



ИНТЕЛЛЕКТУАЛЬНЫЙ  
МЕГАПОЛИС

# МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ



ИТ-класс

В МОСКОВСКОЙ ШКОЛЕ

## НАПРАВЛЕНИЕ РОБОТОТЕХНИКА

## ПРАКТИЧЕСКИЙ ЭТАП

МОСКВА  
2025





ИНТЕЛЛЕКТУАЛЬНЫЙ  
МЕГАПОЛИС

# МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ РАЗРАБОТАНЫ:

**Архипов Максим Викторович**, к.т.н., доцент кафедры «Автоматика и управление»,  
Федеральное государственное автономное образовательное учреждение высшего  
образования "Московский политехнический университет" (Московский Политех)

**Малкаров Артем Юрьевич**, преподаватель кафедры «Автоматика и управление»,  
Федеральное государственное автономное образовательное учреждение высшего  
образования "Московский политехнический университет" (Московский Политех)

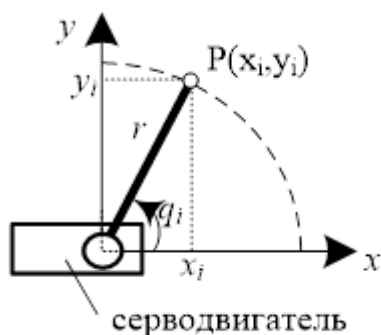
МОСКВА  
2025

## Оглавление

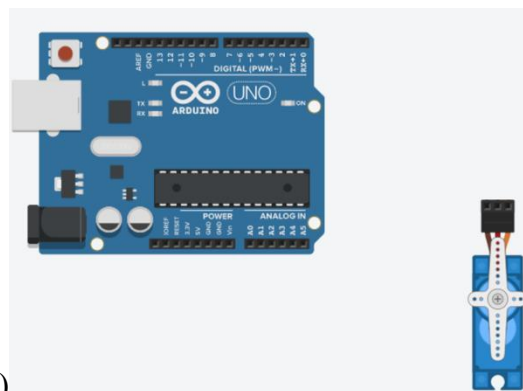
|      |                                                               |    |
|------|---------------------------------------------------------------|----|
| 1.   | Общая формулировка задачи кейса 1 .....                       | 4  |
| 1.1. | Пример выполнения заданий кейса 1 .....                       | 6  |
| 1.2. | Критерии оценивания выполненных пунктов заданий кейса 1.....  | 13 |
| 2.   | Платформа Tinkercad .....                                     | 15 |
| 2.1. | Регистрация .....                                             | 15 |
| 2.2. | Копирование проектов .....                                    | 17 |
| 2.3. | Изменение проектов .....                                      | 18 |
| 2.4. | Эмуляция и отладка.....                                       | 19 |
| 3.   | Общая формулировка задачи кейса 2. ....                       | 21 |
| 3.1. | Пример выполнения заданий кейса 2 .....                       | 24 |
| 3.2. | Критерии оценивания выполненных пунктов заданий кейса 2. .... | 29 |
| 4.   | Платформа Trik Studio.....                                    | 31 |
| 5.   | Инструкция по эксплуатации робота TRIK .....                  | 36 |

## 1. Общая формулировка задачи кейса 1.

Используя компоненты роботизированного конструктора, необходимо собрать схему и разработать программу в среде программирования. В программе реализовать управление вращением балки, прикрепленной к валу серводвигателя. Серводвигатель подключается к любому порту микроконтроллера с выходом ШИМ (~, т.е. широтно-импульсная модуляция). Схематично система показана ниже на рисунке 1а и электрическая схема с убранными соединительными проводами на рисунке 1б. Точка Р с координатами  $x, y$  перемещается по окружности, которую описывает незакрепленный край балки  $r$  в диапазоне от 0 до 180 град. При вращении балка  $r$  образует с осью  $x$  угол  $q$ . Требуется вычислить по произвольно введенным в командную строку координатам  $x, y$  точки Р, угол наклона балки  $r$ , отсчитываемого от оси  $x$  в положительном направлении вращения (т.е. против часовой стрелки). Размерность  $x, y$  – сантиметры. После введения  $x, y$  в командной строке должен выводиться результат – угол  $q$  в градусах, на который должна повернуться балка  $r$ . Вычисления проводить в четырех четвертях плоскости «х-у».



а)



б)

Рис. 1. Схема управления вращением балки серводвигателя: а) схематичная, б) электрическая (соединительные провода убраны)

### Задания.

#### Задание № 1. Электрическая схема

Нарисовать на листе бумаги электрическую схему подключения соединительных проводов, используя условные графические обозначения элементов с маркировкой и нумерацией всех контактов.

#### Задание № 2.1. Сборка электрической схемы

Выполнить монтаж реальной электрической схемы, используя предложенные комплектующие. Подача на схему электропитания производится после проверки всех подключений вместе с проверяющим. По запросу может быть выдана готовая схема подключений (при этом снимается 5 баллов). Эксперт фиксирует результат (с применением фотофиксации).

Альтернативный способ выполнения (участник может определиться с вариантом реализации задания заранее или непосредственно на площадке) задания 2.1 в среде виртуального моделирования Tinkercad (circuits), используя предложенные виртуальные комплектующие. Эксперт фиксирует результат (с сохранением результатов любым способом).

### **Задание № 2.2. Проектирование электрической схемы**

Выполнить разработку электрической схемы и печатной платы в программной среде Fritzing. Файл проекта схемы сохраняется на компьютере.

### **Задание № 3. Блок-схема алгоритма**

Составить подробную блок-схему алгоритма на листе бумаги, включающую разделы инициализации, интерфейсных объявлений и реализации с условными конструкциями.

### **Задание № 4.1. Управляющая программа**

Разработать программу управления сервомотором из робототехнического конструктора. Программа должна обеспечивать выполнение сервомотором базовых движений, перемещать балку на валу сервомотора в нулевое положение, вращать вал мотора в разные стороны без превышения пределов работоспособности.

Альтернативный способ выполнения задания 4.1 в среде виртуального программирования Tinkercad (circuits, code).

Результаты в виде файла с кодом управляющей программы сохраняются на компьютере.

### **Задание № 4.2. Монитор последовательного порта**

В дополнение к пункту 4.1. выполнить разработку части программы по перемещению вала сервомотора на заданный угол. Математический расчет перемещения балки в заданную

точку необходимо выполнить на бумажном бланке. Результаты работы программы должны совпасть с расчетными значениями. Программа должна воспринимать координаты точек маршрута, а в случае ошибочно введенных оператором данных – сообщить в мониторе последовательного порта о требуемом формате ввода данных.

Альтернативный способ выполнения задания 4.2 в среде виртуального программирования Tinkercad (circuits, code).

Результаты в виде файла с кодом управляющей программы должны быть сохранены на компьютере.

### **Необходимое оборудование:**

Микроконтроллер из роботизированного конструктора, сервомотор конечного вращения с балкой любого типа, соединительные провода.

При выполнении альтернативных заданий 2.1, 4.1, 4.2 необходим персональный компьютер с доступом к среде виртуального программирования Tinkercad, установленной программой Fritzting.

## **1.1. Пример выполнения заданий кейса 1**

### **1.1.1. Задание № 1. Электрическая схема**

По данному пункту требуется нарисовать на листе бумаги схему подключения серводвигателя к плате Arduino UNO. Рисунок может быть схематичным используя условные графические обозначения электрических компонентов в виде квадратов и стрелочки, например как на рис. 2.

В схемы должны быть отражены следующие условные обозначения:

- VCC (Питание, красный провод): подключается к пину 5 V на Arduino;
- GND (Земля, черный/коричневый провод): подключается к пину GND на Arduino;
- Signal (Сигнальный, желтый/оранжевый провод): подключается к цифровому пину с поддержкой ШИМ (PWM, обозначается ~). Выберем пин ~9.

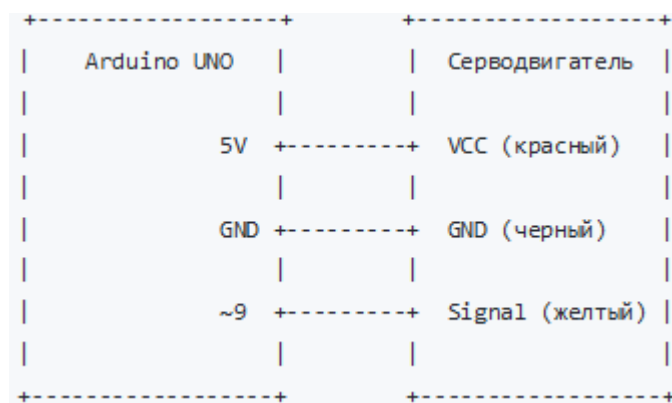


Рис. 2. Схематичное изображение электрической схемы

### 1.1.2. Задание № 2.1. Сборка электрической схемы

Монтаж выполняется на основе предложенных компонентов (Arduino, серводвигатель, соединительные провода) или в среде виртуального моделирования Tinkercad (рис. 3 а, б).

Порядок действий при сборке схемы на реальных компонентах:

1. Взять соединительный провод (например, красного цвета) и подключить один его конец к пину **5V** на плате Arduino, а другой — к контакту **VCC** (питание) серводвигателя.
2. Взять другой провод (например, черного цвета) и подключить один его конец к одному из пинов **GND** на Arduino, а другой — к контакту **GND** (земля) серводвигателя.
3. Взять третий провод (например, желтого цвета) и подключить один его конец к пину **~9** на Arduino, а другой — к сигнальному контакту **Signal** серводвигателя.
4. Схема собрана. Перед подачей питания (подключением USB-кабеля) необходимо провести проверку правильности подключений.

Порядок действий при сборке схемы в Tinkercad:

1. Создать новый проект в Tinkercad Circuits.
2. Перетащить из библиотеки компонентов Arduino UNO R3 и Micro Servo.
3. Соединить проводами соответствующие пины, как описано в задании 1, выбрав для них подходящие цвета.

Непосредственно на мероприятии, доступ в Tinkercad предоставляется организаторами через специальный виртуальный класс. Для подготовки доступ к



Tinkercad участники получают самостоятельно с помощью различных расширений браузера google.

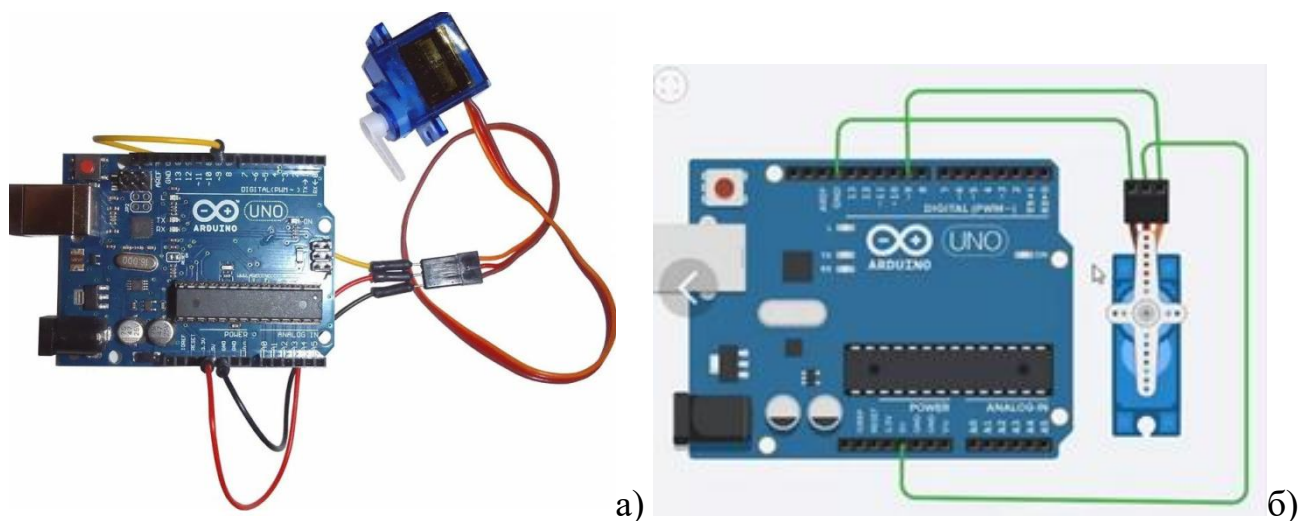


Рис. 3. Схема установки: а) на Arduino и сервомоторе SG90, б) в Tinkercad

Участник может выполнить один вариант решения этого задания, либо собрать схему как на рис. 3 а, либо собрать схему в Tinkercad (рис. 3 б). При выполнении обоих вариантов удвоение баллов не производится.

### 1.1.3. Задание № 2.2. Проектирование электрической схемы

Порядок действий при выполнении задания:

1. Запустить программу Fritzing.
  2. Во вкладке «Макетная плата» перетащить из библиотеки компонентов Arduino Uno и Servo (сервопривод).
  3. Соединить контакты виртуальными проводами в соответствии со схемой из Задания 1.
  4. Переключиться на вкладку «Принципиальная схема». Fritzing автоматически сгенерирует схему. При необходимости можно поправить расположение элементов для лучшей читаемости.
  5. Переключиться на вкладку «Печатная плата» для проектирования РСВ (в рамках данного задания это может быть необязательным, но для полноты выполнения стоит продемонстрировать).
  5. Сохранить проект в файл с именем Фамилия\_Имя\_Fritzing.fzz .
- Созданный проект сохраняется на рабочем столе в именной папке.



Пример реализации задания в Fritzing показан на рис. 4.

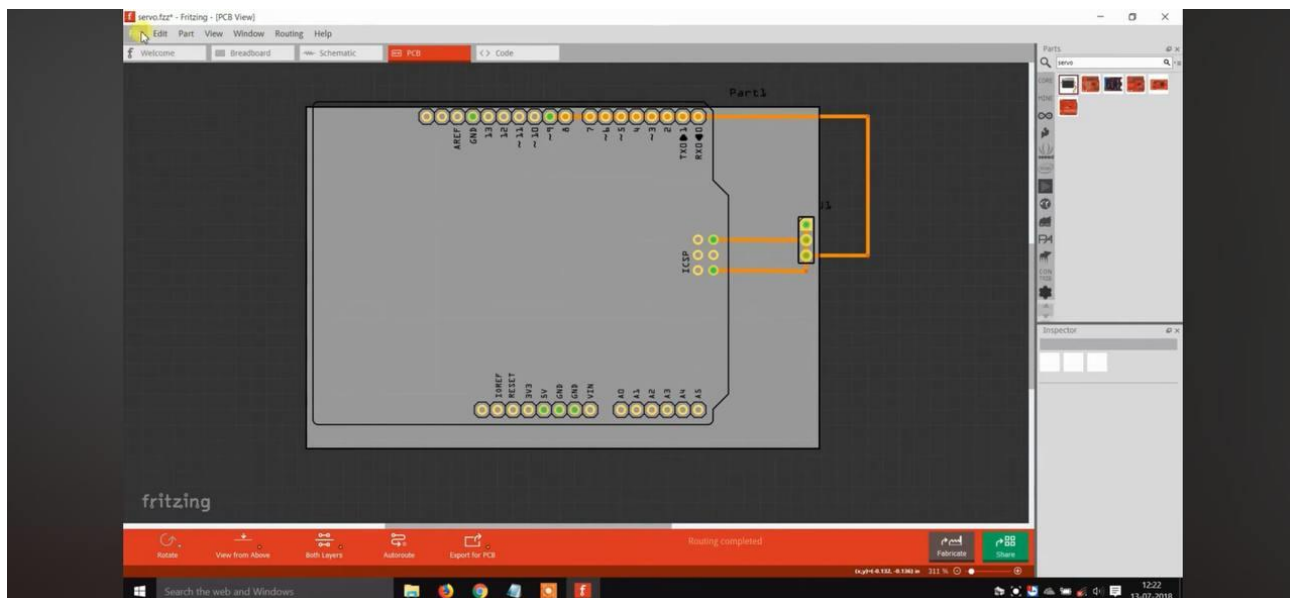


Рис. 4. Схема установки в программе Fritzing

#### 1.1.4. Задание № 3. Блок-схема алгоритма

Блок-схема описывает логику управляющей программы (рис. 5).

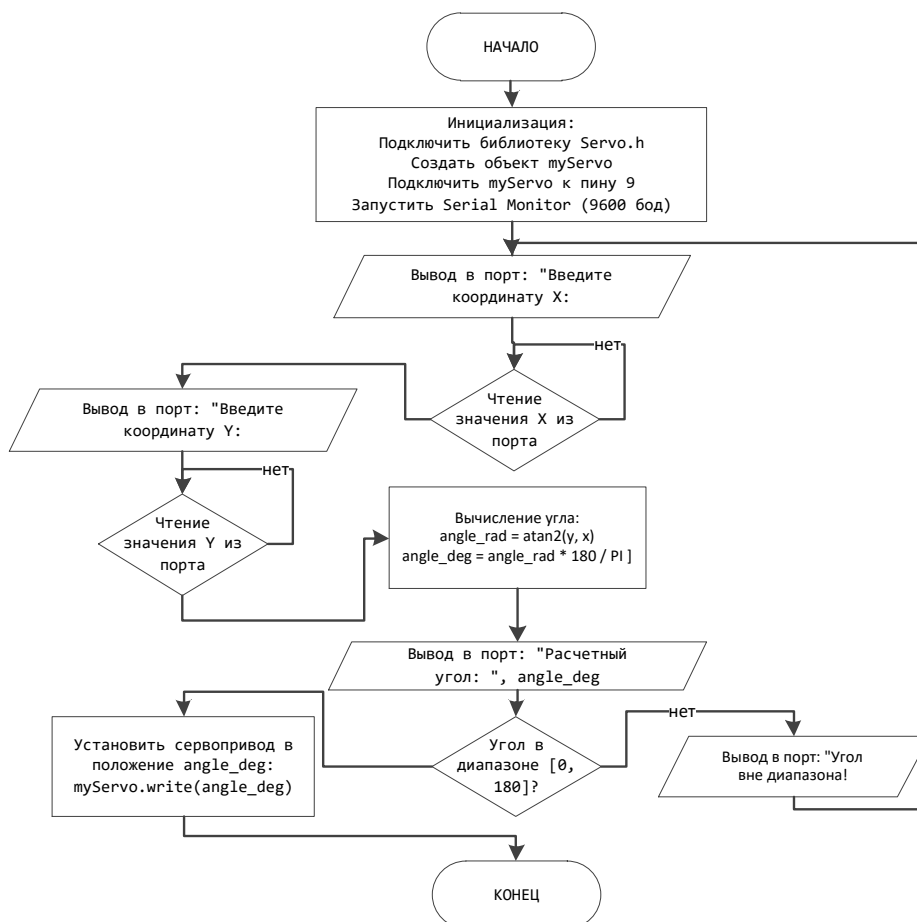


Рис. 5. Блок-схема алгоритма работы

### 1.1.5. Задание № 4.1. Управляющая программа и 4.2. Монитор последовательного порта

Необходимо разработать код управляющей программы, который включает требования по заданиям пунктов 4.1 и 4.2. Программа должна выполнять следующие функции: управление сервомотором, математический расчет для всех четырех четвертей, диалог с пользователем и комментарии.

Математическое обоснование программы. Для нахождения угла вектора (x, y) относительно положительного направления оси X используется арктангенс. Функция `atan2(y, x)` идеально подходит для этой задачи, так как она корректно обрабатывает все четыре квадранта и возвращает результат в радианах в диапазоне  $[-\pi, \pi]$ . Для управления сервомотором результат нужно перевести в градусы (диапазон  $[0, 180]$ ) по формуле:  $\text{градусы} = \text{радианы} * 180 / \text{PI}$ .

Рассмотрим код управляющей программы с комментариями.

Необходимо задать всем используемым портам и значениям имена, это повысит понимание кода и при необходимости упростит его изменение. Для работы с сервомоторами выполним импорт библиотеки `Servo.h` в самом начале программы с помощью директивы `#include`. Математическая библиотека `math.h`.

```
// Подключаем необходимые библиотеки
#include <Servo.h> // Библиотека для управления серводвигателями
#include <math.h> // Математическая библиотека для функции atan2 и константы PI
// --- Задание 4.1: Базовая инициализация ---
```

Объявление и инициализация объектов `Servo` и пинов для светофоров (светодиодов):

```
// Создаем объект для управления сервомотором
Servo myServo;
// Пин, к которому подключен сигнальный провод сервомотора
```

```
const int servoPin = 9;
```

Начальная установка:

```
void setup() {  
  // Инициализация последовательного порта для общения с компьютером  
  Serial.begin(9600);  
  // Привязываем объект сервомотора к пину  
  myServo.attach(servoPin);  
  // Установка сервомотора в начальное (нулевое) положение  
  myServo.write(90); // Установим в среднее положение для старта  
  // Приветственное сообщение для пользователя  
  Serial.println("Система управления сервомотором готова.");  
  Serial.println("-----");  
}
```

Данная программа на Arduino используется для управления сервоприводом. В функции **setup** вызывается функция **servo\_setup** для инициализации сервопривода. Затем в функции **loop** происходит последовательное управление сервоприводом в заданном порядке.

```
void loop() {  
  // --- Задание 4.2: Диалог с пользователем и расчеты --  
  float x, y;  
  // 1. Запрос и чтение координат  
  Serial.println("Введите координату X и нажмите Enter:"); while (Serial.available() ==  
  0) {  
    // Ждем, пока пользователь введет данные  
  } x = Serial.parseFloat();  
  Serial.println("Введите координату Y и нажмите Enter:");  
  while (Serial.available() == 0) {
```

```

    // Ждем, пока пользователь введет данные
} y = Serial.parseFloat();

    // Очистка буфера от лишних символов (например, '\n')
while(Serial.available() > 0) {
    Serial.read();
}

// 2. Математический расчет угла
// Функция atan2(y, x) возвращает угол в радианах для всех 4-х квадрантов
double angle_rad = atan2(y, x);

    // Переводим радианы в градусы
double angle_deg = angle_rad * 180.0 / M_PI;

Serial.print("Введены координаты: X = ");
Serial.print(x);
Serial.print(", Y = ");
Serial.println(y);
Serial.print("Расчетный угол: ");
Serial.println(angle_deg);

// 3. Управление сервомотором
// Сервомотор имеет рабочий диапазон 0-180 градусов.
// Если вычисленный угол находится в этом диапазоне, поворачиваем балку.
if (angle_deg >= 0 && angle_deg <= 180) {
    Serial.print("Угол в рабочем диапазоне. Поворачиваю на ");
    Serial.print((int)angle_deg);
    Serial.println(" градусов.");
    myServo.write((int)angle_deg); // Команда на поворот
} else {
    // Если угол выходит за пределы (например, отрицательный), сообщаем об этом
    Serial.println("Ошибка: Расчетный угол вне рабочего диапазона сервомотора (0-180).");
}

    Serial.println("-----");
delay(1000); // Небольшая задержка перед следующим циклом
}

```

Программный код разрабатывается или в приложении Arduino IDE на компьютере или на платформе Tinkercad. При выполнении обоих вариантов удвоение баллов не производится.

## **1.2. Критерии оценивания выполненных пунктов заданий кейса 1.**

### **1.2.1. Оценка процесса решения схемотехнической части задачи, включая владение практическими навыками:**

- Нарисована электрическая схема, соответствующая условию задачи - **5 баллов**. При наличии ошибок – **3 балла**.
- Собранная электрическая схема, соответствующая условию задачи - **10 баллов**. Если выдана готовая схема - **0 баллов**.
- Спроектирована электрическая схема в программе Fritzing - **15 баллов**. При наличии ошибок – **8 баллов**. При отсутствии файла схемы - **0 баллов**.

### **1.2.2. Оценка и результатов работы программной части:**

- Нарисована блок-схема алгоритма и реализована базовая программа управления сервомотором – **5 баллов**. Реализована только блок-схема или базовый код – **3 балла**.
- Разработан листинг кода, в котором реализовано решение для всех четырех областей плоскости – **до 10 баллов**. Программный код выполняет вычисление не для всех областей – **5 баллов**.

- Программа снабжена текстовыми диалоговыми сообщениями для оператора, который вводит данные и наблюдает результат вычисления. Автор выполнил демонстрацию работы реальной программы и имеется файл кода с комментариями – **15 баллов**. При наличии ошибок – **8 баллов**. При отсутствии файла программы - **0 баллов**.

### **1.2.3. Критерии снижения баллов**

Максимальный балл за выполнение всех заданий кейса 2 – 60 баллов. В каждом задании кейса 1 предусмотрен набор датчиков и исполнительных элементов для микроконтроллера Arduino, сходных по сложности для программирования.

Проверка правильности подключения электронных компонентов в соответствии со схемой проверяется визуально. За правильное проектирование и сборку участник получает максимальный балл. Если допущены незначительные ошибки, но конструкция пригодна к тестированию (пригодность к тестированию определяется правильностью подключения электронных компонентов, отсутствием ошибок в подключении к микроконтроллеру, управляющих линий, отсутствием ошибок подключения модулей при соединении линий 5V, GND), итоговый балл может быть снижен на 1 балл за каждую ошибку. Если при сборке были допущены ошибки, приводящие к выходу компонентов из строя, участник получает 0 баллов.

Правильность разработанного алгоритма проверяется визуально. За алгоритм, отображающий предлагаемый вариант решения и не содержащий логических ошибок, оформленный в соответствии со стандартом, участник получает максимальный балл. Если допущены незначительные ошибки, но алгоритм может быть выполнен, итоговый балл может быть снижен на 1 балл за каждую ошибку. Если алгоритм был выполнен без использования специальных условных графических блоков или оформлен небрежно, итоговый балл может быть снижен на 1 балл. Если алгоритм содержит грубые логические ошибки, участник получает 0 баллов.

Правильность разработанного алгоритма проверяется визуально. За реализацию задания, укладывающуюся в контрольное время и не содержащую ошибок, участник получает максимальный балл. Если допущены ошибки, но задание было выполнено, итоговый балл может быть снижен на 5 баллов за каждую ошибку. Если задание не было выполнено, участник получает 0 баллов. Если задание было выполнено, но с отклонением от требований в задании, итоговый балл может быть снижен, при отсутствии вывода данных в монитор последовательного порта, итоговый балл может быть снижен на 5 баллов. При полном отсутствии вычислений – на 10 баллов.

Для получения максимального балла за практический этап Конкурса необходимо представить папку с фалами по заданиям кейса, содержащим изображения алгоритмов подписанном и оформленном на бумажном бланке, программу в виде блоков или в виде кода для реализации базового задания и задания повышенной сложности. Папка



должна содержать ФИО участника в названии. Необходимо выполнить все задания и не превысить временной лимит, предусмотренный организатором для выполнения заданий.

При реализации заданий в средах Trikstudio проверяются те же критерии, что и при проверке реального робота, с добавлением дополнительного – необходимостью сохранения проектов каждого подзадания отдельным файлом на рабочем компьютере или сетевой папке.

## 2. Платформа Tinkercad

### 2.1. Регистрация

Заходим на страницу <https://www.tinkercad.com/>, нажимаем кнопку “Регистрация” (рис. 6). В связи с ограничением доступа к платформе Tinkercad возможны следующие варианты работы:

- через доступ к виртуальному классу по паролю;
- через установку специального расширения на браузер помогающего против авторизацию.

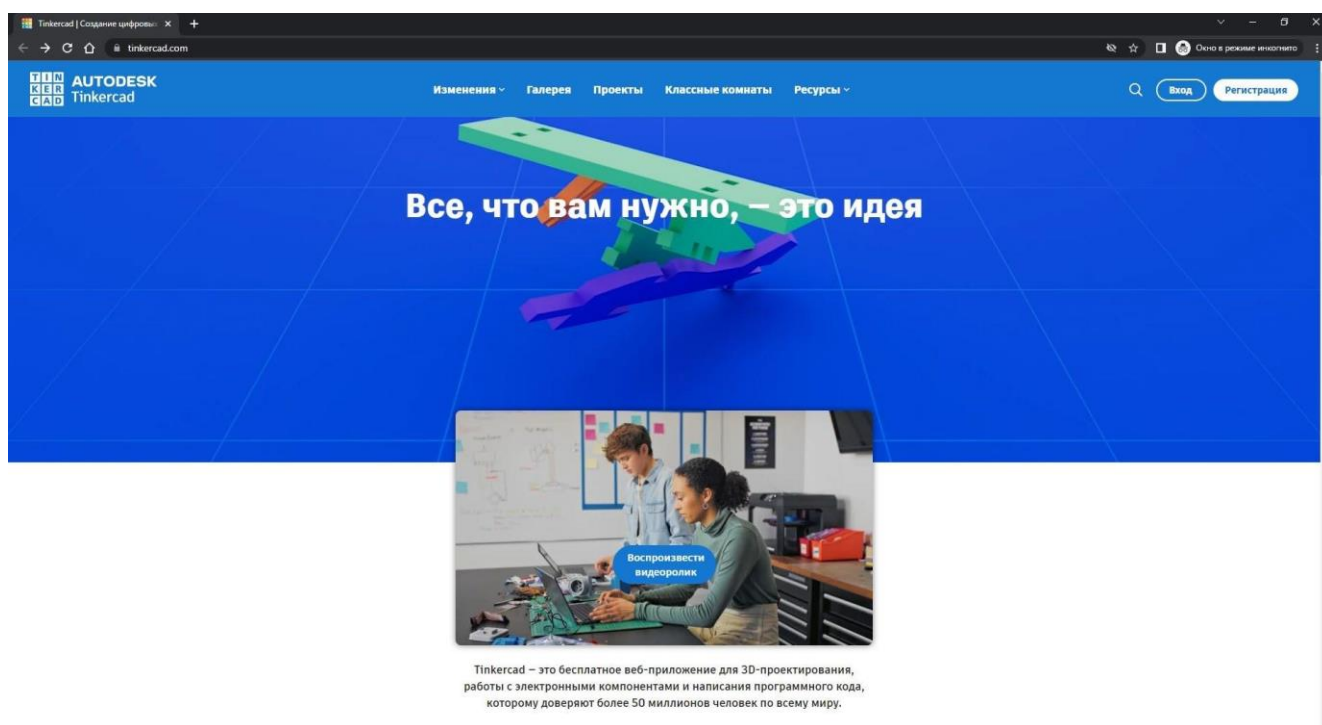


Рис. 6. Начальная страница регистрации

Выбираем вариант “Создать персональную учетную запись” (рис. 7).

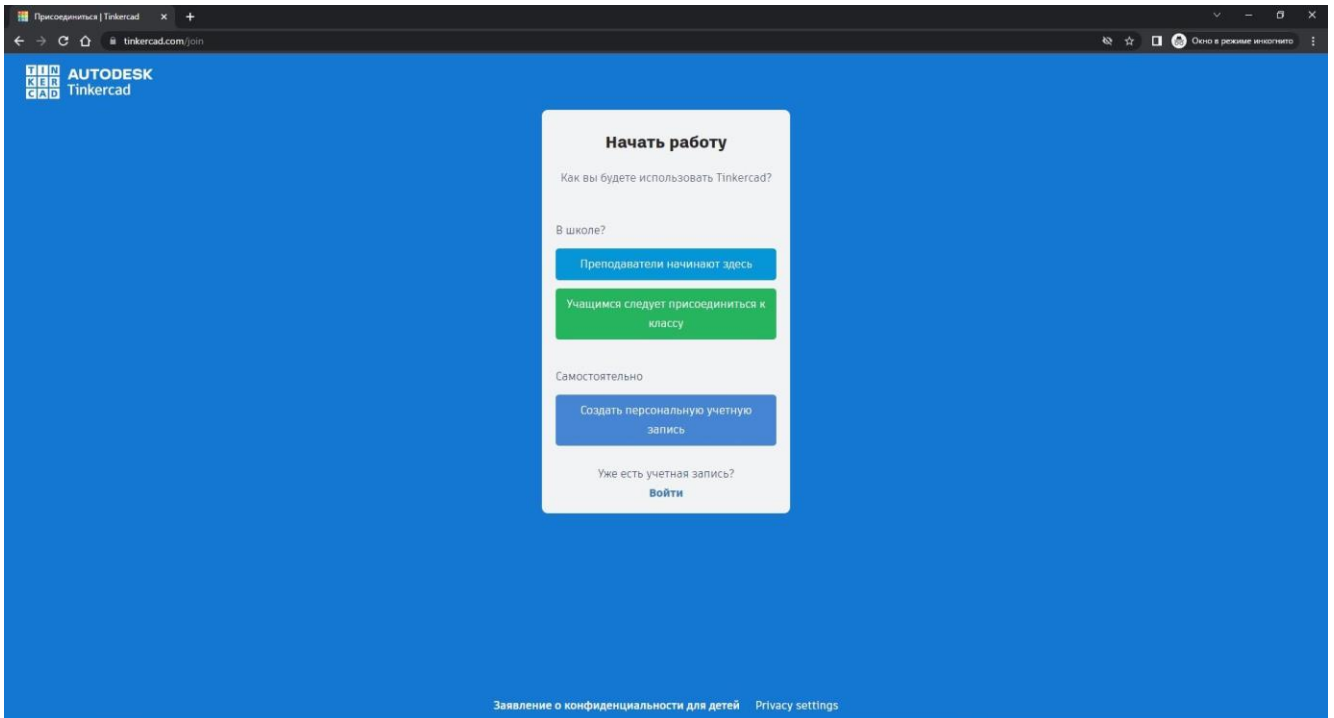


Рис. 7. Страница выбора режима учетной записи

Можно выбрать любой из вариантов регистрации, например с помощью почты (рис. 8).

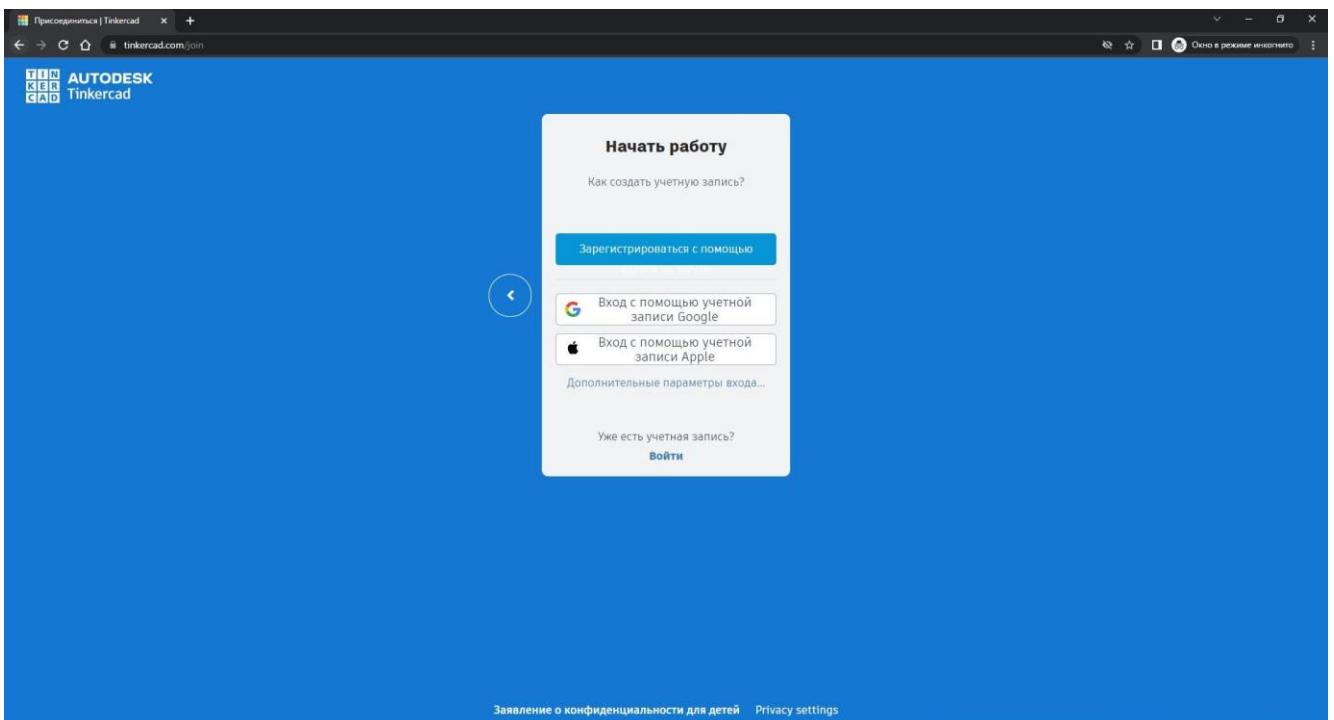


Рис. 8. Регистрация через электронную почту

После регистрации или входа пользователь попадает на главную страницу (рис. 9). На ней располагаются последние проекты – 3Д объекты, цепи и блоки кода.

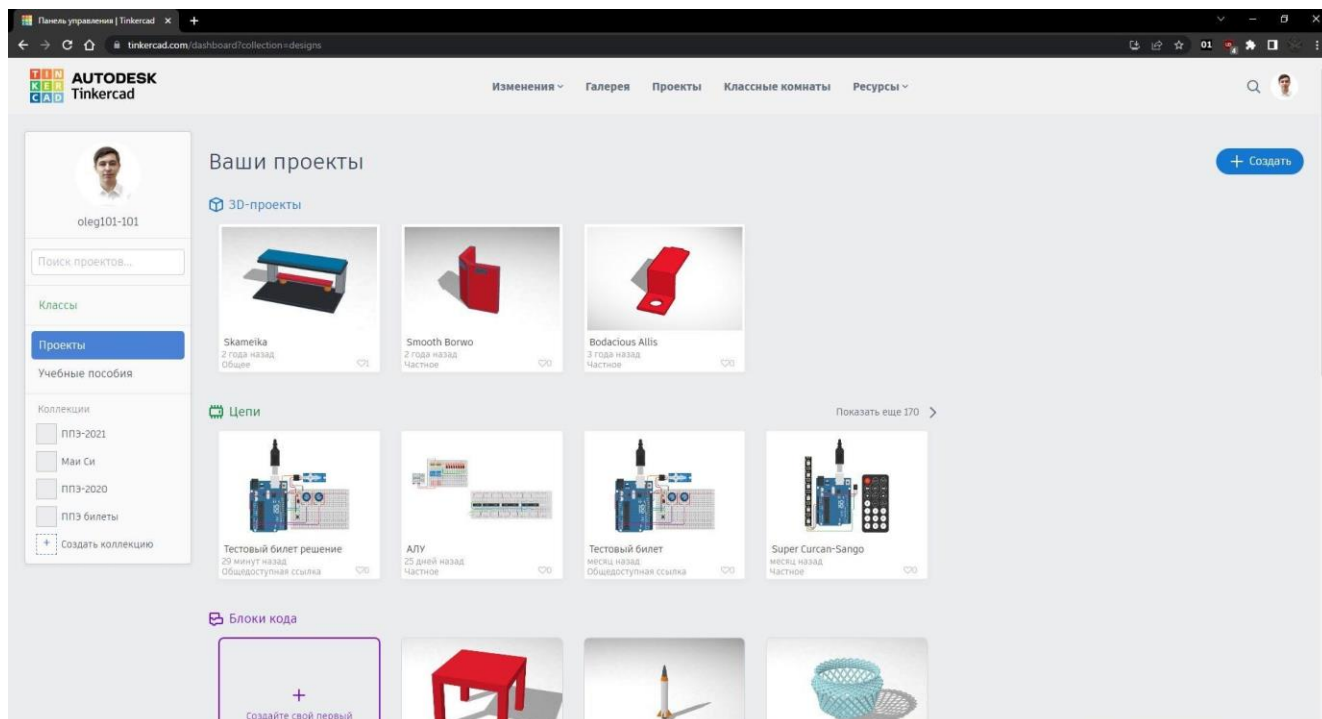


Рис. 9. Главная страница проектов Tinkercad

## 2.2. Копирование проектов

После получения ссылки на проект, или выбора проекта из галереи пользователь может запустить его, чтобы посмотреть на его работу, или скопировать его себе для хранения или внесения изменений с помощью кнопки “Копировать и изменить” (рис. 10).

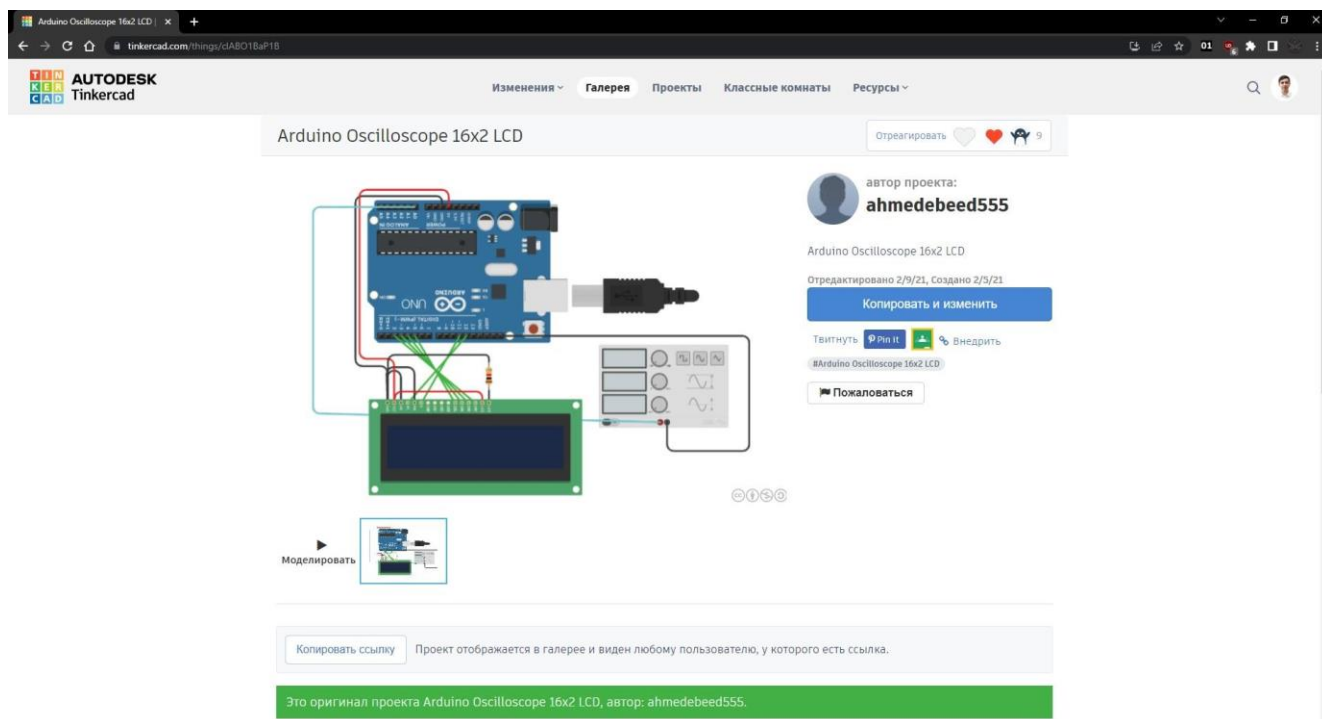


Рис. 10. Страница создания проекта в Tinkercad

## 2.3. Изменение проектов

При работе с проектами пользователь может добавлять необходимые элементы из панели, расположенной справа (рис. 11), элементы достаются перетягиванием “ Drag-and-drop ” или нажатием левой кнопкой на элемент для выбора, переносом в рабочее пространство и повторным нажатием для расположения элемента. По умолчанию выбраны только основные компоненты, можно переключить меню для отображения всех доступных элементов.

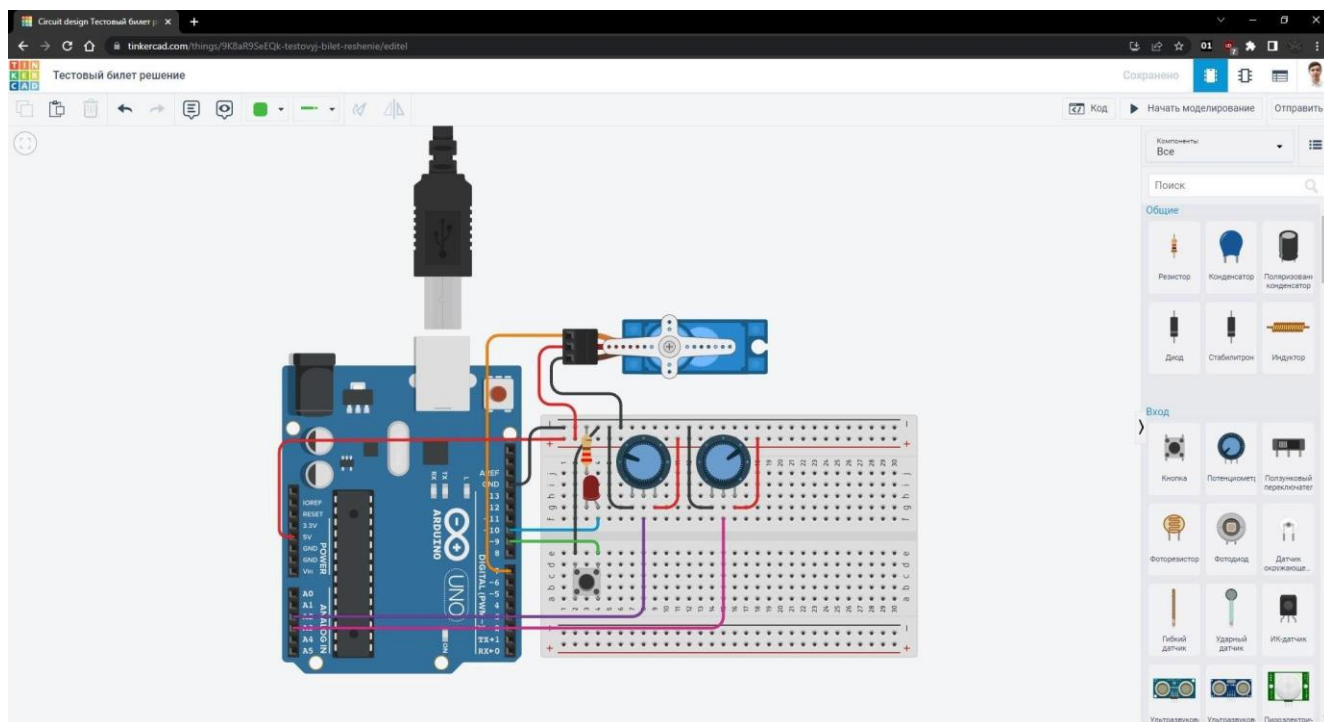


Рис. 11. Меню электрических компонентов

Элементы можно соединять проводами между собой, для создания провода необходимо нажать левой кнопке мыши на возможной точке старта – контакте элемента или макетной платы, после этого провод можно подвести к второй точке подключения и вновь нажать левую кнопку мыши. На проводе можно создавать точки сгиба нажатием левой кнопке мыши на части провода, после его создания.

Для написания программ для микроконтроллеров используется вкладка код, в правом верхнем углу. Tinkercad поддерживает три способа программирования – Scratch-подобными блоками, текстовой программой на языке Си и смешанный формат, при котором код на блоках также показывается в виде кода на Си (рис. 12).

Переключение режима находится в верхней левой части окна с программой.

## 2.4. Эмуляция и отладка

После создания схемы и написания программы можно провести отладку системы, запустив её с помощью кнопки “Начать моделирование” в правом верхнем углу. В режиме моделирования пользователь может взаимодействовать с контроллером через Serial интерфейс, экран расположен в нижней части блока кода. Пользователь также может взаимодействовать с различными устройствами,

например, нажимать на кнопки или вращать потенциометры, нажимая на них левой кнопкой мыши.

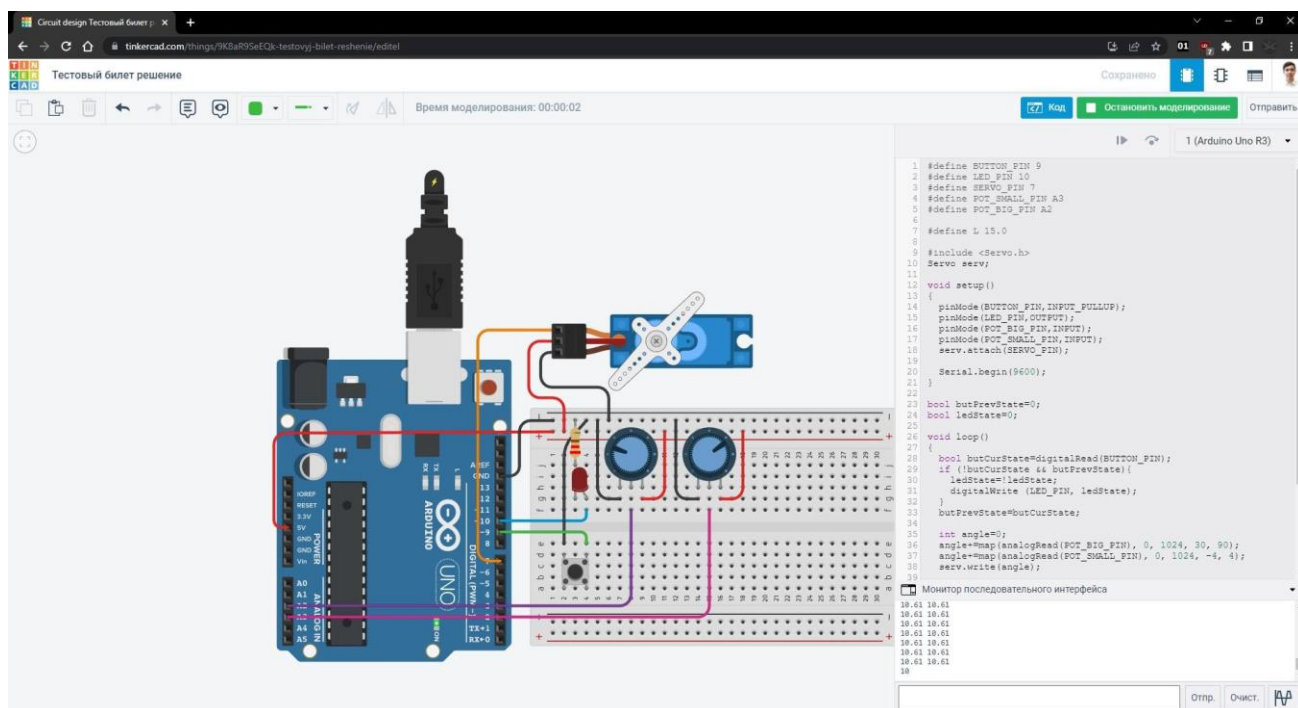


Рис. 12. Меню создания программного кода

Более сложные устройства, например датчик расстояния, предоставляют собственные интерфейсы взаимодействия с ними, интерфейсы активируются при выборе устройства левой кнопкой мыши (рис. 13).

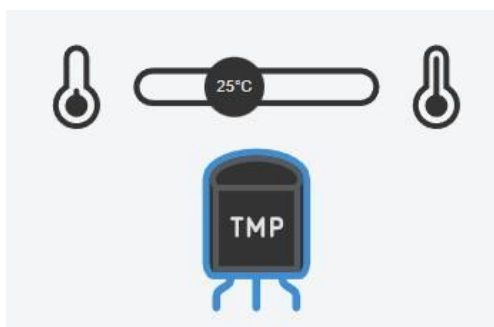


Рис. 13. Интерфейс датчика температуры в Tinkercad

### 3. Общая формулировка задачи кейса 2.

Для мобильного робота собранного из роботизированного конструктора ТРИК, представленного на рис. 14, необходимо разработать программу его перемещения по точкам заданного маршрута. Программа должна по полученным данным вычислить среднюю скорость перемещения робота за заданное время при равномерном перемещении робота по траектории из четырех точек. Необходимо написать программу, в которую включить заданные точки маршрута для того, чтобы программа рассчитала и вывела значения длины маршрута и средней скорости робота. Для обеспечения точности перемещения, колеса робота снабжены энкодерами.



Рис.14. Внешний вид мобильного робота на базе роботизированного конструктора ТРИК

#### Входные данные.

При написании программы необходимо учесть следующие заданные абсолютные координаты точек маршрута:

$x_0, y_0$  – координаты нулевой точки (А) маршрута, м.;

$x_1, y_1$  – координаты первой точки (В) маршрута, м.;

$x_2, y_2$  – координаты второй точки (С) маршрута, м.;

$x_3, y_3$  – координаты третьей точки (D) маршрута, м.;

Робот движется равномерно по траектории маршрута, состоящего из четырех точек. В точках на маршруте робот разворачивается на месте, на необходимый угол, для соответствия траектории, по которой происходит движение. Параметры координат точек маршрута и времени его прохождения задаются в тестовом файле в виде

вещественных чисел. В каждой строке с координатами два числа, первое  $x$ , второе  $y$ . Числа в строке разделены одинарным пробелом. В пятой строке указывается одно число – время. Размерность координат – метры, времени – секунды и миллисекунды (ограничение в два знака после точки). Границы координат не выходят за пределы от 0 до 10 метров по  $x$  и  $y$ .

Пример задания входных данных  $(x_0, y_0)$ ,  $(x_1, y_1)$ ,  $(x_2, y_2)$ ,  $(x_3, y_3)$ ,  $t$  для программы:

0.50 0.60

0.20 0.40

0.30 0.20

0.10 0.60

10

Участники самостоятельно разрабатывают программный код, вносят в него заданные численные значения, проводят промежуточные расчеты и реализуют вывод на дисплей микроконтроллера ТРИК вычисленной длины пройденного пути и скорости. Формулы необходимо самостоятельно преобразовать к программному виду и включить в программу. Обозначения параметров функций и переменных участники выбирают самостоятельно.

### **Выходные данные.**

Программа должна вывести на экран микроконтроллера два числа. В первой строке должно выводиться вычисленная программой длина всего маршрута в виде вещественного числа с двумя знаками после запятой, единица измерения - метры. Во второй строке должно выводиться вычисленная программой скорость робота в виде вещественного числа с двумя знаками после запятой, единица измерения - метры в секунду (м/с). Необходимо предусмотреть только вывод чисел. В качестве разделителя целой и вещественной части – точка. Текст комментариев и единицы измерения выводить не нужно.

Пример того, что должна вывести программа (даны для примера):

1.55



0.05

### **Задания.**

#### **Задание № 1. Вычисление параметров движения**

Выполнить необходимые расчеты параметров движения робота. Расчёты производятся на бумажном бланке. В расчетах необходимо отразить необходимые формулы, промежуточные вычисления и ответ. Подписанный ФИО бланк с расчётами необходимо сдать на проверку.

#### **Задание № 2. Блок-схема алгоритма**

Составить на бумажном бланке подробную и четкую блок-схему алгоритма, включая разделы инициализации, интерфейсных объявлений и реализации с условными конструкциями.

#### **Задание № 3.1. Управляющая программа (в базовом варианте – управление движением по времени)**

Разработать управляющую программу для робота в среде виртуального программирования ТРИК. На мобильном роботе программа должна обеспечивать выполнение: базовых движений, перемещение по точкам, развороты в точках на необходимый угол. Все перемещения в программе реализуются с использованием временных задержек. Определена общая длина пути, пройденного роботом, и его средняя скорость.

#### **Задание № 3.2. Управляющая программа (в усложненном варианте – управление движением по показаниям энкодеров)**

В дополнение к пункту 3.1 в среде ТРИК выполнена разработка части программы с использованием показаний энкодеров при перемещении, математические расчеты перемещения. Выполнен вывод вычисленных показаний общей длины траектории и средней скорости на экран микроконтроллера. Результаты работы программы совпадают с расчетными.

Результаты в виде файла с кодом управляющей программы сохранены на компьютере.

#### **Задание № 4.1. Тестирование в среде ТРИК**

Выполнить отладку и тестирование программ из пунктов 3.1, 3.2 в виртуальной среде ТРИК. В программе ТРИК во вкладке «отладка» в рабочем пространстве нанести рассчитанную траекторию с точками в любом пропорциональном масштабе. Робот должен перемещаться по намеченной траектории без отклонения. На экран контроллера необходимо вывести вычисленные длину пути и среднюю скорость. Проект с отлаженной программой должен быть сохранен на компьютере.

#### **Задание № 4.2. Тестирование на работе**

В дополнение к пункту 4.1 выполнить отладку и тестирование программы 3.1 или 3.2 на реальном мобильном роботе ТРИК. Тестирование программы выполняется проверкой пройденного роботом маршрута, предварительно нанесенного карандашом на лист размером 67,5 на 98 см (лист для флипчарта).

#### **Необходимое оборудование:**

Мобильный робот из роботизированного конструктора ТРИК.

При выполнении альтернативных заданий 3.1, 3.2, 4.1 необходим персональный компьютер с доступом к среде виртуального программирования Trik Studio.

### **3.1. Пример выполнения заданий кейса 2**

Формулировка задачи: Разработать программу для мобильного робота ТРИК, которая по заданным координатам четырех точек (А, В, С, D) и общему времени  $t$  вычисляет и выводит на экран общую длину маршрута и среднюю скорость движения. Робот должен проехать по этому маршруту.

Входные данные (пример):

A ( $x_0, y_0$ ) = (0.50, 0.60)

B ( $x_1, y_1$ ) = (0.20, 0.40)

C ( $x_2, y_2$ ) = (0.30, 0.20)

D ( $x_3, y_3$ ) = (0.10, 0.60)

$t = 10$  секунд

**3.1.1. Задание № 1. Вычисление параметров движения (на бумажном бланке).**

Формулы:

- Расстояние между двумя точками  $P1(x1, y1)$  и  $P2(x2, y2)$  :  $d = \sqrt{(x2-x1)^2 + (y2-y1)^2}$ );

- Общий путь (S):  $S = d(A,B) + d(B,C) + d(C,D)$  . Средняя скорость (V):  $V = S / t$ .

Промежуточные вычисления:

- Расстояние AB:  $d(A,B) = \sqrt{(0.20-0.50)^2 + (0.40-0.60)^2} = \sqrt{(-0.30)^2 + (-0.20)^2} = \sqrt{0.09 + 0.04} = \sqrt{0.13} \approx 0.3606$  м;

- Расстояние BC:  $d(B,C) = \sqrt{(0.30-0.20)^2 + (0.20-0.40)^2} = \sqrt{0.10^2 + (-0.20)^2} = \sqrt{0.01 + 0.04} = \sqrt{0.05} \approx 0.2236$  м;

- Расстояние CD:  $d(C,D) = \sqrt{(0.10-0.30)^2 + (0.60-0.20)^2} = \sqrt{(-0.20)^2 + 0.40^2} = \sqrt{0.04 + 0.16} = \sqrt{0.20} \approx 0.4472$  м.

Итоговые результаты:

- общий путь S, округляем до двух знаков после запятой: 1.03 м.  $S = 0.3606 + 0.2236 + 0.4472 = 1.0314$  м.

Средняя скорость V:  $V = 1.0314 \text{ м} / 10 \text{ с} = 0.10314 \text{ м/с}$ , округляем до двух знаков после запятой: 0.10 м/с.

Ответ для вывода на экран: первая строка: 1.03, вторая строка: 0.10.

### **3.1.2. Задание №2. Блок-схема алгоритма.**

Блок-схема описывает логику управляющей программы (рис. 15).

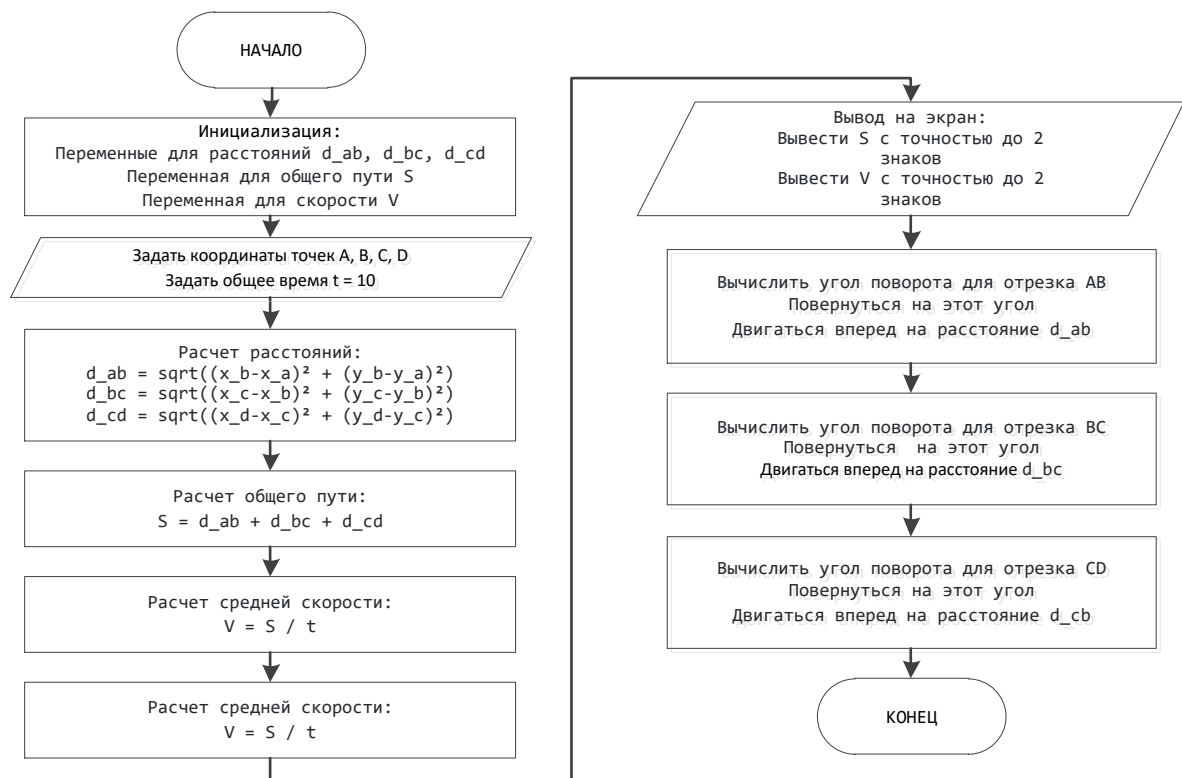


Рис. 15. Блок-схема алгоритма работы

### 3.1.3. Задание №3.1 и №3.2. Управляющая программа (логика для TRIK Studio)

Так как TRIK Studio использует блочное программирование или скриптовые языки (JS/Python), я представлю логику в виде псевдокода, который легко переносится в среду. Так как TRIK Studio использует блочное программирование или скриптовые языки (JS/Python), я представлю логику в виде псевдокода, который легко переносится в среду.

Общая часть (инициализация и расчеты):

```
// --- Инициализация переменных ---
```

```
x0 = 0.50, y0 = 0.60
```

```
x1 = 0.20, y1 = 0.40
```

```
x2 = 0.30, y2 = 0.20
```

```
x3 = 0.10, y3 = 0.60
```

```
t = 10
```

```
// --- Расчеты (как в Задании 1) ---
```

```

dist_ab = sqrt((x1-x0)^2 + (y1-y0)^2)
dist_bc = sqrt((x2-x1)^2 + (y2-y1)^2)
dist_cd = sqrt((x3-x2)^2 + (y3-y2)^2)
total_dist = dist_ab + dist_bc + dist_cd
avg_speed = total_dist / t
// --- Вывод результатов на дисплей ---
brick.display().print(round(total_dist, 2), 1, 1) // Вывод в 1 строку, 1 столбец
brick.display().print(round(avg_speed, 2), 2, 1) // Вывод во 2 строку, 1 столбец

```

Логика движения (Задание 3.1 - по времени, базовое): Этот метод не достаточно точен и требует калибровки. Нужно заранее определить, сколько времени робот должен ехать с определенной мощностью, чтобы преодолеть 1 метр, и сколько времени требуется на поворот на 1 градус.

```

POWER = 30 // Мощность моторов
TIME_PER_METER = 5000 // мс на 1 метр
TIME_PER_DEGREE = 20 // мс на 1 градус
// Функция движения по сегменту
function move_segment(x_start, y_start, x_end, y_end, dist):
// 1. Поворот
angle_rad = atan2(y_end - y_start, x_end - x_start)
angle_deg = angle_rad * 180 / PI
// Предполагаем, что есть функция поворота на абсолютный угол
// или вычисляем относительный поворот
turnToAngle(angle_deg) // Это сложная функция, для простоты считаем ее
существующей
// 2. Движение вперед
time_to_move = dist * TIME_PER_METER
motors.on(POWER, POWER)
script.wait(time_to_move)

```

```

motors.off()

// --- Основной блок движения ---

move_segment(x0, y0, x1, y1, dist_ab)
move_segment(x1, y1, x2, y2, dist_bc)
move_segment(x2, y2, x3, y3, dist_cd)

```

Логика движения (Задание 3.2 - по энкодерам, усложненное):

Этот метод гораздо точнее. Требуется калибровка констант: диаметр колеса и расстояние между колесами.

```

// Калибровочные константы
WHEEL_DIAMETER_M = 0.05 // Диаметр колеса в метрах (5 см)
WHEEL_CIRCUMFERENCE_M = PI * WHEEL_DIAMETER_M // Длина окружности
колёса
TICKS_PER_REVOLUTION = 360 // Сколько "тиков" энкодера на один оборот колеса
TICKS_PER_METER = TICKS_PER_REVOLUTION / WHEEL_CIRCUMFERENCE_M

// --- Функция движения по энкодерам ---
function moveForward(distance_m, power):
    encoder_left.reset()
    encoder_right.reset()
    target_ticks = distance_m * TICKS_PER_METER
    motors.on(power, power)
    while ( (encoder_left.read() + encoder_right.read()) / 2 < target_ticks ):
        script.wait(10) // Маленькая задержка, чтобы не загружать процессор
    motors.off()

// --- Функция поворота по энкодерам (или с гироскопом, если есть) ---
// ... (реализация поворота сложнее, но основана на вращении колес в разные стороны
// на определенное количество тиков энкодера, рассчитанное из геометрии робота)
// --- Основной блок движения (аналогично 3.1, но с вызовом точных функций) ---
turnToAngle(atan2(y1-y0, x1-x0))

```

```
moveForward(dist_ab, 30)
turnToAngle(atan2(y2-y1, x2-x1))
moveForward(dist_bc, 30)
turnToAngle(atan2(y3-y2, x3-x2))
moveForward(dist_cd, 30)
```

### **3.1.4. Задание №4.1 и №4.2. Тестирование**

#### **4.1. Тестирование в среде TRIK (виртуально):**

1. Создать проект в TRIK Studio, реализовать программу из п. 3.2.
2. Перейти во вкладку «Отладка».
3. Включить 2D-модель робота.
4. Нарисовать на виртуальном поле рассчитанную траекторию (линии, соединяющие точки A, B, C, D в нужном масштабе).
5. Запустить программу.
6. Наблюдать за движением виртуального робота. Он должен перемещаться по намеченной траектории без значительных отклонений. На экране контроллера должны отобразиться вычисленные значения 1.03 и 0.10.
7. Сохранить проект с отлаженной программой.

#### **4.2. Тестирование на реальном роботе:**

1. Взять лист флипчарта (или аналогичный большой лист).
2. Начертить на нем систему координат и отметить точки A, B, C, D в масштабе 1:1, соединив их линиями.
3. Загрузить программу из п. 3.2 в память реального робота TRIK.
4. Установить робота в начальную точку A, сориентировав его вдоль оси X.
5. Запустить программу.
6. Фиксировать результат: робот должен проехать по траектории от точки A до D, совершая повороты в точках B и C. Проверяется точность прохождения маршрута. Выведенные на экран значения должны совпасть с ручным расчетом.

### **3.2. Критерии оценивания выполненных пунктов заданий кейса 2.**

#### **3.2.1. Оценка процесса решения расчетной части задачи:**

- Выполнены расчеты траектории движения мобильного робота и скорости, соответствующие условию задачи - **5 баллов**. При наличии ошибок – **3 балла**.

### **3.2.2. Оценка результатов работы программной части:**

- Нарисована блок-схема алгоритма и реализована базовая программа управления роботом – **5 баллов**. Реализована только блок-схема или базовый код – **3 балла**.

- Разработан листинг кода, в котором реализовано решение для всех режимов управления – **до 10 баллов**. Программный код не содержит данные для работы с энкодерами – **5 баллов**.

- Программа снабжена функционалом для работы с энкодерами, которые обеспечивают точное прохождение робота по траектории. Участник выполнил демонстрацию работы программы с вводом вычисленных данных о длине траектории маршрута и средней скорости робота – **15 баллов**. При наличии ошибок – **8 баллов**. При отсутствии файла программы - **0 баллов**.

### **3.2.3. Критерии снижения баллов**

Максимальный балл за выполнение всех заданий кейса 2 – 60 баллов. В каждом задании кейса 2 предусмотрен набор вариантов траекторий движения робота, сходных по сложности для программирования.

Правильность выполненных расчетов проверяется визуально. За расчет, отображающий предлагаемый вариант решения и не содержащий логических ошибок, оформленный в соответствии со стандартом, участник получает максимальный балл. Если допущены незначительные ошибки, но расчет в целом выполнен правильно, итоговый балл может быть снижен на 1 балл за каждую ошибку. Если расчеты не содержат формул или оформлены небрежно, итоговый балл может быть снижен на 1 балл. Если расчеты содержат грубые ошибки, участник получает 0 баллов.

Правильность разработанного алгоритма проверяется визуально. За реализацию задания, укладывающуюся в контрольное время и не содержащую ошибок, участник получает максимальный балл. Если допущены ошибки, но задание было выполнено, итоговый балл может быть снижен на 5 баллов за каждую ошибку. Если задание не



было выполнено, участник получает 0 баллов. Если задание было выполнено, но с отклонением от расчетных значений, итоговый балл может быть снижен на 5 баллов, при отсутствии расчетов длины траектории и средней скорости – снижен на 5 баллов. При отсутствии в программе учета данных с энкодеров, итоговый балл может быть снижен на 15 баллов.

Для получения максимального балла за практический этап Конкурса необходимо представить папку с фалами по заданиям кейса, содержащим изображения алгоритмов подписанных и оформленных на бумажном бланке, программу в виде блоков или в виде кода для реализации базового задания и задания повышенной сложности. Папка должна содержать ФИО участника в названии. Необходимо выполнить все задания и не превысить временной лимит, предусмотренный организатором для выполнения заданий.

При реализации заданий в средах Trik Studio проверяются те же критерии, что и при проверке реального робота, с добавлением дополнительного – необходимостью сохранения каждого подзадания отдельным файлом в именную папку на компьютере или в сетевой папке.

## **4. Платформа Trik Studio.**

### **4.1. Установка программной среды**

Произвести установку бесплатной среды программирования Trik Studio, ссылка на скачивание программы - <https://trikset.com/products/trik-studio?ysclid=mcpcgur1hd527646656>.

### **4.2. Сборка робота**

В окне «Дисплей» в разделе «Порты» установите на порте A1 и A2 датчики расстояния ИК (Рис. 16 а). После их появления, нажмите *ПКМ+Shift* на красный кружок одного из датчиков, чтобы разворачивать его (Рис. 16 б). Потом с помощью удержания на ПМК переместите датчики на роботе (Рис. 16 в).

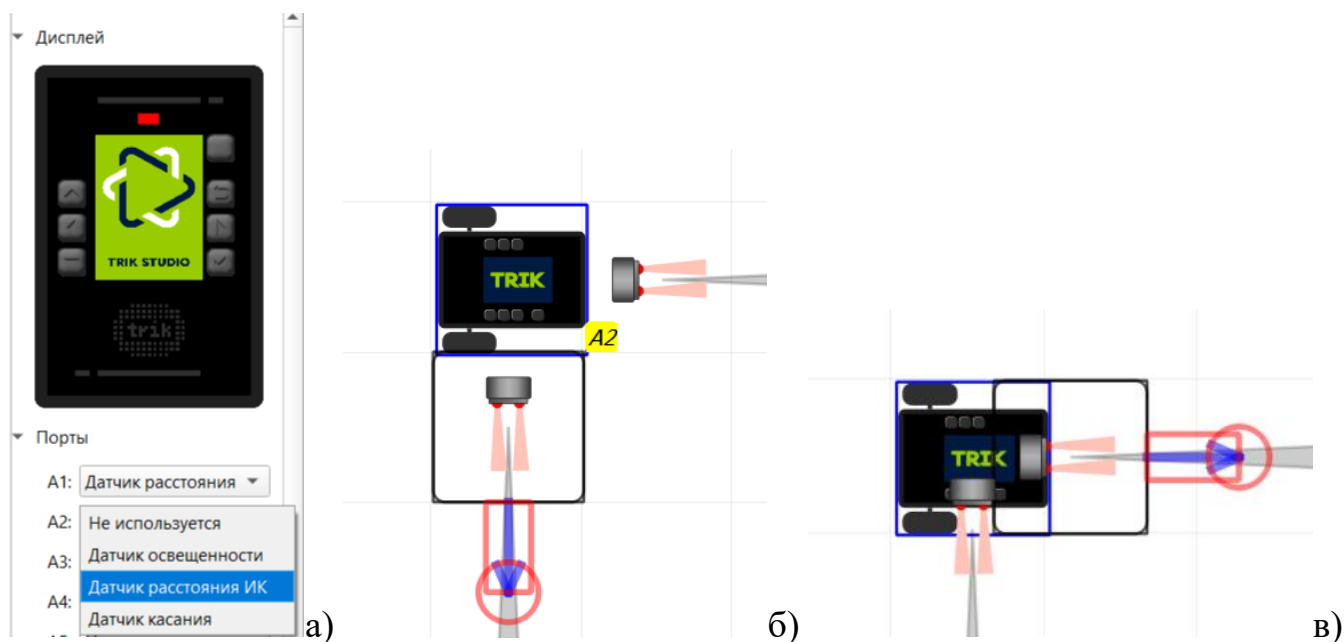


Рис. 16. Сборка робота: а) установка датчиков, б) разворот датчика, в) перемещение датчика

### 4.3. Движения робота

Рассмотрим пример программирования робота, чтобы он мог проехать вперёд на заданное расстояние и поворачивать в разные стороны на 90 градусов. В окне «Дисплей» в разделе «Моторы» установите порт М3 на левое колесо и порт М4 на правое колесо (рис. 17 а, б). Визуально запомните или запишите, какие порты находятся в моторах робота.



Рис. 17. Установка портов моторов (а) и расположение их на роботе (б)

**Движение прямо.**

Откройте раздел «Редактор», чтобы начать писать алгоритм робота. Для того, чтобы робот двигался вперёд на определенное расстояние, используйте блоки «Моторы вперёд» и «Таймер» (рис. 18). В блоке «Моторы вперёд» укажите оба порта мотора и скорость, по которой моторы будут работать. В блоке «Таймер» укажите время задержки, за которое робот будет выполнять действие блока «Моторы вперёд». Без блока «Таймер» программа досрочно завершается.

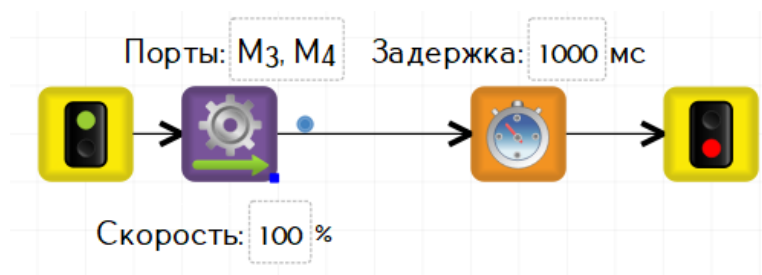


Рис. 18. Алгоритм движения робота вперёд с помощью таймера

### Поворот (плавный).

Для того, чтобы робот поворачивался в конкретную сторону, используется несколько блоков «Моторы вперёд» и измените скорости каждого мотора (Рис. 19).

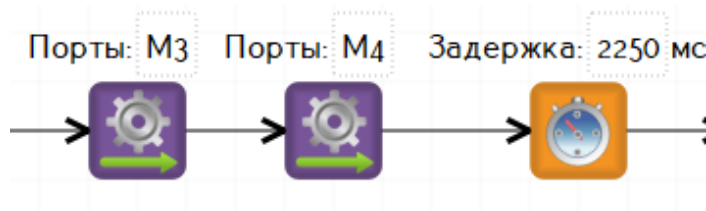


Рис. 19. Алгоритм плавного поворота налево

Для плавного поворота направо (Рис. 20) задается скорость левого колеса М3 на 70%, а скорость правого колеса М4 – на 30%. В блоке «Таймер» поставить время на 2250 мс. Для плавного поворота налево поменять значения скоростей между моторами М3 и М4.

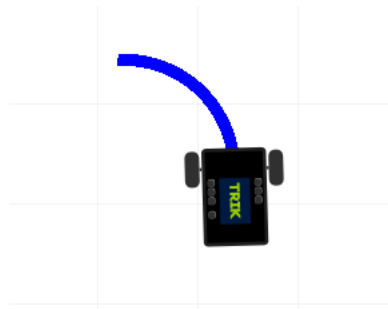


Рис. 20. Плавный поворот направо

Скорости и время поворота корректируются индивидуально под конкретный ваш случай.

### Поворот (на месте).

Для того, чтобы робот поворачивался в конкретную сторону на месте (рис. 21). Поворот на месте направо, используются либо несколько блоков «Движение вперёд», либо с блоком «Движение назад» (рис. 22).

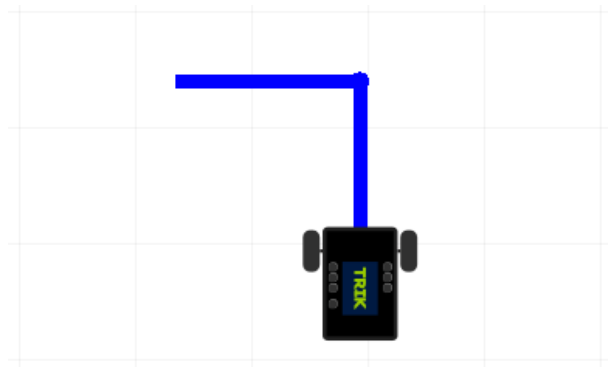


Рис. 21. Поворот на месте направо

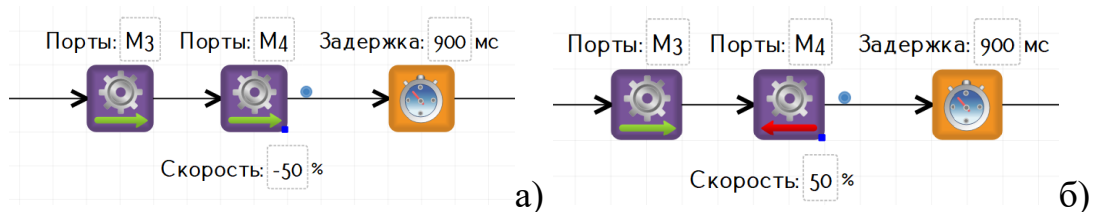


Рис. 21. Алгоритм поворота на месте направо: а) с двумя блоками «Моторы вперёд», б) с блоками «Моторы вперёд» и «Моторы назад»

Для поворота на месте направо (рис. 22) задается скорость левого колеса М3 на 50%, а скорость правого колеса М4 – на -50% (вариант а) или на 50% (вариант б). В блоке «Таймер» поставить время на 900 мс.

### Энкодер.

Возникает проблема, что при разряде аккумулятора робот, используя тайм-модели, будет проходить меньшее расстояние из-за нехватки мощности и его поведение становится непредсказуемым.

Второй способ задать движение робота, это использование встроенных в моторах датчики количества оборотов, или же *энкодеры*. Если задать движение до достижения определенного количества оборотов, то независимо от уровня заряда аккумулятора робот пройдет одинаковое расстояние.

В программе моторы обозначены как М3, М4, а энкодеры – Е3, Е4 (они соответствуют друг другу) (рис. 23).

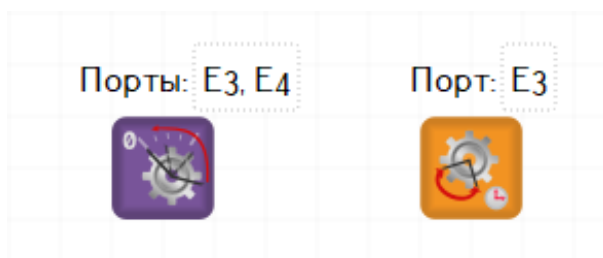


Рис. 23. Моторы (слева) и энкодер (справа)

Для того, чтобы задать новое движение робота (рис. 24), для начала используйте блок «Сбросить показания энкодера» (его нужно задавать каждый раз перед новыми движениями, т.к. робот запоминает показания энкодера за предыдущие движения. Вы можете посмотреть показания энкодеров моторов во вкладке «переменные» с именованием encoder3 и encoder4). Затем построить алгоритм движения и в конце использовать блок «Ждать энкодер». В этом блоке задаете предел оборотов моторов, за которое робот должен проехать, независимо от мощностей моторов.

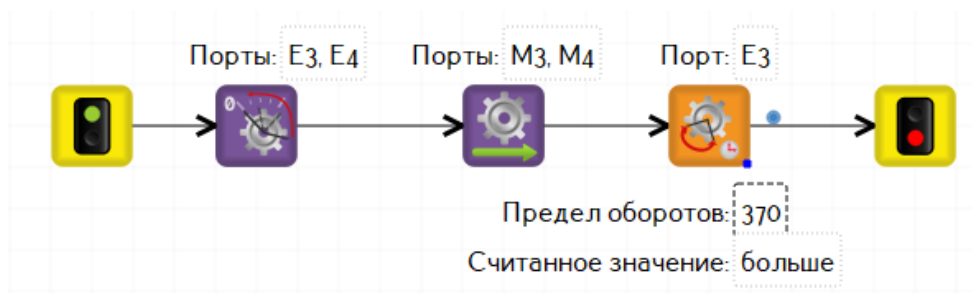


Рис. 24. Движение робота вперёд по данным с энкодера

Считается, что чтобы робот проехал одну клетку, ему нужно выполнить  $\pm 370$  оборотов (рис. 25).

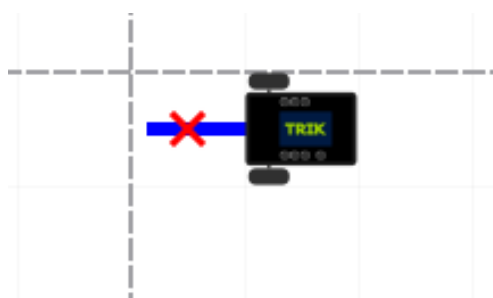


Рис. 25. Перемещение робота на одну клетку

## 5. Инструкция по эксплуатации робота TRIK

### 5.1. Информация о правилах безопасности

Конечно, вот текст из изображения:

**ООО "Кибернетические Технологии"**

**Информация о правилах безопасности**

Контроллер предназначен для выполнения только функций, описанных в инструкции по эксплуатации. Если контроллер не работает в соответствии с инструкцией по эксплуатации, обратитесь в службу технической поддержки. Контроллер нельзя самостоятельно чинить – в этом случае гарантия на контроллер перестаёт действовать. Если вы не согласны с данными условиями, верните контроллер продавцу до начала пользования контроллера.

**Меры безопасности**

- Не засовывайте в корпус контроллера посторонние предметы.

- Не подвергайте контроллер воздействию сильных магнитных или электрических полей.

- Не подвергайте воздействию жидкостей.

- Не используйте повреждённые кабели для работы контроллера.

- Не выбрасывайте прибор с хозяйственными бытовыми отходами. Берегите окружающую среду. Обратитесь в местные органы.

- Не подключайте к работающему контроллеру датчики и моторы.

- Если монитор контроллера разбился, соблюдайте осторожность, чтобы не пораниться осколками стекла и избежать контакта жидкокристаллического вещества с кожей, а также попадания этого вещества в глаза или рот.

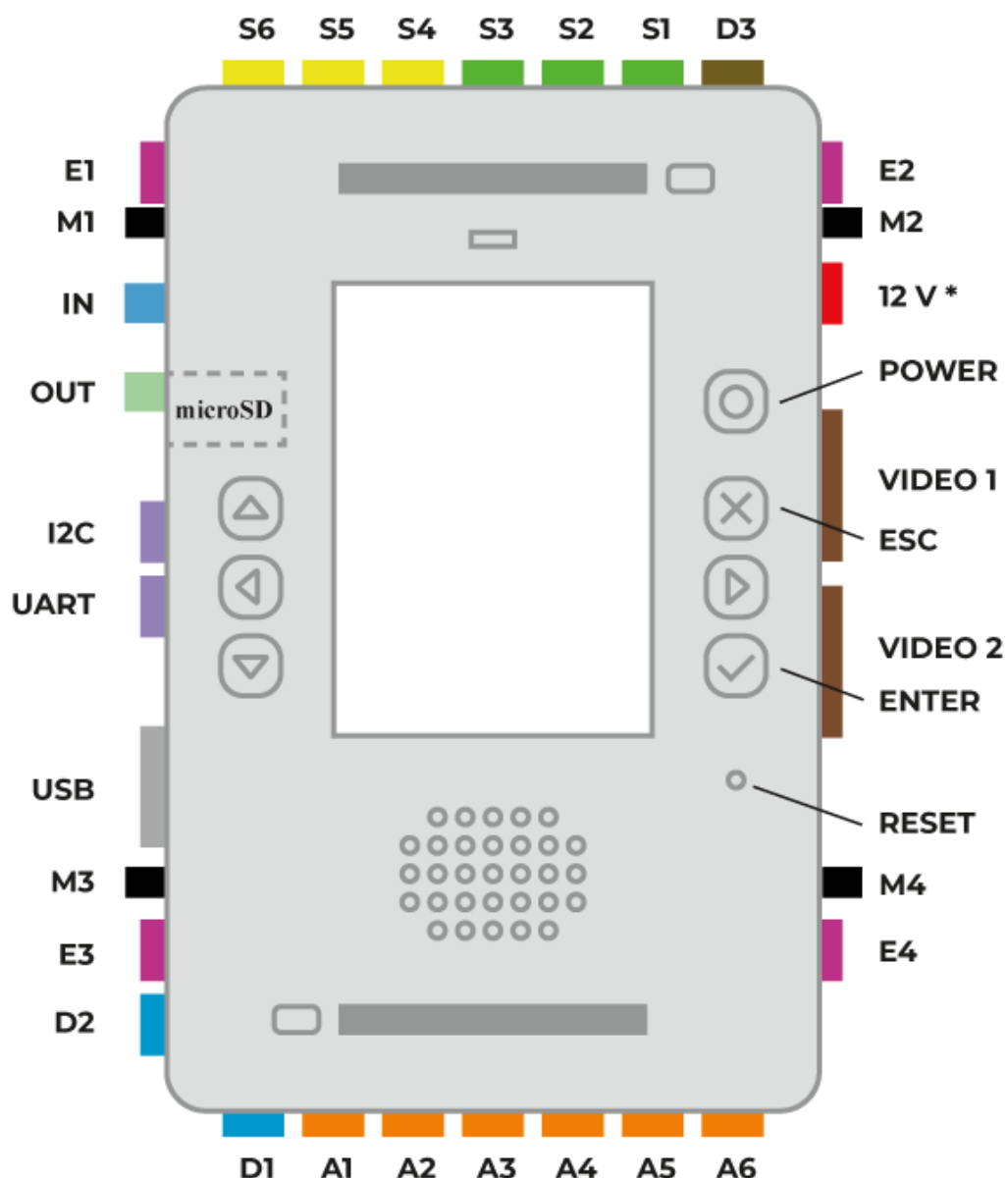
- Соблюдайте осторожность при обращении с батареями. Неправильное обращение с батареями может привести к их протеканию или взрыву. Соблюдайте следующие меры предосторожности при использовании батарей с данным изделием:

- Используйте с данным изделием только рекомендованные батареи (см. «Технические характеристики контроллера TRIK»).

- Не разбирайте батарею и не замыкайте ее контакты.

## **5.2. Описание контроллера TRIK**

На схеме ниже показано расположение основных элементов управления и разъемов на плате контроллера TRIK.



\* Рекомендованное напряжение. Допустимое напряжение: 6–12,8 В

### Общие разъемы и элементы управления:

- **PWR 6-12V:** вход электропитания 6.0 -12.8 V DC / 20A
- **PWR\_ON:** кнопка включения и отключения питания
- **SW\_RESET:** кнопка перезагрузки
- **Разъемы и периферийные устройства центрального процессора**
- **X1, X2:** разъемы подключения модулей расширения и модулей адаптации периферийных устройств



- **video1, video2:** разъемы видеовходов процессора, предназначенные для подключения 2-х моно или 1-го стерео-модуля видеосенсора

- **DISP:** цветной сенсорный TFT дисплей с разрешением 320\*240 пикселей

- **SPKR:** встроенный динамик (250 – 20000 Hz, 1 W ном.)

- **MIC\_L\_IN, MIC\_R\_IN:** разъемы подключения микрофонных входов, левый и правый каналы

- **LINE\_OUT:** линейный стереоаудио выход

- **J2 (USB):** разъем USB 2.0 «тип А», в режиме Host предназначенный для подключения периферийных устройств и флэш-накопителей с интерфейсом USB, а в режим Device (при использовании переходного кабеля, входящего в комплект) – для подключения к компьютеру с целью загрузки и отладки встроенного программного обеспечения.

- **I1 (I2C):** разъем TTL 5V / LVTTTL 3.3 V интерфейса I2C для организации межмашинного взаимодействия в конфигурациях с несколькими контроллерами. Допускает подключение нескольких устройств при использовании специального шлейфа-разветвителя. Разъем может также использоваться, как порт ввода-вывода общего назначения (GPIO) для сенсоров, совместимых по питанию и уровню логических сигналов.

- **U1 (UART):** разъем TTL 5V / LVTTTL 3.3 V для подключения устройств с интерфейсом UART или удаленной консоли компьютера. Разъем может также использоваться, как порт ввода-вывода общего назначения (GPIO) для сенсоров, совместимых по питанию и уровню логических сигналов.

- **SD\_Card:** разъем подключения флэш-карты формата Micro-SD

- **Разъемы периферийного процессора**

- **M1-M4:** выходные разъемы подключения двунаправленных двигателей постоянного тока (6-12V, предельная нагрузка - 2A). Разъемы двигателей имеют аппаратную защиту от превышения предельно допустимого тока 2A. Превышение порога вызывает аварийное отключение соответствующего драйвера двигателя,

повторное включение выполняется периферийным процессором на программном уровне.

- **A1-A6:** двухканальные TTL /Analog 5V + LV гибридные аналого-цифровые разъемы. Канал для подключения цифровых и аналоговых устройств, использующих «3-проводной» или «4-проводной» интерфейс, обеспечивает диапазон измерений 0-5 V. Канал A6 может быть использован, как дополнительный цифровой порт для подключения сенсору с поддержкой «прямого» и «инверсного» режимов двунаправленной связи, а также как альтернативный аналоговый канал с диапазоном измерений 0-3.3 V.

- **E1-E4:** двухканальные TTL 5V цифровые разъемы для подключения периферийных устройств с питанием 5V. Могут также использоваться, как порты ввода-вывода общего назначения (GPIO).

- **S1-S3:** 1-канальные TTL 5V разъемы цифрового ввода-вывода для подключения периферийных устройств с питанием 5V, обеспечивающие доступ к сигналам ЦП ECAP1\_APWM0, ECAP1\_APWM1, ECAP2\_APWM2. Могут также использоваться, как порты ввода-вывода общего назначения (GPIO).

- **D1, D2** – двухканальные TTL 5V разъемы цифрового ввода-вывода с программным управлением для подключения сенсоров, использующих «3-проводной» или «4-проводной» интерфейс. Разъем поддерживает «прямой» и «инверсный» режим двунаправленной связи с сенсором.

- **S4-S6:** 1-канальные TTL 5V разъемы цифрового ввода-вывода для подключения периферийных устройств с питанием 5V, обеспечивающие доступ к сигналам ЦП EPWM1\_A, EPWM1\_B, EPWM0\_B. Могут также использоваться, как порты ввода-вывода общего назначения (GPIO).

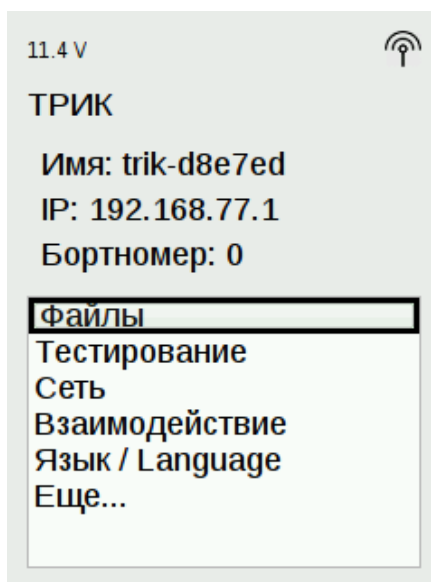
- **D3** – двухканальный TTL 5V разъем цифрового ввода-вывода для подключения периферийных устройств с питанием 5V, обеспечивающий доступ к сигналам RTC\_ALARM и TIMER64\_P0. Может также использоваться, как двухканальный порт ввода-вывода общего назначения (GPIO).

- **BI-LED:** двухцветный (Red/Green) светодиод с программным управлением

- **6 пользовательских кнопок:** «вправо», «влево», «вверх», «вниз», «ввод», «меню». Функциональность кнопок полностью определяется на программном уровне.

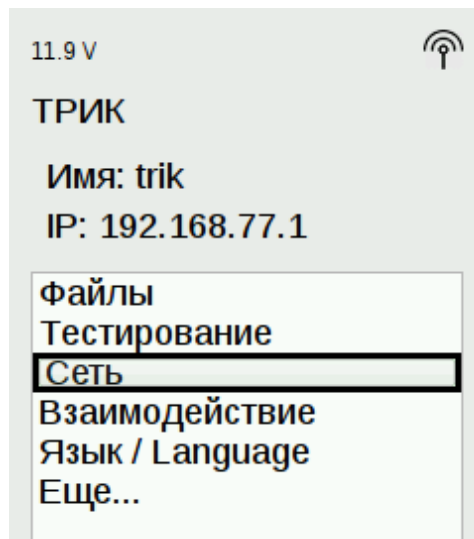
### 5.3. Начало работы с контроллером

1. Вставьте в разъем microSD карточку из комплекта.
2. Подключите к разъёму питания батарейный блок, блок питания или аккумуляторные батареи.
3. Нажмите на кнопку Power.
4. На экране появится надпись и полоска загрузки контроллера.
5. Как только на экране появится системное меню, контроллер готов к работе.



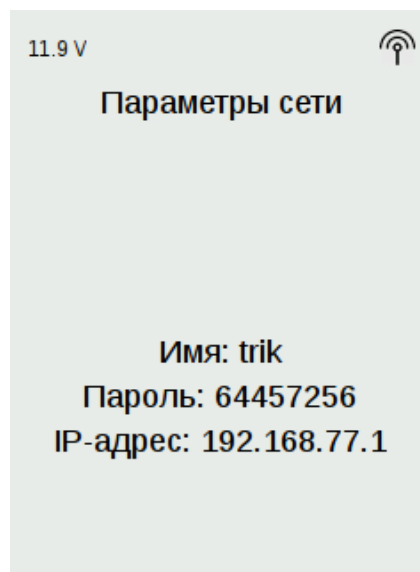
Контроллер может работать как в режиме Wi-Fi точки доступа, так и в режиме Wi-Fi клиента (подключаться к любой доступной Wi-Fi сети).

6. Зайдите на контроллере в меню "Сети".



7. Выберите режим "Wi-Fi точка доступа".

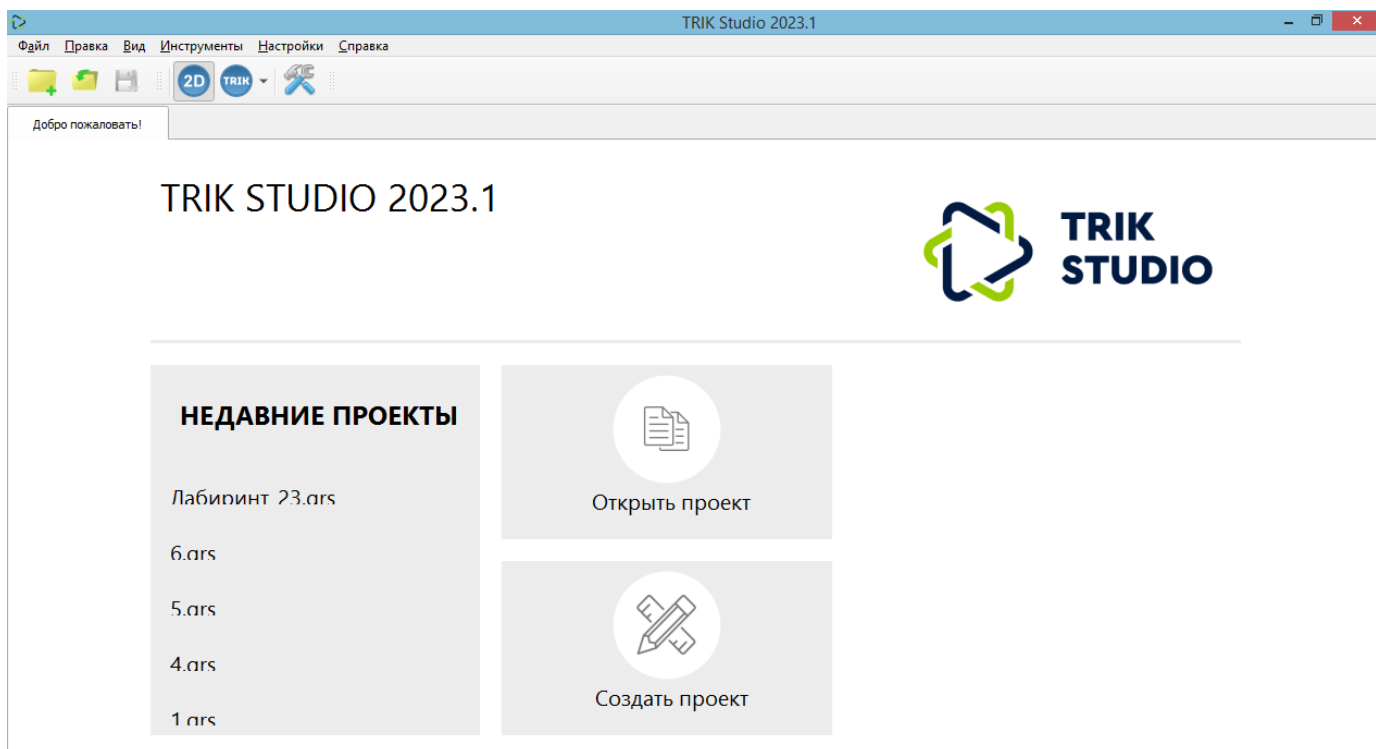
8. Контроллер включится в режим точки доступа. На экране появится название сети (SSID) и пароль для подключения.



9. Подключите компьютер к данной Wi-Fi сети с указанным паролём.

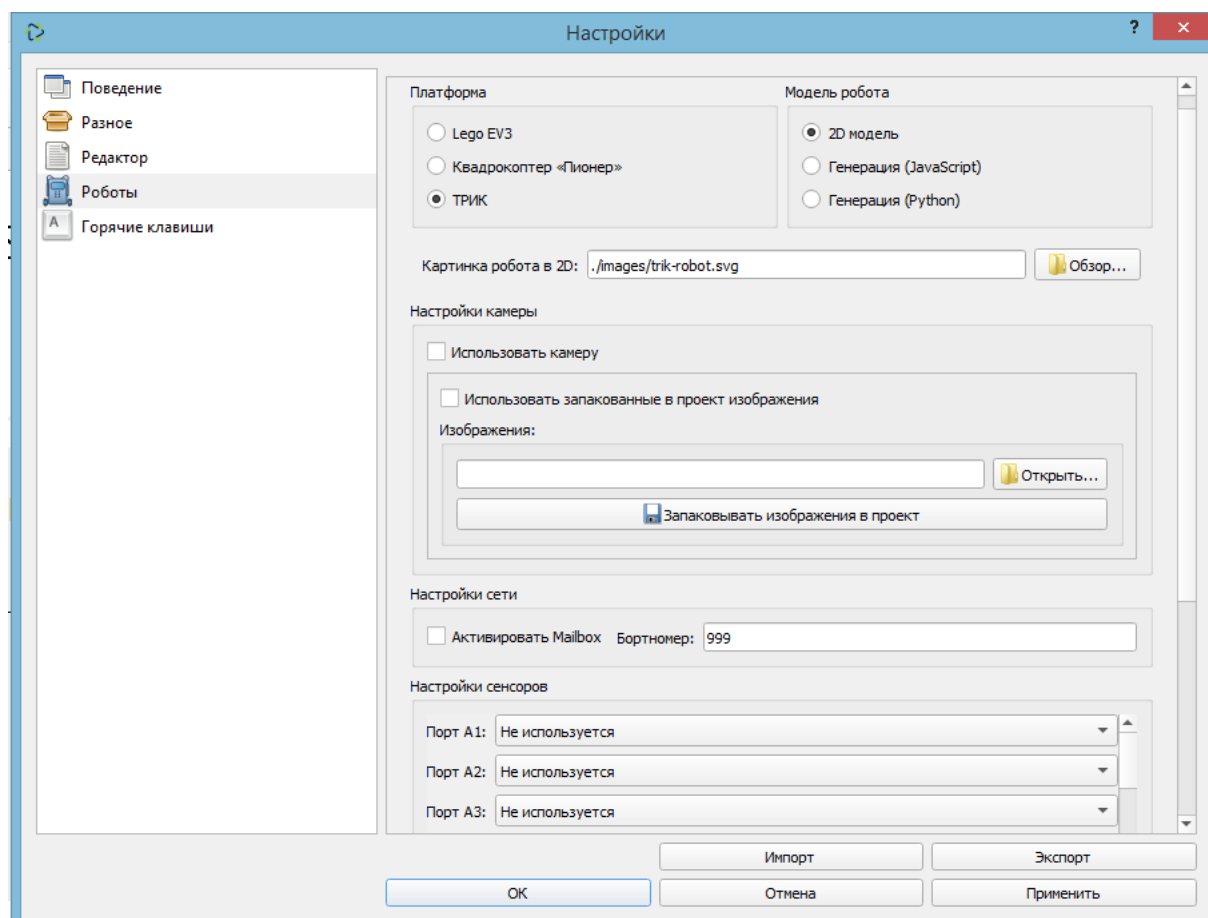
10. Скачайте и установите TRIKStudio с сайта <http://trikset.com>.

11. Откройте TRIKStudio.



12. Откройте пункт меню "Настройки" и выберите пункт "Настройки".

13. Перейдите на вкладку "Роботы".



14. Выберите пункт "ТРИК" в разделе "Тип модели робота".

15. Убедитесь, что в TRIK Studio выбран режим работы с реальным роботом (горит синяя кнопка «TRIK» на панели инструментов).

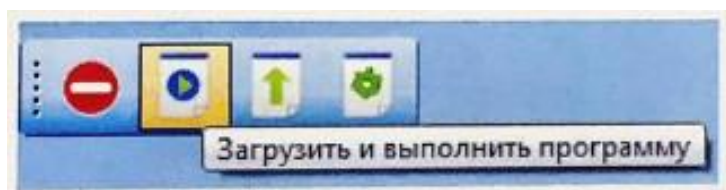


16. В поле "Настройки TCP" введите IP-адрес робота, который отобразился на его экране (по умолчанию 192.168.1.1 в режиме точки доступа).

17. Откройте один из примеров. Выберите меню "Файл" -> "Открыть".

18. Выберите папку "trik" и в ней файл smiles.qrs.

19. Нажмите кнопку "Загрузить и выполнить".



#### 5.4. Подключение робота к существующей WiFi сети

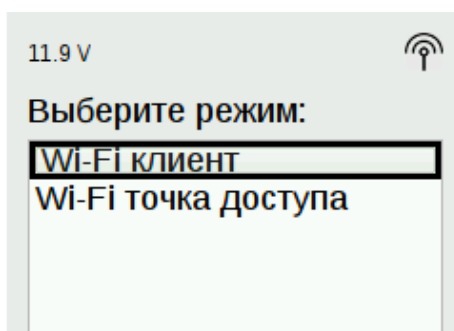
1. Запустите TRIKStudio.

2. Выберите меню "Файл" пункт "Открыть".

3. Перейдите в директорию "trik", а затем "system".

4. Выберите файл configureNetwork.qrs.

5. Включите робота в режиме точки доступа (меню Сеть" -> "Wi-Fi точка доступа").



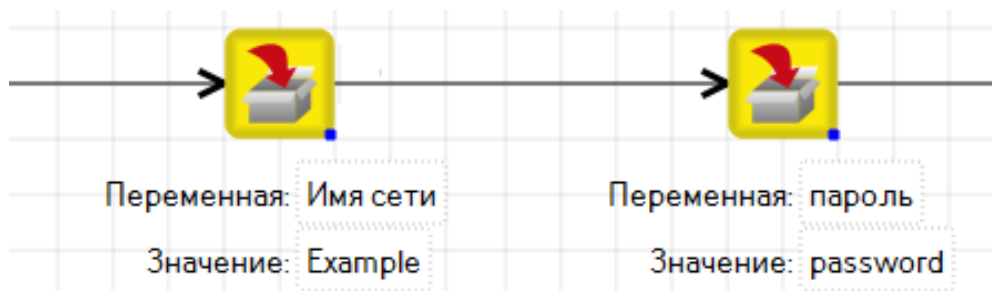
6. переведите TRIK Studio в режим работы с реальным роботом (кнопкой на панели инструментов).



7. Подключитесь к роботу по WiFi (проверить корректность можно кнопкой "Подключиться" на панели инструментов).

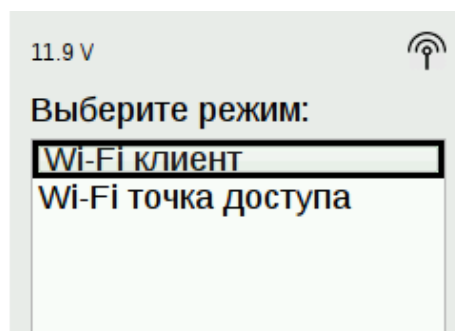


8. Поменяйте значение переменной Имя Сети, введя имя сети (SSID), в кавычках. Поменяйте значение переменной Пароль, введя пароль сети.



9. Запустите программу кнопкой "Загрузить и выполнить программу".

10. После того, как программа отработает (появится меню на роботе), выберите режим "Wi-Fi-клиент" в меню "Сеть".



11. Выберите вновь добавленную сеть и нажмите кнопку "Ввод" на роботе.

12. Робот через некоторое время должен подключиться к сети и показать выданный ему IP-адрес в верхней части окна настройки сети. Рядом с IP-адресом будет отображаться зеленый маркер.

