

# РОБОТИЗИРОВАННАЯ ПЛАТФОРМА

На колесах Бенгта Илона

Выполнил Павловский Иван,  
ученик 10 класса МАОУ “Лицей №38”

Руководители:

Еделев А.Ю.

Еделев Ю.А.

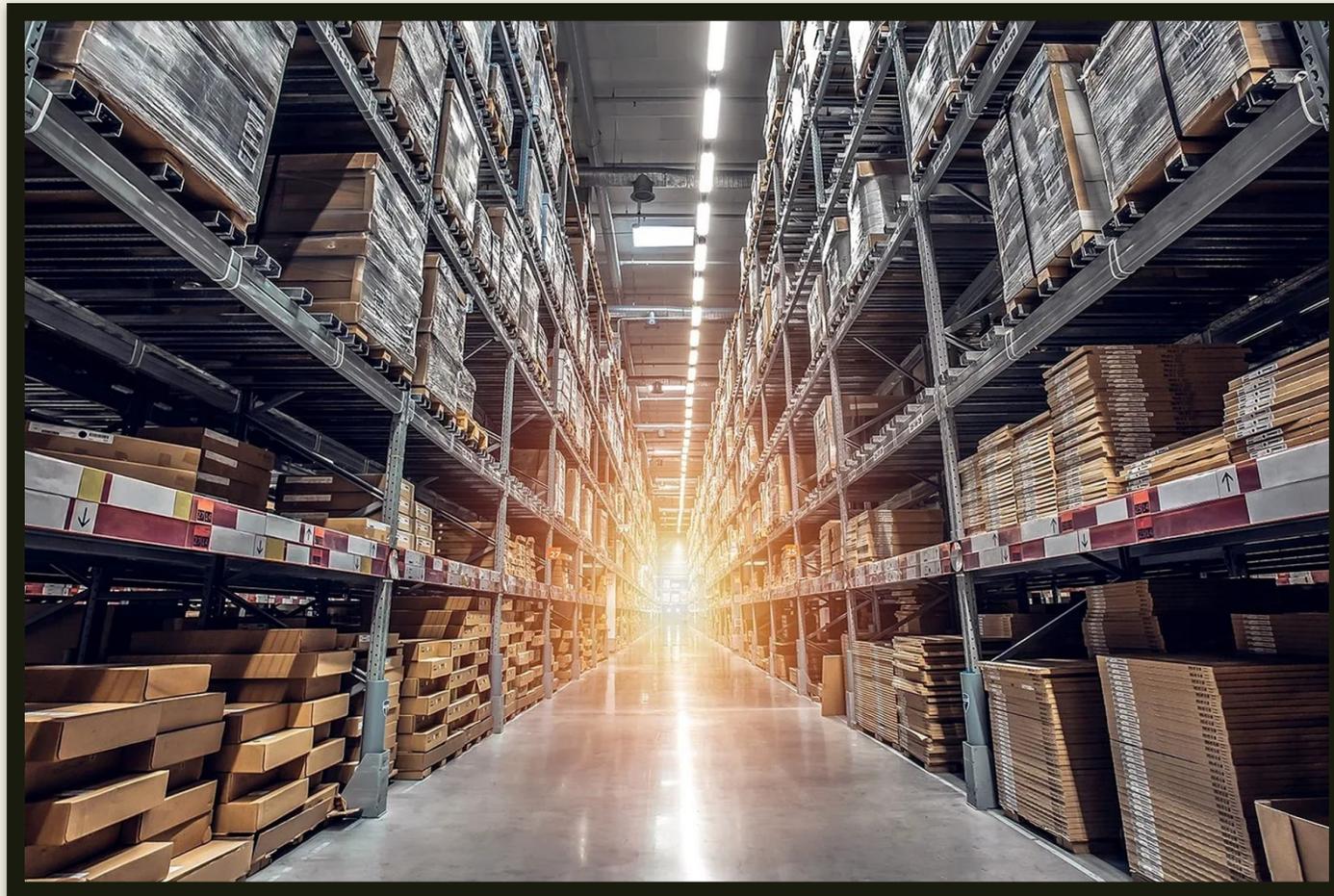
# Актуальность

- Ограниченные пространства для маневров на складских помещениях замедляют транспортировку
- Высокоманевренные платформы стоят значительно дороже стандартных устройств
- Ускорение транспортировки делает её значительно дороже



# Цель

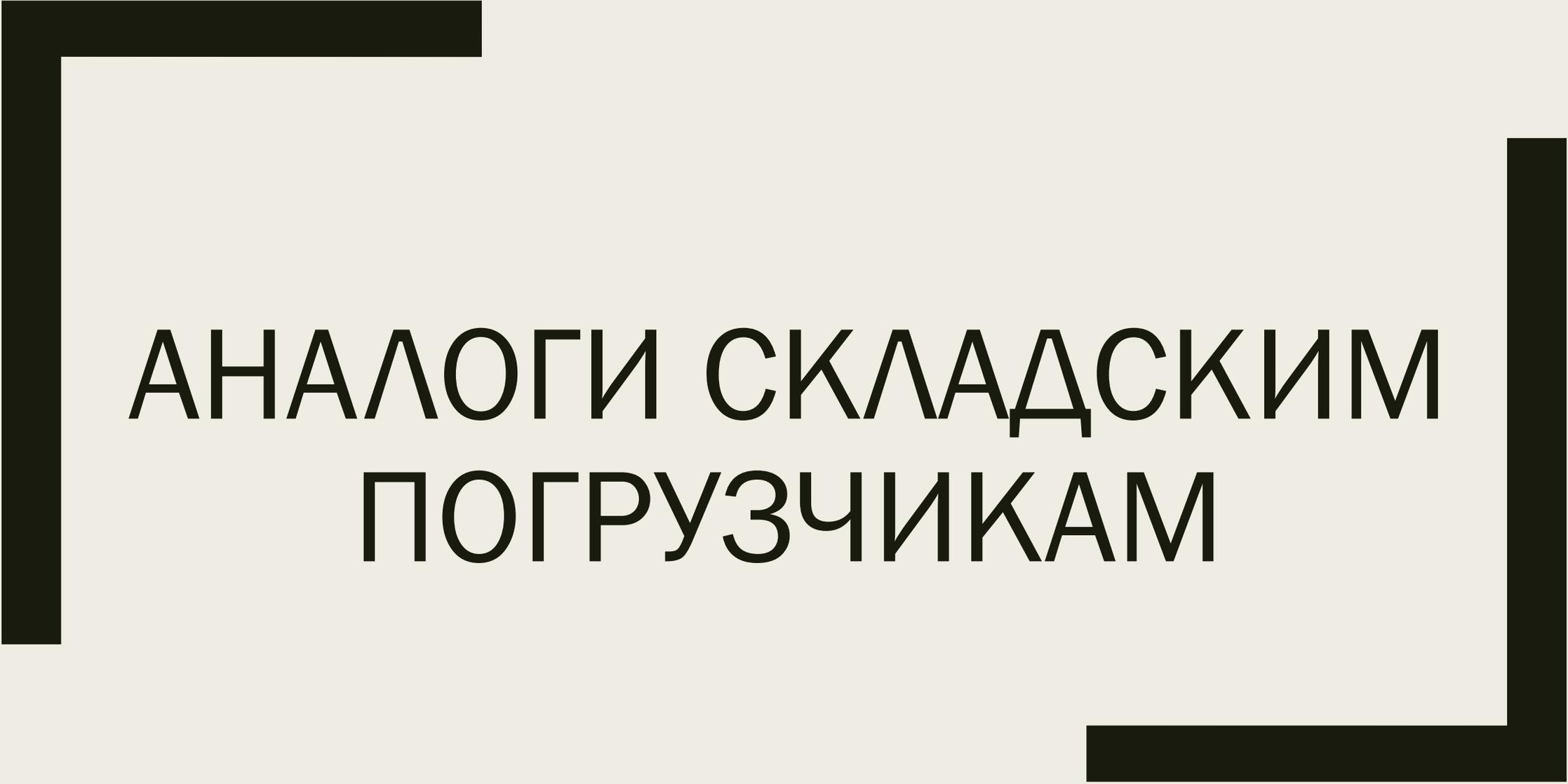
Создать высокоманевренное роботизированное устройство для транспортировки грузов.



# Задачи

- Исследовать аналоги стандартным складским транспортировочным устройствам
- Спроектировать и создать собственное устройство
- Провести опыты и тестирования





# АНАЛОГИ СКЛАДСКИМ ПОГРУЗЧИКАМ



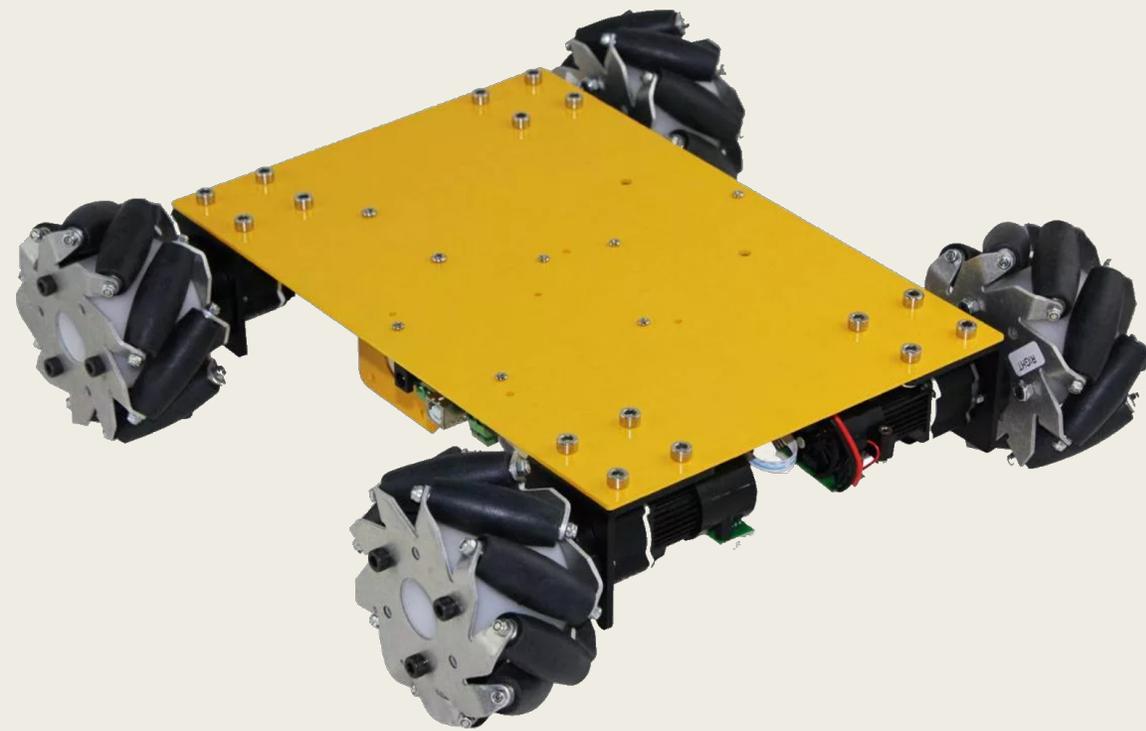
Гусеничная тележка



Тележка с четырьмя независимыми приводами



Тележка с двумя осями  
вращения колеса



Тележка на колесах Илона

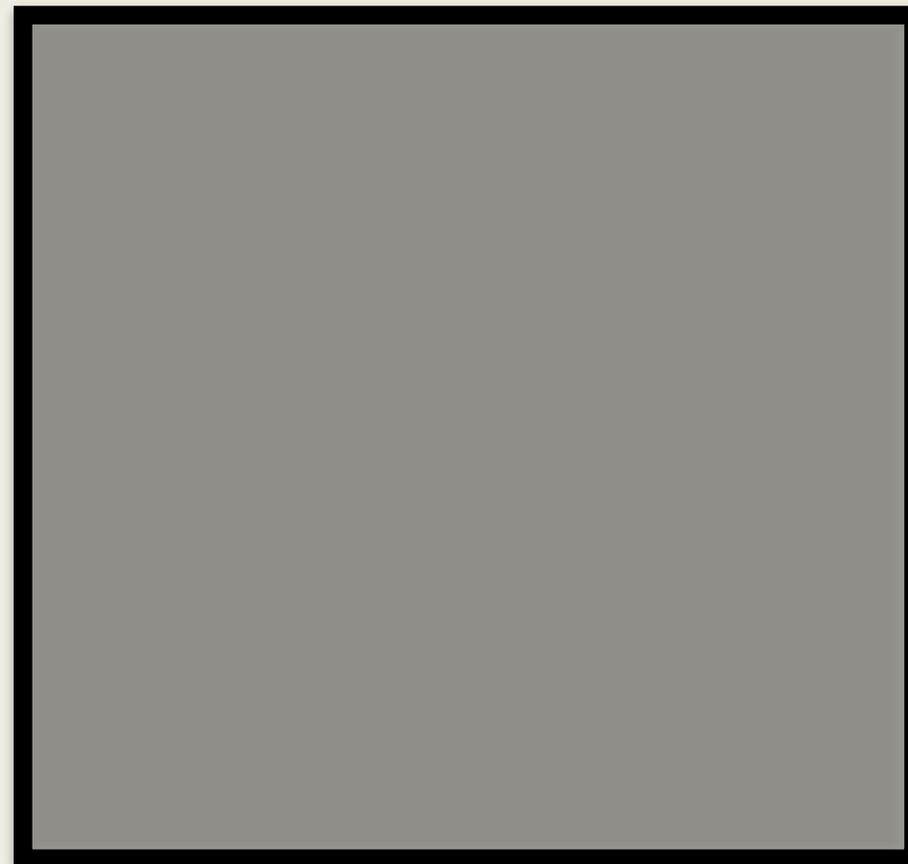


КОЛЕСО ИЛОНА

# Колесо Илона



Колесо



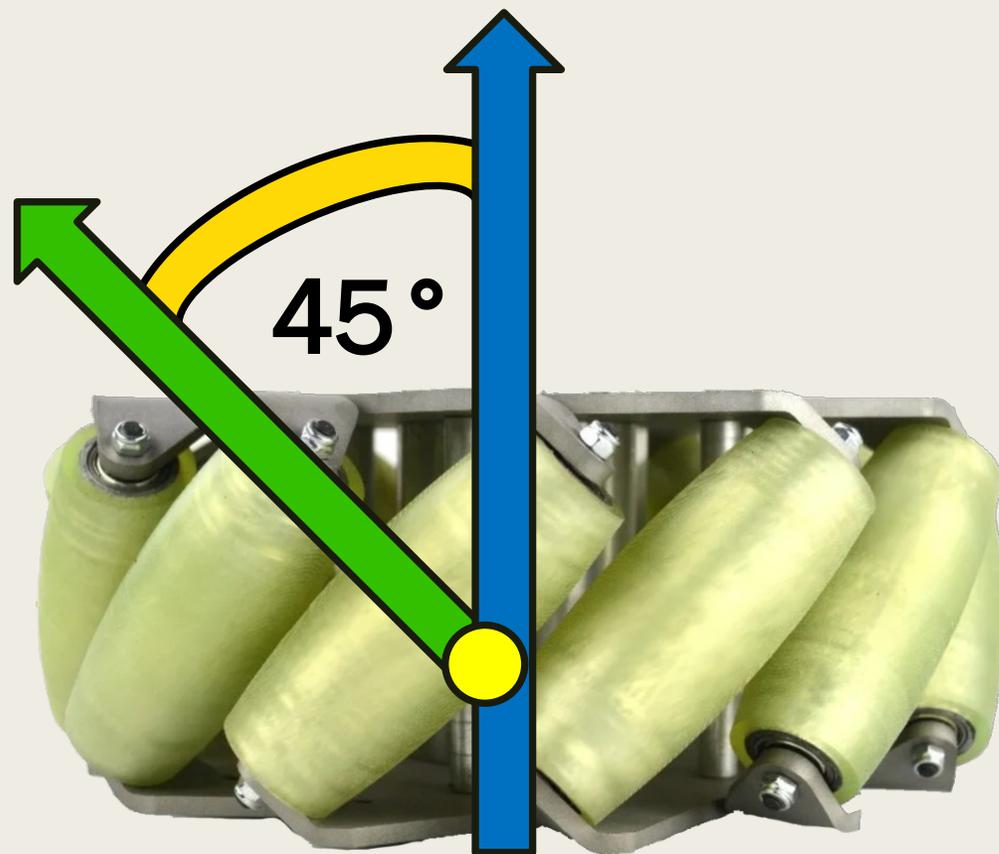
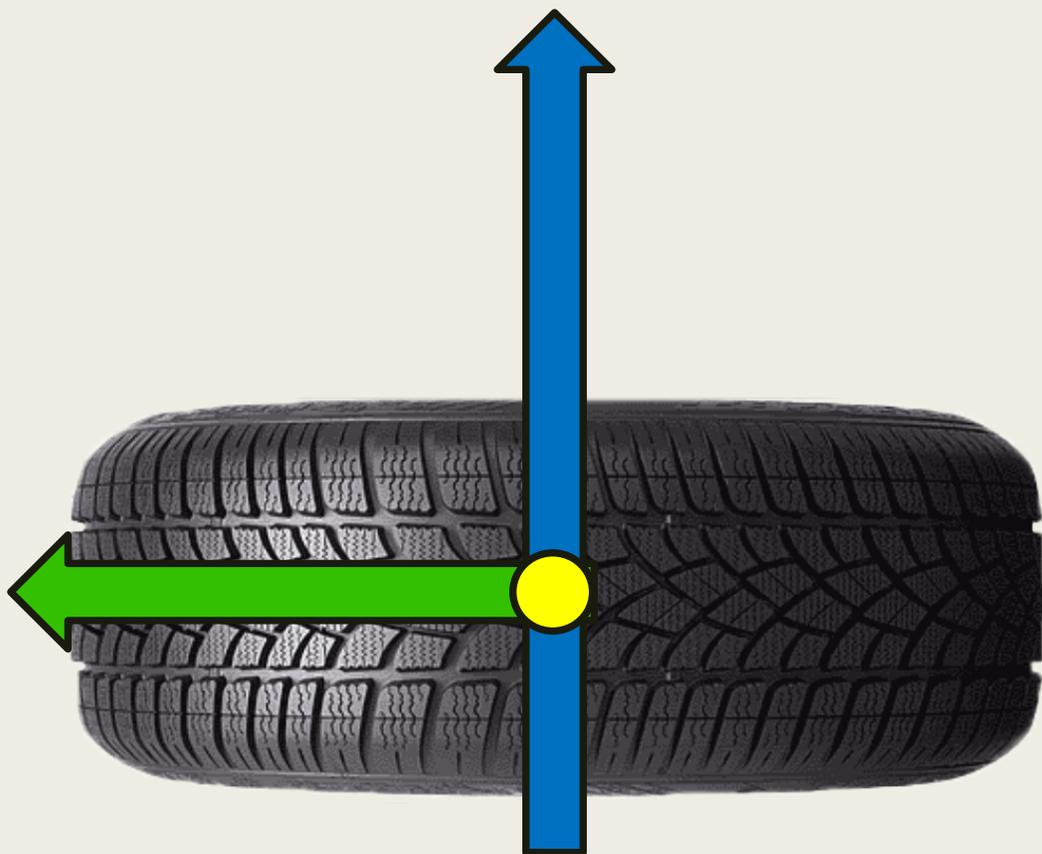
Движение платформы на  
колесах Илона



Ось Вращения



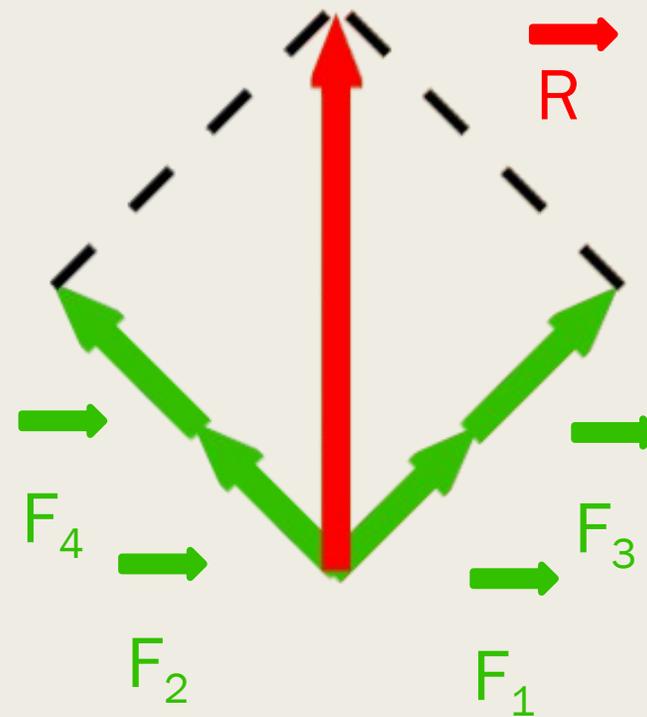
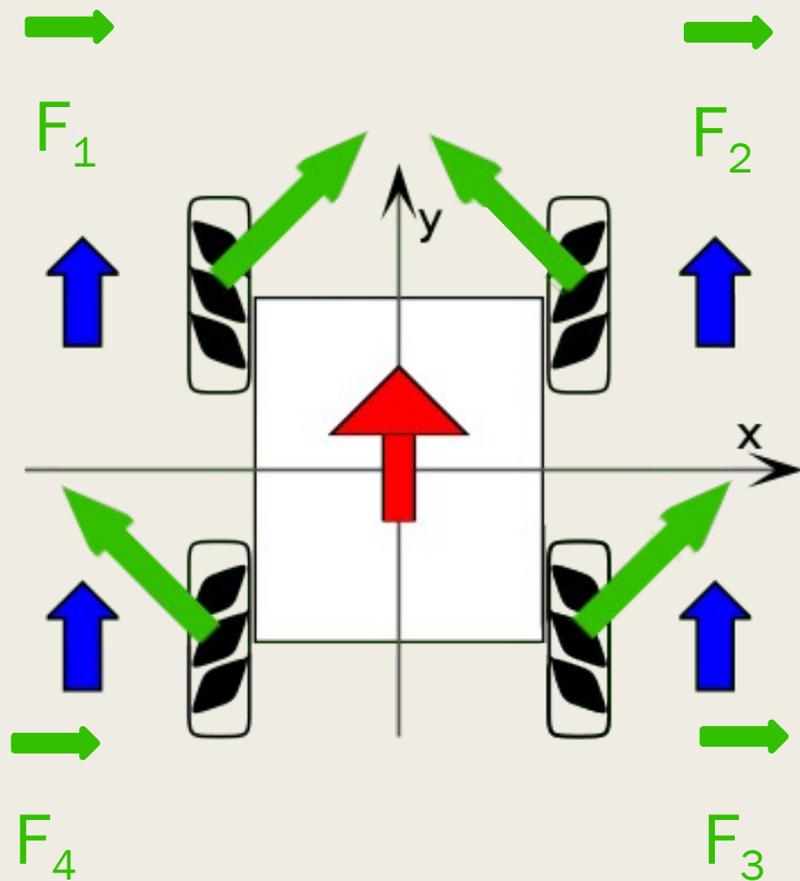
Сила Трения Качения



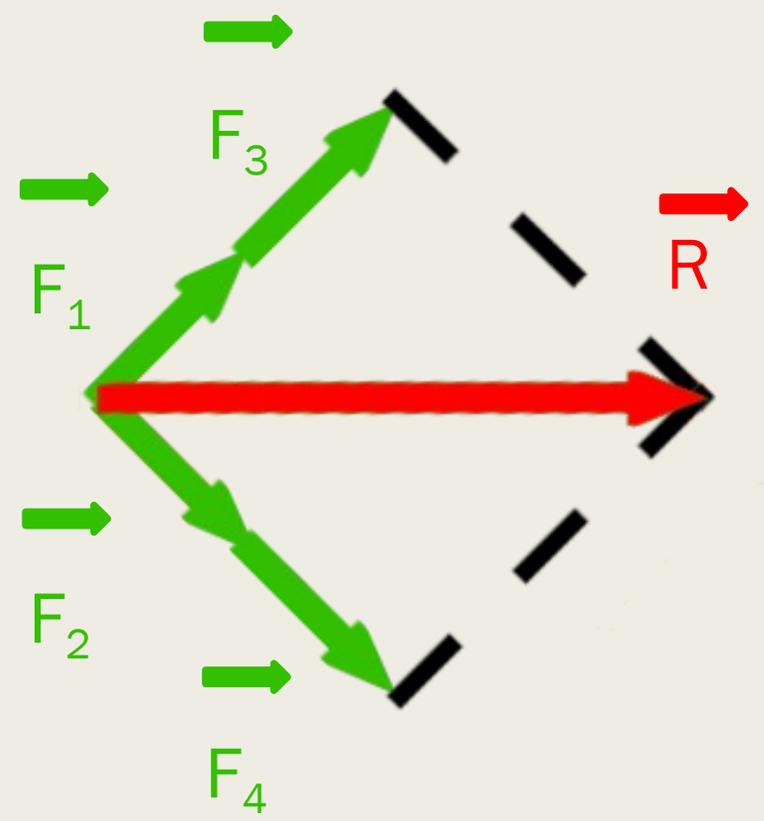
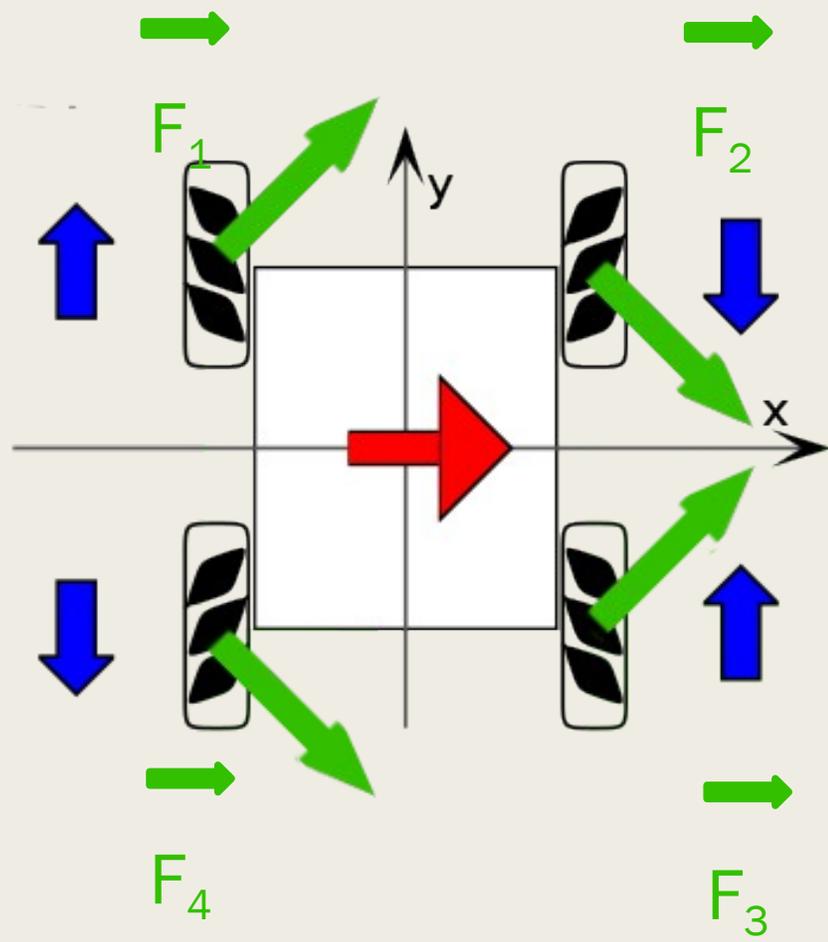
The image features two large, thick black L-shaped brackets. One is positioned on the left side, with its vertical bar extending downwards and its horizontal bar extending to the right. The other is on the right side, with its vertical bar extending upwards and its horizontal bar extending to the left. These brackets frame the central text.

ТЕОРЕТИЧЕСКОЕ  
ОБОСНОВАНИЕ  
ДВИЖЕНИЯ

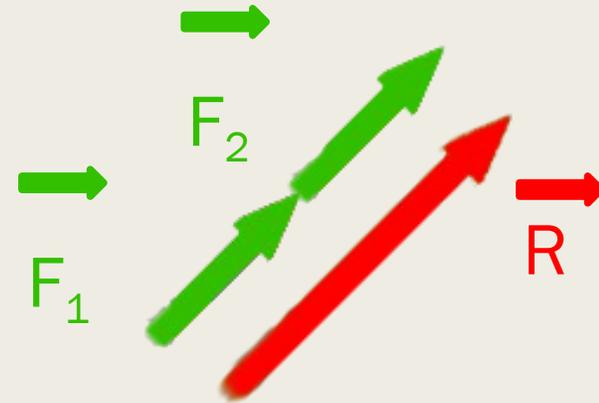
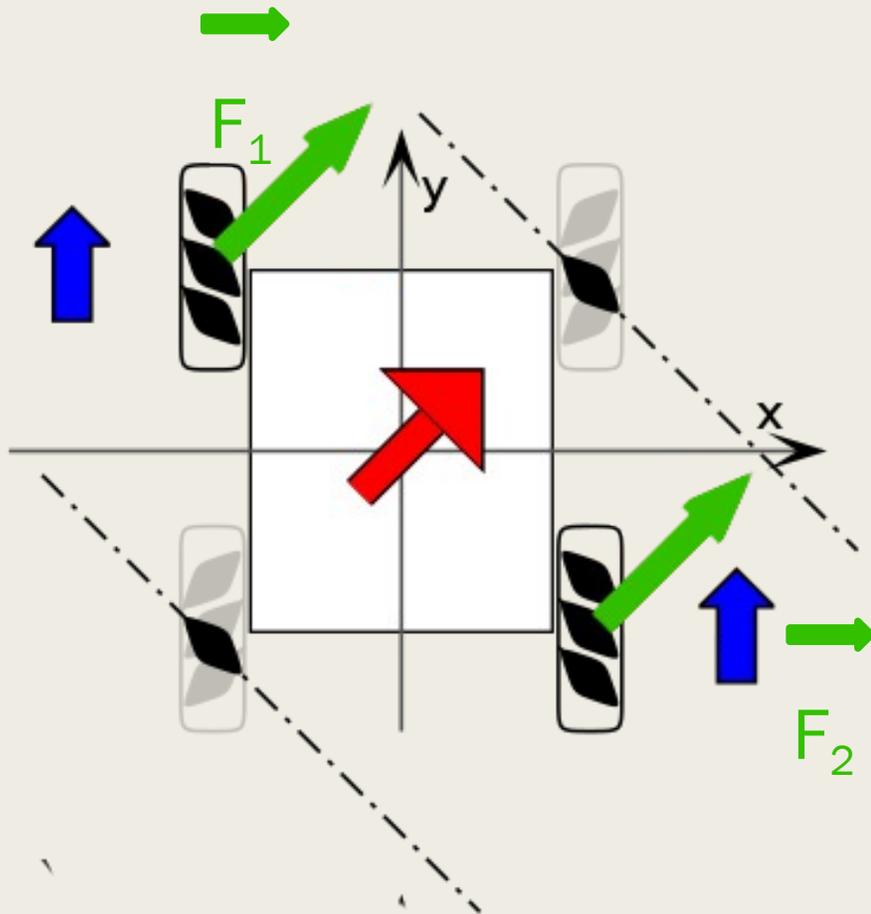
# Движение вперед



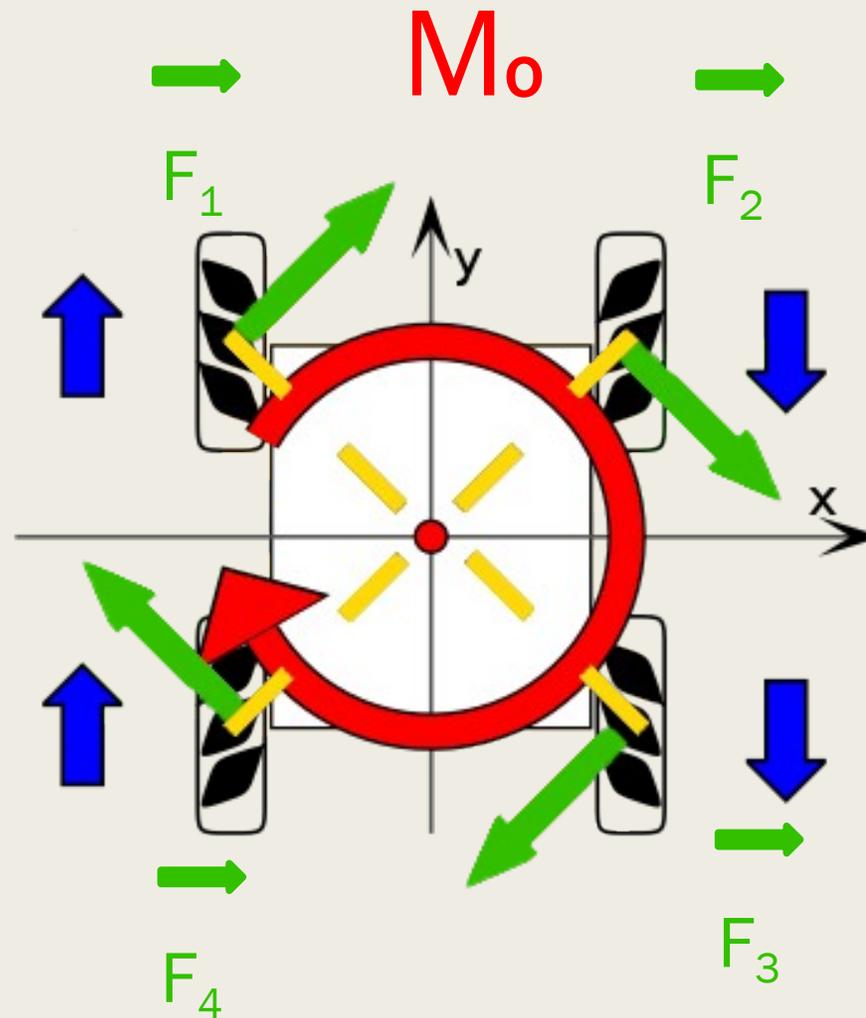
# Движение вправо



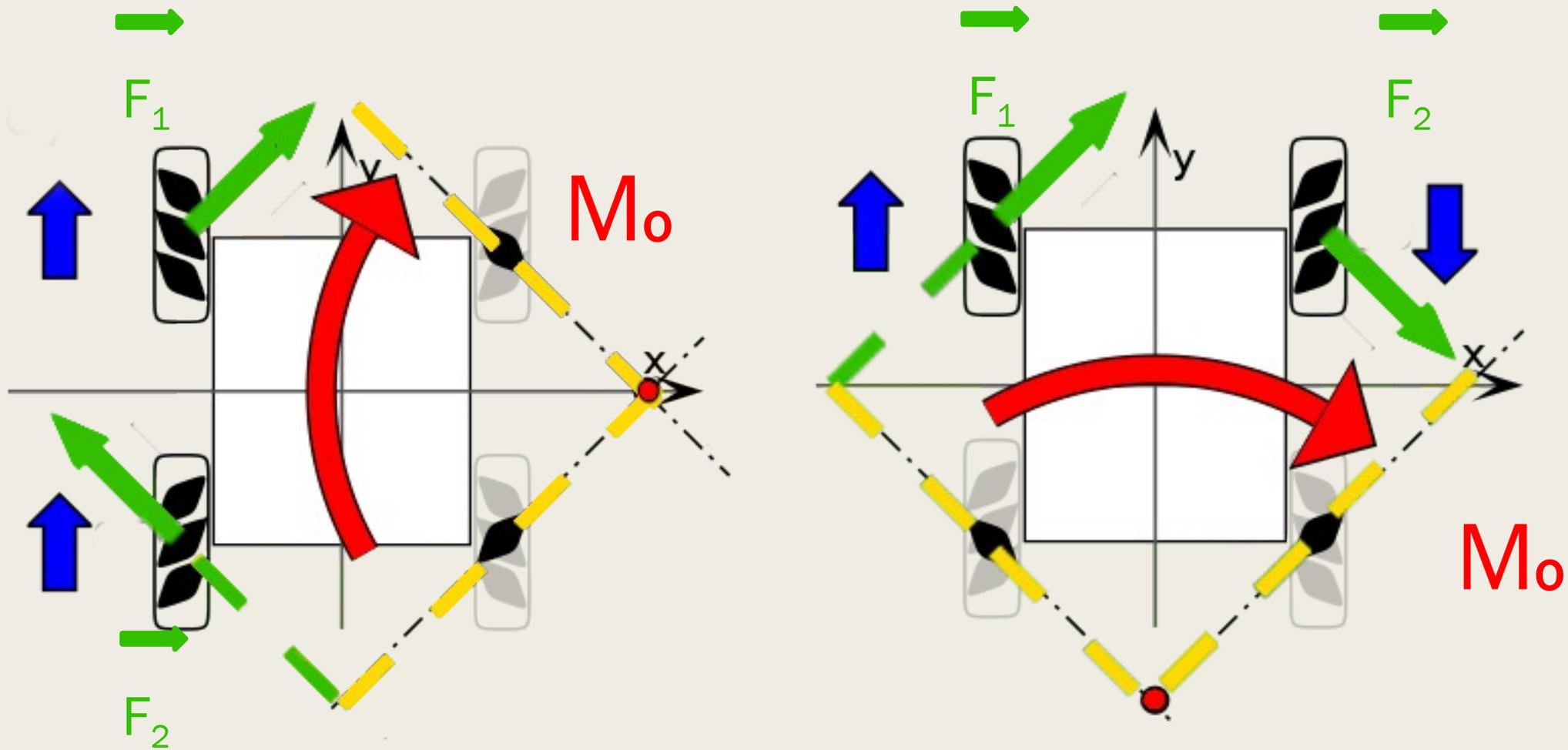
# Движение по диагонали



# Вращение относительно центра



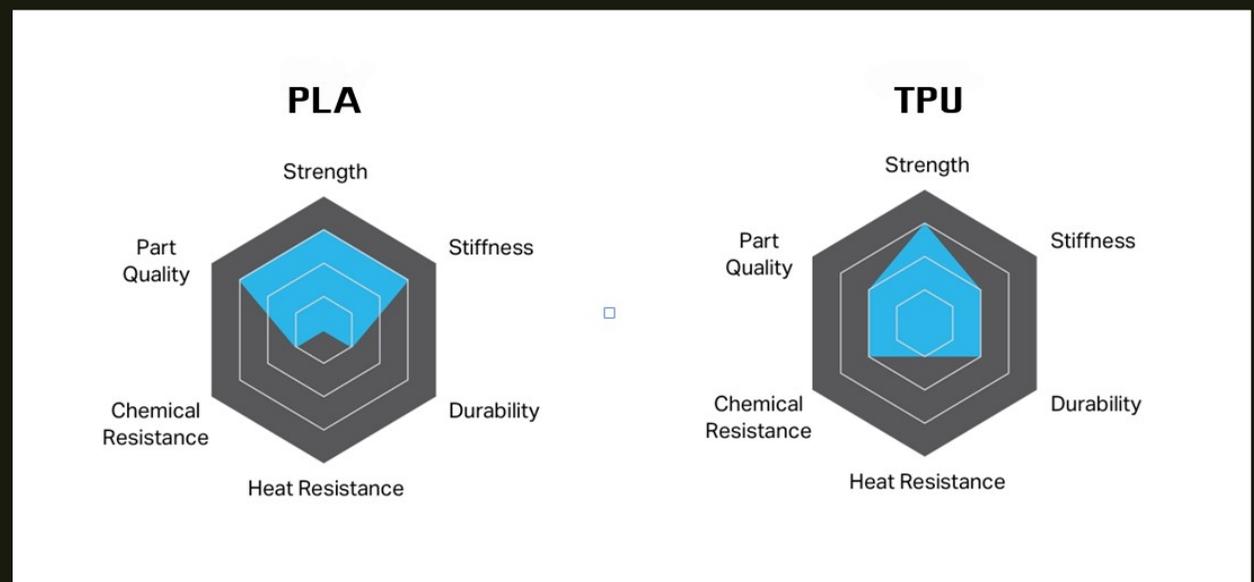
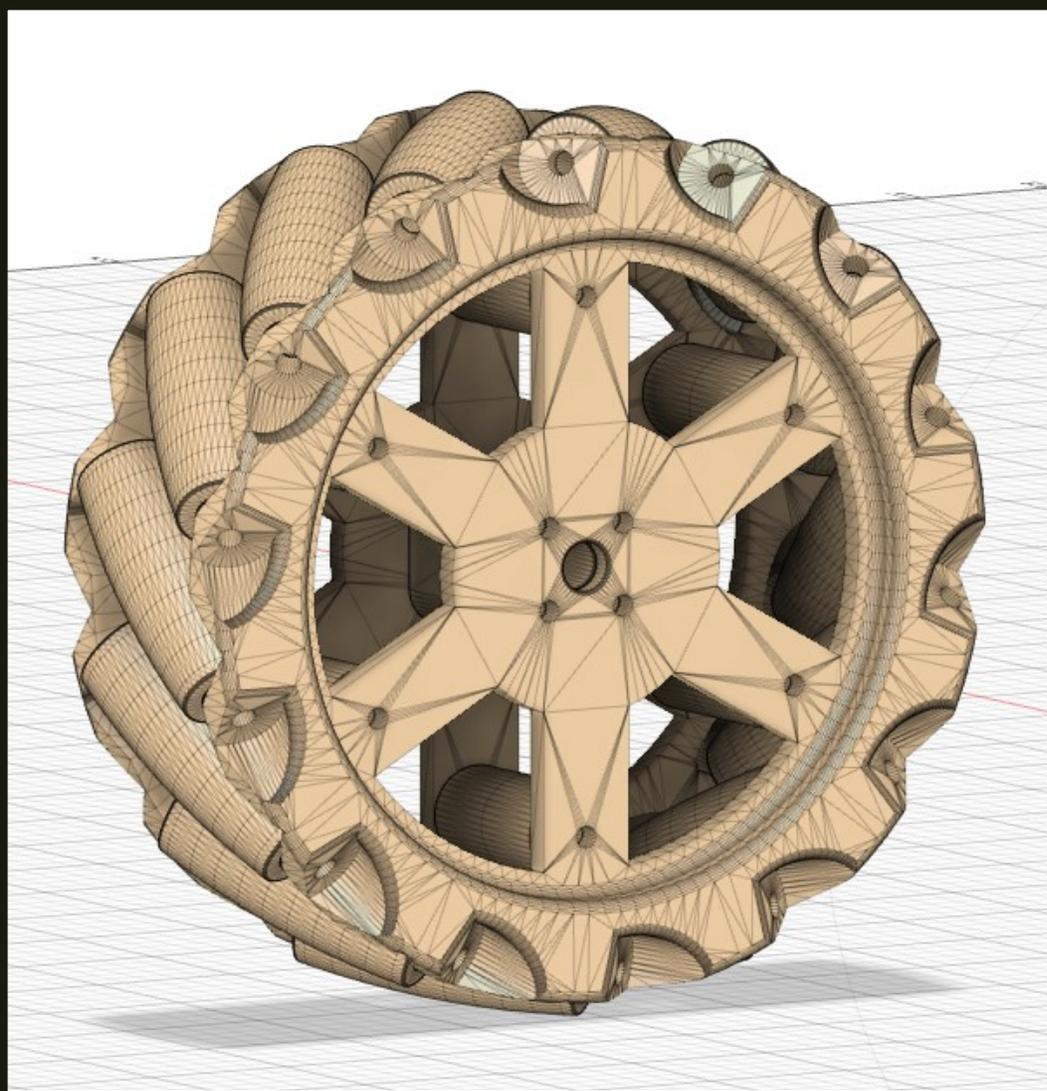
# Вращение относительно точки вне платформы



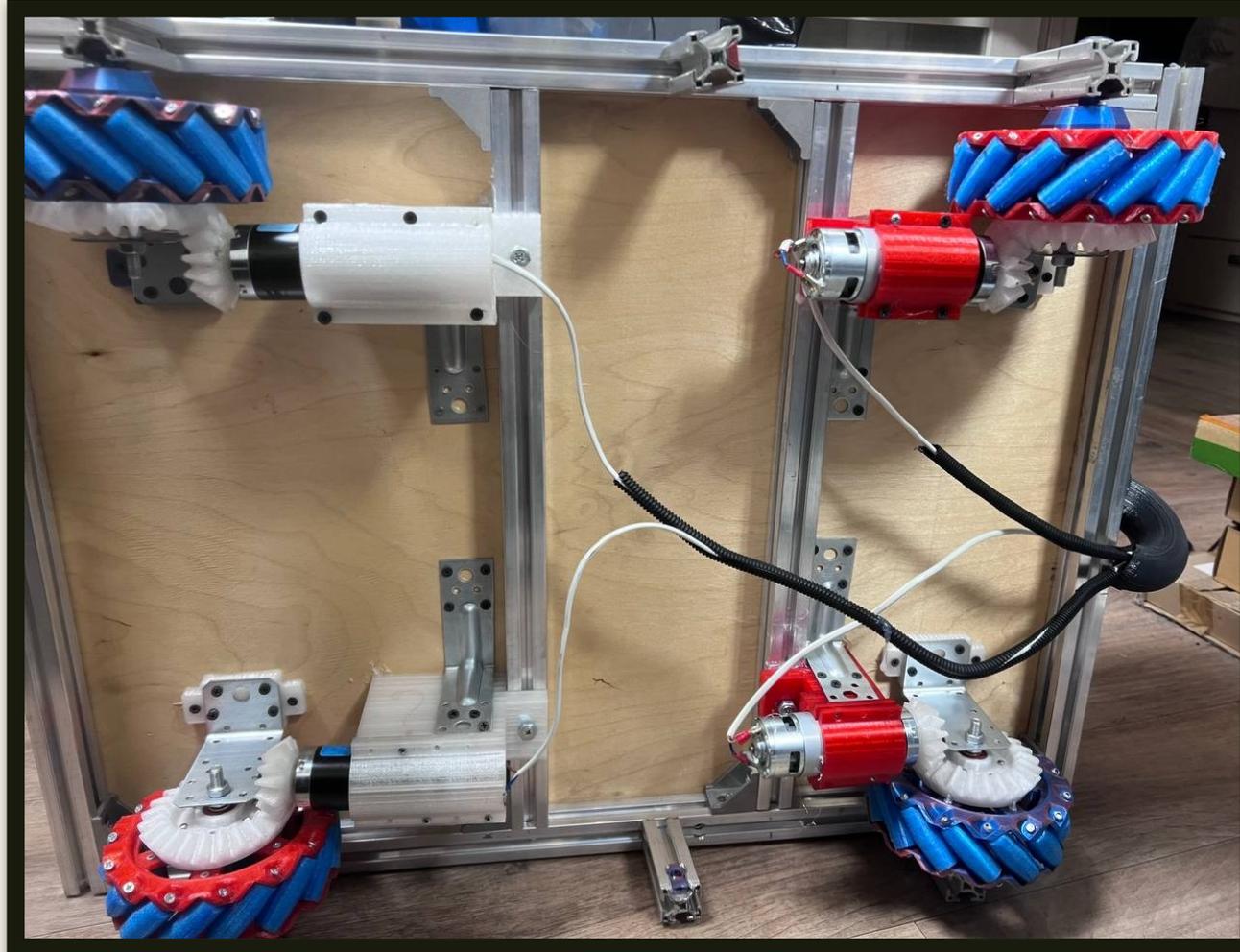


# КОНСТРУИРОВАНИЕ ПЛАТФОРМЫ

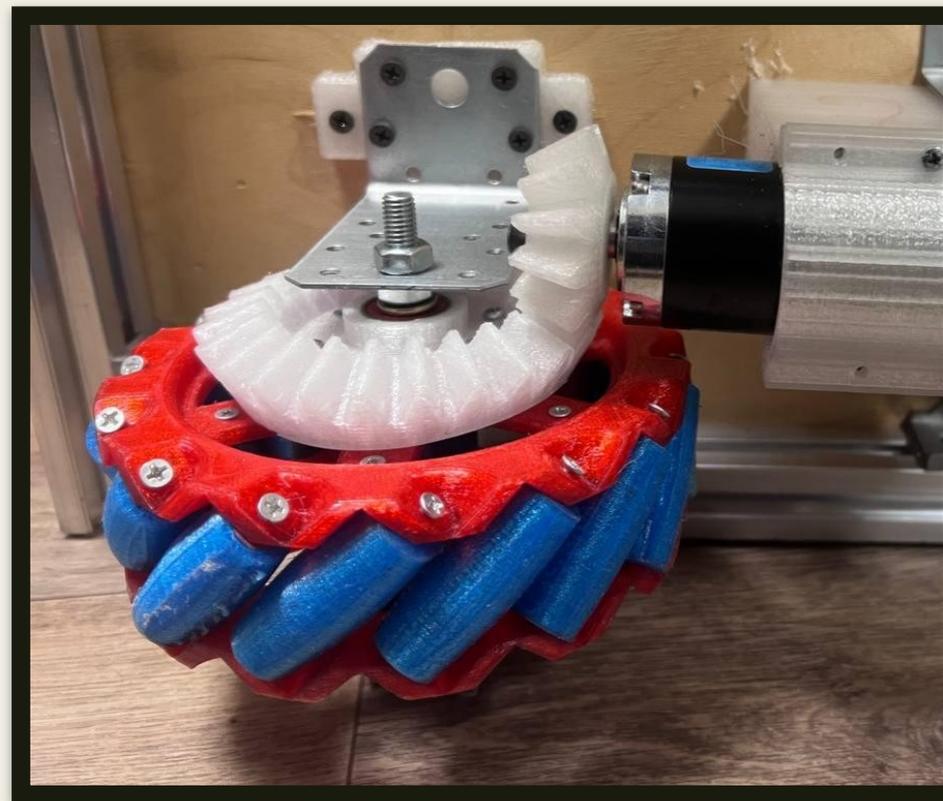
# Моделирование колеса



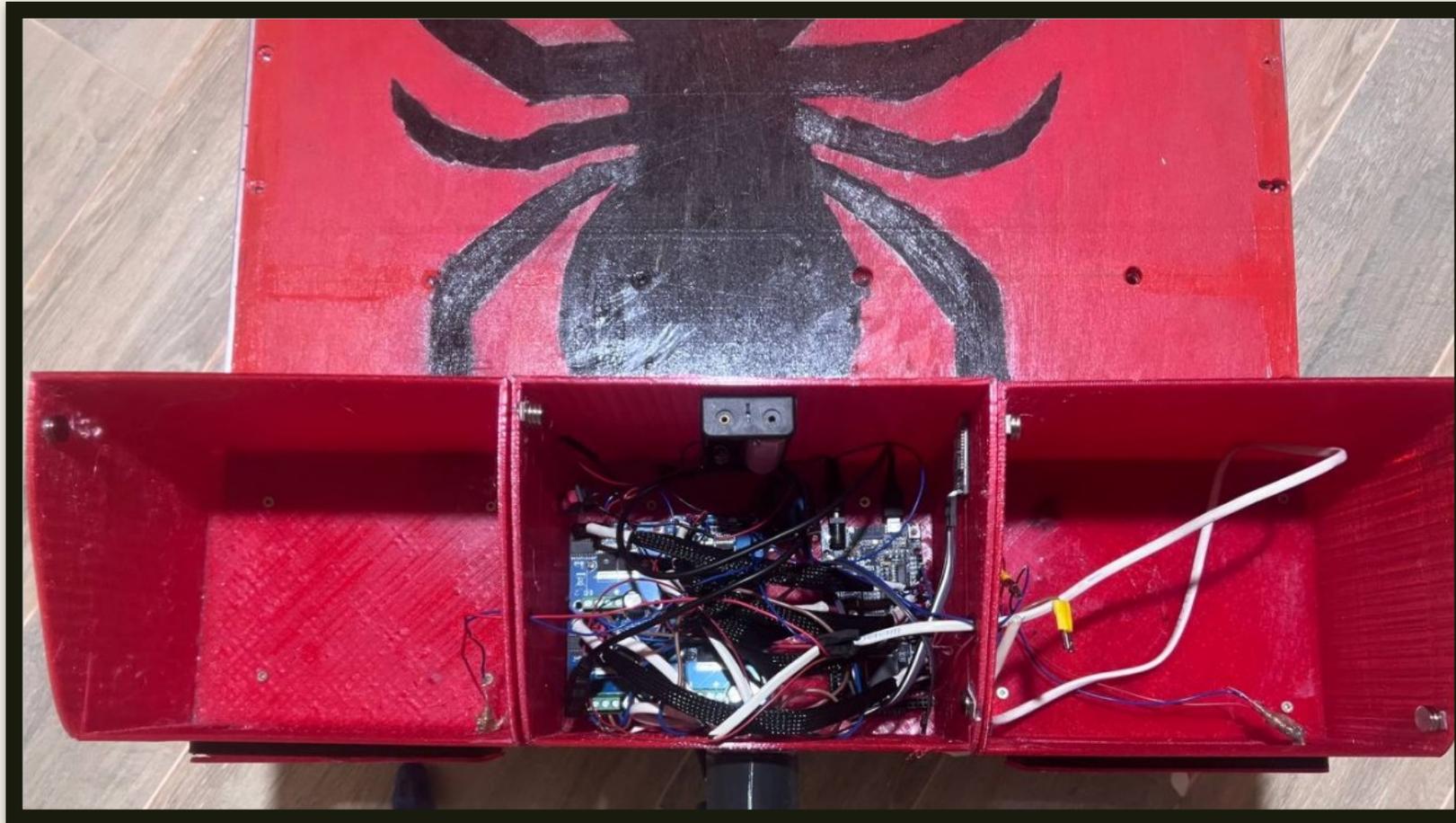
# Конструирование платформы



# Двигатели и передача



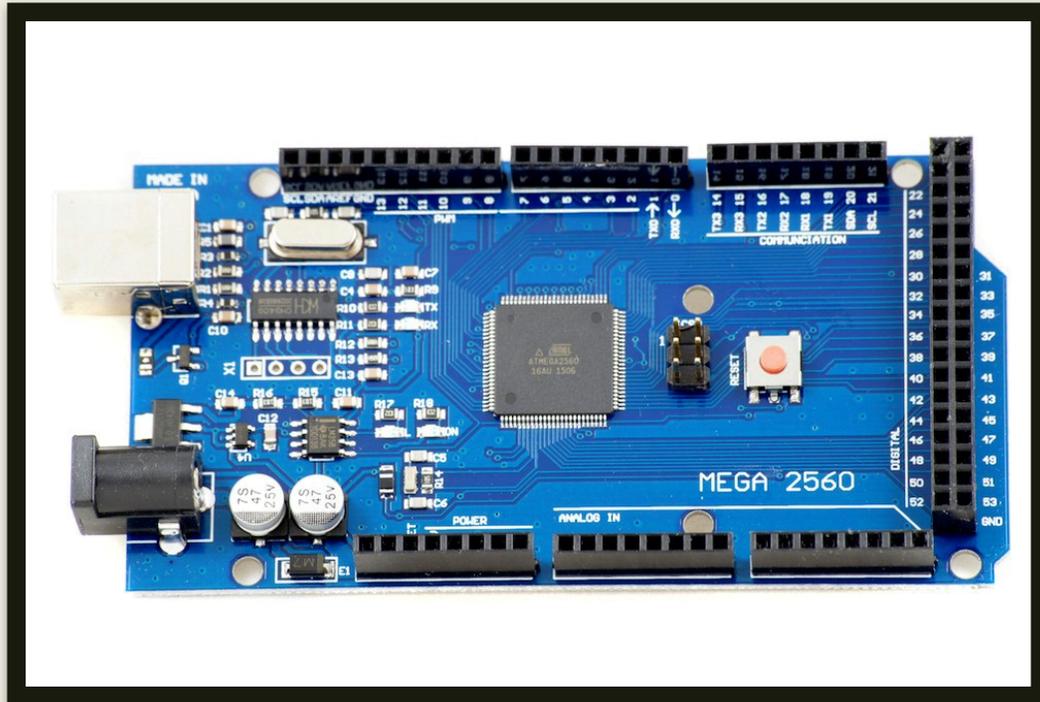
# Отсеки для электроники





# ЭЛЕКТРОННЫЕ КОМПОНЕНТЫ

# Плата Arduino MEGA



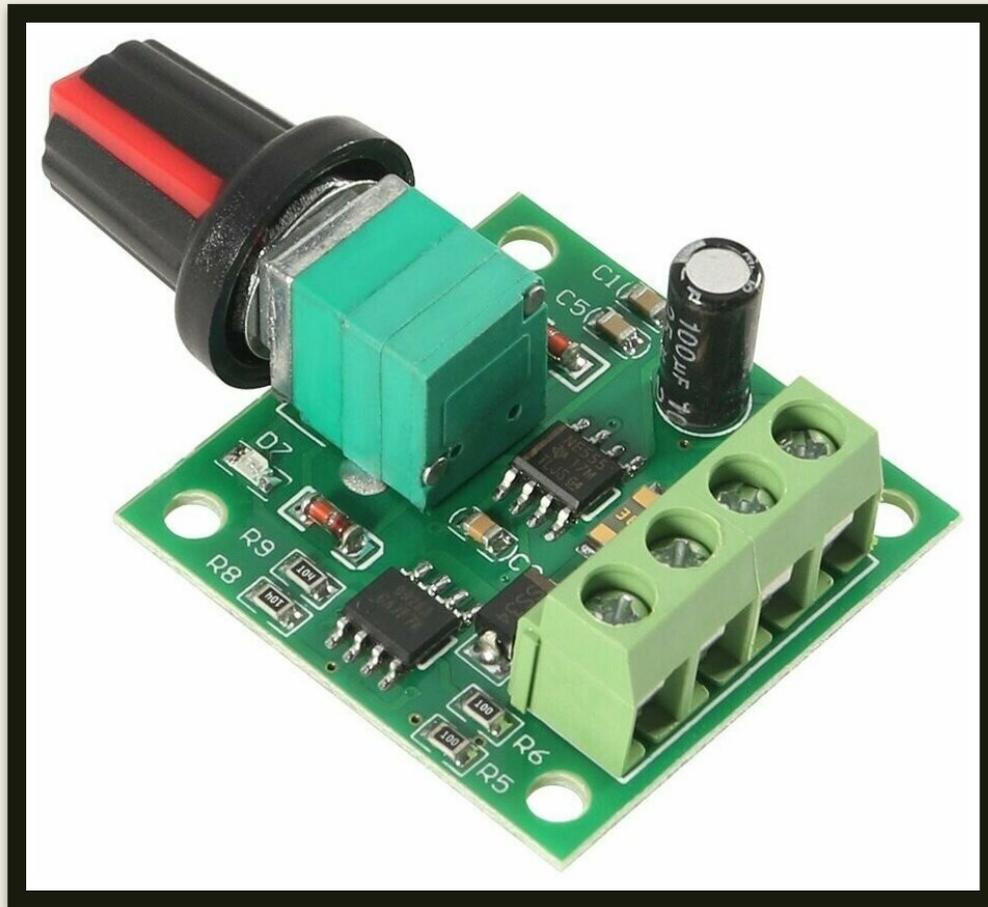
Рабочее напряжение: 5В

Цифровые входы: 54

Аналоговые входы: 16

Входы с поддержкой ШИМ: 14

# ШИМ-контроллеры



Номинальное напряжение: 15В

Номинальный ток: 2А

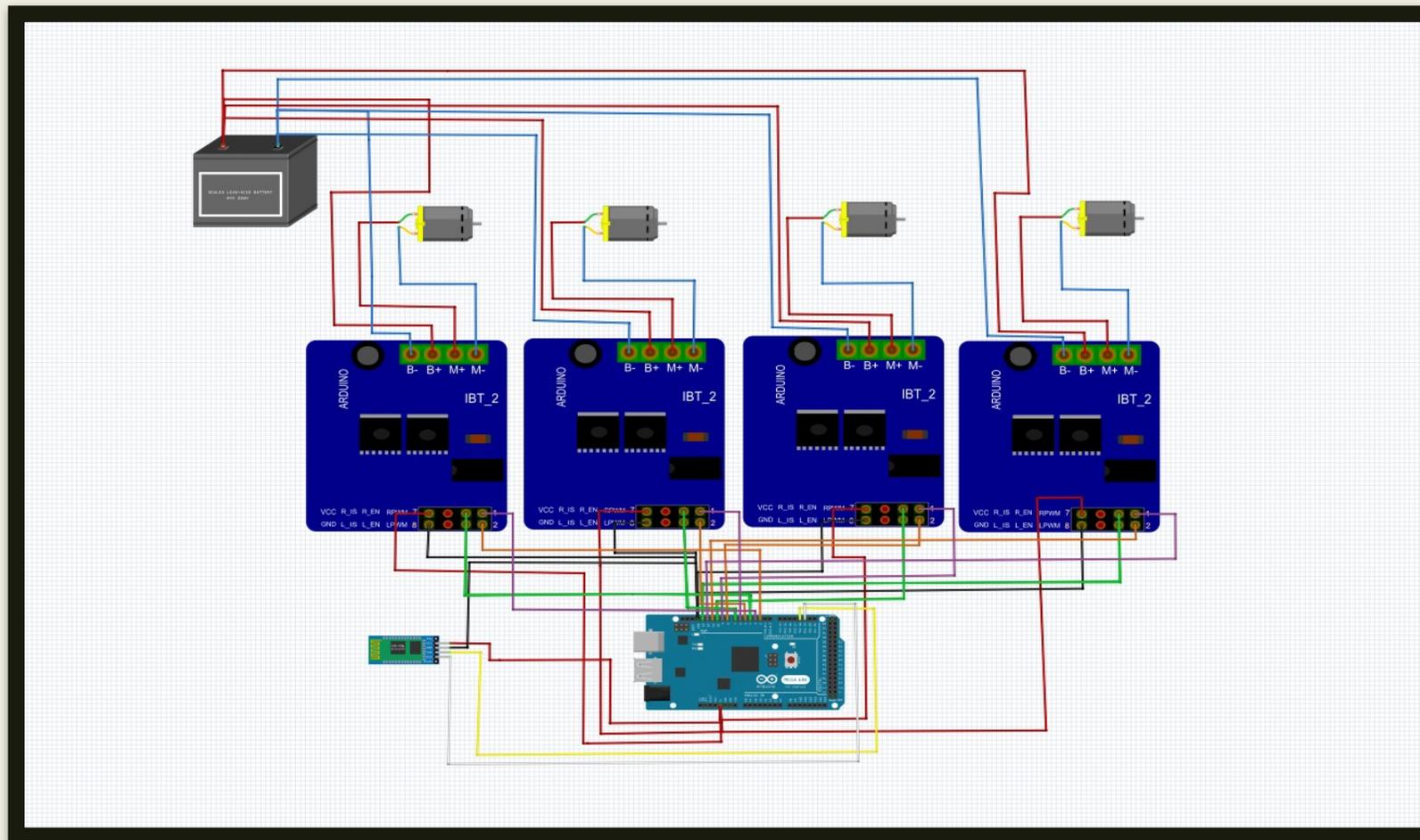
# Аккумулятор



Напряжение: 12В

Сила тока: 6А

# Принципиальная схема



The image features two large, thick black L-shaped brackets. One is positioned in the top-left corner, and the other is in the bottom-right corner. They are oriented towards each other, framing the central text.

ПРОГРАММИРОВАНИЕ

# Движение вперед/назад

```
//////////////////////////////////// Движение вперед //////////////////////////////////////
void forward () {
  ////////////////////////////////////// Мотор Front Right //////////////////////////////////////
  digitalWrite(L_PWM1, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM1, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN1, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

  ////////////////////////////////////// Мотор Rear Right //////////////////////////////////////
  digitalWrite(L_PWM2, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM2, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN2, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

  ////////////////////////////////////// Мотор Rear Left //////////////////////////////////////
  digitalWrite(L_PWM3, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM3, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN3, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

  ////////////////////////////////////// Мотор Front Left //////////////////////////////////////
  digitalWrite(L_PWM4, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM4, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN4, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);
}
}
```

```
//////////////////////////////////// Движение назад //////////////////////////////////////
void back () {
  ////////////////////////////////////// Мотор Front Right //////////////////////////////////////
  digitalWrite(L_PWM1, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM
  digitalWrite(R_PWM1, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
  analogWrite (EN1, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

  digitalWrite(L_PWM2, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM
  digitalWrite(R_PWM2, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
  analogWrite (EN2, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

  ////////////////////////////////////// Мотор Rear Left //////////////////////////////////////
  digitalWrite(L_PWM3, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM
  digitalWrite(R_PWM3, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
  analogWrite (EN3, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

  ////////////////////////////////////// Мотор Front Left //////////////////////////////////////
  digitalWrite(L_PWM4, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM
  digitalWrite(R_PWM4, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
  analogWrite (EN4, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN
}
}
```

# Движение по диагонали

```
//////////////////////////////////// Движение по диагонали назад вправо////////////////////////////////////
void backright (){
  ////////////////////////////////////// Мотор Front Left////////////////////////////////////
  digitalWrite(L_PWM4, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM4, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN4, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

  ////////////////////////////////////// Мотор Rear Right //////////////////////////////////////
  digitalWrite(L_PWM2, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM2, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN2, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);
}

//////////////////////////////////// Движение по диагонали назад влево////////////////////////////////////
void backleft () {
  ////////////////////////////////////// Мотор Front Right////////////////////////////////////
  digitalWrite(L_PWM1, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM
  digitalWrite(R_PWM1, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
  analogWrite (EN1, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

  ////////////////////////////////////// Мотор Rear Left////////////////////////////////////
  digitalWrite(L_PWM3, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM-
  digitalWrite(R_PWM3, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
  analogWrite (EN3, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN
}
}
```

```
//////////////////////////////////// Движение по диагонали вперед влево //////////////////////////////////////
void forwardleft (){
  ////////////////////////////////////// Мотор Front Left////////////////////////////////////
  digitalWrite(L_PWM4, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM4, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN4, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

  ////////////////////////////////////// Мотор Rear Right //////////////////////////////////////
  digitalWrite(L_PWM2, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM2, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN2, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);
}

//////////////////////////////////// Движение по диагонали вперед вправо //////////////////////////////////////
void forwardright(){
  ////////////////////////////////////// Мотор Front Right////////////////////////////////////
  digitalWrite(L_PWM1, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM1, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN1, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

  ////////////////////////////////////// Мотор Rear Left////////////////////////////////////
  digitalWrite(L_PWM3, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
  digitalWrite(R_PWM3, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
  digitalWrite(EN3, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);
}
}
```

# Движение вправо/влево

```
//////////////////// Движение вправо //////////////////////
void right(){
    ////////////////////// Мотор Front Left////////////////////
    digitalWrite(L_PWM4, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM
    digitalWrite(R_PWM4, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN4, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

    ////////////////////// Мотор Rear Left////////////////////
    digitalWrite(L_PWM3, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM3, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN3, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

    ////////////////////// Мотор Front Right////////////////////
    digitalWrite(L_PWM1, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM1, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN1, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

    ////////////////////// Мотор Rear Right //////////////////////
    digitalWrite(L_PWM2, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM
    digitalWrite(R_PWM2, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN2, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN
}
```

```
//////////////////// Движение влево //////////////////////
void left(){
    ////////////////////// Мотор Front Left////////////////////
    digitalWrite(L_PWM4, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM4, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN4, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

    ////////////////////// Мотор Rear Left////////////////////
    digitalWrite(L_PWM3, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM
    digitalWrite(R_PWM3, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN3, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

    ////////////////////// Мотор Front Right////////////////////
    digitalWrite(L_PWM1, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM
    digitalWrite(R_PWM1, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN1, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

    ////////////////////// Мотор Rear Right //////////////////////
    digitalWrite(L_PWM2, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM2, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN2, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);
}
```

# Вращение

```
////////// Вращение вправо //////////  
void superright(){  
    ////////// Мотор Rear Left//////////  
    digitalWrite(L_PWM3, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM  
    digitalWrite(R_PWM3, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM  
    analogWrite (EN3, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN  
  
    ////////// Мотор Front Left//////////  
    digitalWrite(L_PWM4, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM  
    digitalWrite(R_PWM4, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM  
    analogWrite (EN4, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN  
  
    ////////// Мотор Front Right//////////  
    digitalWrite(L_PWM1, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM  
    digitalWrite(R_PWM1, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM  
    digitalWrite(EN1, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);  
  
    ////////// Мотор Rear Right //////////  
    digitalWrite(L_PWM2, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM  
    digitalWrite(R_PWM2, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM  
    digitalWrite(EN2, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);  
}
```

```
////////// Вращение влево //////////  
void superleft(){  
    ////////// Мотор Rear Left//////////  
    digitalWrite(L_PWM3, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM  
    digitalWrite(R_PWM3, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM  
    digitalWrite(EN3, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);  
  
    ////////// Мотор Front Left//////////  
    digitalWrite(L_PWM4, HIGH); // Устанавливаем логическую 1 на входе драйвера L_PWM  
    digitalWrite(R_PWM4, LOW ); // Устанавливаем логический 0 на входе драйвера R_PWM  
    digitalWrite(EN4, HIGH); // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);  
  
    ////////// Мотор Front Right//////////  
    digitalWrite(L_PWM1, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM  
    digitalWrite(R_PWM1, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM  
    analogWrite (EN1, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN  
  
    ////////// Мотор Rear Right //////////  
    digitalWrite(L_PWM2, LOW ); // Устанавливаем логический 0 на входе драйвера L_PWM  
    digitalWrite(R_PWM2, HIGH); // Устанавливаем логическую 1 на входе драйвера R_PWM  
    analogWrite (EN2, 255 ); // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN  
}
```

# Ввод данных

A collection of 12 Scratch-style code blocks arranged in two columns. Each block starts with a 'when' event (button click or slider change) followed by a 'do' action of calling 'BluetoothClient1.SendText' with a specific text value. The text values are: 'F', 'B', 'R', 'U', 'S', 'SL', 'G', 'J', 'H', 'I', 'SR', and 'D'.

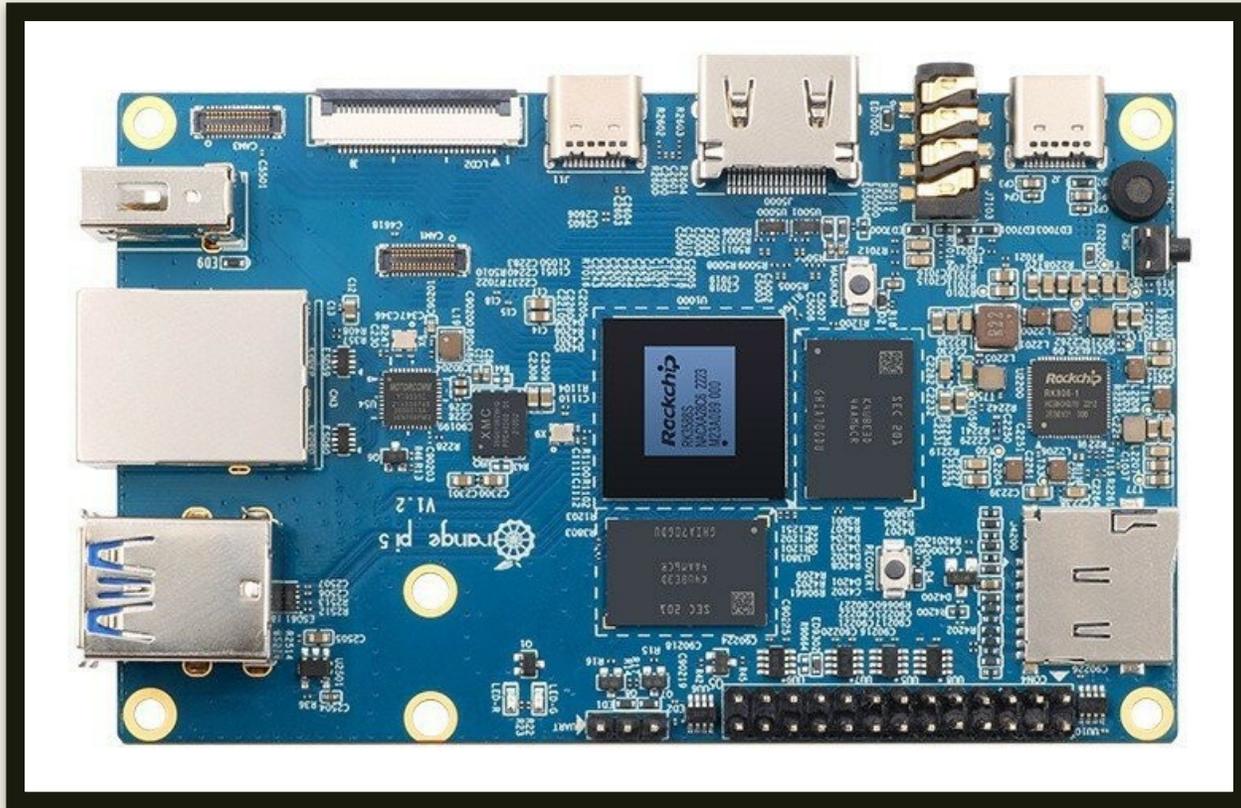
```
void loop() {  
  if (Serial1.available()) {  
    command = Serial1.read();  
    if (command == 'X') {  
      state = 1;  
    } else if (command == 'D') {  
      state = 0;  
    }  
    if (command == 'B') {  
      back();  
    } else if (command == 'F') {  
      forward();  
    } else if (command == 'SR') {  
      superright ();  
    } else if (command == 'SL') {  
      superleft ();  
    } else if (command == 'R') {  
      right ();  
    } else if (command == 'L') {  
      left ();  
    } else if (command == 'G') {  
      forwardleft ();  
    } else if (command == 'I') {  
      forwardright ();  
    } else if (command == 'H') {  
      backright ();  
    } else if (command == 'J') {  
      backleft ();  
    } else if (command == 'S'){  
      stopRobot();  
    }  
  }  
}
```



The image features two large, thick black L-shaped brackets. One is positioned in the top-left corner, and the other is in the bottom-right corner. They are oriented towards each other, framing the central text.

ПЕРСПЕКТИВЫ

# Машинное зрение



Wi-Fi и Bluetooth модули

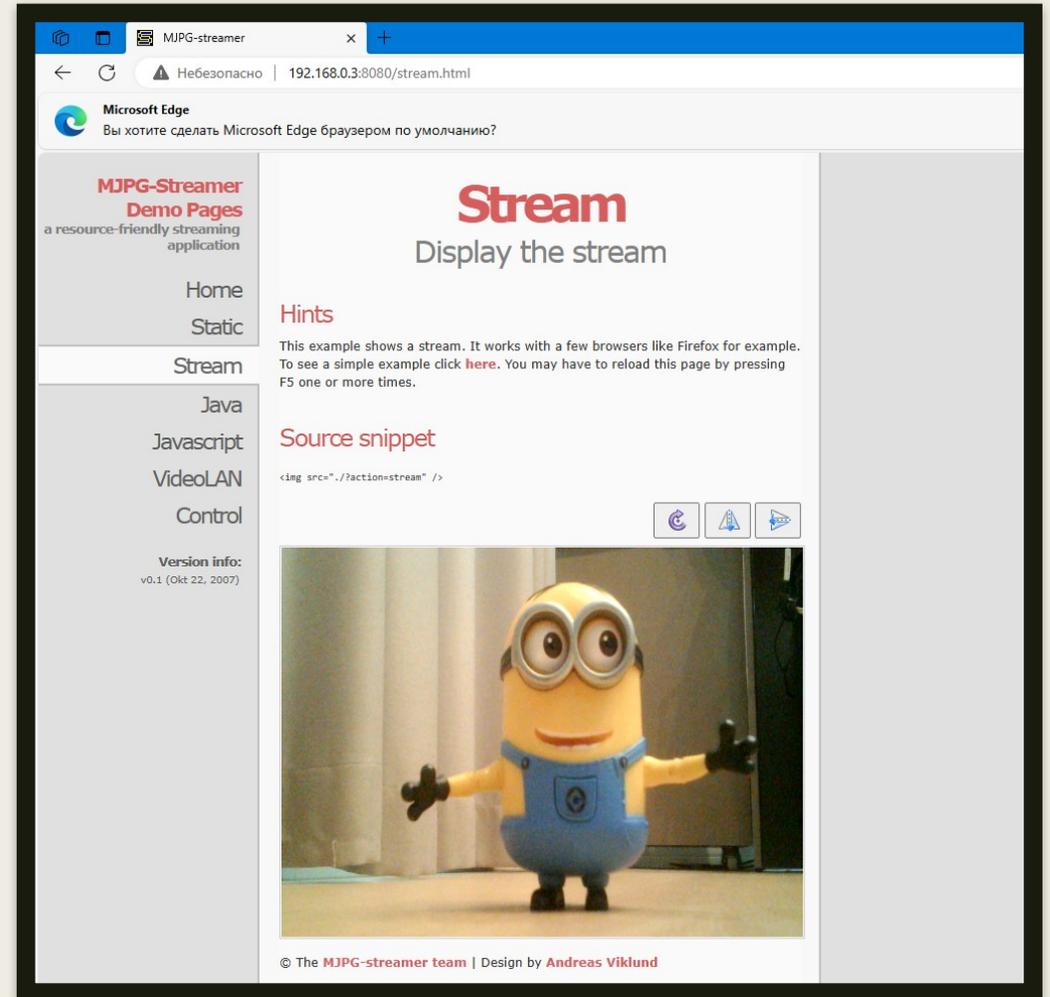
HDMI-вывод

3 USB-порта

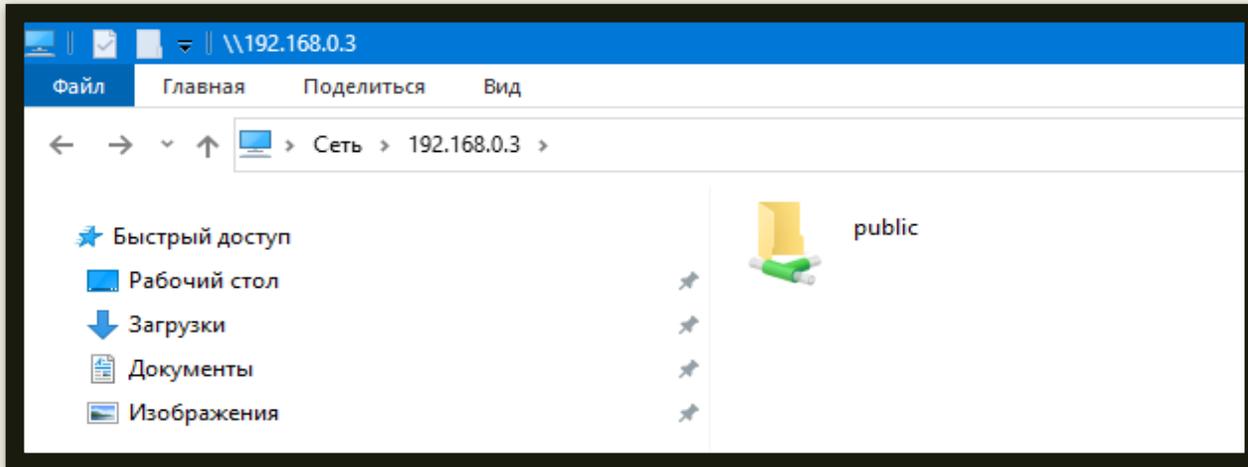
Плата Orange Pi 3 LTS

# Видеотрансляция

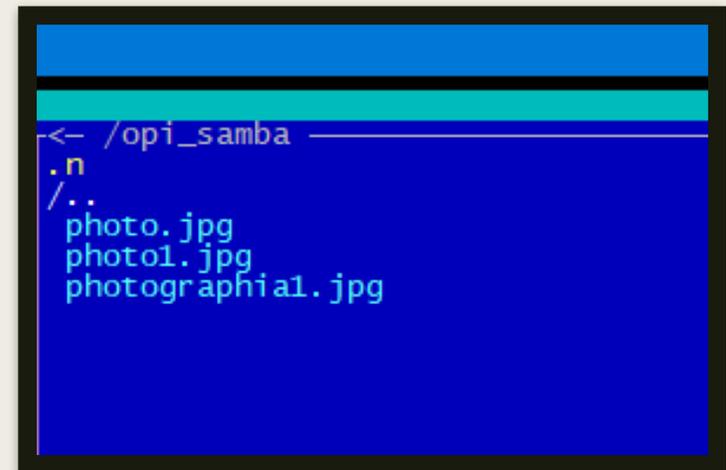
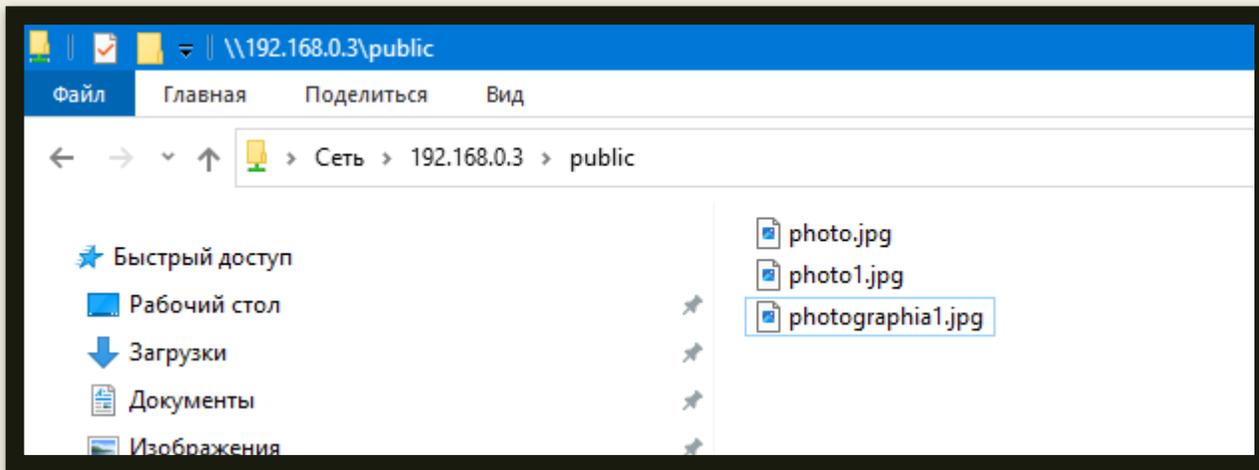
```
mc [root@orangepi3-lts:~/CVbot/python_app/mjpg-streamer/mjpg-streamer-experimental]
Login as: root
root@192.168.0.3's password:
OPi3 LTS
Welcome to Armbian 23.8.3 Bookworm with Linux 6.1.53-current-sunxi64
No end-user support: community creations
System load: 38%      Up time: 0 min
Memory usage: 7% of 2.92G  IP: 192.168.0.3
CPU temp: 49°C      Usage of /: 10% of 29G
RX today: 31.1 MiB
Last login: Sun Oct 15 19:42:47 2023
root@orangepi3-lts:~# mc
root@orangepi3-lts:~/CVbot/python_app/mjpg-streamer/mjpg-streamer-experimental# ./start.sh
root@orangepi3-lts:~# █
```



# Сетевая папка



Путь к папке



Файлы папки на микрокомпьютере и ПК

# СНИМОК

```
← ~/test  
.n  
/..  
.mjpeg2.py.swp  
*camera.py  
camera2.py  
mjpeg2.py  
photo1.jpg  
photographia1.jpg  
*test.py
```

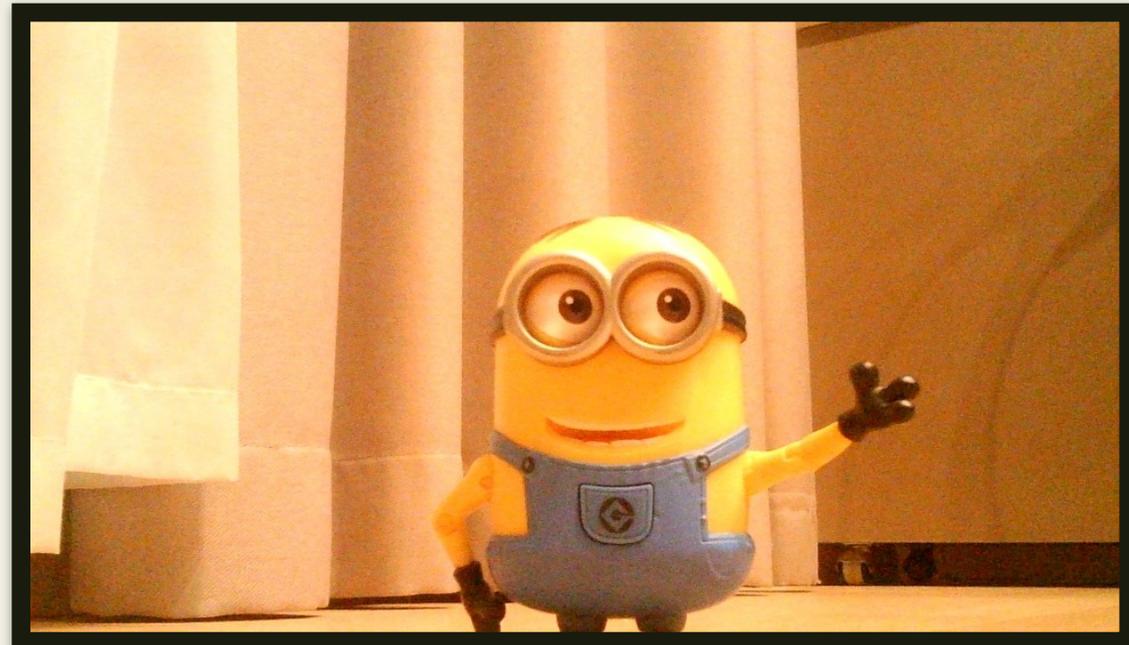
Расположение скрипта и фотографий

```
mc [root@orangepi3-lts]:~/test  
/root/test/test.py [----] 0 L: [ 1+13 14/ 14] *(431 / 431b) <EOF>  
#!/root/flask-app/venv/bin/python  
import cv2  
import numpy  
def frame_generation():  
<----->cap = cv2.VideoCapture(1) #ignore the errors  
#<----->if cap.isOpened():  
<----->cap.set(3,960) #Set the width important because the default will timeout  
<----->cap.set(4,544) #ignore the error or false response #Set the height ignore the errors  
<----->r, frame = cap.read()  
<----->cv2.imwrite("photographia1.jpg", frame)  
frame_generation()
```

Скрипт

```
← /opi_samba  
.n  
/..  
photo.jpg  
photo1.jpg  
photographia1.jpg
```

Сетевая папка



СНИМОК

The image features two large, black, L-shaped corner brackets. One is positioned in the top-left corner, and the other is in the bottom-right corner. They are composed of thick, solid black lines that meet at a 90-degree angle.

**СПАСИБО ЗА  
ВНИМАНИЕ**