

**Государственное бюджетное общеобразовательное учреждение
города Москвы «Школа № 1412»**

**ПРОЕКТНАЯ РАБОТА К XI МИНСКОМУ ОТКРЫТОМУ
РОБОТУРНИРУ РЕГЛАМЕНТ ЮНЫЙ ИНЖЕНЕР ПОД
НАЗВАНИЕМ «INGENUITY»**

Участник:
ученик 8 «Д» класса ГБОУ Школа
№ 1412 XXXXX Дмитрий

Руководители:
Методист ГАОУ ДПО ЦПМ
Кузьмич Василий Михайлович

Учитель ГБОУ Школа № 1412
Швецов Артём Викторович

Москва, 2024

РЕФЕРАТ

Пояснительная записка 26с., 23 рис., 2 источника, 3 приложения.

РОБОТ, МИКРОКОНТРОЛЛЕР, 3D-МОДЕЛИРОВАНИЕ, ПРОГРАММИРОВАНИЕ, СХЕМОТЕХНИКА, МЕХАНИКА, ПЕЧАТНЫЕ ПЛАТЫ

Объектом исследования является регламент «Юный инженер» Московской командной олимпиады по робототехнике (далее – регламент).

Целью исследования является создание робота, который будет выполнять задания регламента наилучшим образом.

Для достижения цели были поставлены следующие задачи:

- 1) Провести анализ регламента и определить функциональные части робота;
- 2) Изучить систему автоматизированного проектирования «Компас-3D» (далее – САПР «Компас-3D»);
- 3) Создать чертеж основания робота (первый уровень) и вырезать его с помощью лазерной резки;
- 4) Смоделировать и напечатать на 3D принтере колеса и формы для заливки шин из силикона. Залить формы шин силиконом;
- 5) Создать механизм сбивания планки с помощью лазерной резки и 3D-печати;
- 6) Разработать в САПР «Компас-3D» и напечатать второй уровень робота, бампер для улучшения работы датчика отражённого света;
- 7) Изучить систему автоматизированного проектирования «KiCad»;
- 8) Научиться подготавливать файлы печатных плат для станка с числовым программным управлением (далее - станок с ЧПУ);
- 9) Создать печатную плату датчика отраженного света для отслеживания линии;
- 10) Разобрать принцип работы драйвера моторов;
- 11) Создать системную печатную плату;
- 12) Изучить язык программирования C++ в среде Arduino IDE;

- 13) Научиться работать с сервоприводом;
- 14) Создать программу, которая выполняет задание регламента наилучшим образом;

В ходе работы были изучены программирование микроконтроллеров, создание электрических плат, основы 3D моделирования и получены соответствующие практических навыки в данных областях.

Для определения метода реализации проекта поэтапно рассмотрен процесс создания робота на Arduino.

В результате проведенного исследования был создан робот, соответствующий всем критериями регламента.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
ОСНОВНАЯ ЧАСТЬ	6
1. Конструкция робота	6
1.1 Первый уровень робота	6
1.2 Второй уровень робота.....	10
2. Электроника робота	11
3. Программирование робота.....	13
ЗАКЛЮЧЕНИЕ.....	15
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	16
ПРИЛОЖЕНИЕ А.....	17
ПРИЛОЖЕНИЕ Б.....	20
ПРИЛОЖЕНИЕ В.....	26

ВВЕДЕНИЕ

Изучив регламент, я пришёл к выводу, что для решения данной задачи достаточно создать заднеприводного робота на дифференциальной платформе, состоящего из самостоятельно изготовленных электронных компонентов.

Данный проект полезен для формирования и отработки различных навыков: программирования, сборки, создания механизмов и печатных плат, 3D-моделирования. В дальнейшем полученные навыки позволят лучше ориентироваться в перечисленных сферах, принимать оптимальные решения в условиях многозадачности.

Помимо вышесказанного данный проект является отличным опытом междисциплинарной работы, что очень важно для современного инженера.

ОСНОВНАЯ ЧАСТЬ

1. Конструкция робота

В ходе работы над проектом был разработан двухуровневый робот: основание с механическими элементами и уровнем управления. В процессе работы над механической частью все детали были сделаны самостоятельно при использовании системы автоматизированного проектирования «Компас 3D», в которой были разработаны все 3D модели и чертежи (приложение А), и 3D принтера.

1.1 Первый уровень робота

Основа. Основа робота (Рисунок 1), содержащая все технологические отверстия, была изготовлена методом лазерной резки из фанеры.

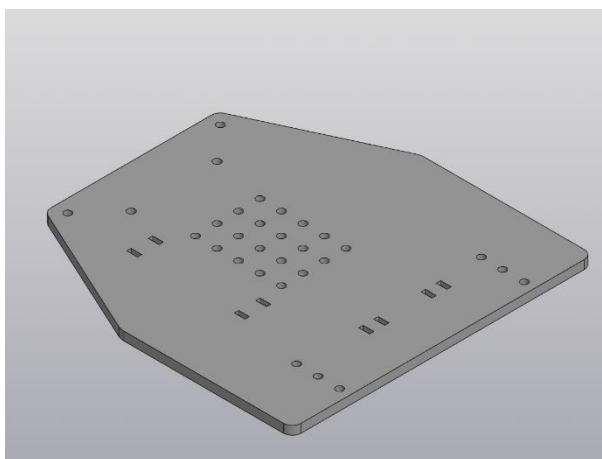


Рисунок 1 – Модель основы робота в САПР «Компас-3D» (первый уровень)

Колеса и шины. Исходя из необходимости крепления датчиков отслеживания линии на определённой высоте, был выбран следующий диаметр колес в 4,5 см (Рисунок 3). Также данный размер колёс обеспечивает оптимальное соотношение между скоростью и плавным управлением

движением робота. Шины были созданы методом литья из силикона в форму (Рисунок 2). Формы и шины были изготовлены самостоятельно в программе «Компас 3D» и распечатаны на 3D принтере.



Рисунок 2 – Форма для шин с залитым силиконом



Рисунок 3 – Колесо с силиконовой шиной

Крепления для моторов. Исходя из анализа регламента было принято решение создавать робота на дифференциальном приводе с горизонтальным креплением моторов. Была разработана специальная деталь для крепления моторов (Рисунок 4.1, Рисунок 4.2). Данная деталь также является основой крепления для второго уровня, поэтому технические отверстия присутствуют на обеих сторонах.

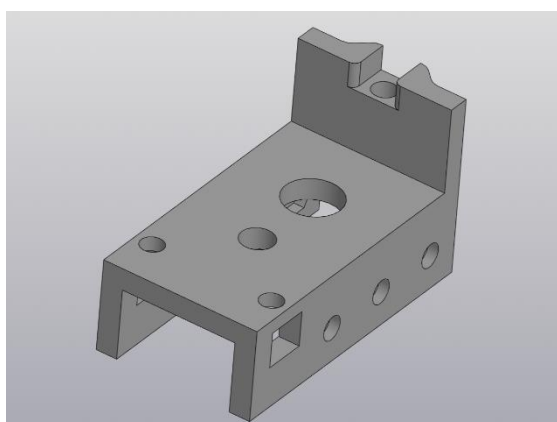


Рисунок 4.1 – Модель крепления моторов (вид сверху)

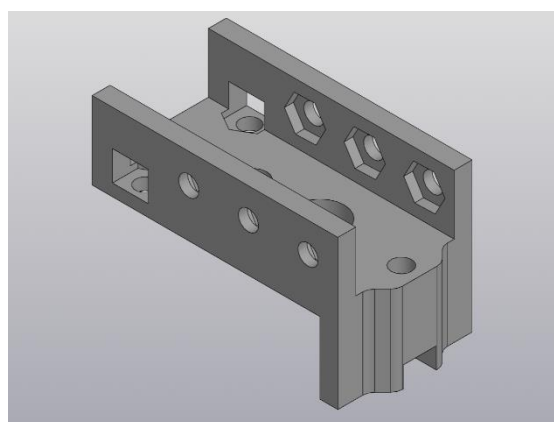


Рисунок 4.2 – Модель крепления моторов (вид снизу)

Механизм сбивания (Рисунок 5.1, Рисунок 5.2). Для приведения механизма сбивания планки в движение был выбран сервопривод sg90. Механизм сбивания состоит из трех составных частей. Первая часть (собственно, сбивалка) закрепляется на вспомогательной детали – уголке, что позволяет механизму поворачиваться и основную нагрузку концентрировать на уголке. Вторая часть соединяет первую и третью часть, обеспечивая их подвижность. Третья часть соединяется со второй и закрепляется на вале качалки сервопривода. Все части механизма соединены между собой болтами, для улучшения скольжения механизма использовались прокладки из робототехнического набора «Трик».

Впоследствии выяснилось, что при сбитии роботом планки, она падает на корпус, что приводило к разрушению объектов на поле. Это сподвигло сделать версию 2 (Рисунок 6.1, Рисунок 6.2) механизма сбивания, которая включает в себя сбивалку с «язычком», которая кидает планку перед роботом.

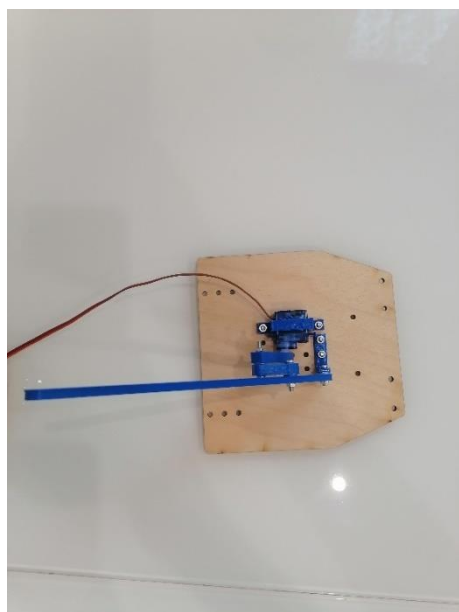


Рисунок 5.1 – Версия 1 механизма сбивания планки (вид сверху)

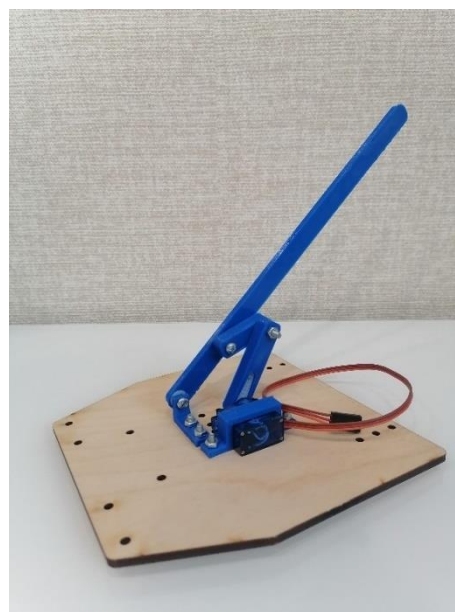


Рисунок 5.2 – Версия 1 механизма сбивания планки (вид сбоку)



Рисунок 6.1 - Версия 2 механизма сбивания планки (вид с левого боку)



Рисунок 6.2 - Версия 2 механизма сбивания планки (вид спереди)

Бампер для улучшения работы датчика отражённого света (Рисунок 7). Данная деталь крепится на четырёх болтах к основе робота, проходящих через технологические отверстия датчика отражённого света, и обеспечивает защиту от механических повреждений, например, при падении планки. Также бампер обеспечил лучшую работу датчика отражённого света, так как защищает от влияния стороннего света.

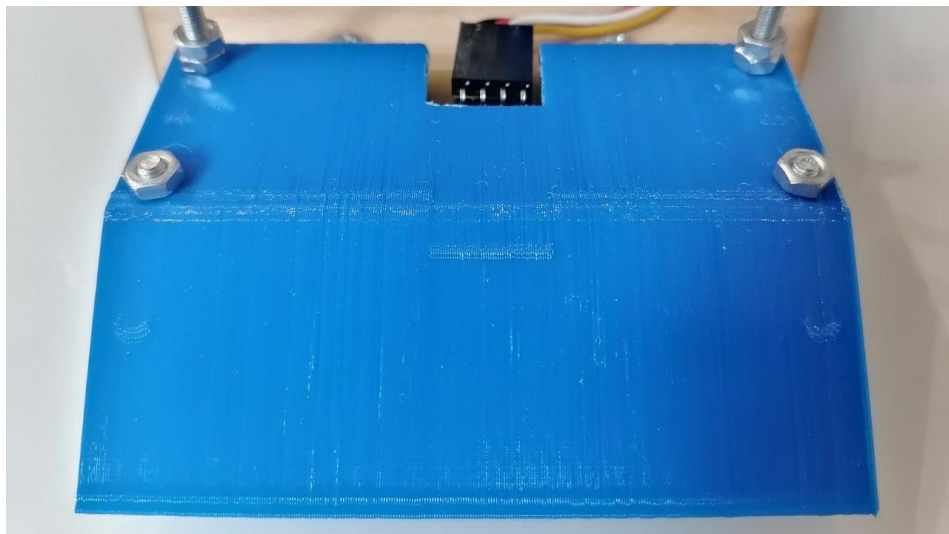


Рисунок 7 – Бампер для улучшения работы датчика отражённого света (вид спереди)

Третья точка опоры (Рисунок 8). Для придания устойчивости дифференциального робота была разработана третья точка опоры. В качестве материала для компонента, контактирующего с поверхностью поля были использованы фетровая накладка, которая позволяет улучшить скольжение и не наносит вред соревновательному полигону.



Рисунок 8 – Третья точка опоры

1.2 Второй уровень робота

В целях безопасности и экономии места питание робота и тумблер включения были вынесены на заднюю стенку робота между первым и вторым уровнем. Задняя стенка робота, выполненная в форме уголка (Рисунок 8). Уголок имеет все функциональные отверстия: под тумблер питания и для крепления батарейного блока.

Системная плата робота (Рисунок 9). Данная плата содержит все необходимые разъёмы для подключения электронных устройств, а также имеет технологические отверстия для крепления.



Рисунок 8 – Задняя стенка робота



Рисунок 9 – Системная плата

2. Электроника робота

Все печатные платы произведены на основе текстолита с помощью станка PureLogic с ЧПУ (Рисунок 10.1, Рисунок 10.2). Подготовка файлов для станка с ЧПУ происходила при помощи программы FlatCam. Все принципиальные электрические схемы представлены в приложении Б

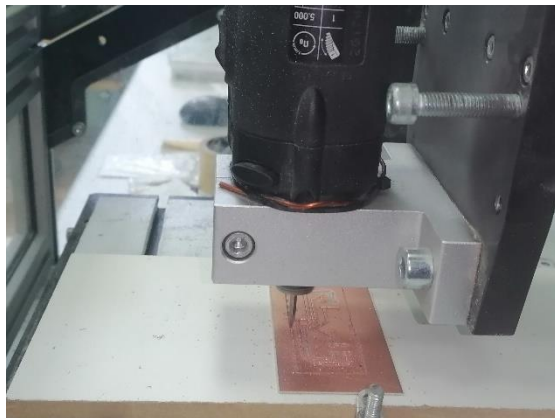


Рисунок 10.1 – Изготовление печатной платы на станке с ЧПУ

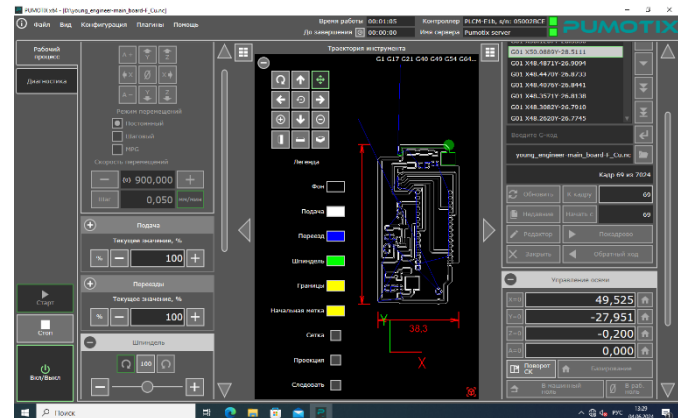


Рисунок 10.2 – Программа для управления станка с ЧПУ

Датчик отраженного света. Изначально, было изготовлено два датчика отражённого света (Рисунок 11.1) для отслеживания линии на полигоне. Данный датчик был изготовлен методом пайки на паечной плате. Была использована стандартная обвязка. Для создания датчика отраженного света в качестве основного компонента был использован фоторезистор, получающий отраженный свет с красного светодиода. Впоследствии была изготовлена

отдельная плата для движения по линии (Рисунок 11.2), включающая в себя функционал двух датчиков из прошлой версии платы. Новая плата была изготовлена на текстолите с помощью станка с ЧПУ.

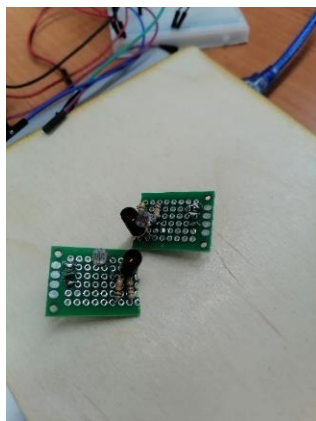


Рисунок 11.1 – Старые датчики отраженного света

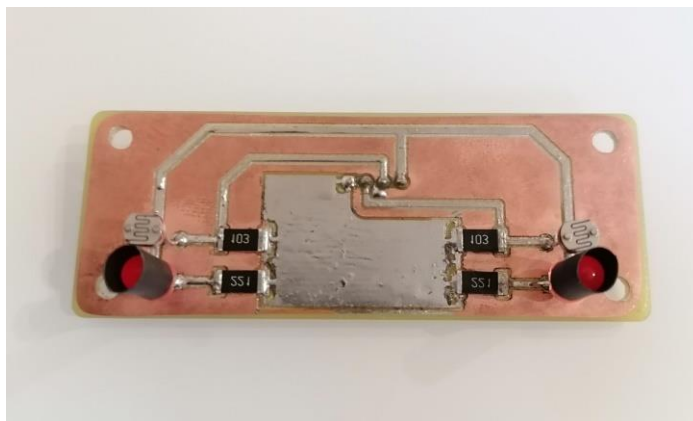


Рисунок 11.2– Новая плата с датчиками для движения по линии

Системная плата. В первой тестовой версии робота все компоненты робота соединялись между собой проводами и это больше напоминало «паутину». В ходе практического применения было обнаружено, что во время движения микроконтроллеру не хватает питания из-за того, что сервопривод питался от Arduino Nano и это приводило к перезагрузке микроконтроллера. Поэтому было принято решение использовать преобразователь напряжения AMS1117-5.0, берущий ток напрямую из аккумуляторов. Драйвер моторов (Рисунок 12.1) представлял собой отдельную плату, изготовленную на паечной макетной плате, со стандартной обвязкой.

Исходя из выше сказанного было принято решение сделать системную плату (Рисунок 12.2, Рисунок 12.3), объединяющую все компоненты: микроконтроллер Arduino Nano, драйвера моторов, представляющие собой MOSFET-транзисторы, преобразователь напряжения, выводы для подключения внешних устройств. Это новшество значительно улучшило коммуникацию компонентов.

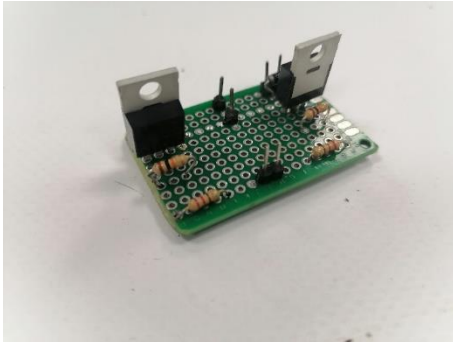


Рисунок 12.1 – Старый драйвер моторов

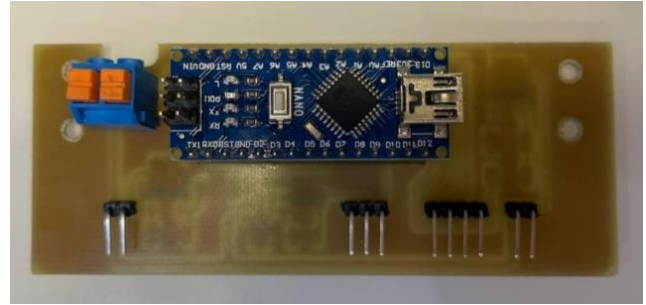


Рисунок 12.2 – Системная плата (вид сверху)

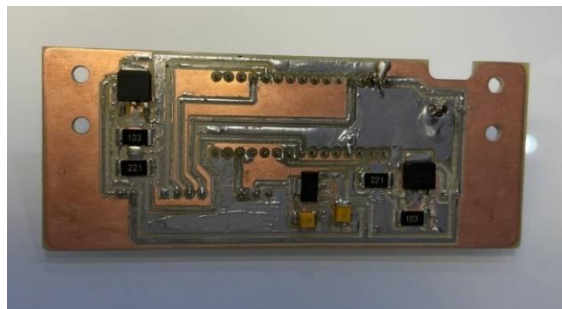


Рисунок 12.3 – Системная плата (вид снизу)

3. Программирование робота

Программа для микроконтроллера была написана в среде программирования Arduino IDE. В ходе работы был изучен язык C++, была создана библиотека (код представлен в приложении В) для управления мотором, которая позволяет с помощью понятного человеку простого числового набора от 0 до 100 процентов управлять мощностью электродвигателя (приложение В, рисунок В1). Для управления сервоприводом была выбрана стандартная библиотека Servo. После серии экспериментов было подобрано время для поднятия и опускания механизма сбивания планки и данный таймер был реализован с помощью встроенной функции millis (Рисунок 13.1). Для отслеживания линии роботом используется пропорционально-дифференциальный регулятор движения (Рисунок 13.2).

```

21
22     if (millis() <= 25000) {
23         servo.write(130); // Опустить механизм, чтобы не сбить первую планку
24     } else {
25         servo.write(90); // Поднять механизм, чтобы сбить вторую планку
26     }
27

```

Рисунок 13.1 – Текст участка программы, отвечающего за управление сервоприводом по таймеру

```

29
30     int left = analogRead(A1); // Читаем показания с левого датчика
31     int right = analogRead(A0); // Читаем показания с правого датчика
32
33     left = map(constrain(left, 500, 880), 500, 880, 0, 100); // Перевод показаний с левого д
34     right = map(constrain(right, 400, 870), 400, 870, 0, 100); // Перевод показаний с правого .
35
36
37     int error = (right - left) * 0.5; // Вычисляем ошибку П-регулятора (разница между
38     int u = error + 0.9 * (error - lastError); // Добавляем к ошибке П-регулятора дифференциал
39     lastError = error; // Записываем показания с датчиков в прошлую ош
40
41     motor.move(speed - u, speed + u); // Подаем на моторы скорость с управляющим воздействием
42

```

Рисунок 13.2 – Текст участка программы, отвечающего за движение по линии

ЗАКЛЮЧЕНИЕ

В результате работы был создан робот (Рисунок 14), выполняющий задания регламента наилучшим образом, что подтверждено дипломом I степени Московской командной олимпиады по робототехнике от 21 апреля 2024 года.

В процессе работы была достигнута поставленная цель и решены следующие задачи:

- изучена система автоматизированного проектирования «Компас-3D»;
- созданы чертежи и 3D модели оснований робота первого и второго уровней, колес, форм для заливки шин, датчика отраженного света и корпуса для него, механизмов;
- напечатаны на 3D принтере колеса, формы для заливки шин, основа для второго уровня;
- изучен принцип работы с сервоприводом;
- изучен язык программирования C++.
- изучена система автоматизированного проектирования «Kicad»;
- изучен способ создания печатных плат с помощью станка с ЧПУ;

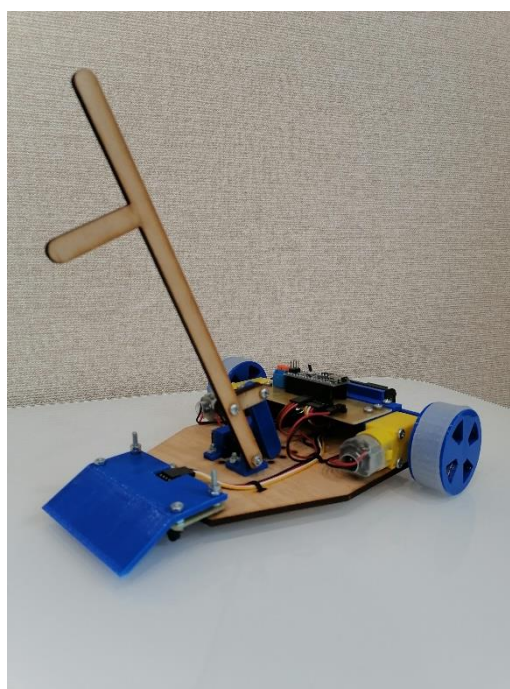


Рисунок 14 – Робот (вид сбоку)

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. База знаний Америки: [сайт]. – URL: <http://wiki.amperka.ru/> (дата обращения 15.02.2024). – Текст: электронный.
2. Arduino documentation: [сайт]. – URL: <https://docs.arduino.cc/programming/> (дата обращения 15.02.2024). – Яз. англ. – Текст: электронный.

ПРИЛОЖЕНИЕ А

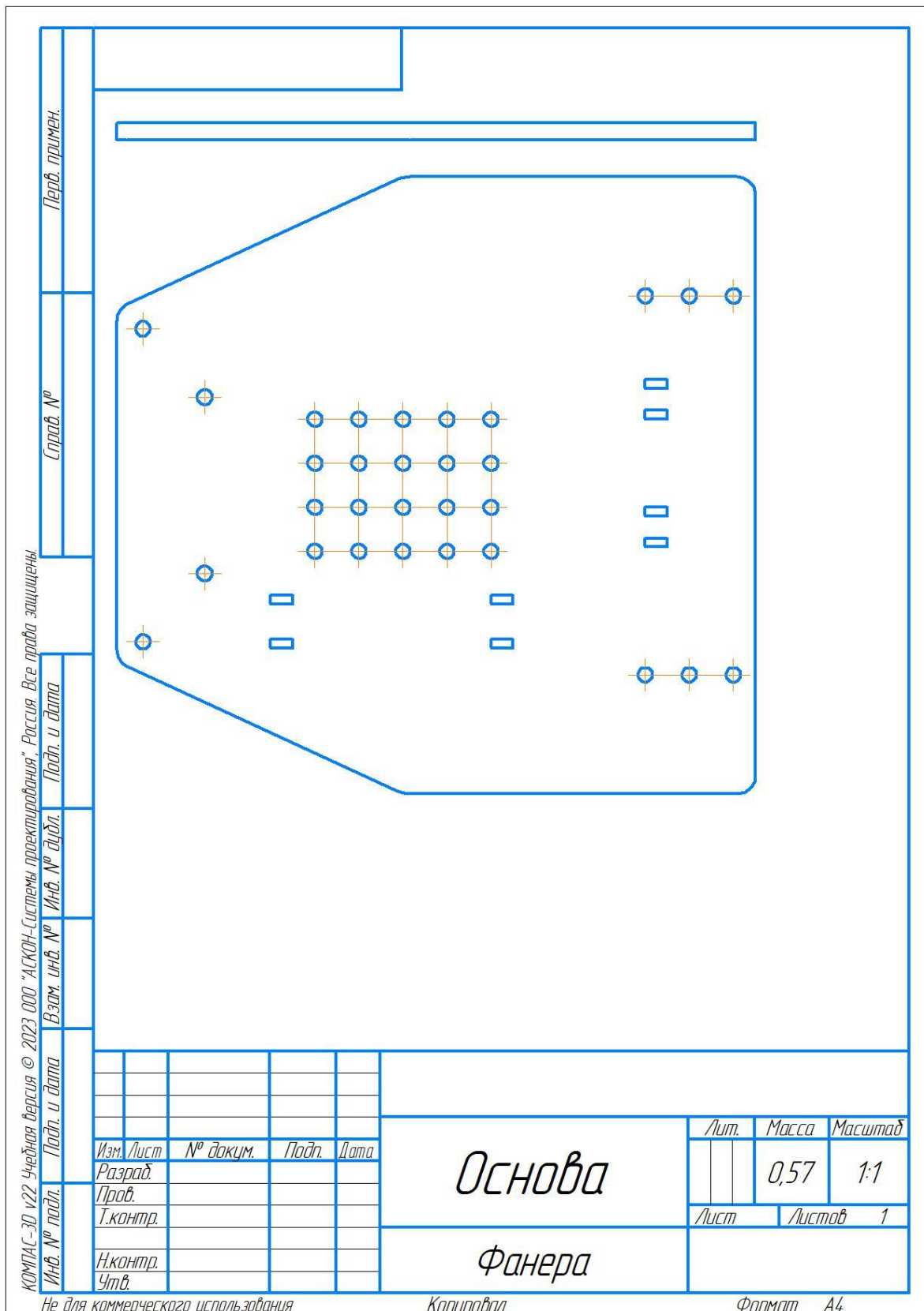


Рисунок А.1 – Чертеж основания (первый уровень)

КОМПАС-3D v22 Учебная версия © 2023 ООО "АСКОН-Системы проектирования", Россия. Все права защищены.
 Инв. № подл. Подл. и дата. Инв. № дддл. Подл. и дата. Инв. №. Взам. инв. №. Справ. №. Перв. примен.

Изм.	Лист	№ докум.	Подп.	Дата	Лит.	Масса	Масштаб
Разраб.						0,01	1:1
Проб.					Лист	Листов	1
Т.контр.					Крепление сервопривода		
И.контр.					PLA Пластик		
Утв.							

Не для коммерческого использования
Копировал
Формат A4

Рисунок А.2 – чертеж крепления сервопривода

КОМПАС-3D v22 Учебная версия © 2023 ООО "АСКОН-Системы проектирования", Россия. Все права защищены.
 Инв. № подл. Подл. и дата. Инв. № дддл. Подл. и дата. Инв. № Взам. инв. № Инв. № дддл. Подл. и дата. Справ. № Перв. примен.

Изм.	Лист	№ докум.	Подп.	Дата				
					Сбивалка	Лит.	Масса	Масштаб
Разраб.							0,09	1:1
Проб.						Лист	Листов	1
Т.контр.								
И.контр.					Фанера			
Утв.								

Не для коммерческого использования
Копировал
Формат A4

Рисунок А.3 – Чертеж сбивалки

ПРИЛОЖЕНИЕ Б

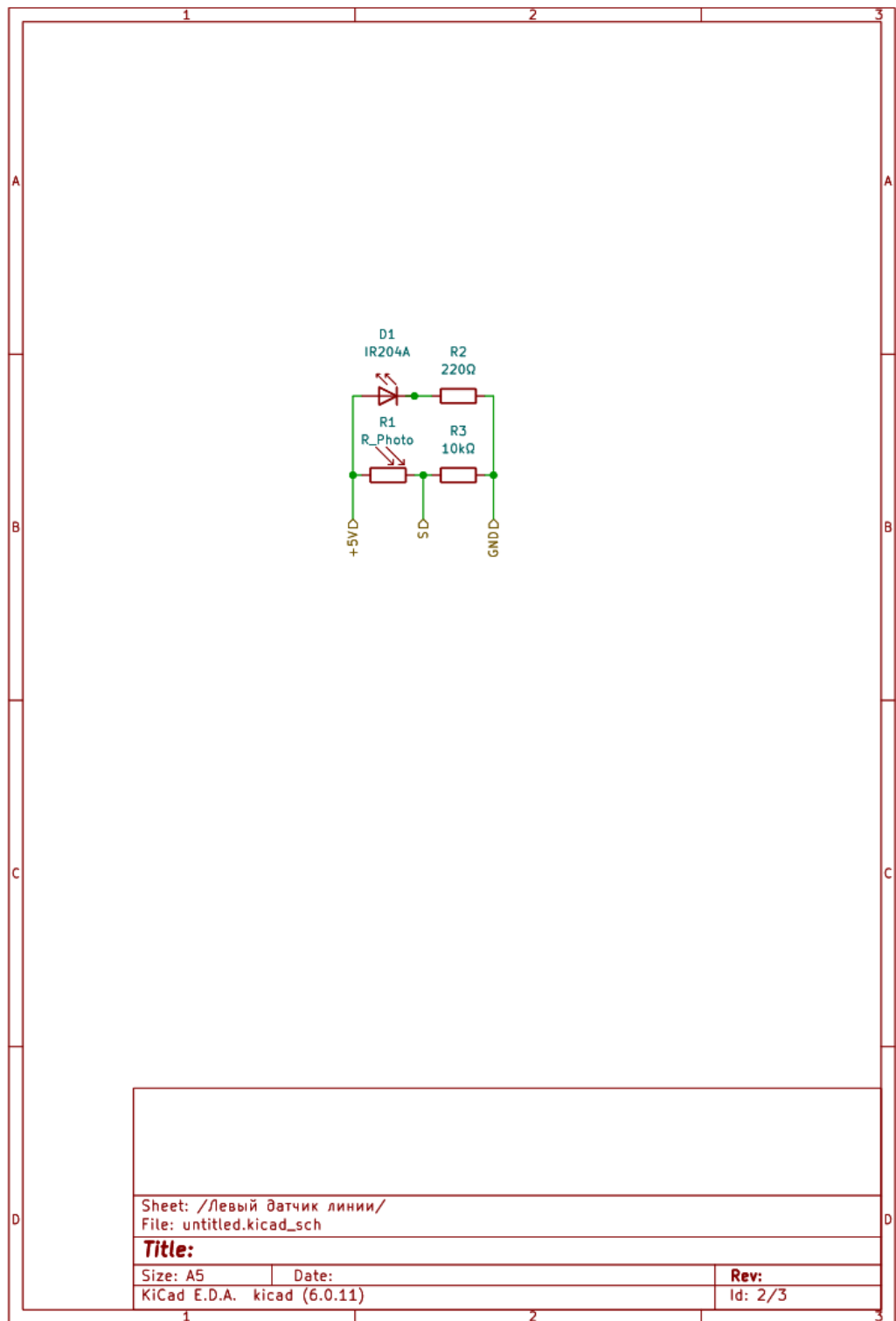


Рисунок Б.1 – Принципиальная схема старого датчика отражённого света

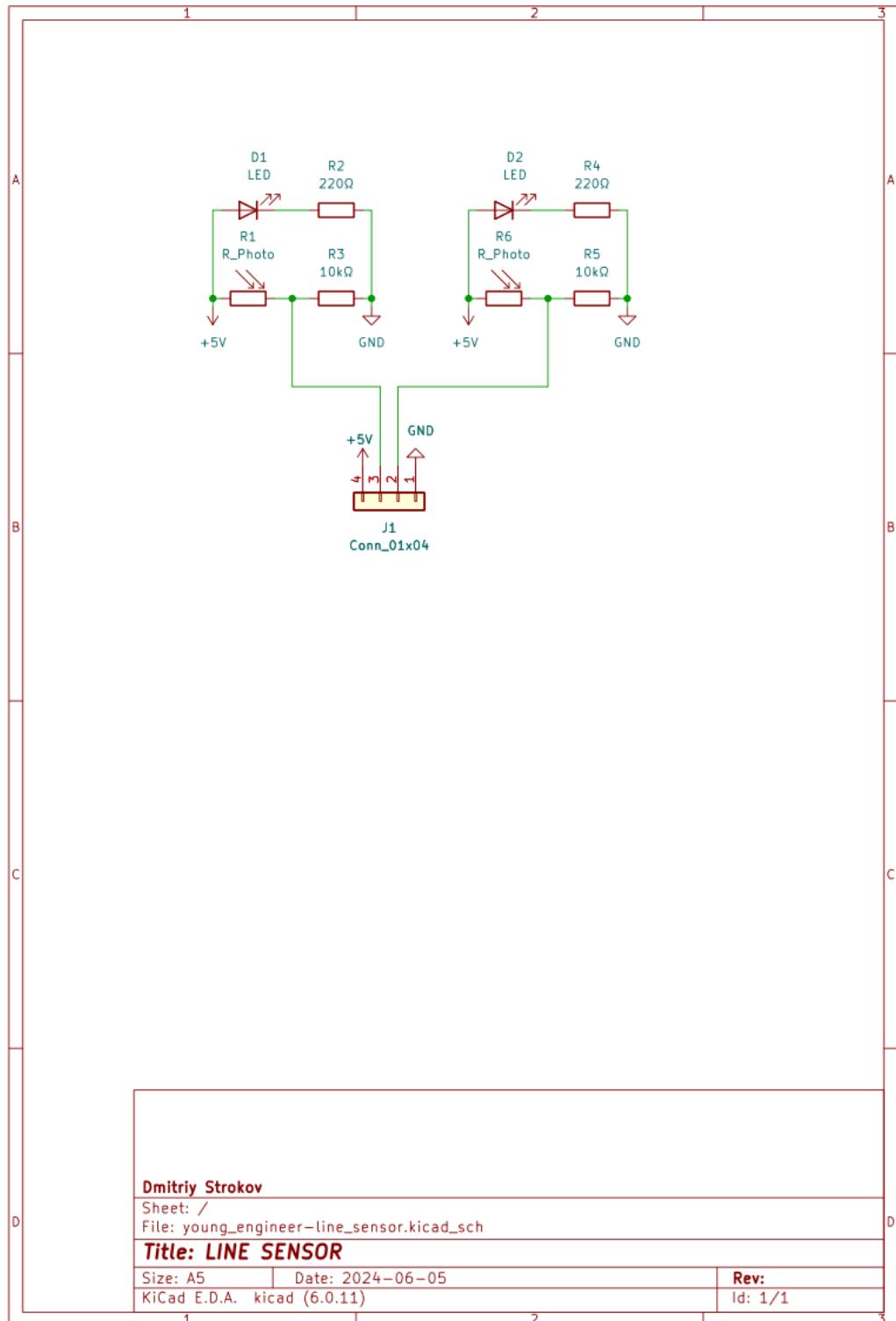


Рисунок Б.2 – Принципиальная схема нового датчика отраженного света

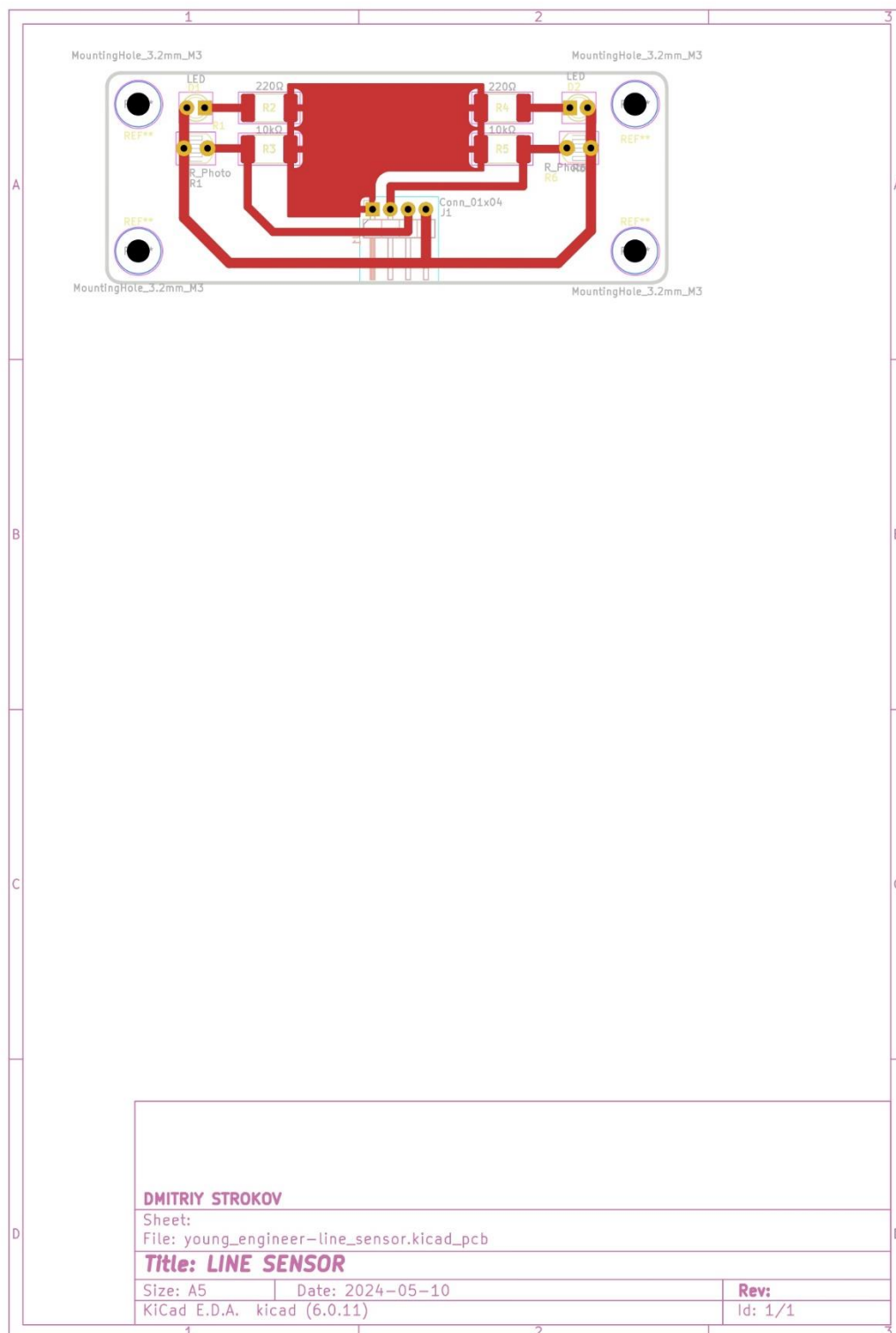


Рисунок Б.3 – Схема печатной платы нового датчика отраженного света

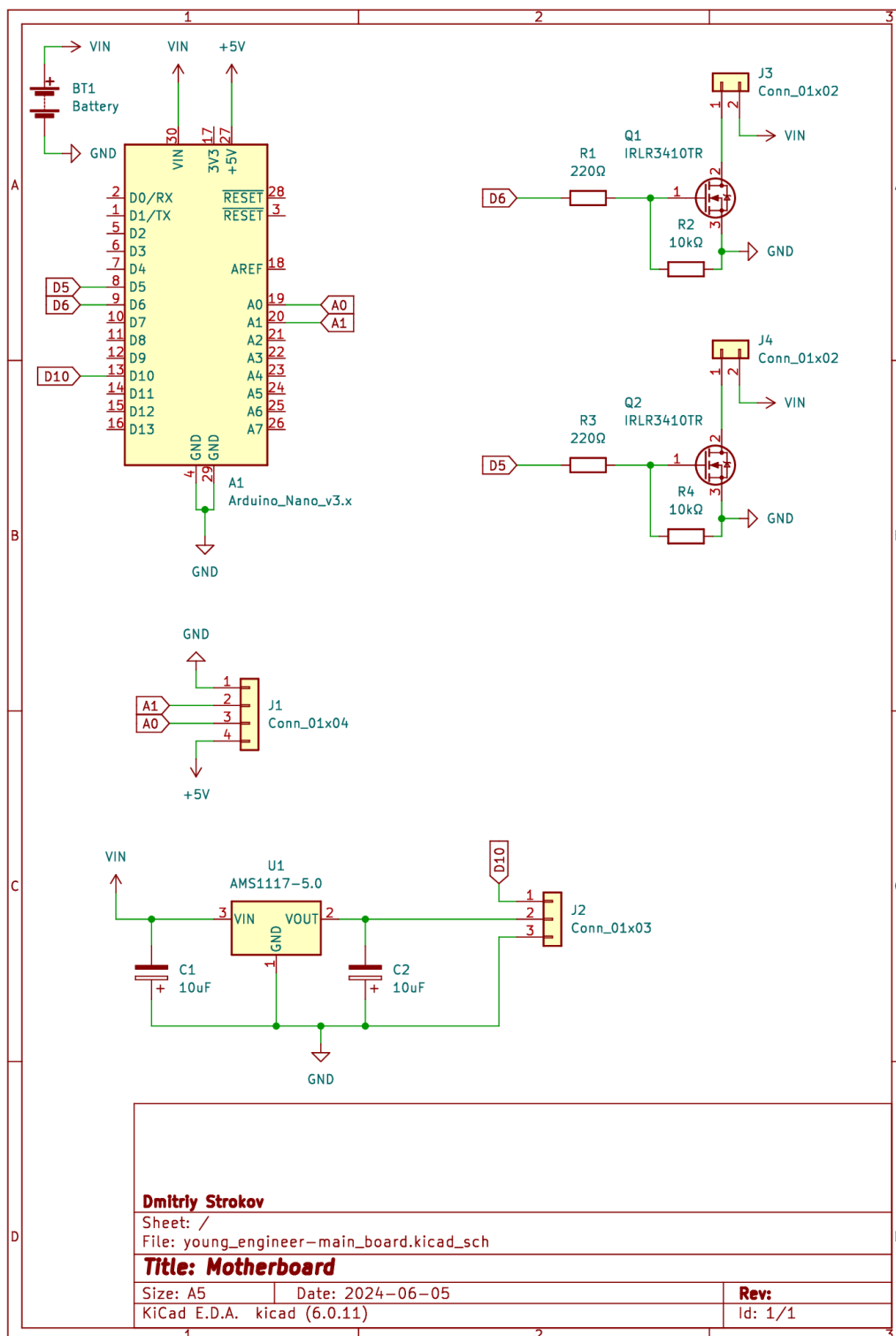


Рисунок Б.4 – Принципиальная схема системной платы

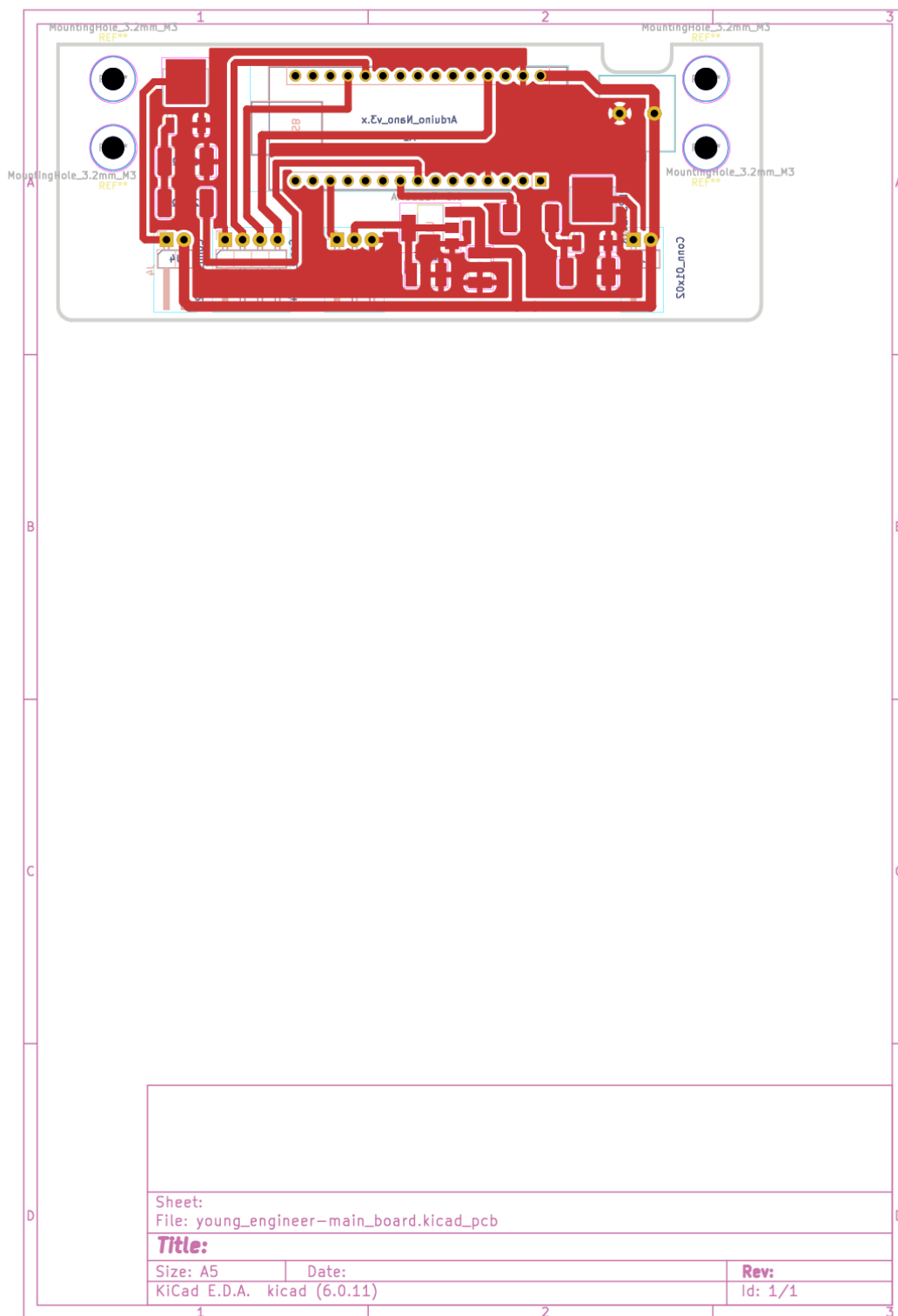


Рисунок Б.5 – Схема печатной системной платы

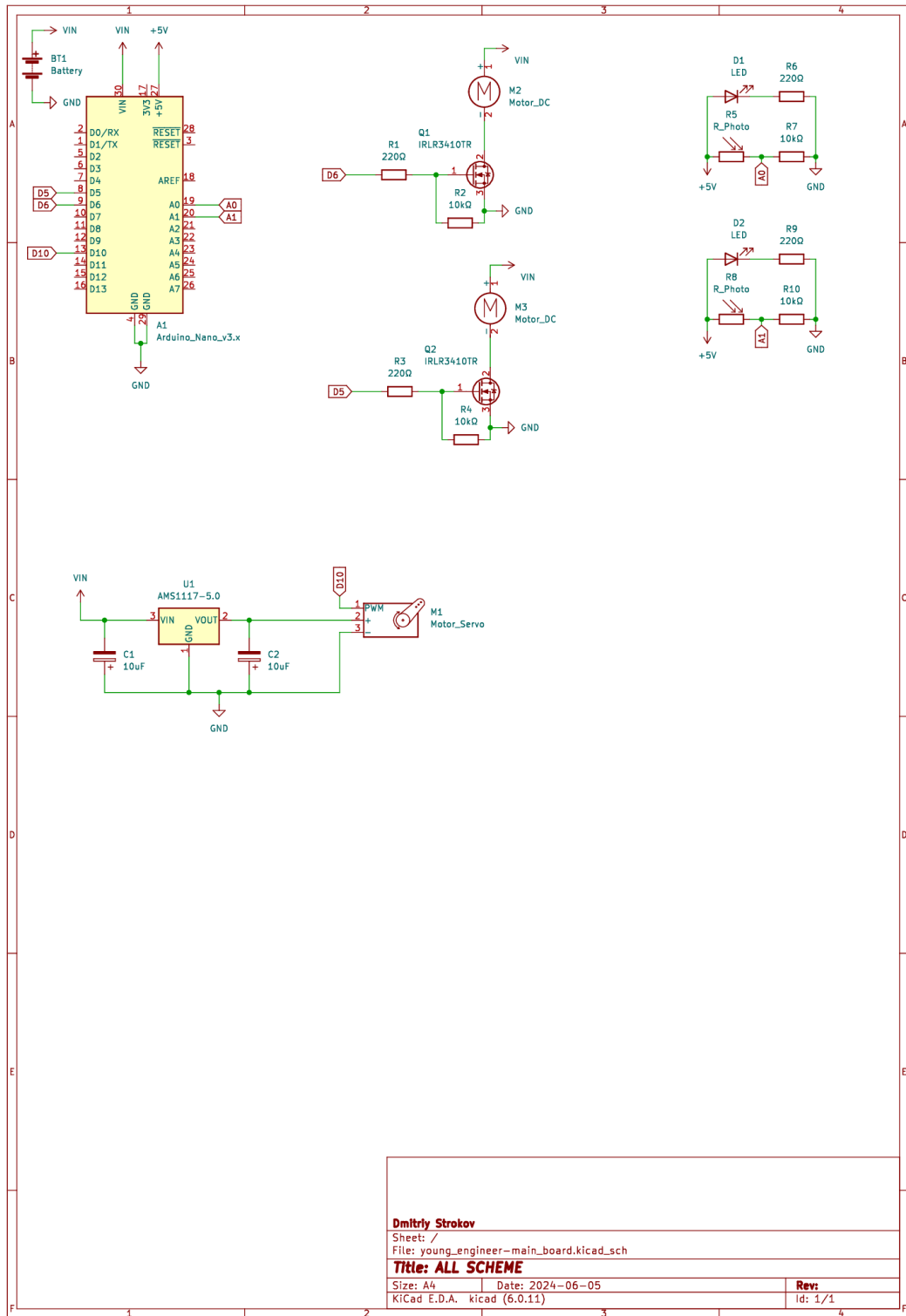


Рисунок Б.6 – Принципиальная схема всего робота

ПРИЛОЖЕНИЕ В

```
young_engineer.ino  move_control.h
1  #pragma once
2  // Строчкой раньше запретили подключение библиотеки более 1 раза
3  #include <Arduino.h> // Подключаем стандартную библиотеку
4
5  class Motor { // Описываем класс библиотеки
6  public:
7      Motor();
8      void attach(byte leftPin, byte rightPin); // говорим компилятору, что должна существовать функция attach с параметрами пинов левого и правого моторов
9      void move(int leftSpeed, int rightSpeed); // говорим компилятору, что должна существовать функция move с параметрами скорости левого и правого моторов
10 private:
11     int _leftPin; // локальная переменная, хранящая пин левого мотора
12     int _rightPin; // локальная переменная, хранящая пин правого мотора
13     int _realLeftSpeed; // настоящее значение подающееся на пин левого мотора
14     int _realRightSpeed; // настоящее значение подающееся на пин правого мотора
15 };
16
17
18 Motor::Motor() {} // Пустой класс-обёртка
19
20 void Motor::attach(byte leftPin, byte rightPin) { // функция для инициализации портов моторов
21     _leftPin = leftPin; // записываем параметр (пин левого мотора) в локальную переменную класса
22     _rightPin = rightPin; // записываем параметр (пин правого мотора) в локальную переменную класса
23
24     pinMode(_leftPin, OUTPUT); // инициализация пина левого мотора как пин для подачи сигнала
25     pinMode(_rightPin, OUTPUT); // инициализация пина правого мотора как пин для подачи сигнала
26 }
27
28 void Motor::move(int leftSpeed, int rightSpeed) { // функция для задачи скорости скорости моторов
29     _realLeftSpeed = map(constrain(abs(leftSpeed), 0, 100), 0, 100, 0, 255); // переводим скорость левого мотора из диапазон от 0 до 100 в диапазон от 0 до 255
30     analogWrite(_leftPin, _realLeftSpeed); // задача скорости левого мотора
31
32     _realRightSpeed = map(constrain(abs(rightSpeed), 0, 100), 0, 100, 0, 255); // переводим скорость левого мотора из диапазон от 0 до 100 в диапазон от 0 до 255
33     analogWrite(_rightPin, _realRightSpeed); // задача скорости правого мотора
34 }
35
36
```

Рисунок В.1 – Текст кода библиотеки для управления моторами