчком в пустое поле или клавишей **I**

02

Твой проект

03

Твой патч

Нажми на кнопку с бургером . Здесь можно открыть, переименовать или удалить патч

04

Библиотеки

В библиотеках разные ноды разложены по тематике и назначению

05

Ноды

Ноды — их можно перетащить на рабочее поле мышкой.

Project Browser

- My Project
- main
- xod/bits
- xod/color
- xod/common-hardware
- xod/core
- xod/datetime
- xod/debug
- xod/gpio
- analog-read
- digital-read**
- digital-read-pullup
- digital-write
- pwm-write

Inspector

main

Description

main x

led

xod/common-hardware/led

Drives a generic single color dimmable LED. The conversion between luminance and LED power is biased so that change in brightness on

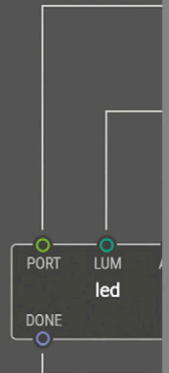
xod/core/fade

Lineary animates an internal value toward target value 'TARG' with a rate 'RATE'. Use the node to smoothen LED switching, motor starting,

led

xod/common-hardware

Drives a generic single luminance and LED range 0...1 is percept errors: — Invalid port



Кстати!

Когда ты вновь откроешь XOD, ты сможешь найти свой сохраненный проект с помощью
File → Open Project...

НОДЫ, ПИНЫ, ИНСПЕКТОР

Нода

Нода — визуальное представление физических устройств, например светодиода, или функций, например арифметических действий.

Led — это простая нода управления одноцветным светодиодом. У этой ноды есть два пина ввода: `PORT` и `LUM`¹.

Пины

Маленькие цветные круги на нодах называются пинами². Они делятся на пины ввода и пины вывода. Пины ввода всегда находятся сверху, а пины вывода — снизу.

Пины используются для передачи данных между нодами. Ноды обрабатывают эти значения и выполняют какое-либо действие или выводят результат. В этом примере нода `led` передаёт значение яркости светодиода нашему микроконтроллеру `Arduino`. Изменять значения пинов можно с помощью боковой панели `Inspector`. Инспектор появляется на левой панели под списком проектов, если выделить какую-нибудь ноду.

¹ `LUM` — светимость, яркость света (от англ. *luminosity*).

² Пин — штифт, штырь (от англ. *pin*). В этом контексте — место для подключения.

01

Port

Это номер пина, к которому физически подключен наш светодиод на плате

02

LUM

Значение светимости светодиода. На этом месте у других устройств будут другие показатели — угол поворота сервопривода, текст на дисплее, нота зуммера и т. д.

03

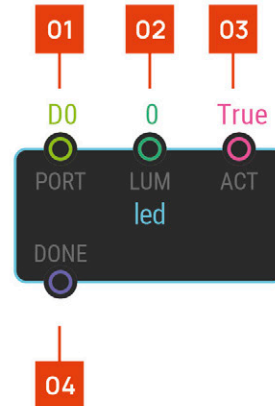
ACT

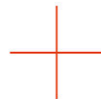
Этот пин регулирует обновление значения. С ним разберёмся позже

04

DONE

Нам не пригодятся.





Как менять значения

- 01** Кликни на ноду, значения которой хочешь изменить. Сейчас это нода `Led`.
- 02** В левой панели внизу появился инспектор.
- 03** Поменяй значение `LUM` с 0 на 1. 1 означает, что светодиод будет светиться на 100 %.
- 04** Проверь значение `PORT` — там должен быть указан пин `D11`, к которому мы подключили светодиод на странице 11.

Пришло время попросить Arduino!
Переворачивай страницу.



Project Browser

My Project

main

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

xod/debug

Inspector

led

xod/common-hardware/led

PORT

D11

LUM

1

ACT

True

DONE

pulse

Label

Description

main x

02

04

03

01

Deployment



03 ЗАГРУЗКА В ARDUINO

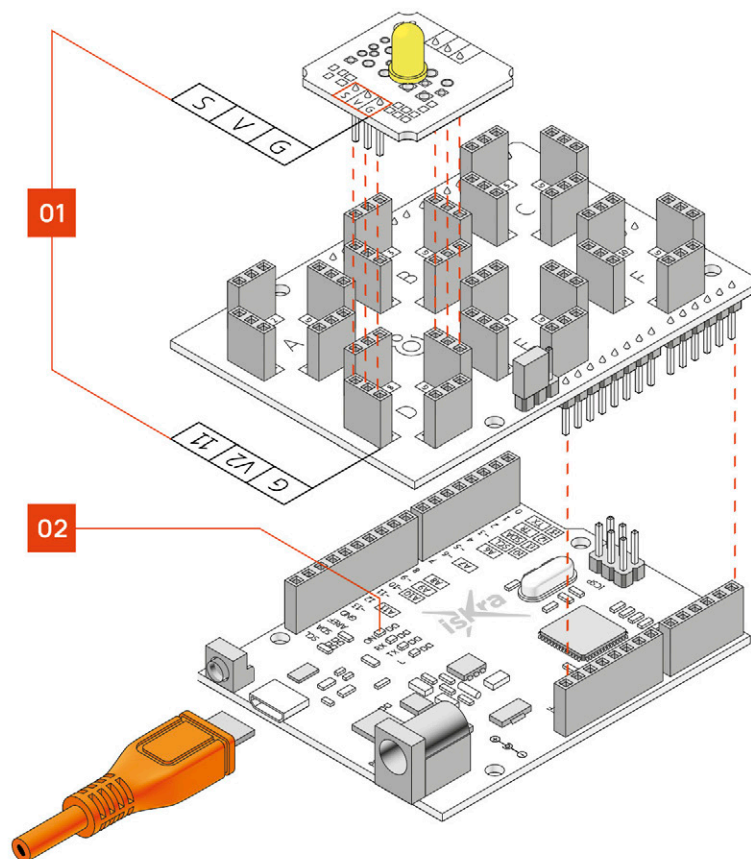
Сейчас мы загрузим наш патч в Arduino

01

Светодиод подключён в сектор D в пин 11.
Следи за положением контактов SVG!
Если неправильно соединить контакты, можно
испортить плату!

02

Проверь подключение Arduino к компьютеру
кабелем USB. На плате должен гореть диод ON.



Важно!

- Разные USB-порты твоего компьютера могут работать по-разному. Поэтому, если при первой загрузке кода в Arduino у тебя возникнет проблема, первым делом попробуй разные USB-порты.
- Всегда давай плате полностью включиться. Подожди 5–10 секунд после подключения к питанию, пока светодиод L не перестанет пульсировать.

Загрузка патча

01 Нажми **Deploy → Upload to Arduino** в главном меню (Переместить → Загрузить в Arduino).

02 Выбери модель платы, у нас это Arduino Leonardo. Во втором окне должен автоматически определиться порт компьютера, к которому она подключена.

03 Нажми **Upload** (Загрузить).

Поскольку мы впервые загружаем патч на нашу плату, XOD предложит скачать необходимые инструменты и настройки.

Arduino dependencies missing
You have to install these package first
Arduino AVR Boards

Download &
Install

Жми на кнопку и подожди пару минут, пока программа скачает нужные файлы. После этого повтори пункты 2—4.

Если всё сделано правильно, светодиод загорится.

04 Щёлкни по этой вкладке, и в открывшемся окне ты сможешь наблюдать за процессом компиляции и загрузки. Если что-то пойдёт не так, программа даст описание ошибки.

Если всё сделано правильно, светодиод загорится.

Восточный, у нас проблемы!

Возникли проблемы с загрузкой? Дай нам об этом знать: amperka.ru/page/service-center. Опиши свои действия, ожидания и результат. Мы поможем.

File Edit Deploy View Help

Project Browser Show Code for Arduino

My Project Upload to Arduino...

Simulate

Connect Serial...

Upgrade Arduino packages ...

01

Upload project to Arduino

Board model:

Arduino Leonardo

Update

Serial port:

COM4

Refresh

☐ Debug after upload

Upload

02

03

04

Deployment

Installer Compiler Uploader Serial Tester

Uploading compiled code to the board...

avrdude: Version 6.3, compiled on Jan 17 2017 at 12:00:53

Copyright (c) 2000-2005 Brian Dean, <http://www.bdmicro.com/>

Copyright (c) 2007-2014 Joerg Wunsch

System wide configuration file is "C:\Users\admin\xod_packages_\packages\arduino\tools\

ШИМ И ЗНАЧЕНИЯ ПИНОВ

Отрегулировать яркость диода или другую величину можно двумя путями. Первый способ — изменить значение в программе. Мы можем ввести любое число между 0 и 1. Например, если введём 0.6, светодиод будет светить на 60 %.

Второй путь — привязать значение светимости диода к выводу другого устройства. Если подключить термометр, led будет гореть тем ярче, чем выше температура. А если потенциометр, мы сможем регулировать свет руками.

Немного теории:

Тип чисел, у которых есть дробная часть, в программировании называют `float` — числа с плавающей точкой. Записать в XOD их можно так:

- `0.03`
- `.03` (без нуля)
- `3e-2` ($3 \times 10^{-2} = 0.03$ — научное обозначение)

ШИМ, широтно-импульсная модуляция — так называется трюк, с помощью которого мы можем имитировать аналоговый сигнал. Диоду нужно 2,5 вольта, чтобы светиться вполсилы, но микроконтроллер может выдавать только 0 или 5 вольт. ШИМ позволяет переключать сигнал между 5 и 0 вольт в нужном соотношении тысячи раз в секунду. Поскольку наш глаз не способен заметить столь частое мерцание, нам кажется, что диод горит наполовину. Именно так работают современные мониторы компьютеров и экраны смартфонов. Пины, которые так умеют, отмечены знаком  на плате.

Собери тестовую схему:

01

Потенциометр

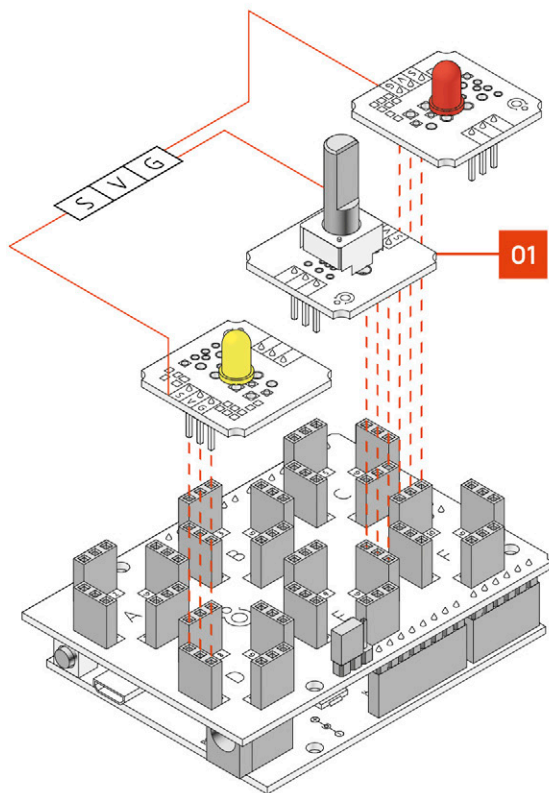
Потенциометр может регулировать яркость светодиодов и другие величины. Поворот ручки регулирует, какое напряжение он отдаёт контроллеру. Подключи его к пину A1 в секторе E.

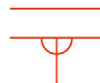
02

Вставь один из светодиодов в пин 10 в секторе F, а второй в пин 11 в секторе D.

03

Возьми в привычку всегда проверять подключение модулей. Если перепутать пины, можно что-нибудь испортить.





01 Выдели ноду `led` и поменяй значение `LUM` на 0.6.

02 С помощью поиска добавь ещё одну ноду `led`.

03 Переименуй её, чтобы не путаться: выдели её и в инспекторе в поле `Label` напиши любое слово, например `Red led`.

04 Добавь ноду для потенциометра. В XOD она называется `pot`.

05 Соедини пин потенциометра `VAL`¹ с пином `LUM Red led`.

06 Не забудь проверить значения пинов `PORT` на нодах. Они должны совпадать с номерами пинов, к которым подключены устройства. Загрузи патч в Arduino.

Теперь покрути ручку потенциометра в разные стороны. Один светодиод будет менять свою яркость. Обрати внимание на другой — он горит заметно менее ярко, чем когда его значение `LUM` было 1.

¹ `VAL` — значение (от англ. `value`).

04

pot

xod/common-hardware/pot

Reads values from a generic potentiometer. Basically a thin wrapper around 'analog-input' for a simpler learning experience. Possible errors:

xod-dev/servo/example-00-pot

pot

xod/common-hardware/pot

Reads values from a generic wrapper around 'analog-input'. Possible errors: — Invalid port

01

05

03

02

АРИФМЕТИКА

Вспомнить арифметику никогда не поздно! Даже если ты летишь с огромной скоростью на другую планету в консервной банке, а от вакуума тебя защищает пара сантиметров обшивки корабля. По крайней мере, так считает наш программист Жора.

В библиотеках  `xod/core` и  `xod/math`



хранятся ноды для разных математических и тригонометрических расчётов. Для самых простых вычислений есть ноды `multiply` (умножить), `add` (прибавить), `subtract` (вычесть), и `divide` (разделить).

Нода `multiply` (умножить) умножает числа на пинах ввода (`IN1`, `IN2`) и передаёт результат на пин вывода.

01

Добавь ноду `multiply`.

02

Соедини её левый пин ввода с пином потенциометра `VAL`.

03

Задай ноде `multiply` значение правого пина. Например, 2.

04

Не забудь проверить пины подключения модулей!

05

Загрузи патч в Arduino.

06

Поверни ручку потенциометра. Светодиод на пине 10 будет максимально ярким уже на половине оборота. Это происходит из-за того, что нода `multiply` умножает введённые значения на 2 и передаёт их на свой пин вывода. Попробуй другие арифметические ноды.

Project Browser

My Project

main

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

xod/debug

Inspector

multiply

xod/core/multiply

IN1 linked

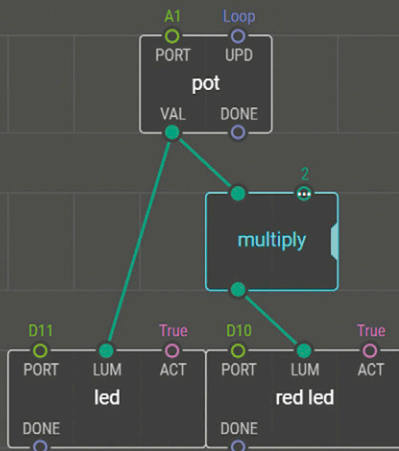
IN2 2

OUT computed

Label

Description

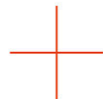
main x



Кстати!

Чтобы не потерять свои труды, не забывай сохранять их. Это можно сделать через **File → Save**. Если вдруг забудешь сохранить проект, при выходе программа напомнит тебе об этом.

Deployment



СЕРВОПРИВОД

Сервопривод — необходимый прибор для нашей колонии

Любой выход на поверхность Марса сопряжён с рисками и не всегда возможен. Системы жизнеобеспечения защитного костюма могут не спасти от сильной бури или зимней ночи, когда температура может опуститься ниже -100°C . С помощью сервопривода мы сможем автоматизировать движение панелей солнечных батарей и не настраивать их руками. Проверим его.



- 01 Выбери ноду `servo`.
- 02 Задай значение пина `PORT = D10`.
- 03 Соедини пин ввода `VAL` с пином вывода потенциометра или другой ноды, передающей значения в диапазоне 0–1.
- 04 Загрузи патч в Arduino.
- 05 Вращай ручку потенциометра в разные стороны и смотри, как сервопривод следует за ней!



В XOD нода для управления сервоприводом называется `servo`. Ты найдёшь её в библиотеке `xod-dev/servo/servo`. Пин `VAL` принимает значения от 0 до 1 и поворачивает сервопривод на угол от 0 до 180°.

Кстати!

В XOD очень много нод. Чтобы самостоятельно понять назначение каждой ноды, прочитай страницу справки. Посмотреть её можно двумя способами:

- 01 Выбери нужную ноду на патче или в списке. Открой панель справки (Helpbar) через **View → Toggle Helpbar** или нажми клавишу H на клавиатуре в английской раскладке.
- 02 Щёлкни по значку книги в **Inspector** или **Project Browser**, чтобы открыть ту же справку в веб-браузере.



Project Browser



My Project

main

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

xod/debug

Inspector

servo

xod-dev/servo/servo

PORT

D10

VAL

linked

ACT

True

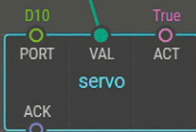
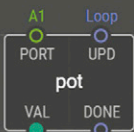
ACK

pulse

Label

Description

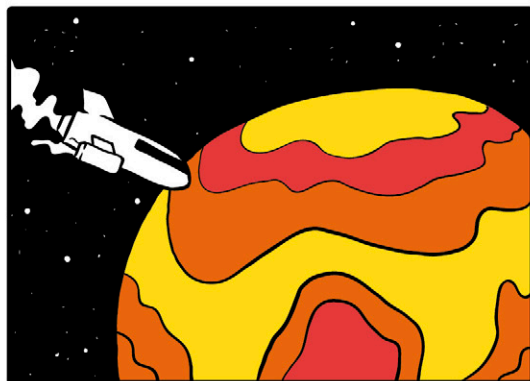
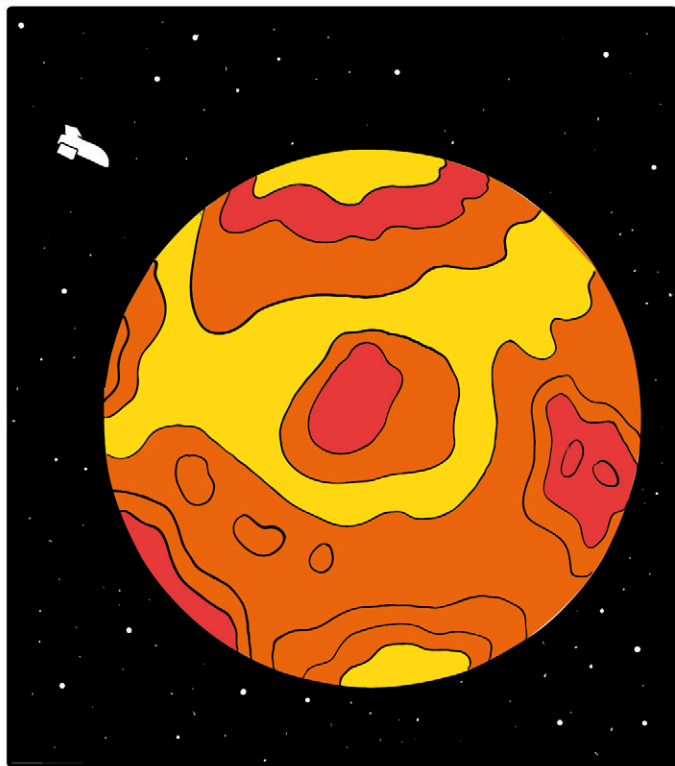
main x



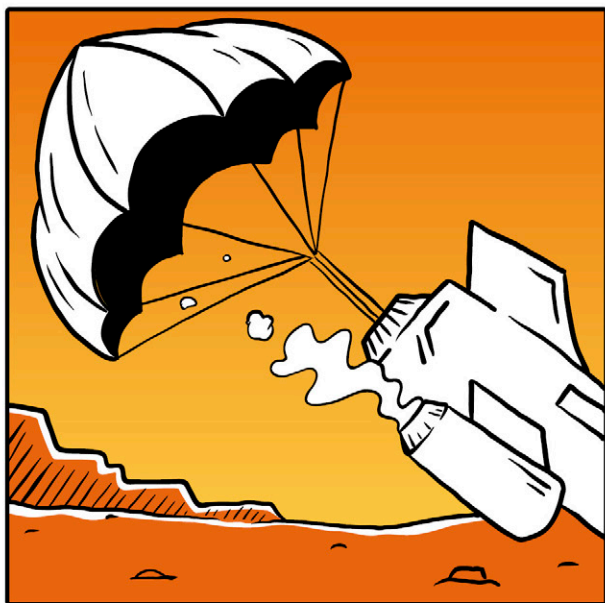
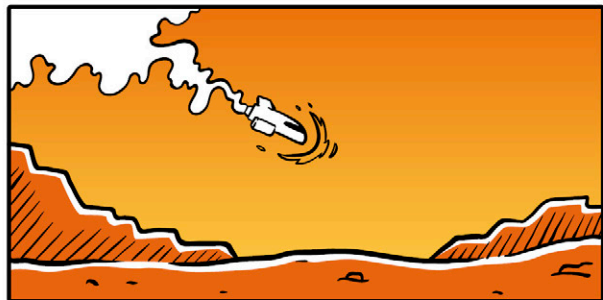
Кстати!

Чтобы удалить ноду или соединительный «провод», выдели их мышкой и нажми на клавиатуре кнопку **Delete** или **Backspace**.

Deployment







МАРС НАШ!

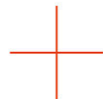
Что ж, мы всё-таки сели. Все остались целы и невредимы. Если бы наш программист не смог исправить ошибку, из-за которой не отстреливались парашюты, нам пришёл бы конец. Атмосфера Марса крайне разрежена — давление здесь в 160 раз меньше земного. Чтобы затормозить перед посадкой, нужно использовать и двигатели, и парашюты.

Мы приземлились на плато Солнца, рядом озеро Феникс. Озера с лебедями здесь, конечно

же, нет. Это вулканическое плато, но название красивое. Мы находимся достаточно близко к экватору. По меркам Марса здесь не очень холодно, особенно летом: примерно $+5^{\circ}$ днём и около -60° ночью.

Мы — первые люди в истории, которые видят Марс собственными глазами. Первые люди в истории, которые ступили на поверхность другой планеты. Давайте разгружать корабль, очень хочется есть.





ДАТЧИК СВЕТА

Наш первый сол¹ на Марсе обернулся суетой и неразберихой

У нас есть энергия в аккумуляторах корабля, но это неприкосновенный запас на случай непредвиденных обстоятельств. Поэтому нам нужно поторапливаться с установкой солнечной батареи. К сожалению, сервопривод, который должен был поворачивать её к свету, разбился при посадке. Придётся ворочать её руками. Для начала проверим, уцелела ли сама батарея.

1 Сол — марсианские сутки, составляют 24 часа 40 минут.



Собери схему:

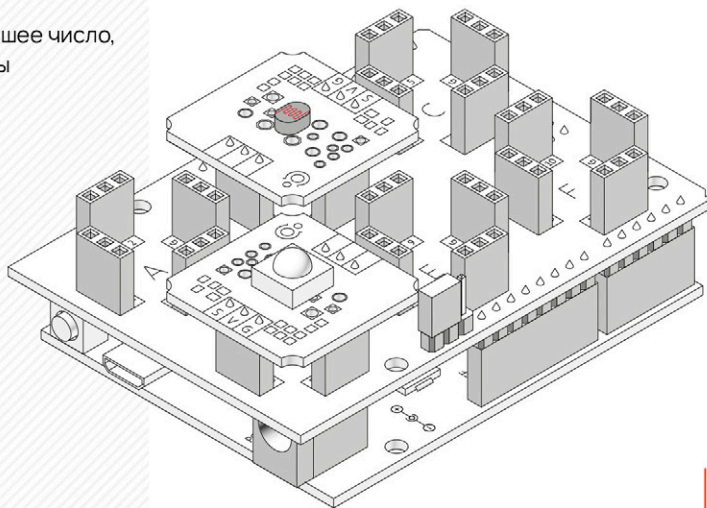
Подключи датчик освещённости в пин A2, а сверхъяркий светодиод в пин 11.

Не направляй сверхъяркий светодиод в глаза.
Он действительно яркий.

Датчик освещённости — важный элемент управления солнечной панелью. С помощью него можно определить, насколько яркий свет падает на батарею, и повернуть её так, чтобы получать максимум энергии. Датчик состоит из фоторезистора — его сопротивление зависит от интенсивности света.

На свой пин VAL он подаёт тем большее число, чем меньше света получает. И чтобы мерить не «темноту», а освещённость, мы перевернём его значения.

В этом нам поможет нода map¹. Она умеет сопоставлять два разных ряда значений.



¹ Map — карта (англ.).

В этом контексте — отображение.



01 Для фоторезистора используем ноду `analogSensor`.
Добавь её на патч.

В ХОД нет отдельной ноды для датчика света, поскольку он примитивен и работает так же, как многие другие сенсоры.

02 Добавь ноду `map`. Она живёт в библиотеке `map` **xod/core**.

03 Подключи пин сенсора VAL к пину X ноды `map`. Это пин для исходной величины, которую будем преобразовывать.

04 `Smin` и `Smax` — диапазон значений, которые мы ожидаем получить от сенсора. У нашего датчика это значения от нуля до единицы. 0 означает, что датчик полностью освещён ярким светом, а 1 — что датчик в полной темноте.

05 `Tmin` и `Tmax` — тот диапазон значений, в который мы хотим превратить исходный. Теперь, когда датчик будет в полной темноте, нода `map` будет передавать дальше значение близкое к нулю. А на свету нода `map` передаст именно освещённость.

Трюк: поставь здесь `Tmin = 1` и `Tmax = 0`. Это «перевернёт» показания датчика и превратит тьму в свет.

06 Подключи нижний пин `map` к VAL пину светодиода и загрузи патч в Arduino.

Попробуй закрыть свет фотодатчику.
Наблюдай за светодиодом.



Project Browser



My Project

main

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

xod/debug

Inspector

map

xod/math/map

X

linked

Smin

0

04

Smax

1

05

Tmin

1

Tmax

0

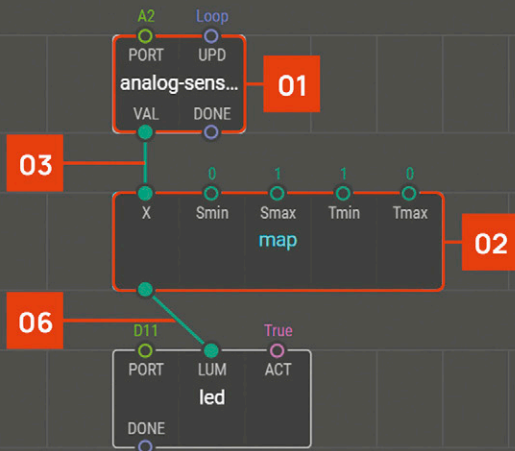
OUT

computed

Label

Description

main x



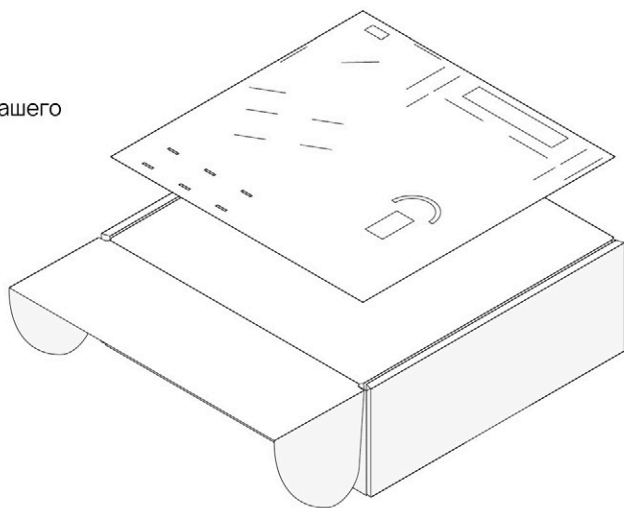
Deployment

ФУНДАМЕНТ

**Колония сама себя
не построит. Разгрузимся
и разметим площадку
под будущие строения.**

01

Достань всё из коробки и налей
на крышку наклейку с разметкой нашего
лагеря, как показано на рисунке.

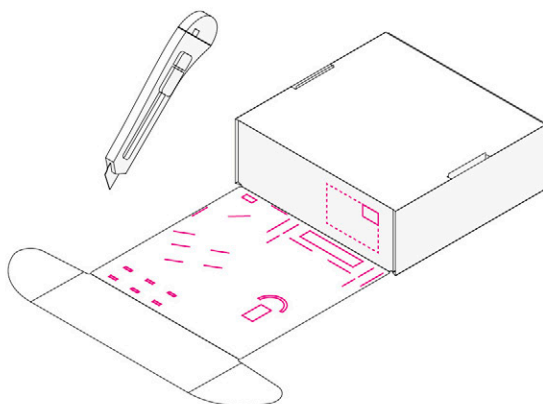


02 Открой и переверни коробку, положи её перед собой как на рисунке. Так будет гораздо удобнее.

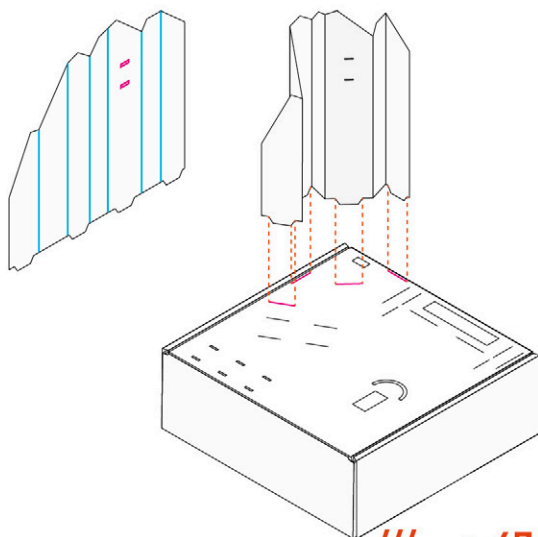
03 Возьми канцелярский нож и аккуратно сделай **отверстия** и **разрезы** на крышке коробки и задней стенке.

Не режь ножом поверх стола!
Подложи кусок картона или доску, которую не жалко.

Если ты никогда не пользовался канцелярским ножом, попроси помощи у коллеги.



04 Согни скалу по **линиям** и вставь ее в прорези в коробке, как показано на рисунке.

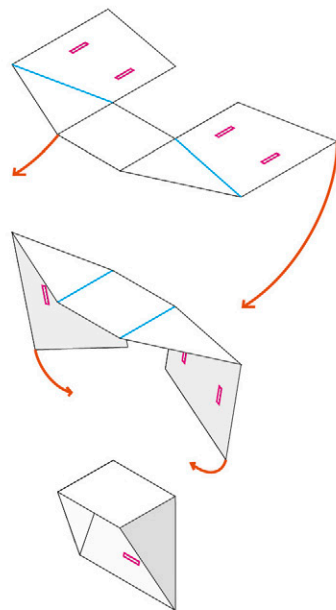


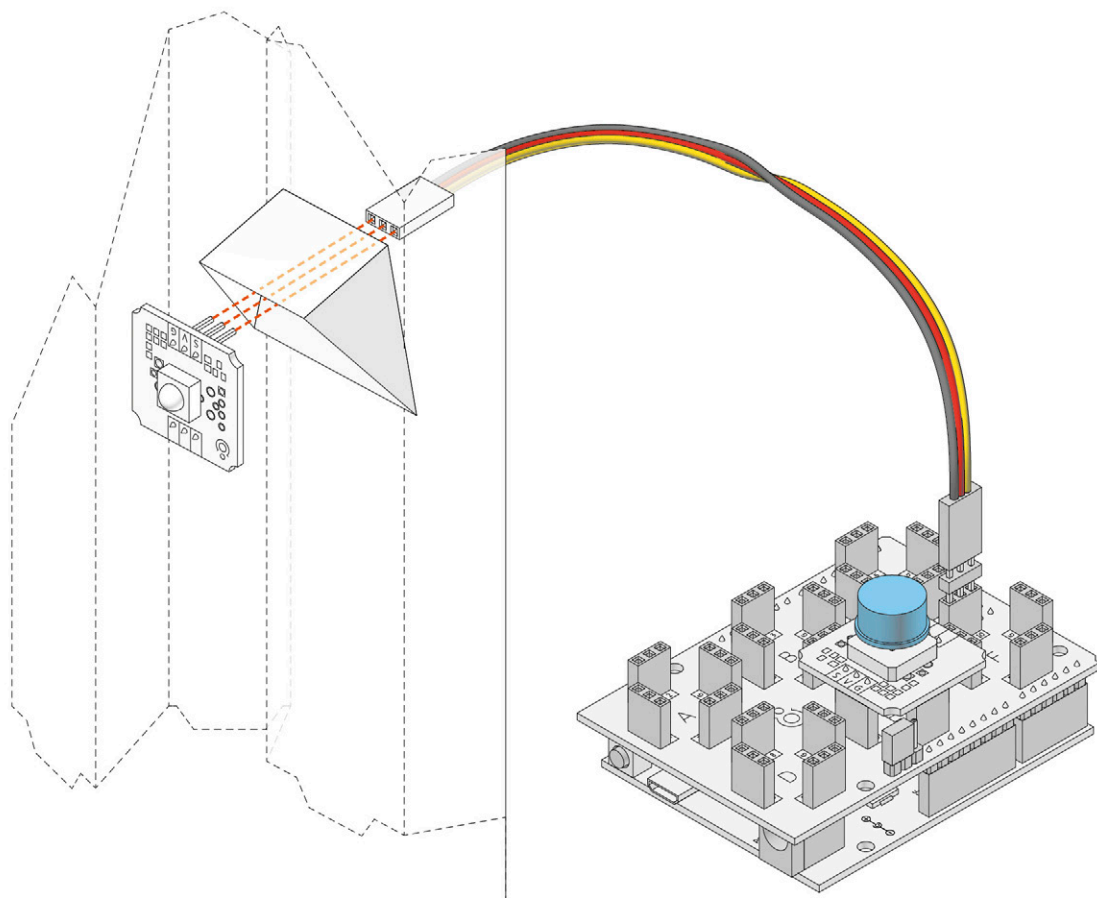
КНОПКИ И FLIP-FLOP



Пока мы возились с солнечной батареей, начало темнеть. У нас ещё много дел: из-за жёсткой посадки наши пожитки разбросало по округе и нам приходится их собирать. Установим прожектор, чтобы в тёмное время суток он освещал округу.

- 01 Достань козырёк для прожектора и согни его по линиям как на рисунке.
- 02 Возьми сверхяркий светодиод и вставь его в прорези: сначала в козырёк, потом в скалу.
- 03 Подключи шлейф одним концом к светодиоду, а другой — к плате в пин 10. Используй переходник папа-папа.
- 04 Подключи кнопку в пин 12.







- 01 Открой новый патч (**File → New Patch**). Предыдущий нам ещё пригодится.
- 02 Достань ноды `led` и `button`. Обе живут в `xod/common-hardware`.
- 03 Ноду `flip-flop` можно найти в библиотеке  `xod/core` или с помощью поиска.

Наша кнопка пропускает ток, только когда зажата. Поэтому нам потребуется ещё один элемент — `flip-flop`¹. `Flip-flop` действует как виртуальный переключатель. Он переключается, когда получает импульс на пин `TGL`. С его помощью мы сможем включать и выключать прожектор.

- 04 Соедини пины `PRS`² и `TGL`³, пины `MEM`⁴ и `VAL`, как на рисунке.
- 05 Проверь номера портов и загрузи патч в Arduino.
- 06 Нажми кнопку. Если всё правильно — прожектор будет включаться и выключаться по нажатию кнопки. Попробуй убрать из схемы `flip-flop` и посмотреть, что изменится.

1 `Flip-flop` — перевертыш, переворот (англ.).
2 `PRS` — нажимать (от англ. *press*).
3 `TGL` — переключать (от англ. *toggle*).
4 `MEM` — память (от англ. *memory*). В данном контексте — запомненное значение, как на калькуляторе.



Project Browser

My Project

main

new

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

Inspector

button

xod/common-hardware/button

PORT D12

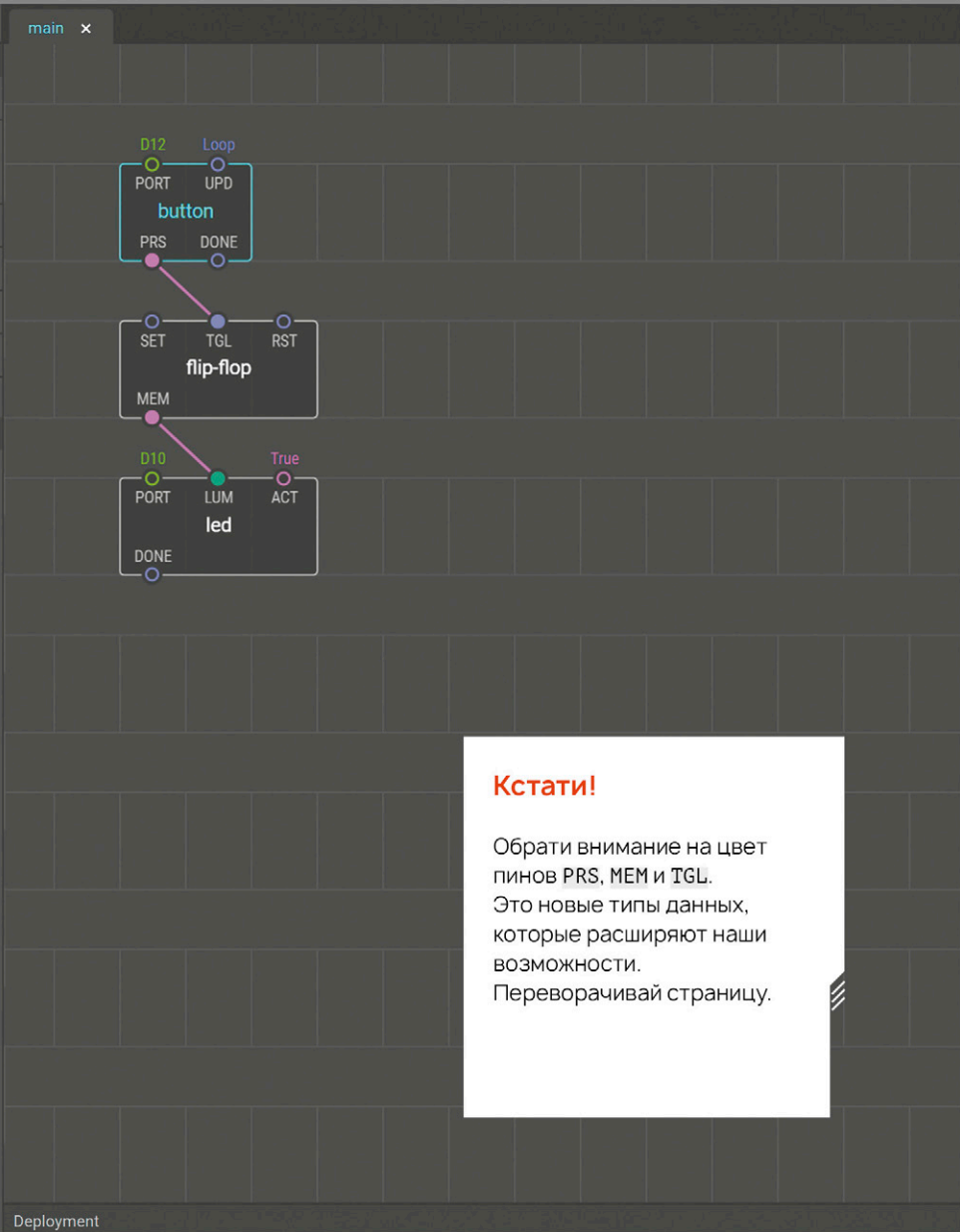
UPD Continuously

PRS False

DONE pulse

Label

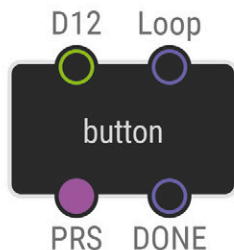
Description



ЛОГИЧЕСКИЕ НОДЫ



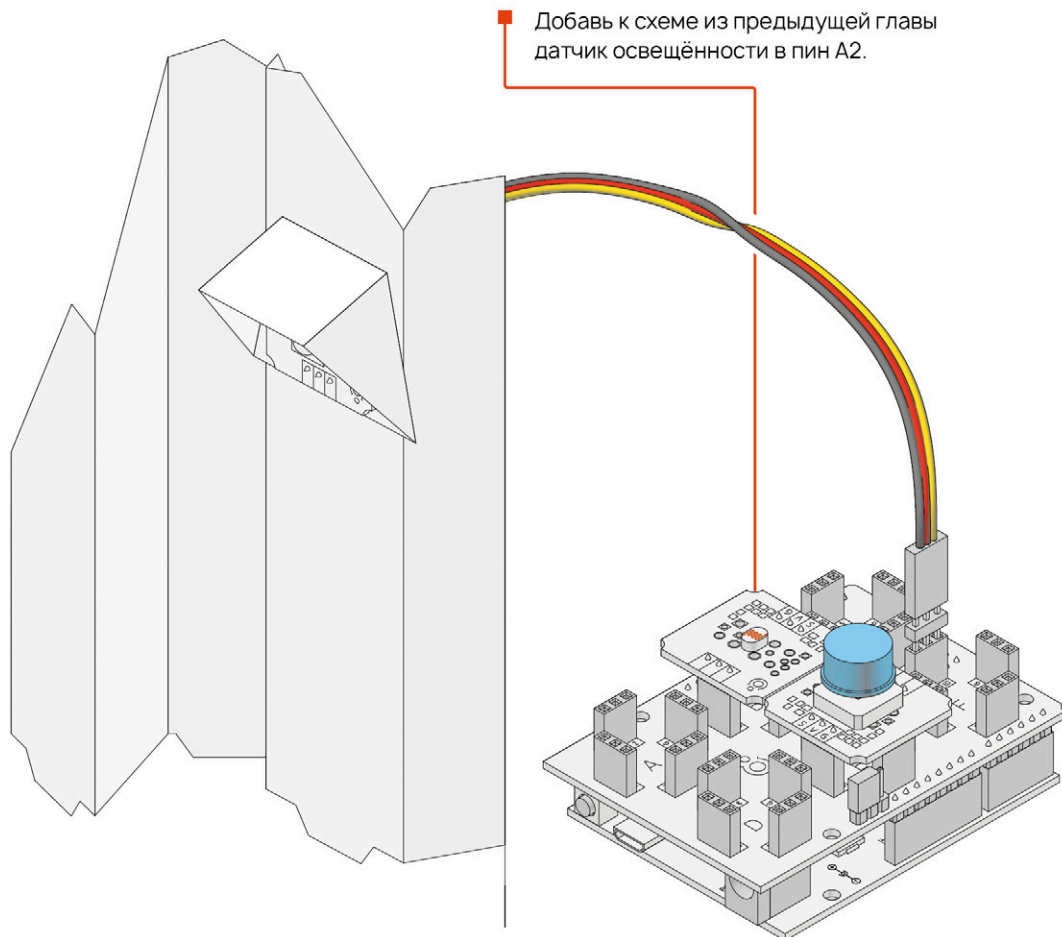
Было бы здорово, если бы наш прожектор включался сам, когда наступает темнота. Но при этом нужно оставить возможность включать его нажатием кнопки. Для этого создадим логическую схему.



Нода `button` имеет розовый выходной пин `PRS`. Этот пин возвращает новый тип данных `boolean` — логические.

Данные этого типа могут принимать только два значения: истина (`true`) или ложь (`false`). Нода `button` возвращает значение `true`, когда кнопка нажата, и `false`, когда не нажата.





Boolean — это круто

Их можно использовать для построения сложных логических систем. В XOD есть много нод для этих целей — они расположены в `xod/core`.

`and` (и) возвращает `true` (истина), если оба входных пина (`IN1` и `IN2`) `true`;

`or` (или) возвращает `true`, если хотя бы один из входных пинов получает `true`;

`xor` (исключающее или) возвращает `true`, если только один из пинов получает `true`;

`not` (нет) инвертирует входное значение;

`nand` (и + нет) инвертирует результат `and`;

`nor` (или + нет) инвертирует результат `or`.

Помимо ноды `or`, нужно добавить условие «темноты», при котором прожектор будет включаться.

Для сравнения чисел можно использовать ноды `greater` (больше) и `less` (меньше) из `xod/core`. Эти ноды сравнивают значение, полученное на пин `IN1`, со значением на пине `IN2`. Если условие выполняется, передают дальше `true`.

Соберем патч

01 Собери патч. Часть этого патча мы собирали в предыдущем опыте, а часть можно скопировать из патча главы 09. Для этого открой вкладку с тем патчем, выдели все элементы и нажми в главном меню **Edit → Copy**, потом перейди на вкладку текущего патча и нажми **Edit → Paste**. Пока все скопированные элементы выделены, их можно перетащить в удобное место.

02 Лишние связи между нодами и `led` можно удалить, выделив их и нажав клавишу `backspace`.

03 Добавь ноды `or` и `less` и соедини их как на рисунке.

04 Выбери ноду `less` и впиши ей значение 0.5. Когда начнёт смеркаться, показания датчика, переданные через ноду `map`, упадут ниже 0.5. Нода `less` «замечит» это и передаст дальше значение `true`, что и включит прожектор.

05 Загрузи патч в Arduino. Понаблюдай за поведением прожектора, нажимая на кнопку и закрывая фотодатчик от света. Попробуй заменить ноду `or` на `and` или `xor`.



Project Browser



My Project



main

new

xod/bits



xod/color



xod/common-hardware



xod/core



xod/datetime



Inspector



less



xod/core/less



IN1

linked

IN2

0.5

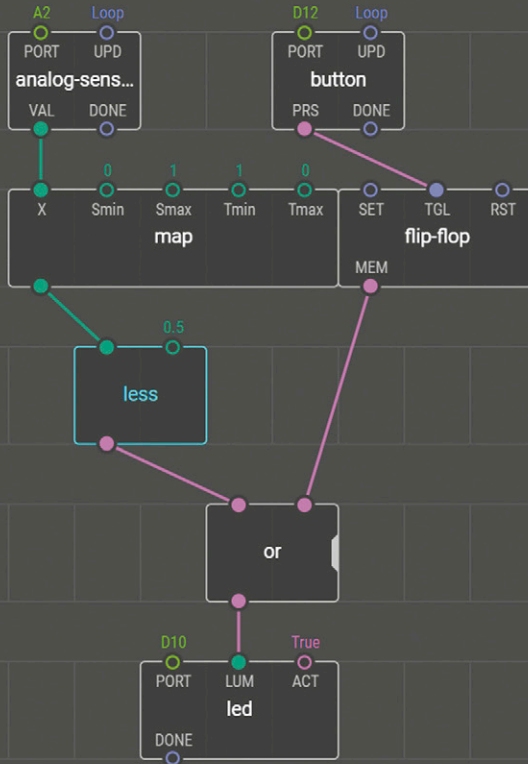
OUT

computed

Label

Description

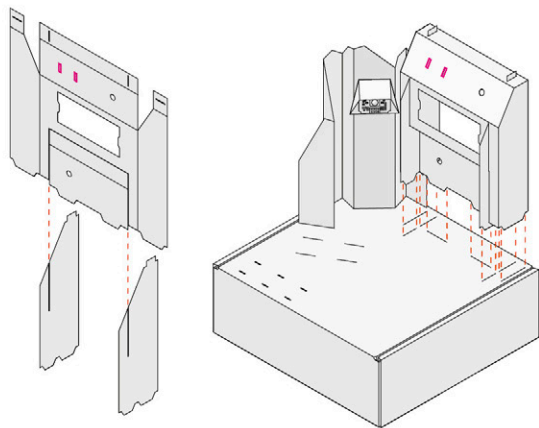
main x



Deployment

НАШ НОВЫЙ ДОМ

Уже несколько месяцев мы живём втроём в тесноте нашего корабля. На каждый килограмм груза, который мы взяли с собой, пришлось несколько десятков килограммов топлива, что влетело в копеечку нашим спонсорам. Поэтому основной кампус нашей колонии тоже не очень просторный. Но там хотя бы есть нормальный туалет.



01

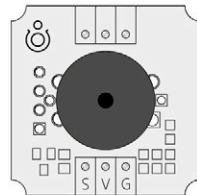
Собери кампус и поставь его на фундамент.

02

Вставь светодиоды в кампус и подключи их шлейфами к плате.

03

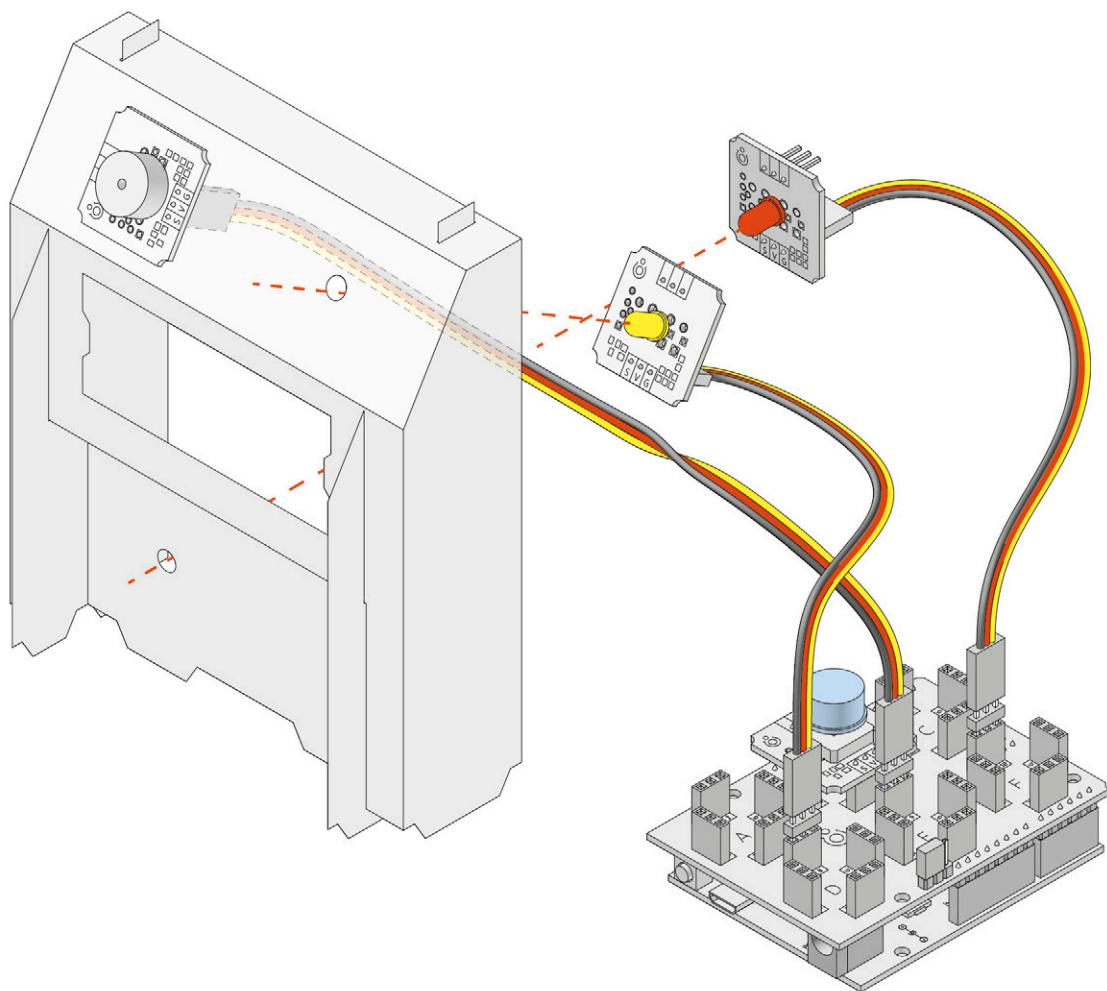
Это называется зуммер¹ — он умеет пищать. Установи его в **отверстия** в крыше кампуса и подключи в пин 9 с помощью шлейфа и переходника.



04

Подключи кнопку в пин 6.

1 Зуммер — жужжалка (от нем. summer).



ТИП ДАННЫХ: ПУЛЬС

Земля уже знает, что мы живы — вокруг Марса полно спутников. Но нам так и не удалось восстановить связь с ЦУПом¹. Кроме того, нужно найти способ передать сигнал в случае чрезвычайной ситуации. Соберём систему из мигающих фонарей, которые будет видно со спутника.

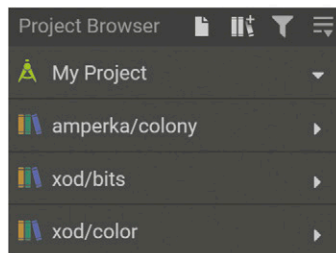
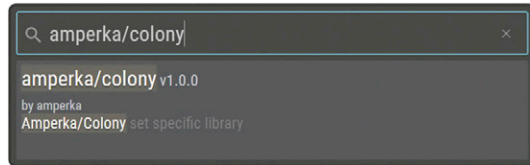
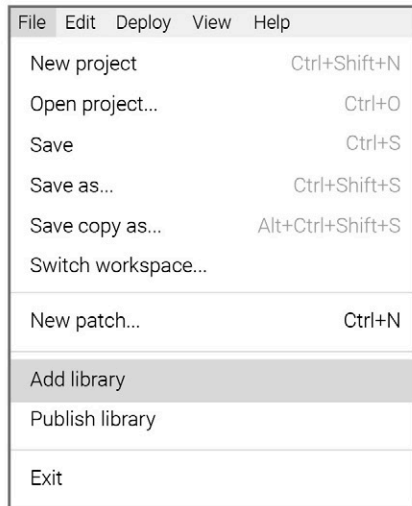
Для подачи и получения сигналов на ногах есть фиолетовые пины `UPD`², `TGL`, `TICK`³ и другие. Они используют тип данных `pulse`⁴. Все значения между нодами передаются непрерывно, даже если они не изменились. `Pulse` позволяет управлять этим. С помощью него мы можем указать, как часто нода будет обновлять своё значение, например заставить светодиод мигать.

Нода `clock`⁵ посылает импульс раз в указанный промежуток времени. Если соединить её с нодой `flip-flop`, мы сможем получить мигалку.

-
- 1 ЦУП — Центр Управления Полётами.
 - 2 `UPD` — обновление (от англ. `update`).
 - 3 `TICK` — идти, тикать (англ.).
 - 4 `Pulse` — импульс, пульс (англ.).
 - 5 `clock` — часы (англ.).



Для работы с зуммером нам нужна нестандартная библиотека. Чтобы скачать её из интернета, нажми **File → Add Library**. В открывшемся окне напиши: `amperka/colony`. Дважды щёлкни по выпавшему варианту, и библиотека скачается сама. Другие библиотеки можно найти на сайте xod.io/libs.



01 Вытащи ноду `clock` и поставь значение пина `IVAL` = `0.3`. Это то, как часто будут перемигиваться диоды — раз в 0.3 секунды.

02 Ноду `alarm` можно найти в библиотеке  **amperka/colony**. Установи значение `DUTY` = `0.5`. Это пауза между «пищаниями» зуммера в секундах. Пин `T` — длина сигнала. `FREQ` — частота звука, то есть нота.

03 Соедини ноды и проверь значение портов.

04 Попробуем расшифровать получившийся алгоритм. Удобно начинать «читать» его снизу.

05 Каждый `led` подключён к `and`. Это значит, что для того, чтобы светодиод включился, должно совпасть два условия.

- Условие 1: мы включили тревогу. То есть нажали кнопку и переключили выходное значение `flip-flop 1` в `true`, которое и передалось на `and`-ноды.

- Условие 2: должно приходить значение `true` от `flip-flop 2`, которым управляет нода `clock`. От часов `true` приходит в течение 0.3 сек., потом 0.3 сек. приходит `false`.

- Нода `not` нужна, чтобы светодиоды мигали не одновременно, а по очереди.

06 Загрузи патч в Arduino и проверь, всё ли верно работает.

07 Теперь поменяй ноду `alarm` на ноду `X`. Загружай патч в Arduino, нажимай на кнопку, чтобы включить тревогу, и переверачивай страницу.

PIN

D9

FRQ1

440

FRQ2

660

T

0.5

DUTY

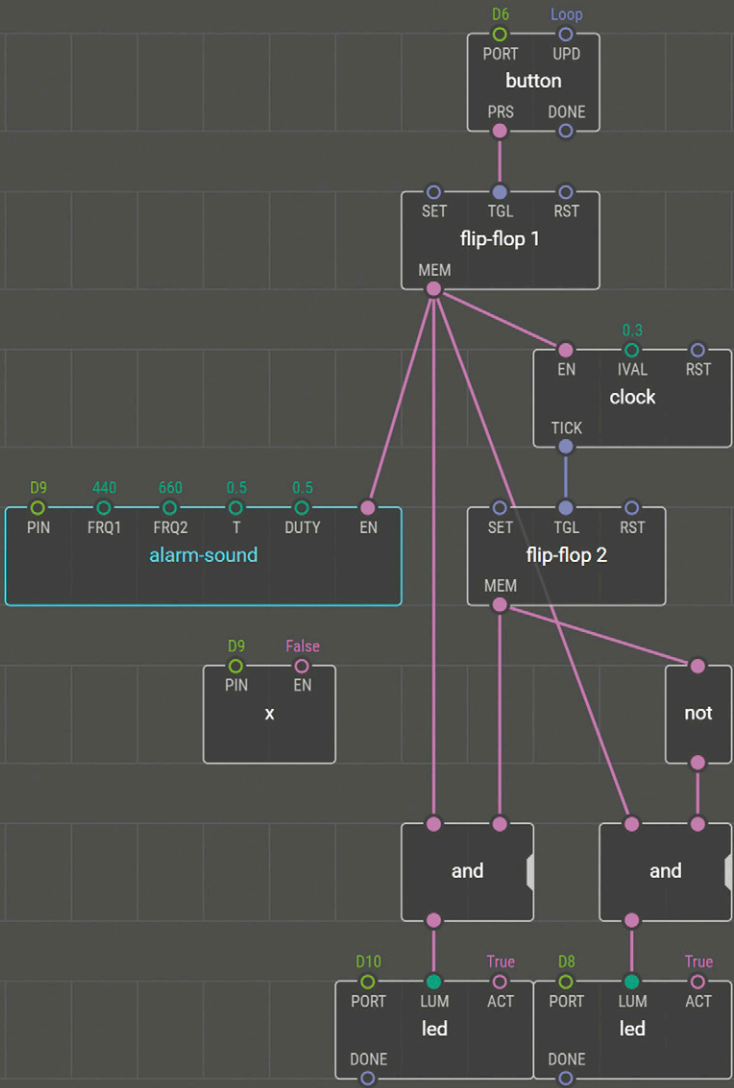
0.5

EN

linked

Label

Description



СОЛ 12

КЭП, ПРОСНИСЬ!
ТАМ ЧТО-ТО ЕСТЬ!

ЧТО?
ГДЕ Я?!

МЫ НА МАРСЕ!
И МЫ ТУТ НЕ ОДНИ!!!

ЧТО?
ТЫ В СВОЁМ
УМЕ?

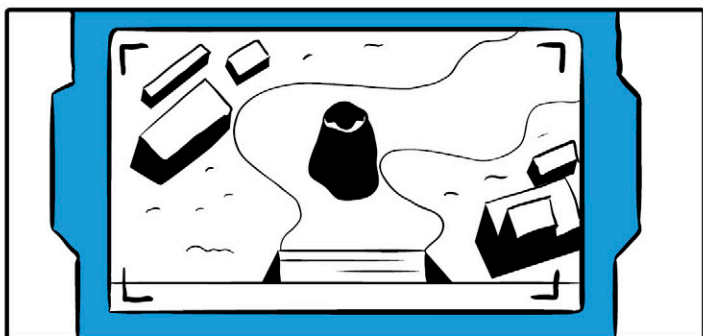
ВОТ ЗДЕСЬ БЫЛО КАКОЕ-ТО ДВИЖЕНИЕ!
КАКОЙ-ТО СИЛУЭТ...

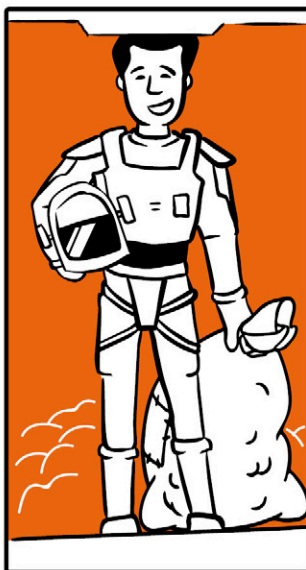
ТЕБЕ НЕ
ПОКАЗАЛОСЬ?

И ЧТО ВЫ
ВООБЩЕ
ДЕЛАЛИ ТУТ
ВДВОЁМ?

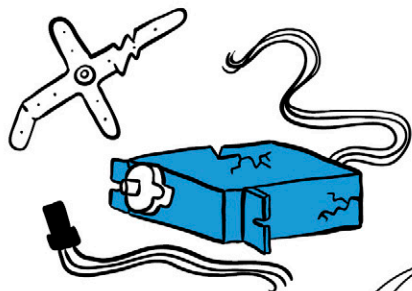
ВОТ, СМОТРИТЕ!

ДЫРУ МНЕ В СКАФАНДР, ОНО ПРИБЛИЖАЕТСЯ!





ЭТО НАШ МОТОР ДЛЯ СОЛНЕЧНОЙ
БАТАРЕИ. ПОСТРАДАЛ ПРИ ПОСАДКЕ.
НО ПОСТОЙ, КАК ТЫ...



О, ЭТО Я МИГОМ ПОЧИНЮ!

ДАЙ-КА МНЕ
ЭТОТ КЛЮЧ...

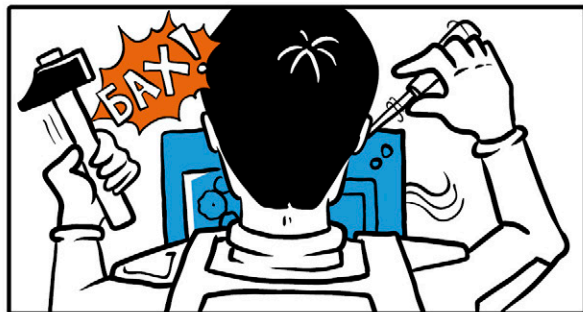
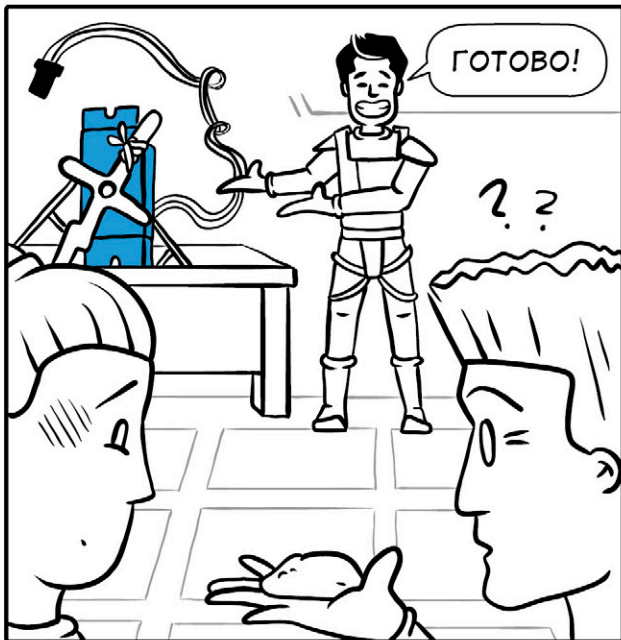


МНЕ НУЖЕН МОЛОТОК...

...И НЕМНОГО КАРТОШКИ



ГОТОВО!



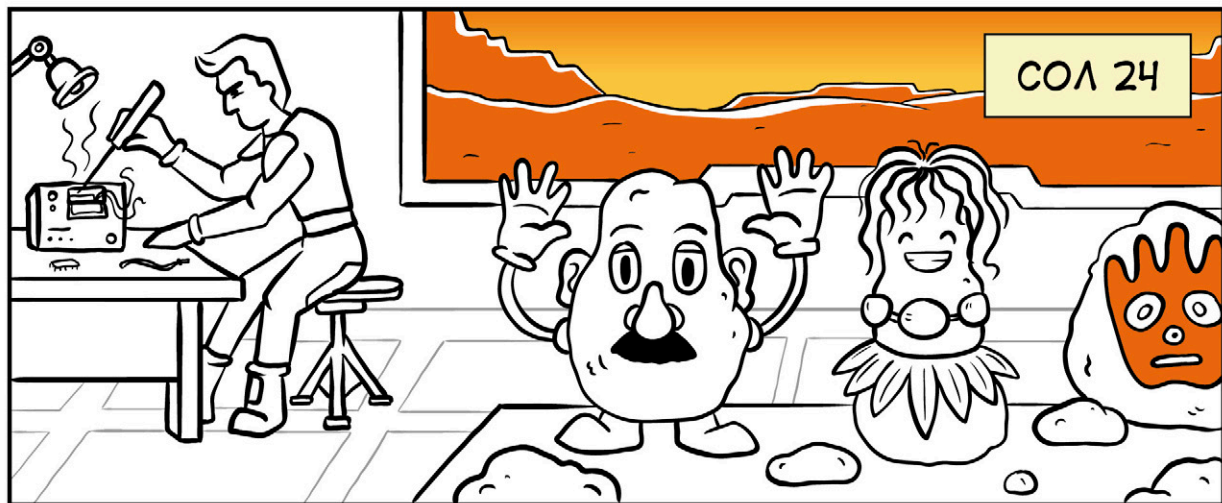
МЫ НЕ ОДНИ

Появление Дэма Мэйтона на пороге нашего кампуса посреди ночи стало полной неожиданностью. Мы думали, что мы совершенно одни на Марсе, и тут в дверь постучали.

Дэм Мэйтон был первым человеком, которого отправили на Марс, но его космический корабль взорвался во время снижения. До этого момента на протяжении года его считали погибшим. Но оказалось, он вовремя понял, что с кораблём что-то не так. У него были считанные минуты на спасение жизни, и он успел выскочить наружу верхом на марсоходе, который был снабжён собственным парашютом.

Мэйтону повезло, что его жилой модуль был доставлен грузовым кораблём заранее. Ему пришлось выживать в одиночку. За этот год он приспособился сам и научился приспособивать любую аппаратуру под любые цели. Даже вырастил три урожая картошки. И... неудивительно, что после всего пережитого он немного тронулся умом.

Мы надеемся, Дэм ещё придёт в себя. Пока он проводит большую часть времени с разными железяками и разговаривает со своим урожаем. Зато он починил наш сервопривод, и мы можем установить солнечную батарею!

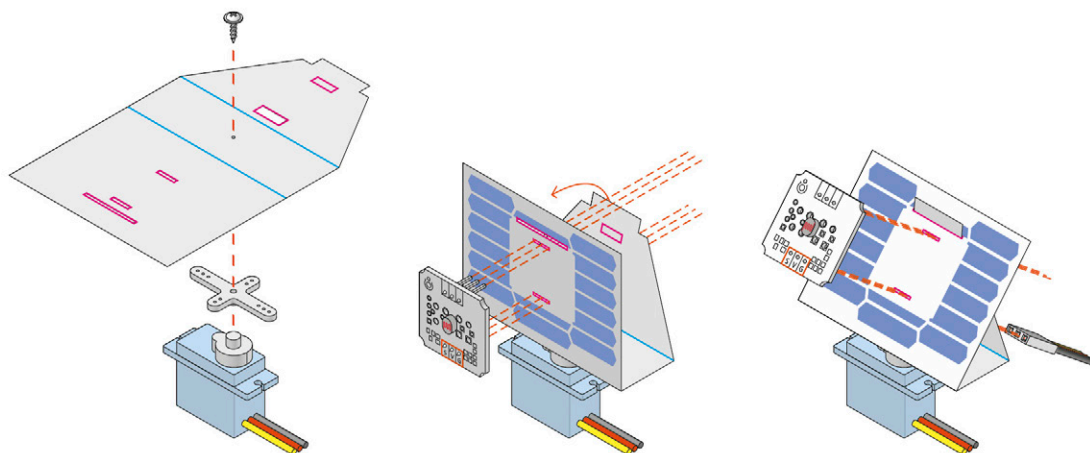


Соберем солнечную батарею

- 01** Возьми отвёртку, сервопривод, крестовую качельку и солнечную панель. Вкрути саморез в отверстие в солнечной панели, дальше в качельку и в сервопривод, как на рисунке.

- 02** Согни панель по линиям. Продень язычок в верхнее отверстие.

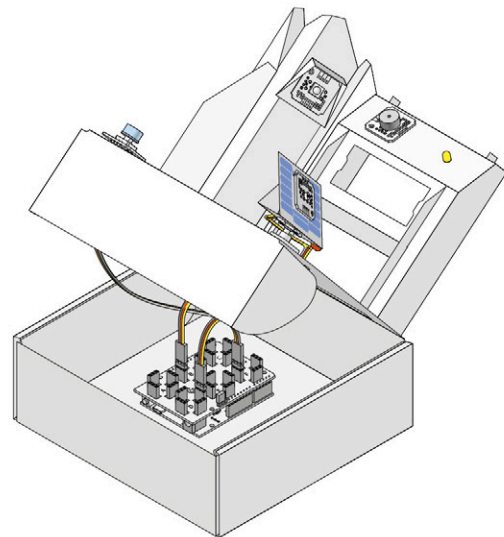
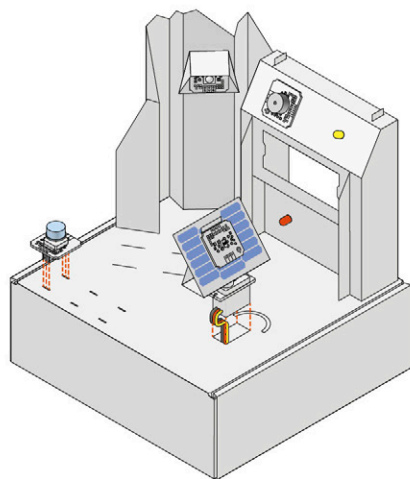
- 03** Вставь датчик освещённости в отверстия и подключи к нему сзади шлейф.

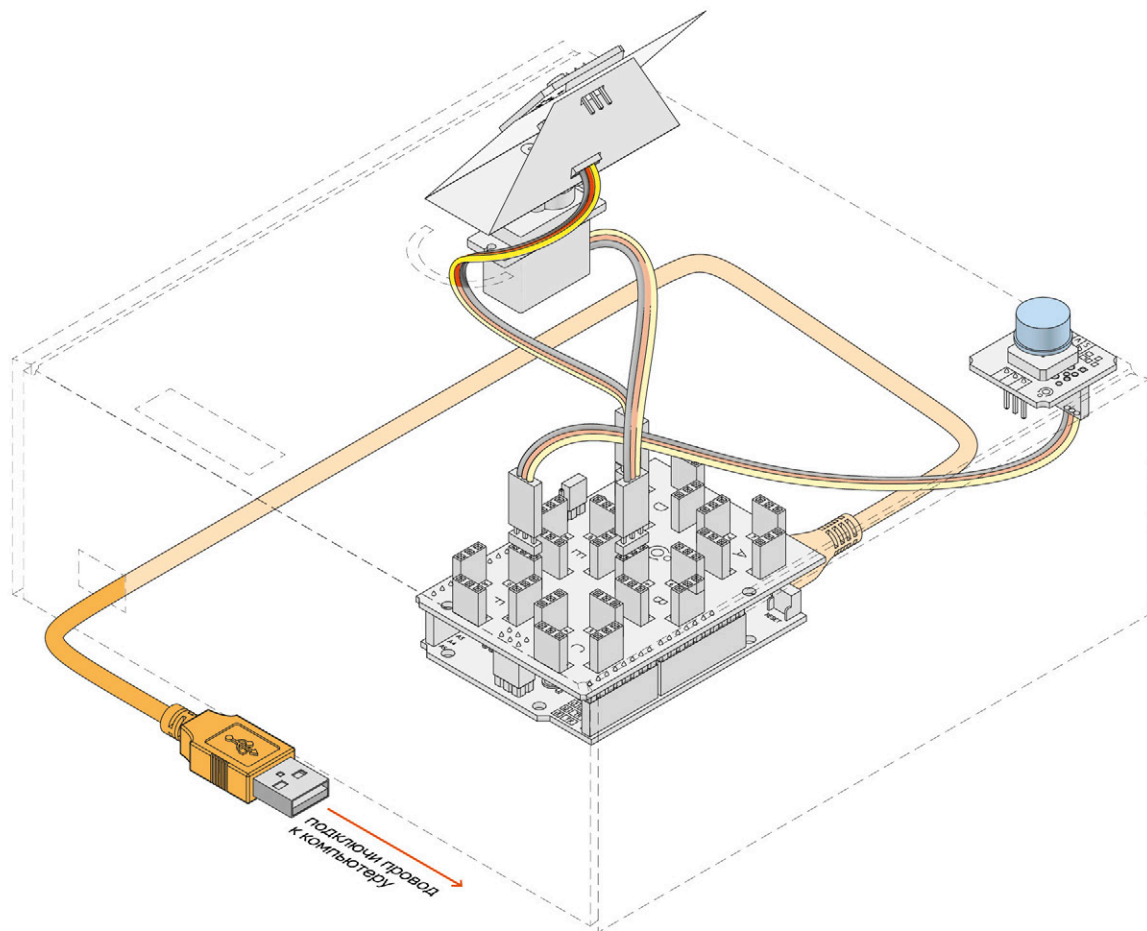


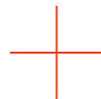
- 04** Вставь сервопривод в отверстие в коробке так, чтобы шлейф оказался внутри.
- 05** Продень шлейф датчика освещённости в дуговое отверстие внутрь коробки.
- 06** Вставь кнопку в коробку, в слот V1-module. Открой коробку и подключи с обратной стороны шлейф к кнопке.
- 07** Положи внутрь коробки контроллер и подключи к нему шлейфы:

- Кнопка — в пин 13
- Фотодатчик — в пин A3
- Сервопривод — в пин 6

- 08** Проверь номера пинов и контакты! Помни: чёрный провод должен быть подключён к пину G — земля.
- 09** Подключи к контроллеру провод питания.







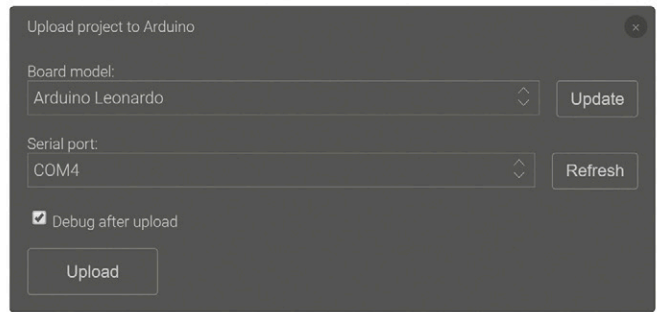
DEBUG

Марс находится дальше от Солнца, чем Земля, и получает меньше солнечной энергии. Наша солнечная панель должна уметь получать максимально возможное количество солнечного света, иначе не хватит энергии на обеспечение всех систем колонии. Для этого она должна поворачиваться вслед за движением Солнца по небосклону в течение дня.

Наша задача — поиск точки максимальной освещённости. Перед тем как создавать умный алгоритм, который будет искать эту точку самостоятельно, проведём небольшой эксперимент и понаблюдаем за значениями освещённости.



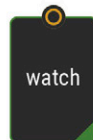
У XOD есть режим работы, который поможет тебе исправлять ошибки и наблюдать за всеми нюансами работы твоих алгоритмов, — `debug`¹. Он позволяет управлять значениями на патче и смотреть, что получается в режиме реального времени. Чтобы включить `debug`, нужно в окошке, которое появляется перед загрузкой патча, поставить галочку в строке `Debug after upload`.



Нода `tweak` подключается к пину ввода любой ноды и позволяет вводить значение с клавиатуры.



Нода `watch` подключается к пину вывода и показывает его текущее значение.



¹ Debug — отладка, процесс исправления ошибок в коде программы (англ.).



- 01 Собери патч как на рисунке. Ноды `watch` и `tweak-number` ты найдёшь в библиотеке `xod/debug`.
- 02 Теперь загрузи патч в Arduino, не забыв поставить галочку в строке `Debug after upload`.
- 03 Нода `watch` показывает текущее значение освещённости.
- 04 Нода `tweak` приняла значение 0, и ты можешь его менять. Для этого выдели её и в инспекторе поставь курсор в поле `OUT`. Теперь ты можешь ввести туда значение или увеличивать и уменьшать его с помощью стрелок «вверх» и «вниз» на клавиатуре — так быстрее.
- 05 Прогони сервопривод от 0 до 180° несколько раз, добавляя и убирая источники света.



Project Browser

My Project

main

amperka/colony

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

xod/debug

xod/gpio

xod/i2c

xod/json

xod/math

Inspector

tweak-number

xod/debug/tweak-number

OUT

0

Label

Description

04

Deployment

Installer Compiler Uploader Serial Tester

=== Debug session started ===

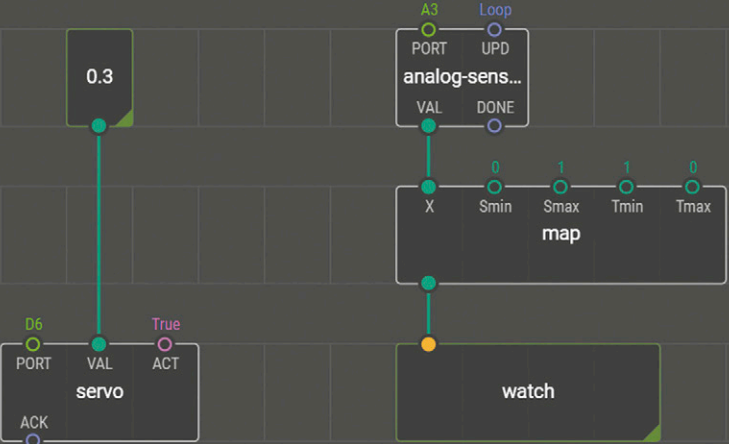
Deployment

main

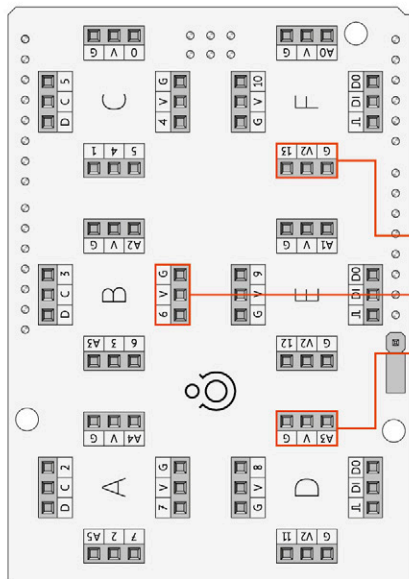
main

Program changed

Stop



ПОИСК МАКСИМУМА



Проверь, так ли всё
подключено

кнопка

сервопривод

датчик освещенности

Чтобы довести дело до конца, надо заставить солнечную панель выполнять эти действия самостоятельно.

Алгоритм следующий:

- | | | | |
|-----------|---|-----------|---|
| 01 | Поворачиваем сервопривод в положение 0°. | 05 | Переходим к пункту 3. |
| 02 | Запоминаем значение, которое передаёт фотодатчик, и угол поворота. | 06 | Таким образом, когда сервопривод повернётся до конца, то есть до 180°, в памяти будет записано значение угла, при котором освещённость была максимальной. |
| 03 | Поворачиваем сервопривод на несколько градусов. | 07 | Поворачиваем сервопривод на этот угол. |
| 04 | Сравниваем текущее значение фотодатчика с тем, что было записано в прошлой позиции. Если новое значение больше предыдущего, перезаписываем значение угла. Если не больше, ничего не делаем. | | |

Сделать такой алгоритм с помощью XOD можно.
И наш программист Жора уже сделал его за нас, спасибо ему за это.

Это нода `max-value`¹ из библиотеки  **amperka/colony**.

Если ты дважды кликнешь по её названию в раскрывающемся списке библиотеки, откроется патч этой ноды.

Попробуй понять, как Жора воплотил этот алгоритм.



- 01** Пин `VAL` принимает значения аналогового сенсора.
- 02** Пин `G0` получает команду к началу цикла в виде сигнала от кнопки.
- 03** В пин `STP`² впиши количество шагов, которые сделает сервопривод на пути от 0° до 180°. Например, если поставить значение 12, то сервопривод будет поворачиваться на 15° за шаг.
- 04** Пин `ANG`³ отдаёт угол, на который надо повернуться сервоприводу. А пин `MEM` — угол с максимальным значением освещённости.
- 05** Проверь номера портов и загрузи патч в Arduino.
- 06** Теперь нажми кнопку и наблюдай за тем, как солнечная панель выполняет наш алгоритм.

-
- 1 `Max-value` — максимальное значение (от англ. `maximum value`).
 - 2 `STP` — шаг (от англ. `step`).
 - 3 `ANG` — угол (от англ. `angle`).





Project Browser



My Project

main

amperka/colony

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

xod/debug

xod/gpio

xod/i2c

xod/json

xod/math

Inspector

max-value

amperka/colony/max-value

VAL

linked

STP

12

EN

linked

MEM

0

ANG

0

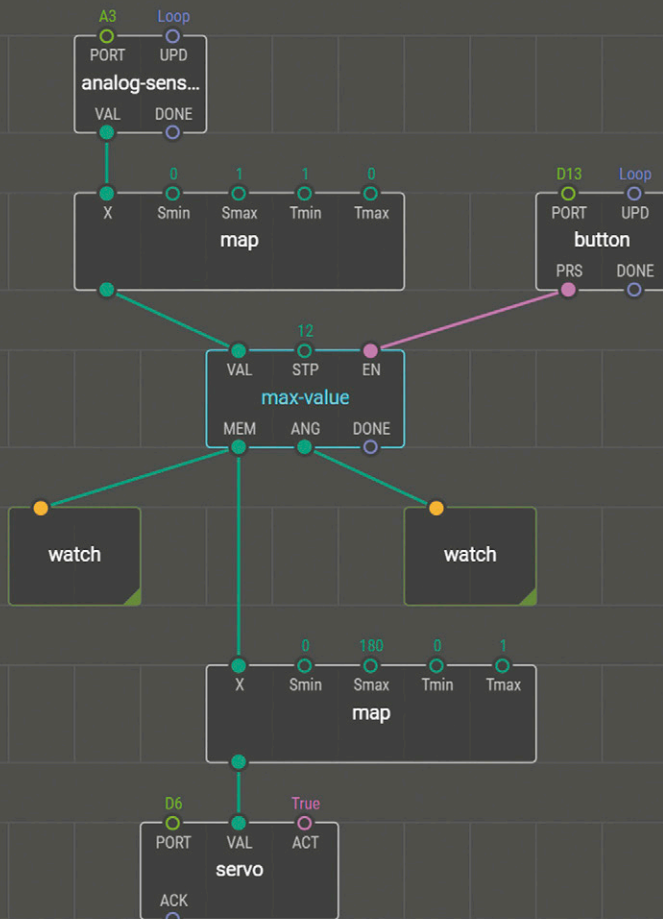
DONE

pulse

Label

Description

main x



Deployment

13

ДИСПЛЕЙ

Подключим и проверим дисплей — наше информационное табло. Для работы с ЖК-дисплеем нам понадобится нода `lcd-melt-rus`¹.

Отключи Arduino от питания. Достань табло и подключи его проводами к Arduino, как на схеме. Обрати внимание на то, как и куда подключаются провода. Если ошибиться, можно испортить табло.

Для вывода текста на экран используй ноды `constant-string`². Эти ноды содержат и передают данные типа `string`.

`String` — это просто другое название для строки обычного текста. Для обозначения пинов и соединений с этим типом данных в XOD используется золотой цвет.

-
- 1 В XOD есть стандартные ноды для LCD-дисплеев. Из них для этого дисплея подойдёт `text-lcd-16x2-i2c`. Но она не умеет выводить кириллицу — русские буквы. Поэтому специально для этого набора мы сделали ноду `lcd-melt-rus`.
 - 2 `Constant` — постоянный, неизменный, константа (англ.). `String` — строка (англ.).

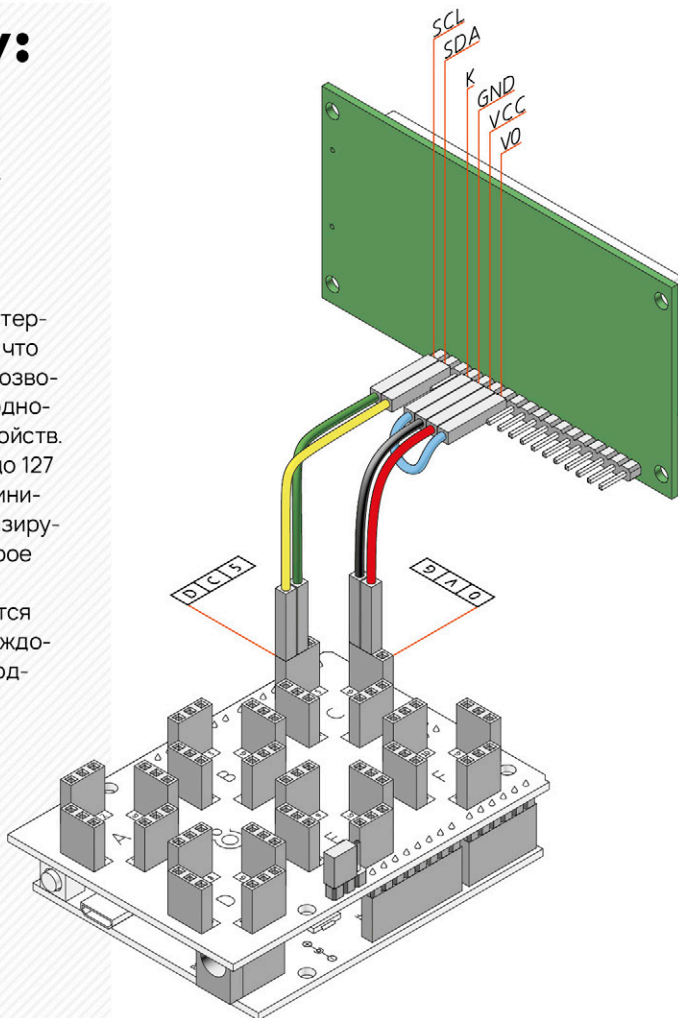
Собери схему:

Возьми одиночные провода мама-папа и подключи их к плате и дисплею, как на рисунке. Обязательно перепроверь. При неправильном подключении дисплей может выйти из строя.

Кстати

Мы подключили дисплей с помощью интерфейса I²C. Читается как «ай-ту-си» или, что более грамотно, «ай-квадрат-си». I²C позволяет с помощью всего двух проводов одновременно управлять множеством устройств. Можно подключить последовательно до 127 модулей. Пин D или SDA передаёт и принимает данные, а пин C или SCL синхронизирует работу. У каждого устройства, которое поддерживает I²C, есть адрес.

Микроконтроллер быстро переключается между адресами и раздаёт команды каждому устройству по очереди. Провода, подключённые в пины V и G, — это питание и земля.





- 01 Сохрани предыдущий патч, он пригодится, и создай новый.
- 02 Присоедини пин VAL обеих нод `constant-string` к пирам ввода L1 и L2 на ноде дисплея. L1 и L2 обозначают первую и вторую строку текста на дисплее.
- 03 Установи пин VAL одной ноды `constant-string` на «Скоро обед!», а второй ноды — на «В меню: картошка». Наш дисплей может выводить 2 строки по 16 символов.

Ещё раз проверь, правильно ли ты подключил дисплей, и загрузи патч в Arduino. На экране должны появиться введённые тобой строки.

Project Browser

My Project

main

amperka/colony

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

Inspector

constant-string

xod/core/constant-string

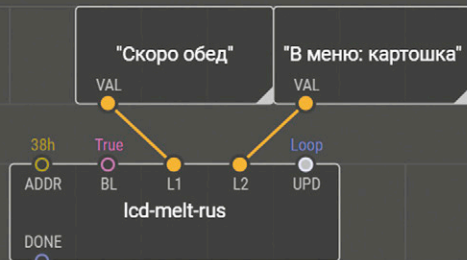
VAL

" "

Label

Description

main x



Deployment

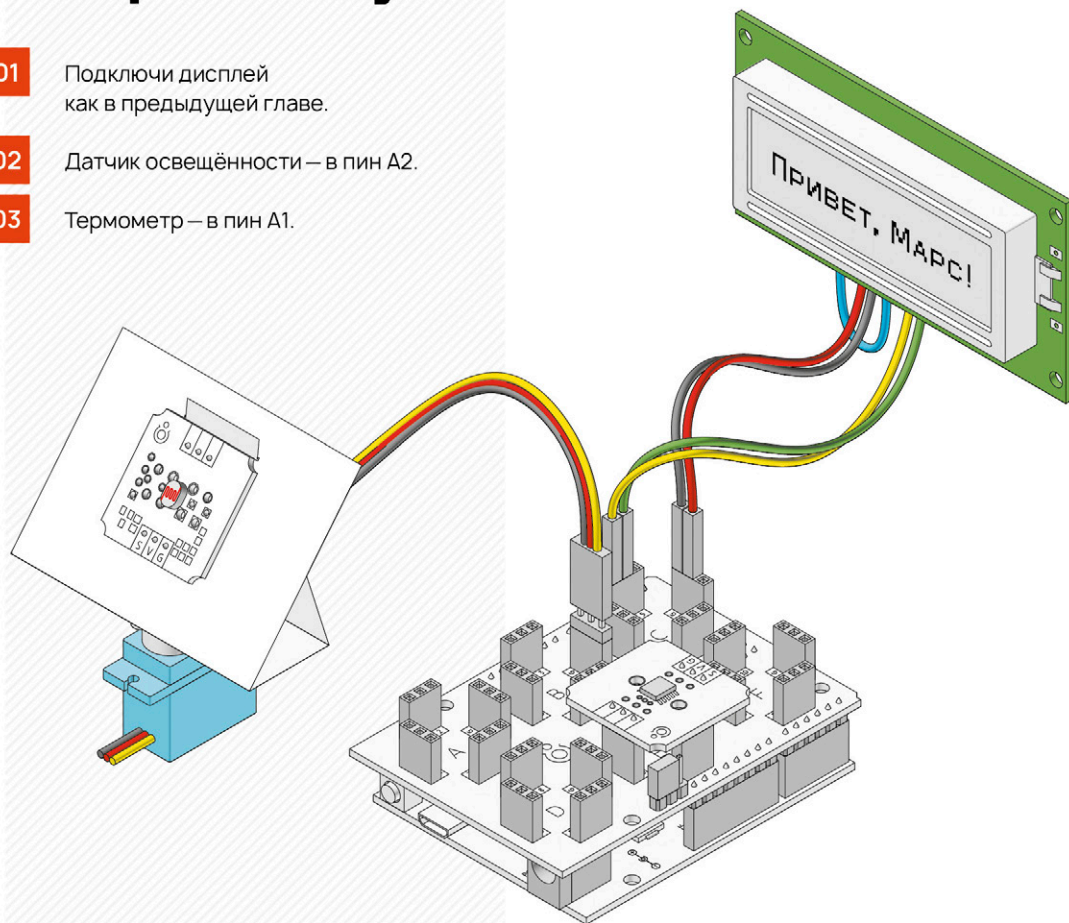
14 ВЫВОД ПЕРЕМЕННЫХ

Утром нашего 40-го сола Дэм Мэйтон, как обычно, встречал рассвет на ближайшем холме. Вернувшись, он сообщил нам, что близится пылевая буря. Как он это понял, мы не знаем, но дело серьёзное.

Пылевые бури на Марсе — настоящая напасть. Хотя они и не такие плотные, как в пустыне Сахара, они всё равно создают множество проблем. Пыль наносит ущерб оборудованию, забиваясь во все щели, и затмевает солнечный свет. Порой температура падает ниже -100° . Пыль поднимается постепенно, и заметить, что буря началась, можно только по приборам — температуре и освещённости. Выведем температуру и освещённость на табло.

Собери схему:

- 01 Подключи дисплей как в предыдущей главе.
- 02 Датчик освещённости — в пин A2.
- 03 Термометр — в пин A1.



Пины L1 и L2 дисплея принимают только по одному значению типа `string`. Если мы хотим вывести несколько значений в одну строку, нам нужно их объединить до подачи на дисплей. Для этого есть нода `concat`¹. Она принимает два или более значения и объединяет их в одну строку.



- 01** Достань на патч-борд две ноды `concat`, ноду для термометра `thermometer-tmp36` и для фотодатчика `analog-sensor`.
- 02** Чтобы «перевернуть» показания фотодатчика, нам нужна нода `map` (подробнее о ноде `map` в главе 07).
- 03** В нодах `concat` в пин IN1 напиши: «Темп. = » и «Свет = ». Не забудь поставить пробелы для красоты.
- 04** Соедини ноды и проверь номера портов.
- 05** Загружай патч в Arduino. Исследуй, как изменяются показания датчиков в разных условиях.

¹ `Concat` — сцеплять (от англ. concatenate).

main x

Project Browser

My Project

main

amperka/colony

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

Inspector

concat

xod/core/concat

IN1

IN2

OUT

Label

Description

A2

Loop

PORT

UPD

analog-sens...

VAL

DONE

A1

Loop

PORT

UPD

thermomete...

Tc

DONE

X

Smin

Smax

Tmin

Tmax

map

0

1

1

0

c-to-f

Свет =

concat

Темп. =

concat

38h

True

Loop

ADDR

BL

L1

L2

UPD

lcd-melt-rus

DONE

Кстати!

Нода термометра выдает на свой пин Tc температуру в градусах Цельсия. Если тебе ближе по духу господин Фаренгейт, используй ноду c-to-f.

Deployment

15

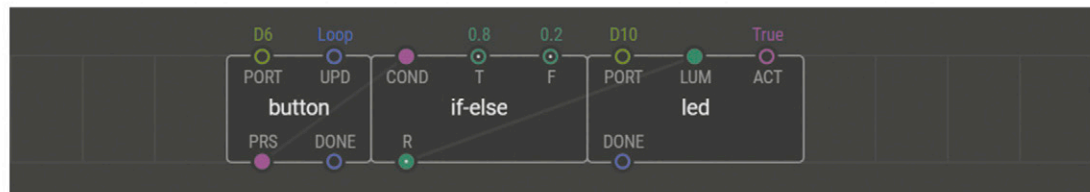
НОДА IF-ELSE

Автоматизируем подачу сигнала тревоги на табло. Тревогу будем включать на основе показаний наших сенсоров. Когда станет очень холодно и темно — значит, буря началась.

Чтобы сделать такое условие, используем ноду `if-else`¹. Если пин ввода этой ноды, `COND`², получает `true`, нода передаёт на свой пин вывода значение, которое установлено в пине `T`. Если же

нода получает `false`, она передаёт дальше значение, указанное в пине `F`.

На картинке — пример, можешь его собрать. Пока мы нажимаем кнопку, она передаёт значение `true` ноде `if-else`. Нода `if-else` выбирает значение 0.8 и отдаёт его на `led`. Светодиод горит на 80 %. Отпускаем кнопку, нода `if-else` получает `false` и передаёт диоду значение 0.2.

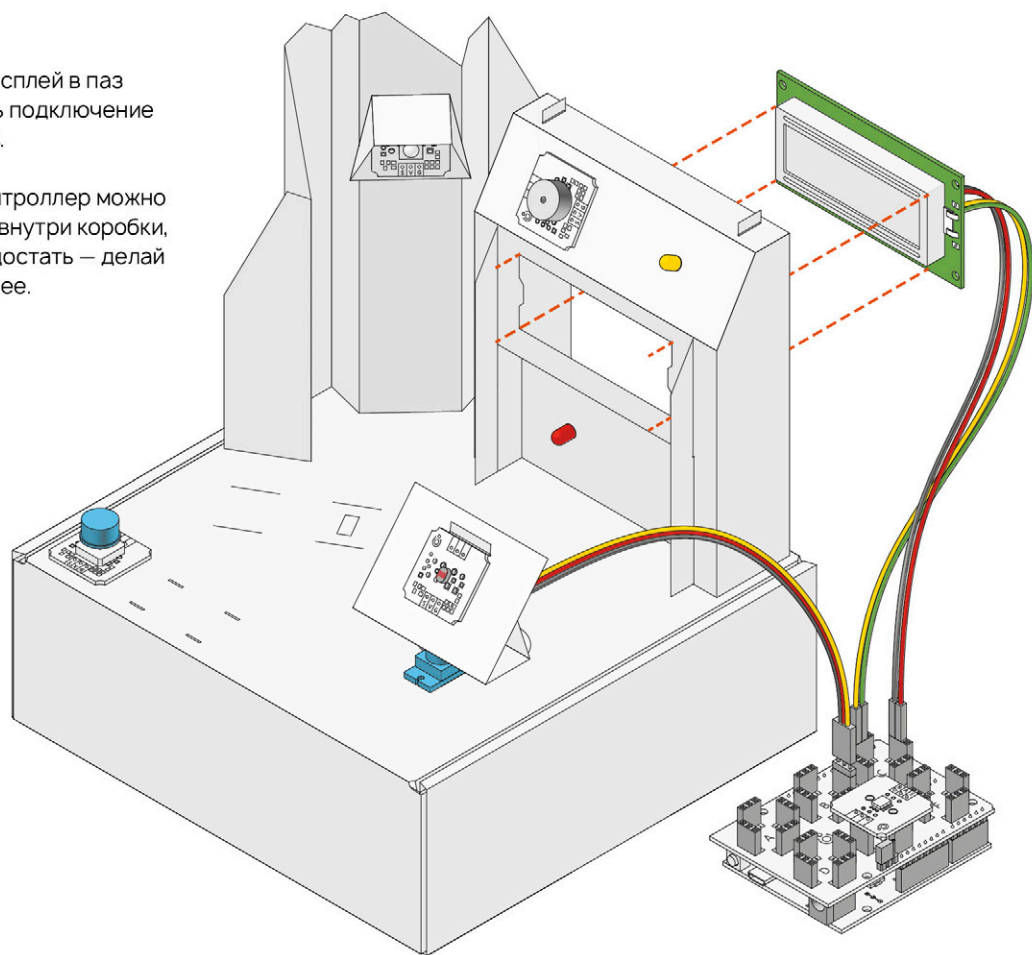


1 `If-else` — если-иначе (англ.).

2 `COND` — условие (от англ. condition).

Вставь дисплей в паз
и проверь подключение
проводов.

Микроконтроллер можно
оставить внутри коробки,
а можно достать — делай
как удобнее.



01 Собери патч. Он несложный, не пугайся.

02 Здесь мы убрали ноды `constant-string` и написали их значения напрямую в поля `true` и `false` нод `if-else`, чтобы сократить количество нод в патче. Это удобно, когда ты заканчиваешь писать программу и уже точно знаешь, какие значения будут у этих нод. Обрати внимание: поскольку нода `if-else` воспринимает данные разных типов, ей надо дать понять, что мы вписываем именно `string`. Для этого поставь кавычки.

03 Расставь значения: в ноде `less-1` пороговое значение: 0.4, в `less-2`: 16 °C. Нода `map`, как и в предыдущих опытах, нужна для того, чтобы перевернуть значения фотодатчика.

04 У ноды `clock` на пине `IVAL` поставь период 1–2 секунды.

05 Проверь номера портов и загрузки в Arduino. Алгоритм таков:

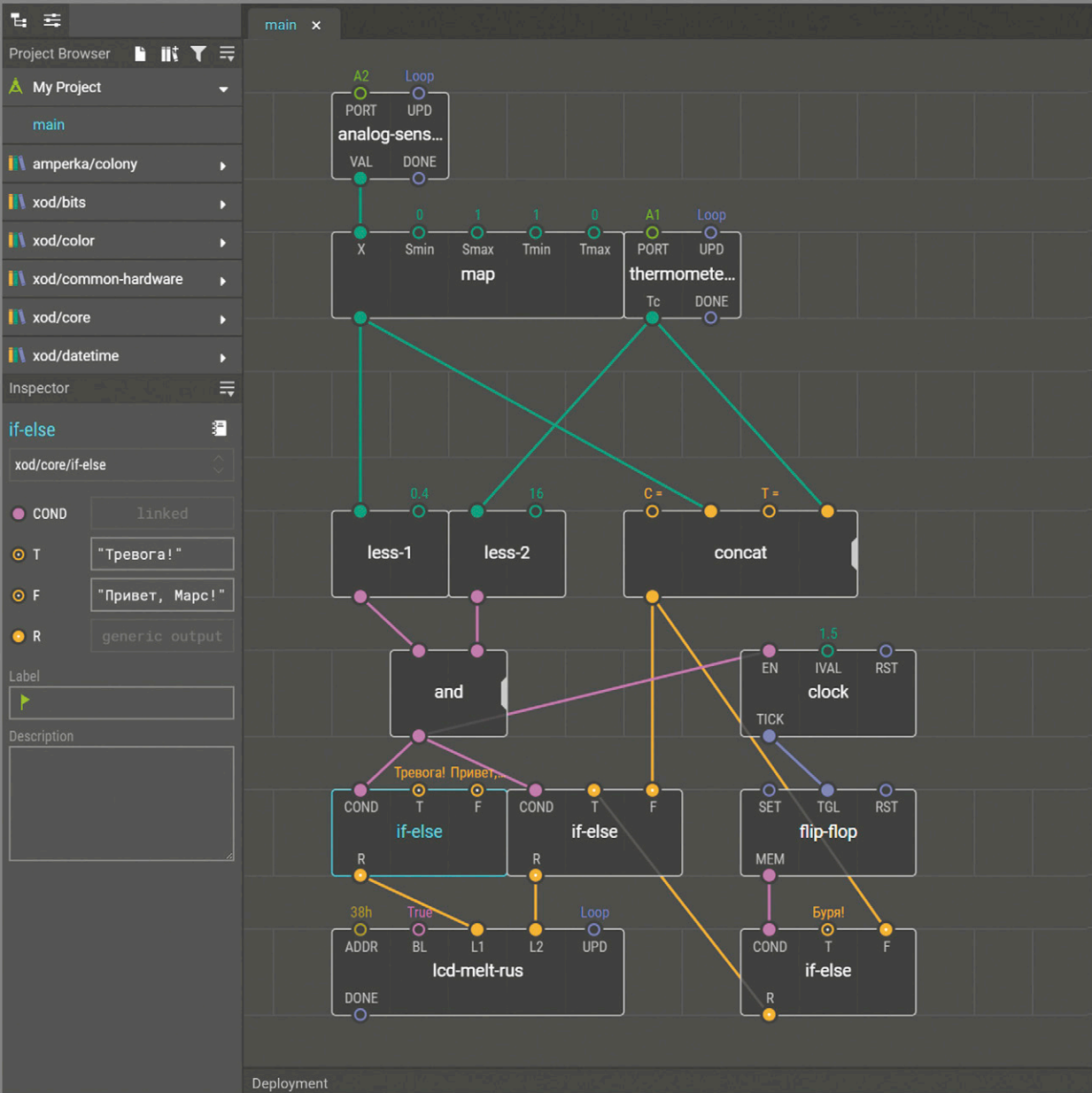
- Если освещённость и температура одновременно падают ниже определённого уровня, нода `and` передаёт значение `true` и включает тревогу.

- Датчики передают свои значения на длинную ноду `concat`, которая соединяет всё в одну строчку. Чтобы «удлинить» ноду и добавить ей новые пины ввода, потяни мышкой за светлый выступ справа.

- Три ноды `if-else` переключают табло в режим тревоги, и на нем выводится тревожный текст. Когда всё в порядке, дисплей выводит строку «Привет, Марс!» и показания датчиков. По тревоге первая строка — «Тревога!», а вторая мигает между значениями датчиков и словом «Буря». Именно за этим нам нужны `clock` и `flip-flop`.

06 Чтобы охладить термометр, возьми ложку и положи её в морозилку. Подожди 10 минут, приложи ложку к термометру и заслони пальцем фотодатчик. Когда термометр остынет, включится тревога.

07 Сохрани этот патч. Он пригодится для финальной схемы.



БУРЯ БЛИЗКО!

Сегодня наши приборы зафиксировали слабое, но устойчивое снижение освещённости. Это значит, Мэйтон был прав, и в течение следующих двух солов нас накроет настоящая марсианская пылевая буря.

Это будет испытанием для систем жизнеобеспечения нашей колонии. День превратится в мрачные красные сумерки, температура может упасть ниже -100°C . Пыль электризуется от трения и будет налипать на все поверхности, в том числе на наши солнечные батареи. Это существенно снизит выработку энергии, а может, и вовсе сведёт к нулю.

Из-за отсутствия связи с Землёй, мы не знаем, сколько продлится буря. Нам повезёт, если она небольшая и закончится через неделю. Но примерно раз в 5 лет здесь бывают гигантские штормы. Они закрывают целые континенты и могут длиться целый месяц и дольше.

Мы готовы к этому: запасли достаточно энергии в аккумуляторах, а Мэйтон нашёл старый марсоход в кратере неподалёку и снял с него радиоизотопный генератор. Этого должно хватить.

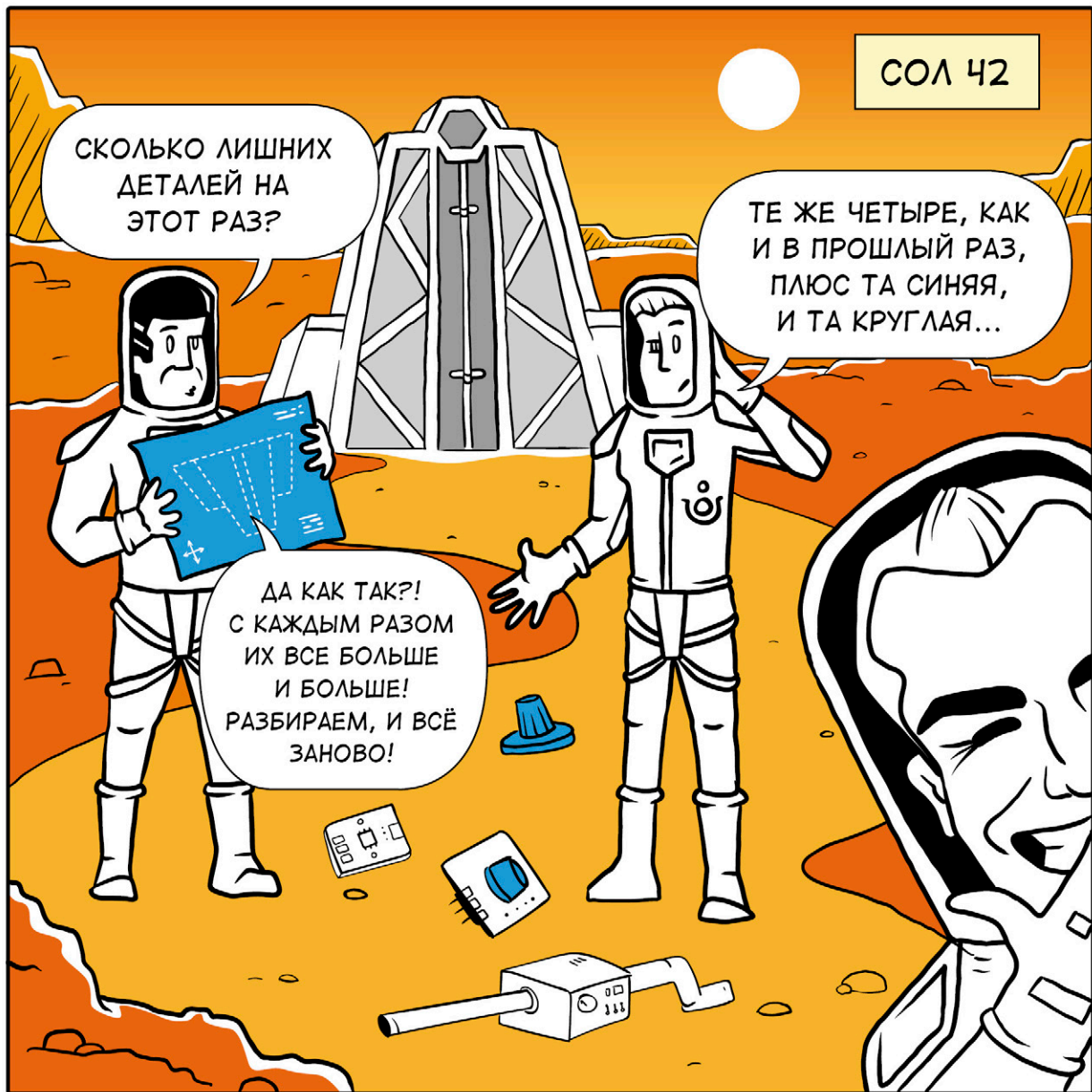
Последняя наша проблема — вода. Она нужна не только для питья. Мы используем её для гигиенических процедур, добычи кислорода, для дыхания, а теперь ещё и для поливки нового урожая Дэма Мэйтона. Кислород мы добываем пропуская через воду электрический ток, который разбивает молекулы воды на водород и кислород.

Несмотря на эффективные системы регенерации и очистки воды, она всё равно расходуется. Под нами есть разведанные запасы жидкой воды, и мы прихватили с Земли добывающую станцию с буром. Нам обязательно нужно установить её перед бурей, иначе, если непогода затянется, мы окажемся в трудном положении.

СКОЛЬКО ЛИШНИХ
ДЕТАЛЕЙ НА
ЭТОТ РАЗ?

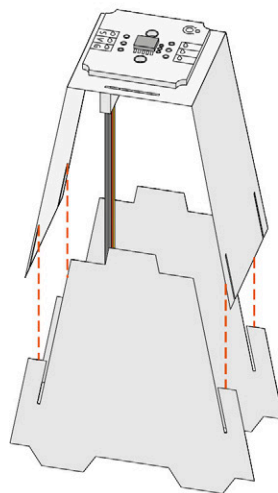
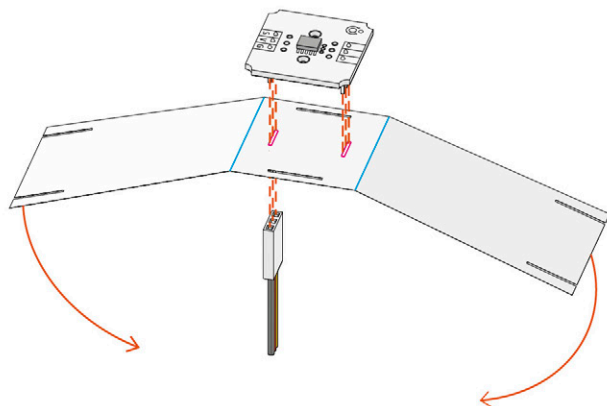
ТЕ ЖЕ ЧЕТЫРЕ, КАК
И В ПРОШЛЫЙ РАЗ,
ПЛЮС ТА СИНЯЯ,
И ТА КРУГЛАЯ...

ДА КАК ТАК?!
С КАЖДЫМ РАЗОМ
ИХ ВСЕ БОЛЬШЕ
И БОЛЬШЕ!
РАЗБИРАЕМ, И ВСЁ
ЗАНОВО!



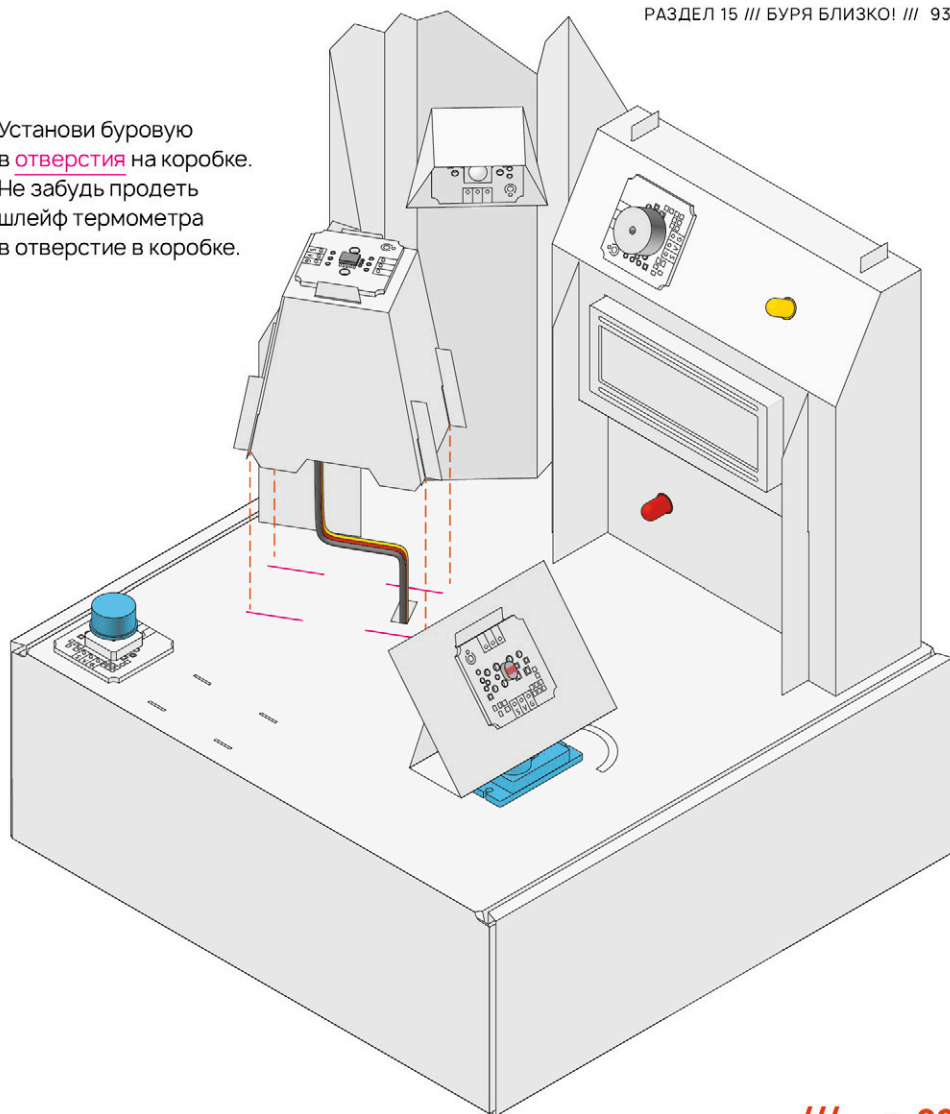
02 Вставь термометр в **пазы** на крыше буровой установки и подключи к нему снизу шлейф.

03 Согни деталь по **линиям** и вставь в пазы в стенах, как на рисунке.



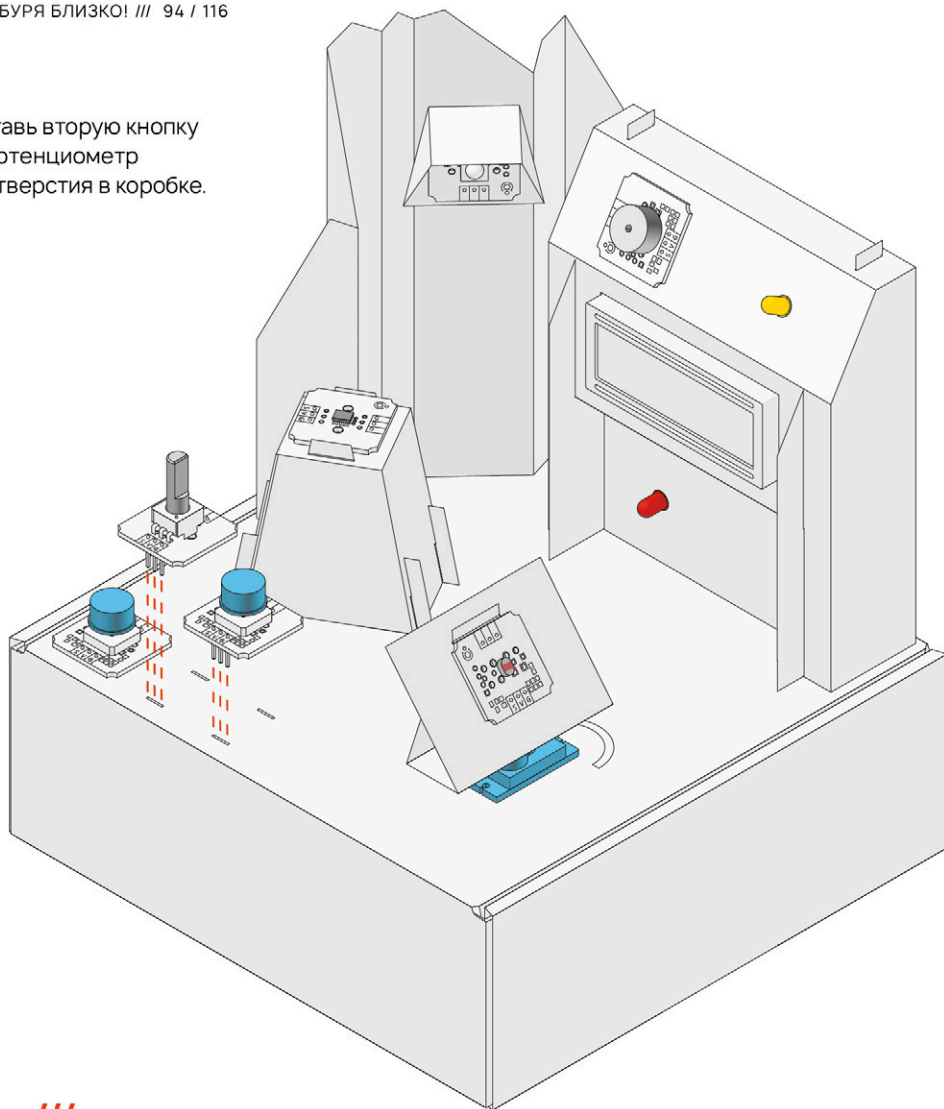
04

Установи буровую в **отверстия** на коробке.
Не забудь продеть шлейф термометра в отверстие в коробке.



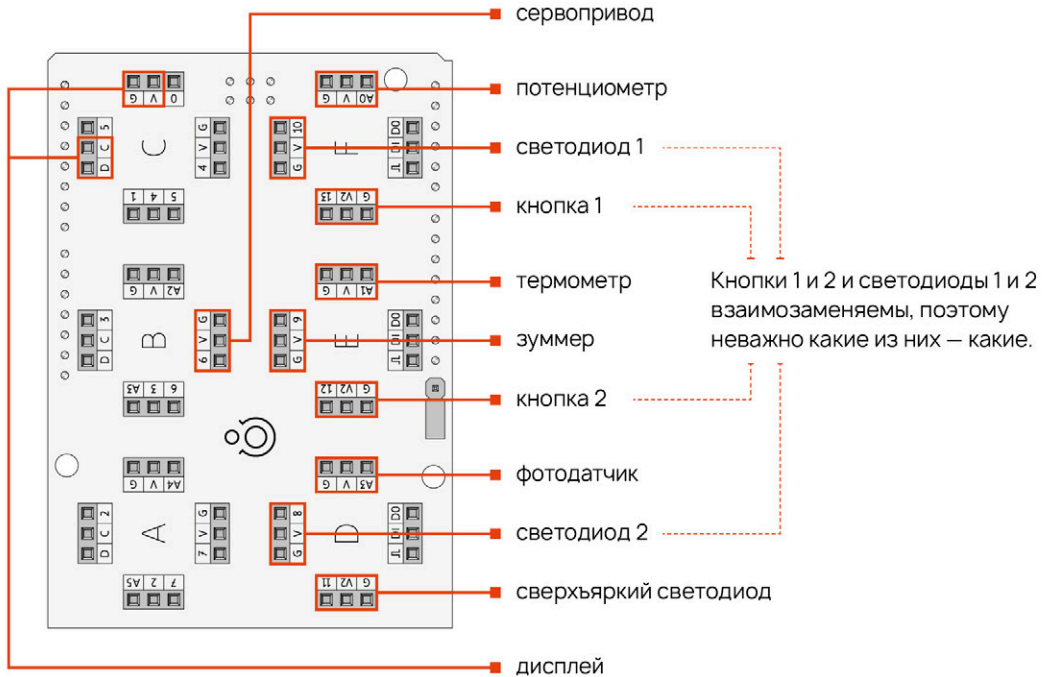
05

Вставь вторую кнопку
и потенциометр
в отверстия в коробке.



06

Возьми все шлейфы и подключи все модули к плате, как на схеме:



07

Ура! Мы наконец достроили колонию. Проверь подключение модулей и переворачивай страницу.

16 СОЗДАНИЕ СОБСТВЕННЫХ НОД

Программы, которые управляют многими устройствами, могут состоять из десятков и сотен нод и занимать кучу места. А также их довольно тяжело «прочитать». В XOD предусмотрен способ избежать этой проблемы. Из любого патча ты можешь сделать ноду и использовать её в других патчах!

- 01 Открой патч своего проекта из главы 10 — тот, где мы делали тревогу с зуммером.
- 02 Достань ноду `input-boolean` из библиотеки  `xod/patch-nodes`. Это нода входного значения типа `boolean`. Это то же самое, что и кружки на нодах.
- 03 Удали `button` и `flip-flop`.
- 04 Соедини пины, которые «остались бесхозными», с нодой `EN`.
- 05 Достань из той же библиотеки ноду `input-number` и подключи её к пину `IVAL` ноды `clock`. Так мы сможем настроить это значение, когда будем использовать этот патч как ноду.
- 06 Для удобства переименуй пины входа как на картинке.
- 07 Переименуй патч в `panic`.

Project Browser

My Project

main

amperka/colony

xod/bits

xod/color

xod/common-hardware

xod/core

xod/datetime

Inspector

input-number

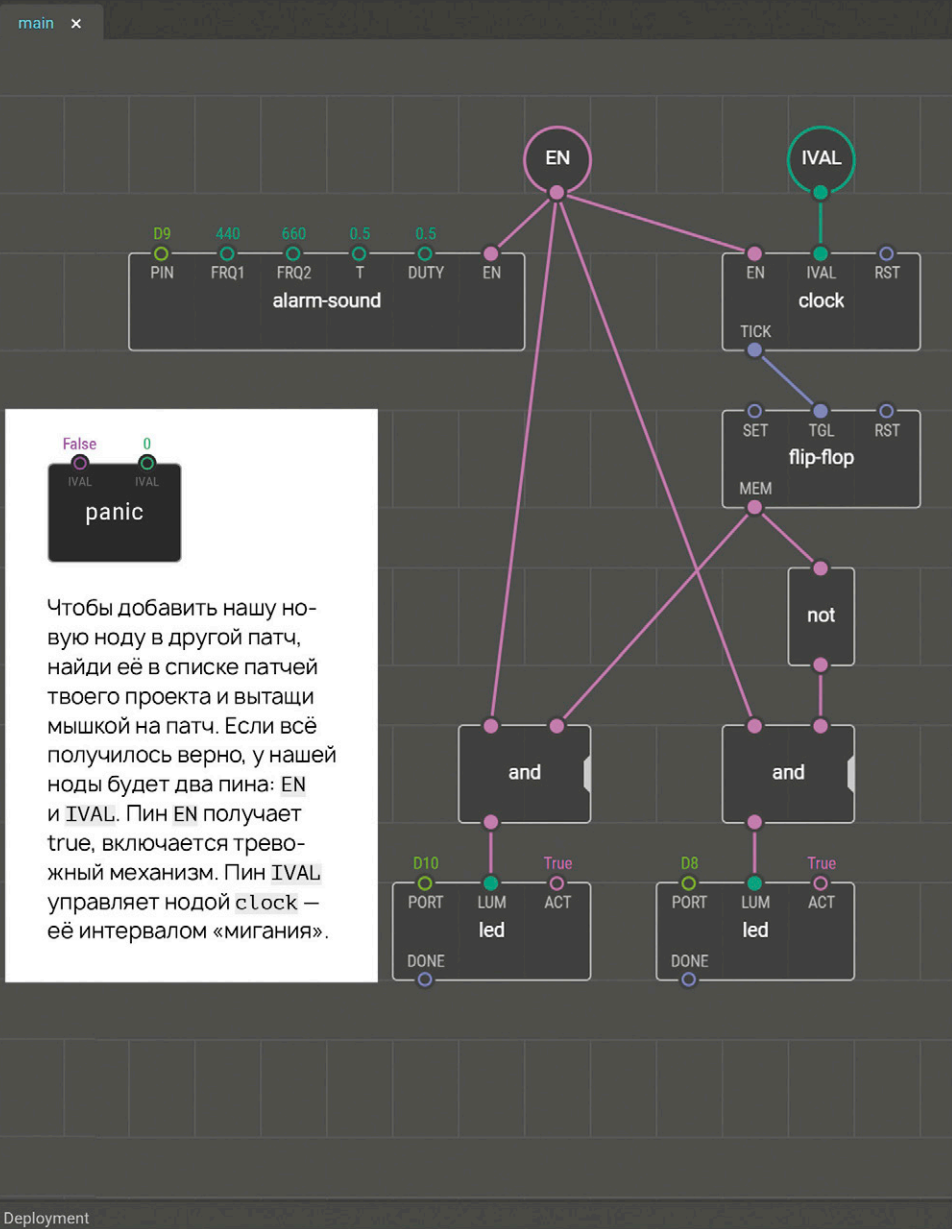
xod/patch-nodes/input-number

OUT 0

Label

Description

Deployment



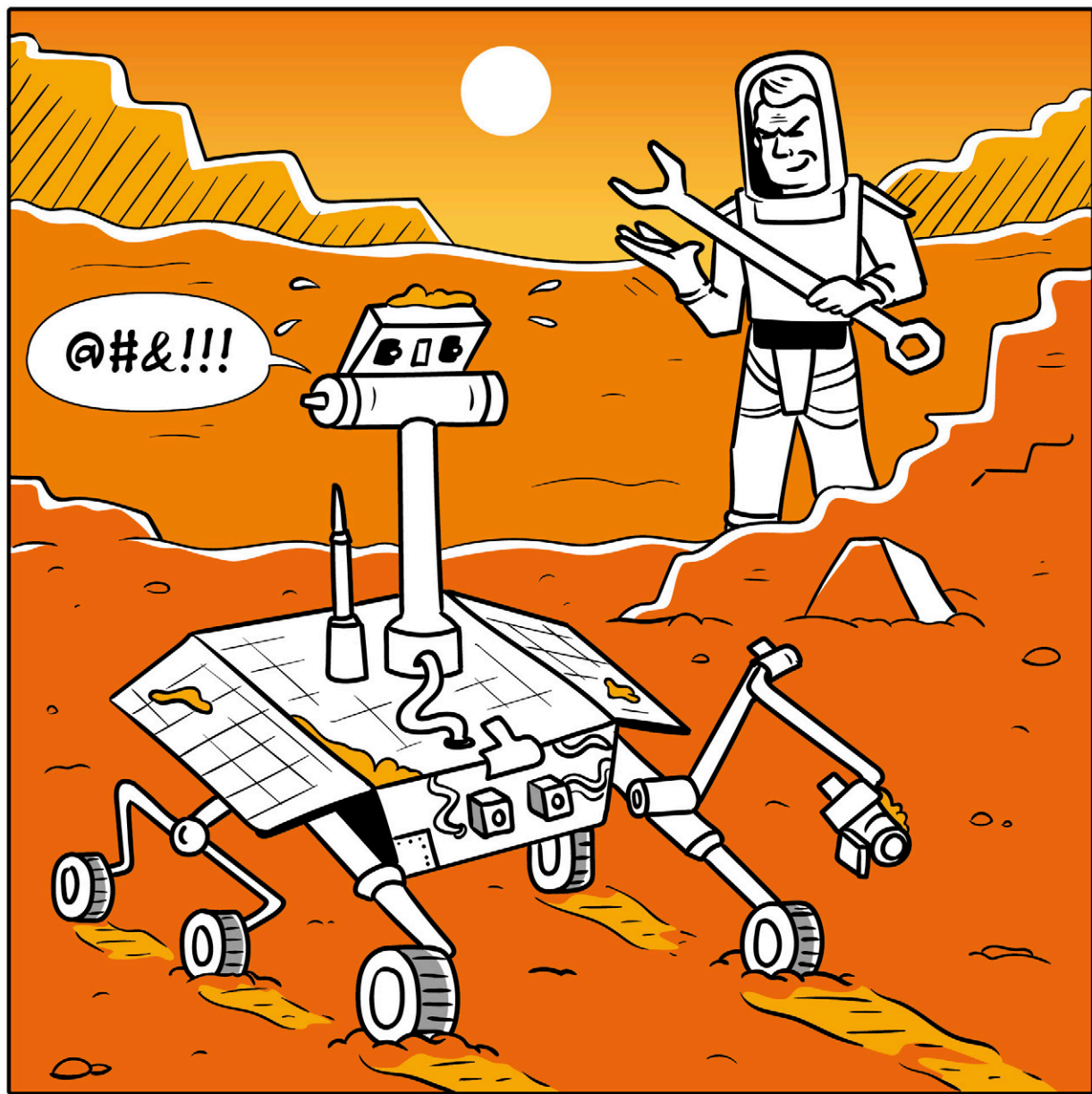
Чтобы добавить нашу новую ноду в другой патч, найди её в списке патчей твоего проекта и вытащи мышкой на патч. Если всё получилось верно, у нашей ноды будет два пина: EN и IVAL. Пин EN получает true, включается тревожный механизм. Пин IVAL управляет нодой clock — её интервалом «мигания».

НОДА ОПОВЕЩЕНИЯ

Высокие температуры также могут стать проблемой. Обычно на Марсе холодно, и для работы любых движущихся механических деталей техники требуется нагрев. Тепло обеспечивают маленькие обогреватели. Иногда эти обогреватели могут застрять во включённом состоянии и продолжать греть, вызывая перегрев.

Так случилось с марсоходом «Оппортьюнити». На его второй сол на Марсе отказал выключатель обогревателя одного из суставов его руки — манипулятора. Несмотря на все усилия инженеров NASA, проблема со временем усугублялась и, в конечном итоге, «Оппортьюнити» не смог больше складывать манипулятор. Так он и ездил задом с вытянутой «рукой» последние 11 лет своей миссии.

Поэтому при критическом повышении температуры нам тоже важно быть бдительными.



Доработаем функционал оповещения и сделаем из этого ноду:

- 01** Собери патч как на схеме. Можно от-крыть патч из главы 15 — это сэкономит время.
- 02** Добавь три ноды `input-number` и одну `input-boolean` и подключи их как на картинке.
- 03** Пин `EN`, как и в прошлой главе, отвечает за включение тревоги.
- 04** Пин `L` будет получать значение освещённости от фотодатчика, а пин `T` — температуру в градусах от термометра.
- 05** `Tmax` — критическая температура, при которой мы выводим на табло уведомление «Опасный нагрев».
- 06** Переименуй пины ввода для удобства, а сам патч назови `show-info`.
- 07** Если хочешь, подключи к пинам нод `if-else` с текстом для дисплея ноды `input-string`.
- 08** Когда мы завершим главную программу управления колонией, попробуй приложить к термометру палец или нагрей ложку под лампой.

Project Browser



My Project

main

amperka/colony

xod/bits

xod/color

xod/common-hardware

xod/core

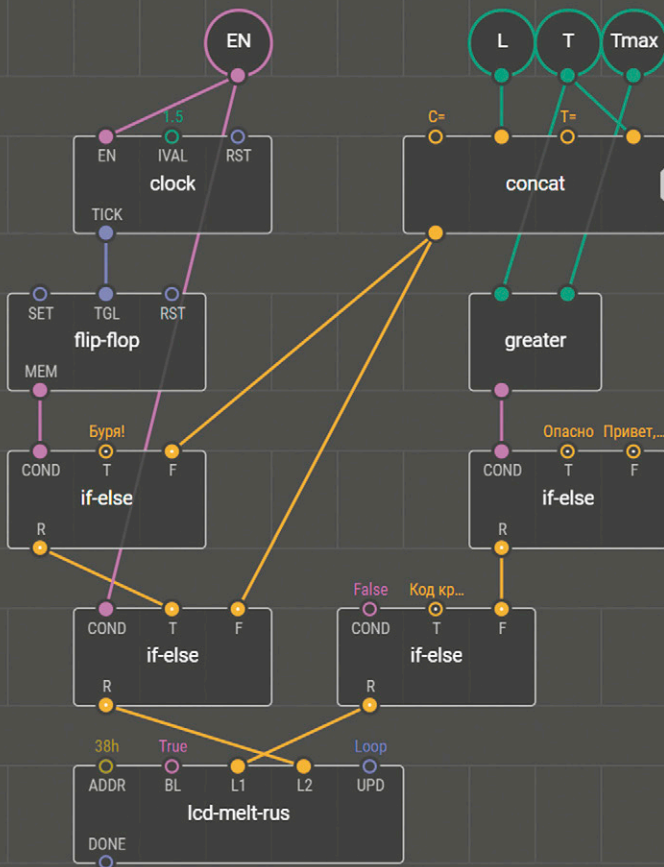
xod/datetime

Inspector

main

Description

main x



Deployment

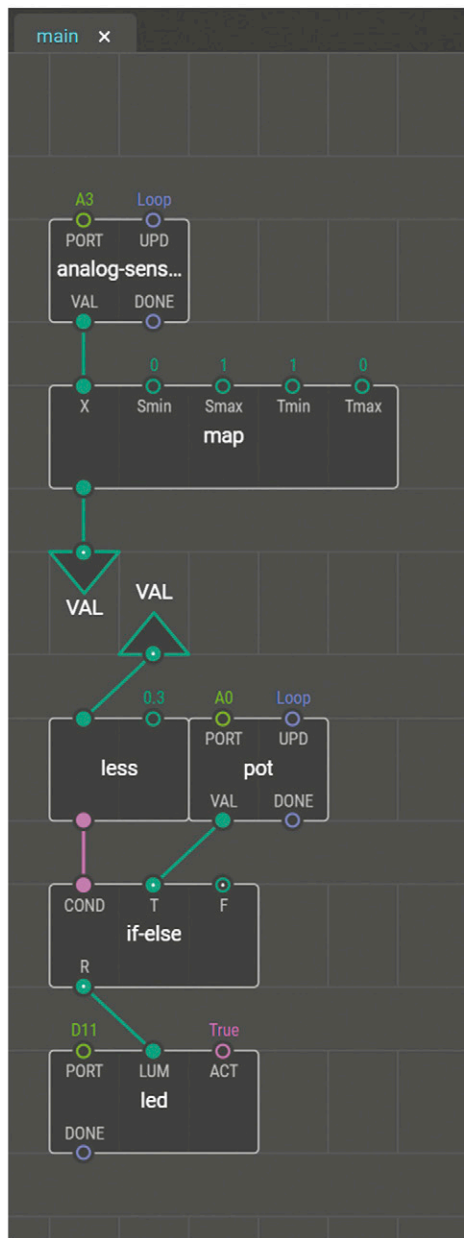
ПОСЛЕДНИЕ ШТРИХИ

Осталось собрать всё воедино, чтобы получить главную программу управления колонией. Она будет управлять всеми нашими устройствами на основе показаний датчиков и команд от кнопок и потенциометра.

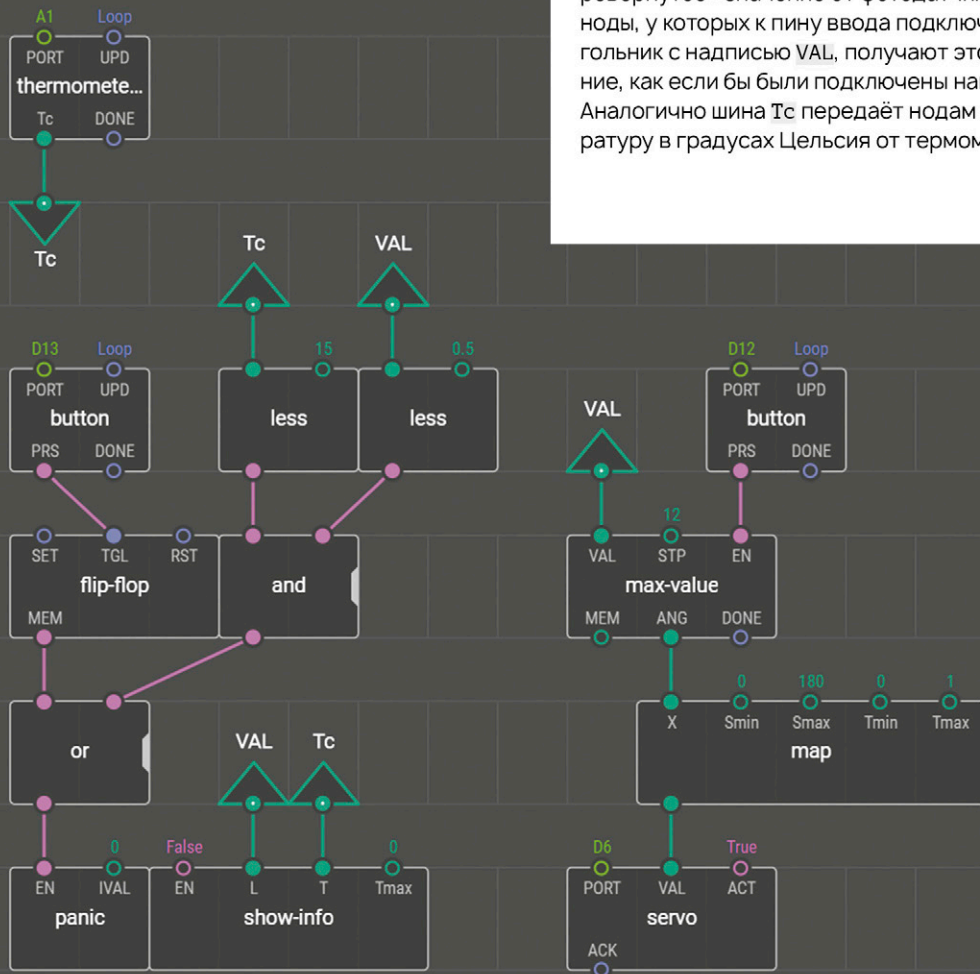
Здесь мы используем новый элемент интерфейса XOD — шину¹. Это способ отображения соединений между пинами, который никак не влияет на работу программы, но делает её проще для восприятия. Шина удобна, когда один пин вывода подключён сразу ко многим пинам ввода других нод, и «провода» к ним тянутся через весь патч.

В нашем примере так получилось с нодой `map` и нодой термометра. Соедини все ноды, как ты делаешь это обычно. Затем выдели соединения, которые хочешь превратить в шину, и нажми на **Edit → Convert Selected Links to Buses**. Прodelай эту операцию дважды: для ноды `map` и для ноды `thermometer-tmp36`.

¹ Шина (от англ. bus) — пучок проводов, который соединяет функциональные части компьютера или другого электронного устройства.

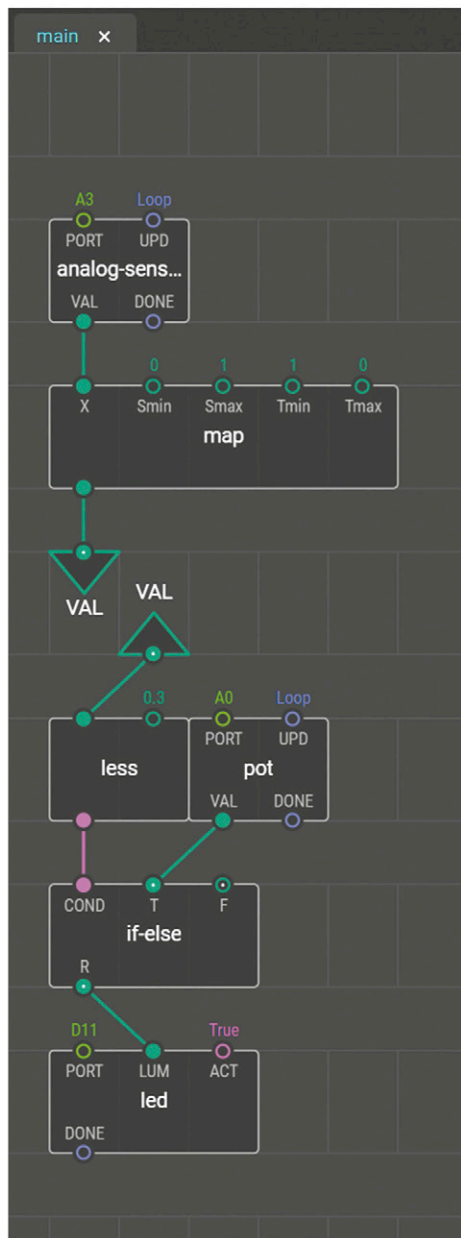


На этом патче две шины: VAL и Tc. VAL подключена к пину вывода ноды тар и передаёт «перевёрнутое» значение от фотодатчика. Все ноды, у которых к пину ввода подключён треугольник с надписью VAL, получают это значение, как если бы были подключены напрямую. Аналогично шина Tc передаёт нодам температуру в градусах Цельсия от термометра.



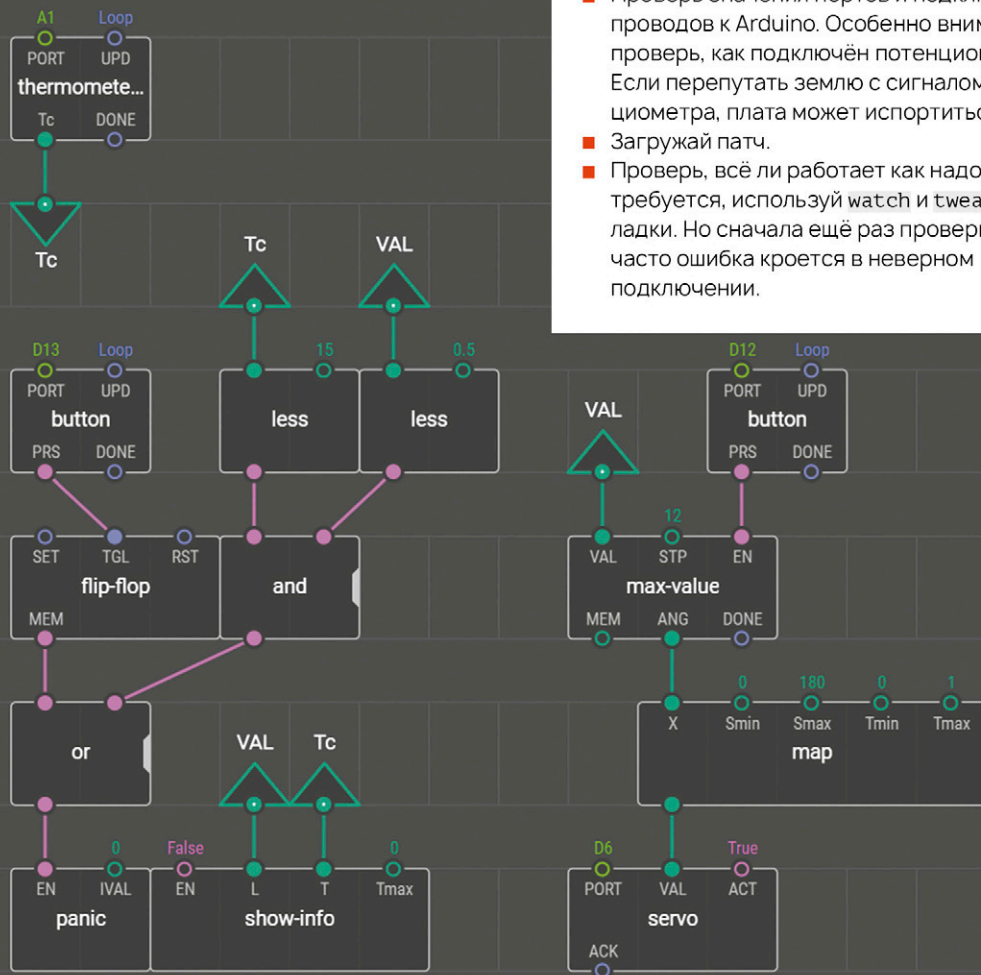
Алгоритм программы состоит из тех патчей, которые мы собирали в предыдущих главах. Освежим его:

- 01** Сервопривод вращает солнечную панель в поисках оптимального положения с помощью ноды `max-value`. Этой ноды нужно ввести значение в пин `STP` — количество шагов, которые сделает сервопривод на пути от 0 до 180°. Чем больше шагов, тем дольше процесс, но тем выше точность. Механизм приводится в действие нажатием кнопки в пине 12.
- 02** Тревога включается автоматически, когда освещённость и температура падают ниже значений в нодах `less`, или вручную — нажатием кнопки в пине 13.
- 03** Ноды `panic` необходимо ввести значение для пина `IVAL` — частоту мигания светодиодов в секундах.
- 04** Если температура превысит значение `Tmax` ноды `show-info`, на дисплей выводится предупреждение о возможном перегреве техники.
- 05** Проектор включается, когда становится темно, его яркость регулируется потенциометром.



Собери патч

- Проверь значения портов и подключение проводов к Arduino. Особенно внимательно проверь, как подключён потенциометр. Если перепутать землю с сигналом у потенциометра, плата может испортиться.
- Загружай патч.
- Проверь, всё ли работает как надо. Если потребуется, используй watch и tweak для отладки. Но сначала ещё раз проверь провода — часто ошибка кроется в неверном подключении.



СОЛ 42

СКОЛЬКО
ЛИШНИХ
ДЕТАЛЕЙ?

17...

ЛАДНО, ВКЛЮЧАЙТЕ УСТАНОВКУ.
ЭТО ДЕТАЛИ ИЗ СТАРОГО РОВЕРА,
Я ИХ ПОДБРОСИЛ

ВЫ ТАК ЗДОРОВО
СОБИРАЛИ
И РАЗБИРАЛИ!

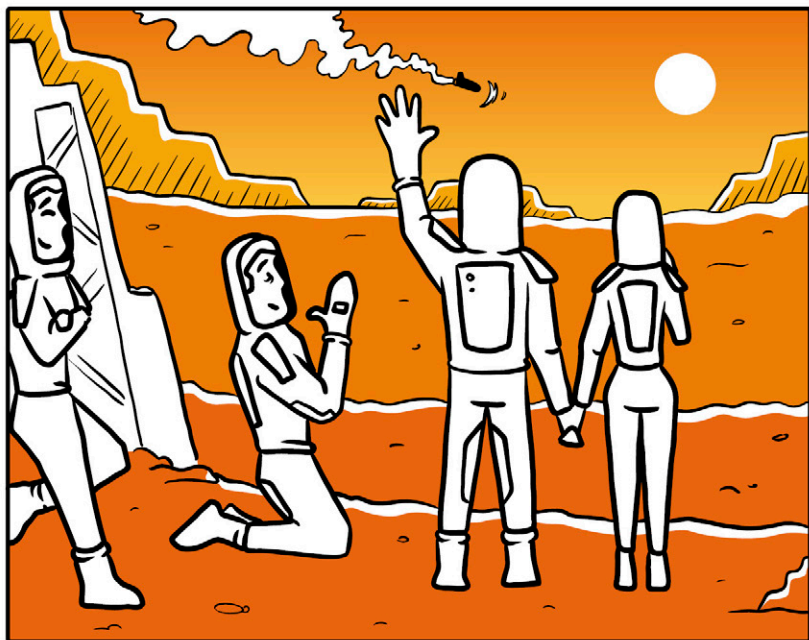
КЭП, ОСТЫНЬ! КЭП, ДАВАЙ
НЕ БУДЕМ ЕГО УБИВАТЬ!
ПОЙДЕМ В ДОМ, ЧАЮ
ЗАВАРИМ, ПОИГРАЕМ?

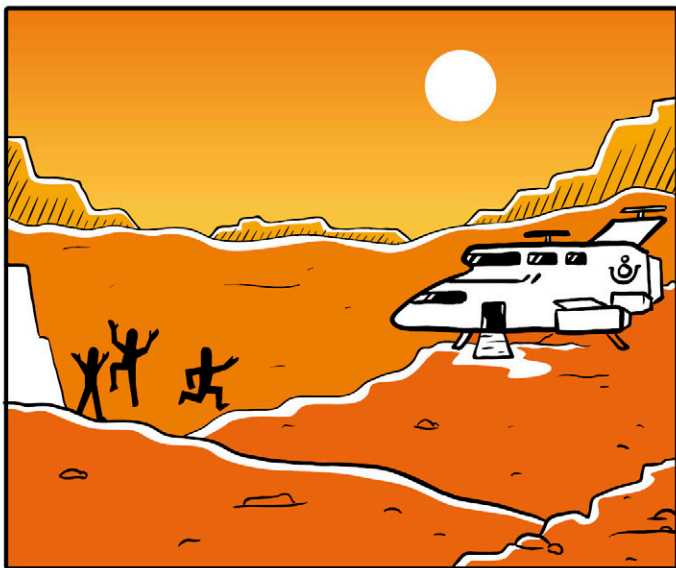
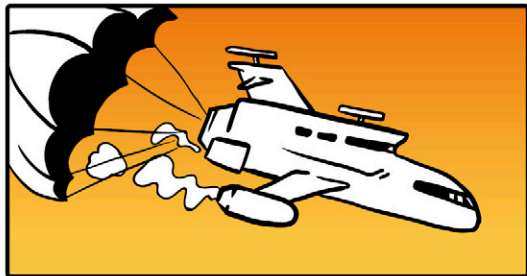
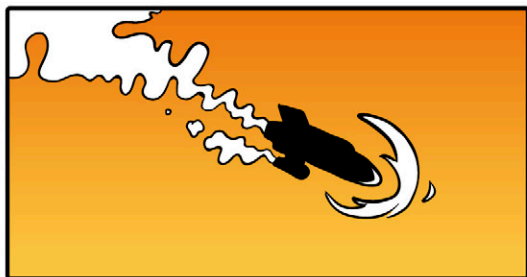
СПУСТЯ НЕКОТОРОЕ ВРЕМЯ

ИТАК, БУРЯ НАЧАЛАСЬ! СЛУШАЙ МОЮ
КОМАНДУ! ЗАДРАИТЬ ЛЮКИ! ВЫХОД НА
ПОВЕРХНОСТЬ СТРОГО ПО ДВОЕ! КАЖДЫЕ
ТРИ ЧАСА ОЧИЩАТЬ СОЛНЕЧНЫЕ БАТАРЕИ
И ПРОВЕРЯТЬ ПОКАЗАНИЯ!

СОЛ 50





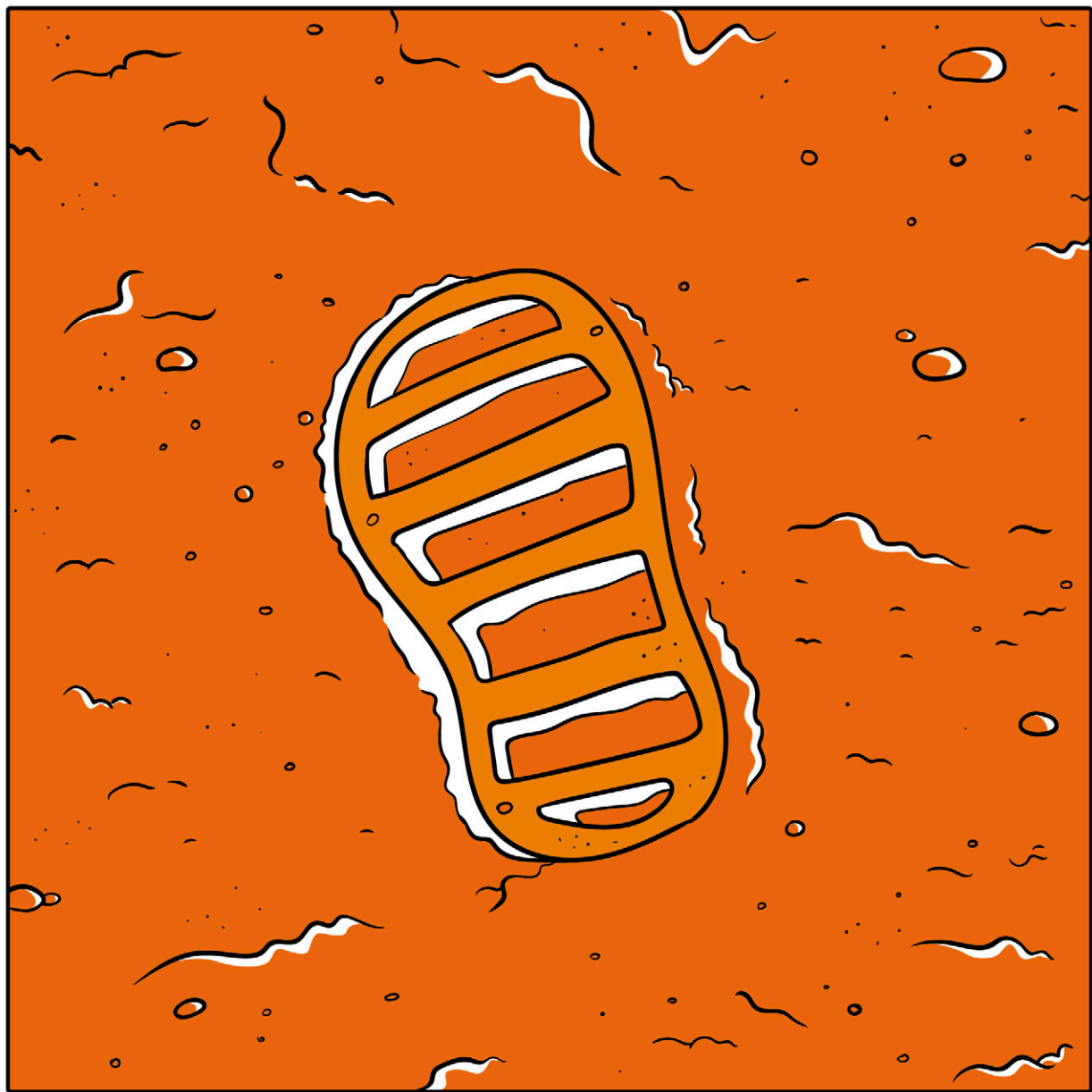


НОВОЕ НАЧАЛО

За исключением программиста Жоры, весь остальной груз, прибывший в нашу колонию с Земли на 74-й сол, оказался одной большой хорошей новостью. Наконец-то! Во-первых, Жора был не единственным пассажиром, и мы испытали огромное облегчение, когда узнали об этом. С ним прибыли микробиолог, два инженера и врач. Во-вторых, у нас появилась долгожданная связь с Землёй. Мы смогли поговорить со своими родными и близкими. Мы узнали, что о наших приключениях снимают фильм, пишут книгу и, конечно, рисуют аниме, а нас самих считают национальными героями.

История Дэма Мэйтона произвела настоящий фурор. Специалисты космических агентств всего мира тут же бросились разрабатывать план его доставки на Землю, но он неожиданно заявил, что решил остаться. «Ничто так круто не хрустит на зубах, как оксид железа из здешней почвы», — сообщил он журналистам.

А ещё у нас наконец-то появилась оранжерея, портативная атомная электростанция, второй жилой блок для кампуса и игровая приставка. Впереди много трудностей и опасностей, но жизнь налаживается. Главное — бодрость духа и надёжные товарищи.



ЧТО ДАЛЬШЕ

Поздравляем! Благодаря тебе команда колонистов выжила и справилась со всеми трудностями.

Мы изучили среду программирования XOD IDE, и теперь ты можешь программировать микроконтроллеры самостоятельно. Придумай свои улучшения для колонии или воспользуйся нашими идеями:

- Добавь датчики газов, чтобы внутри кампуса всегда был правильный состав воздуха.
- Заберись в самую глубь ноды X и попробуй написать свою мелодию для зуммера.
- Замени солнечную батарею на настоящую, снабди её дополнительными датчиками освещённости и улучши программу так, чтобы панель сама следила за самым ярким источником света.
- Собери свою большую и классную марсианскую колонию и придумай новые строения — ветряной электрогенератор с коллекторным мотором, оранжерею и настоящий купол, под которым будет маленький город с улицами, зданиями и деревьями.
- Построй свой марсоход, управляемый или автономный.



Мы, команда «Амперки», надеемся, что тебе понравился наш набор. Подробные инструкции ты найдёшь на нашем сайте wiki.amperka.ru.

Новые наборы и компоненты можно заказать в нашем магазине amperka.ru.



vk.com/amperkaru



facebook.com/amperka.ru



instagram.com/amperkaru



twitter.com/amperka

