Проект на тему Многофункциональный SCARA Манипулятор

Выполнил ученик 8А класса
МБОУ Гимназии №11
Шкуринский Михаил
2022-2023 г.

Цели и задачи

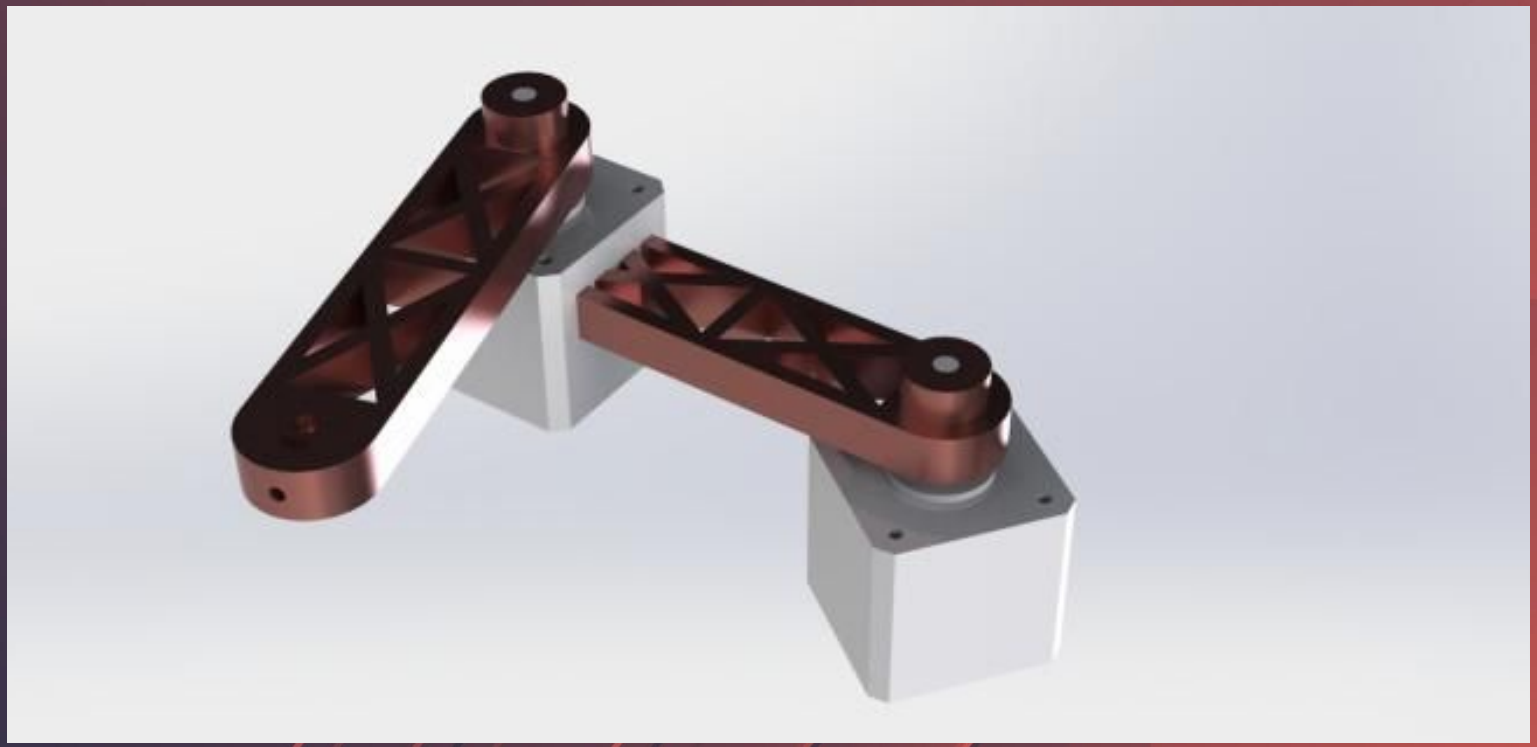
Цель:
Собрать рабочий прототип манипулятора
Задачи:

1. Определится с механикой манипулятора
2. Спроектировать и собрать модель
3. Запрограммировать
4. Испытать модель
5. Выявить недостатки и цели на будущее
6. Сделать вывод

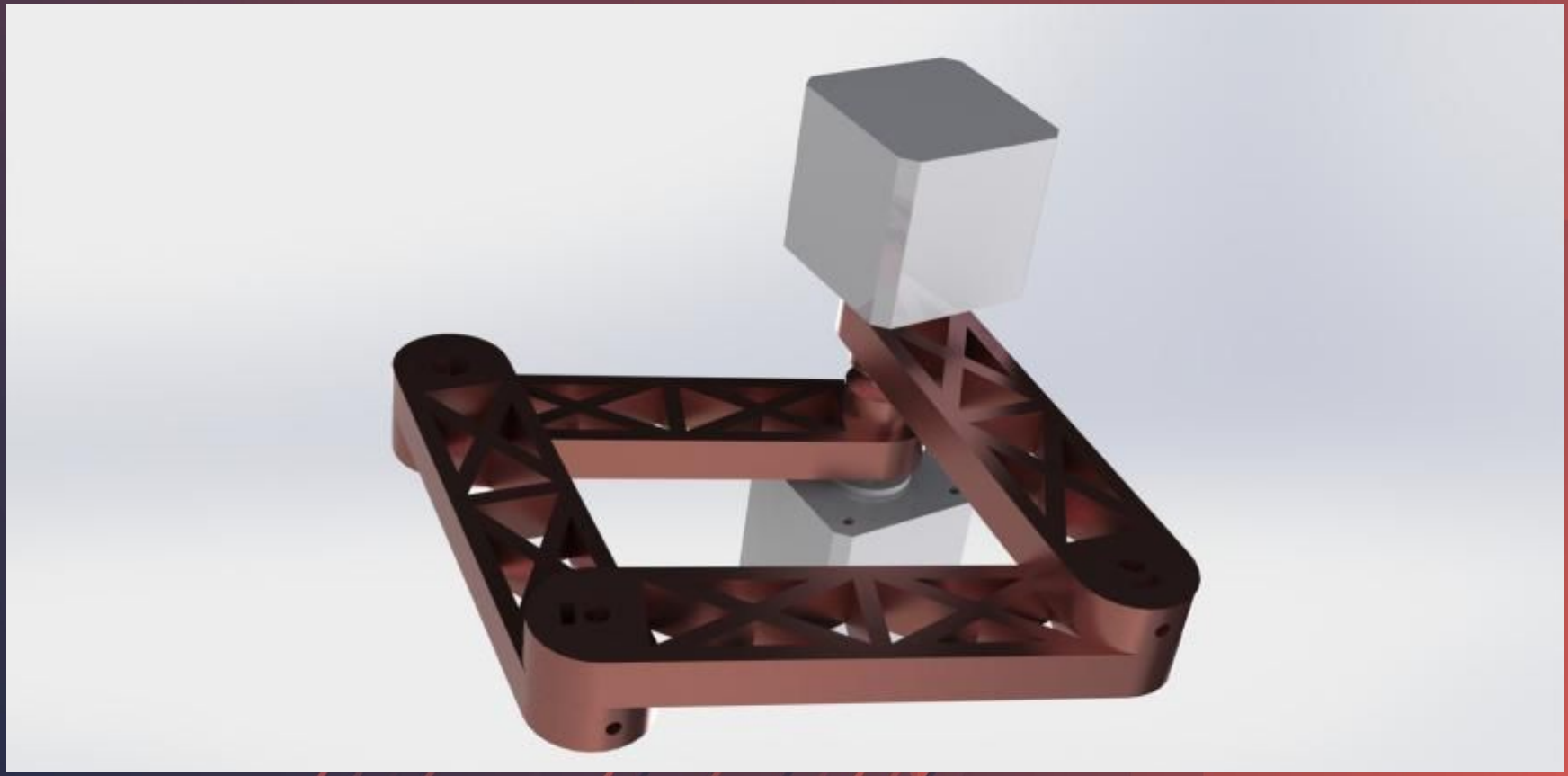
Механика манипулятора

Механику я выбирал по нескольким критериям:

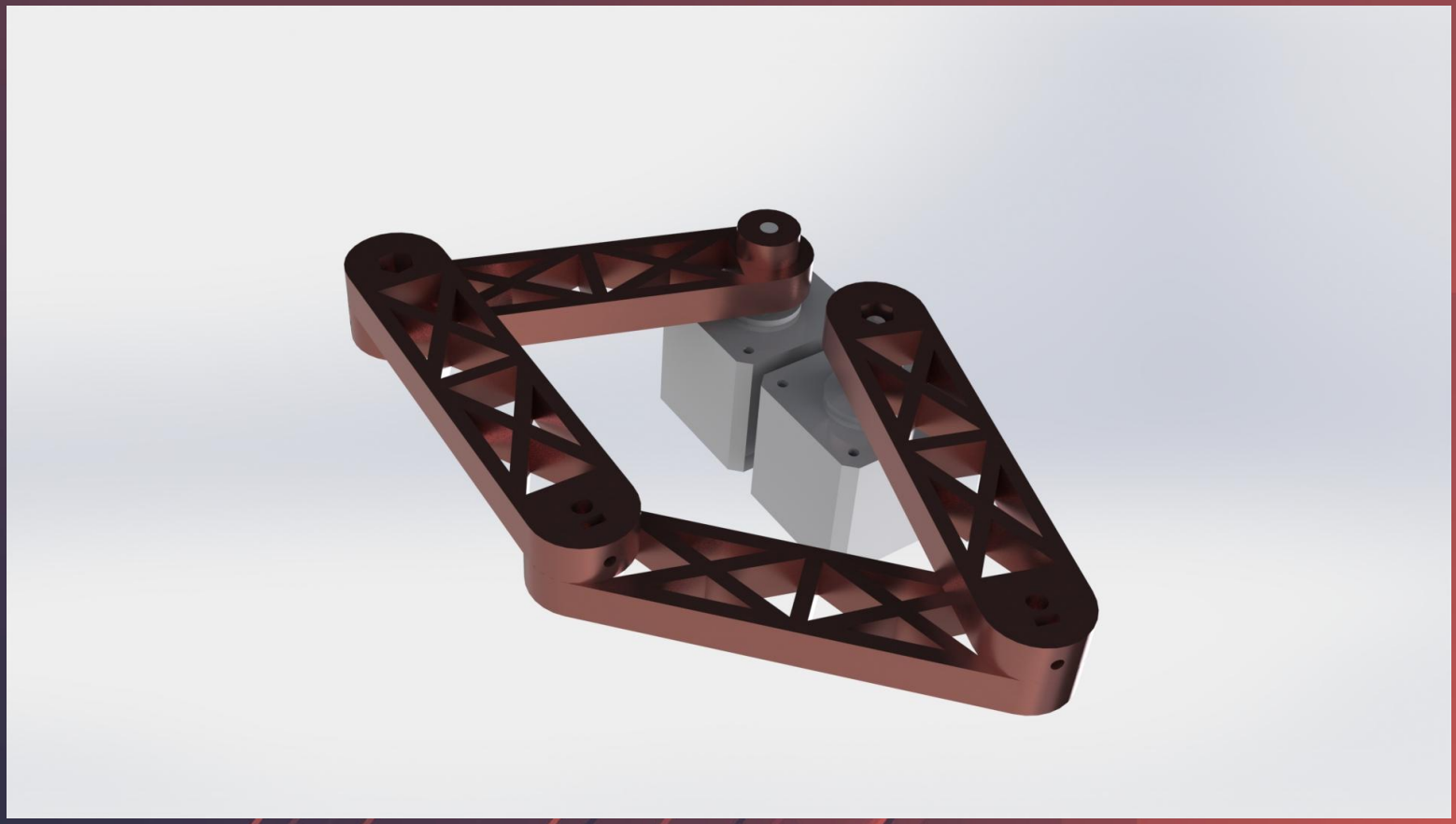
1. Легкость сборки
2. Удобство расчётов углов по координатам
3. Без использования ремней
4. Прочность



1) Сам SCARA-робот, оригинальное исполнение. Один шаговый мотор приводит в движение внутренний рычаг, на котором установлен второй шаговик для привода наружного рычага. Промышленные SCARA-роботы используют именно этот вариант. Скорость перемещения у них невероятно быстрая. Для домашнего манипулятора использовать такой вариант проблематично, так как жесткости печатных деталей будет недостаточно.



2) Это уже Morgan SCARA. Отличия от оригинального SCARA : шаговые моторы установлены стационарно, что значительно облегчает конструкцию из рычагов. Именно этот вариант в итоге я и выбрал.



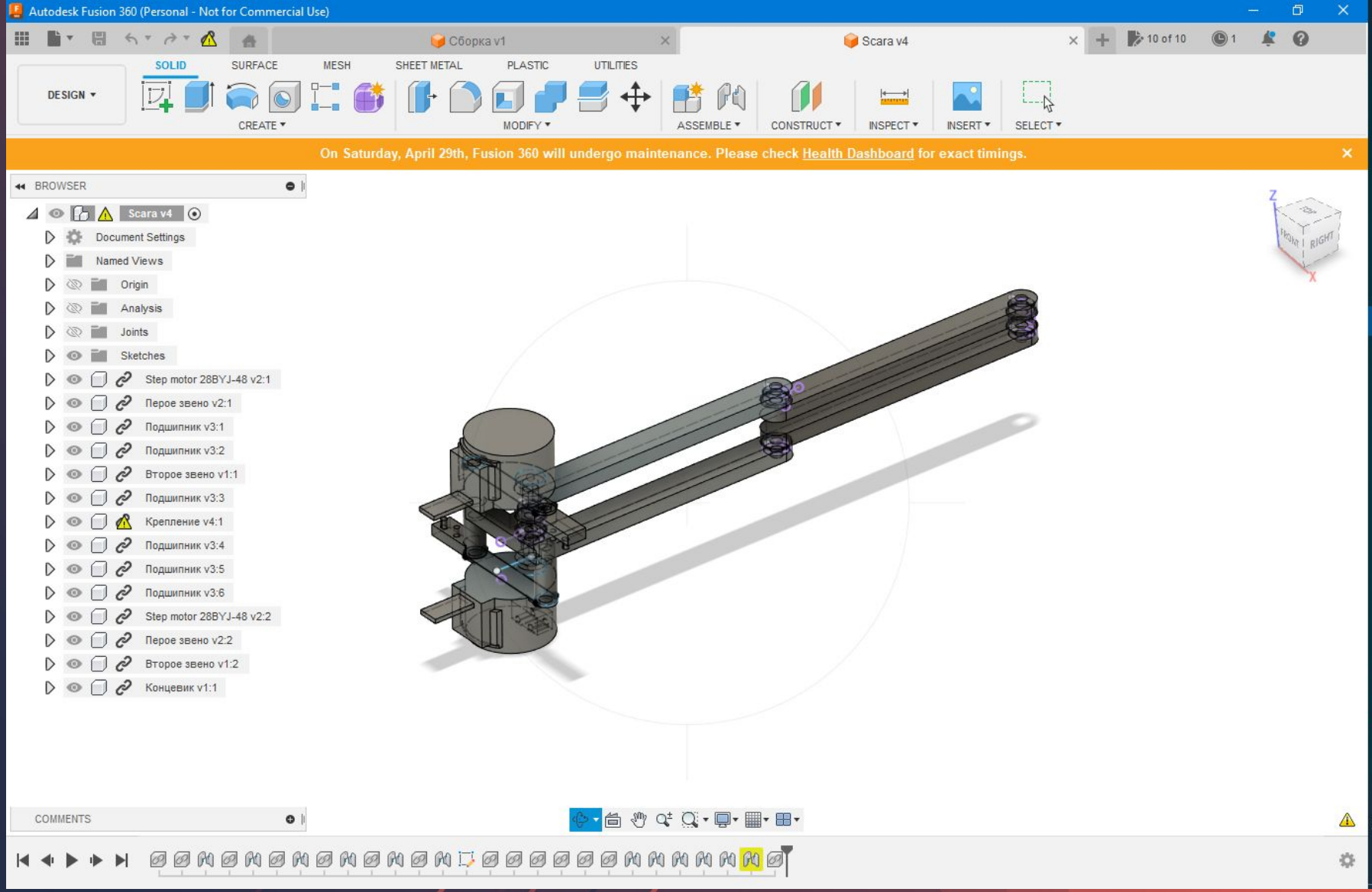
3) Parallel Scara .В отличии от Morgan Scara внутренние рычаги находятся не на одной оси вращения. На такой механике продаётся 3д принтер SkyOne.

Комплетующие для сборки

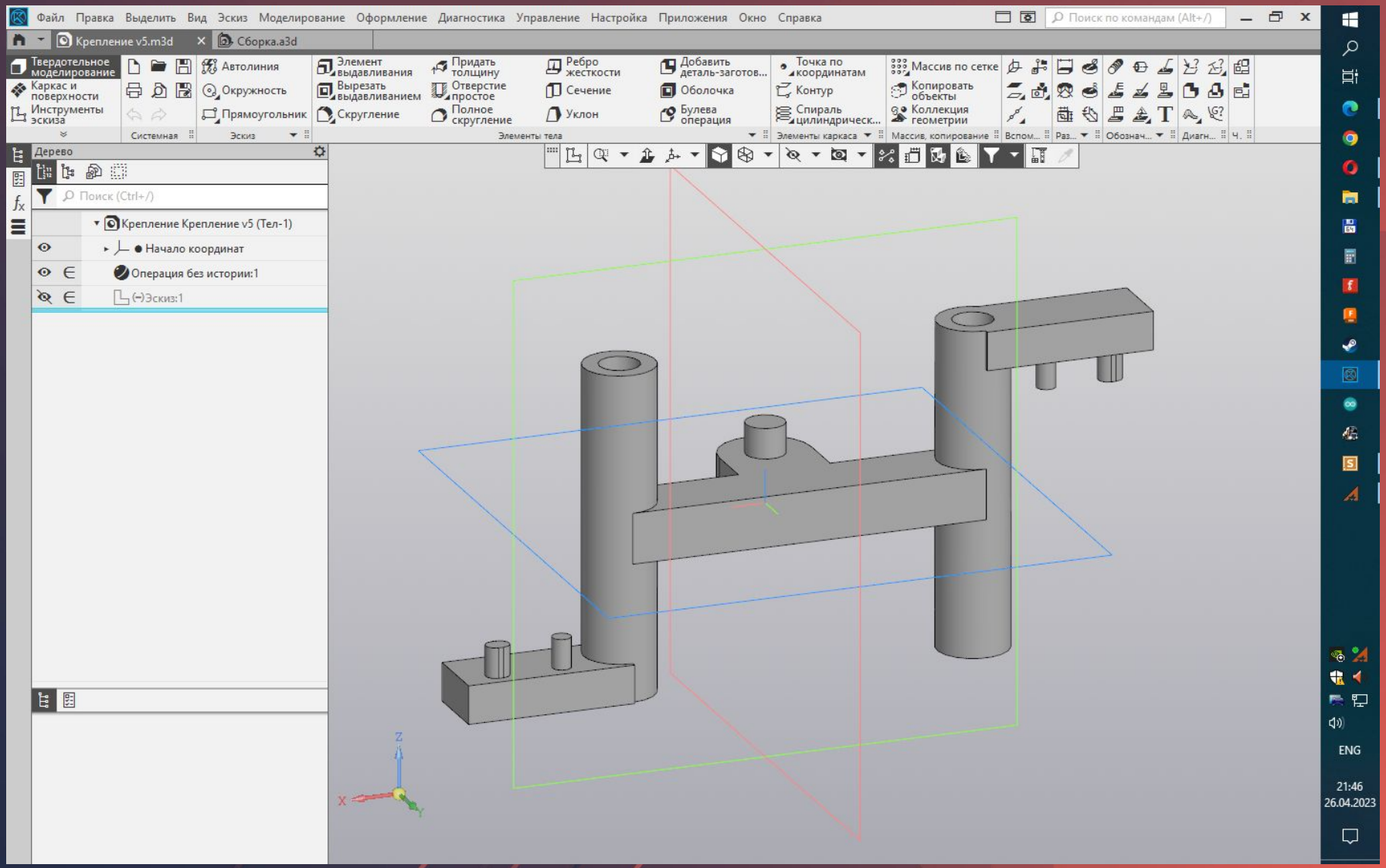
1. Плата Arduino Mega Pro
2. Два шаговых мотора 28BYJ-48 с драйвером ULN2003
3. Модуль SD-reader для Arduino
4. OLED дисплей 0.91″ 128×32 точки
5. Сервопривод
6. Неодимовые магниты
7. Подшипники
8. Концевики
9. Детали напечатанные на 3д принтере
10. Соединительные провода

Программы

- КОМПАС 3D и Fusion 360 для создания деталей.
- Arduino IDE для программирования
- Fritzing для создания схемы подключения компонентов



Сборка из деталей в Fusion 360



Деталь для крепления двух половинок в КОМПАС 3D

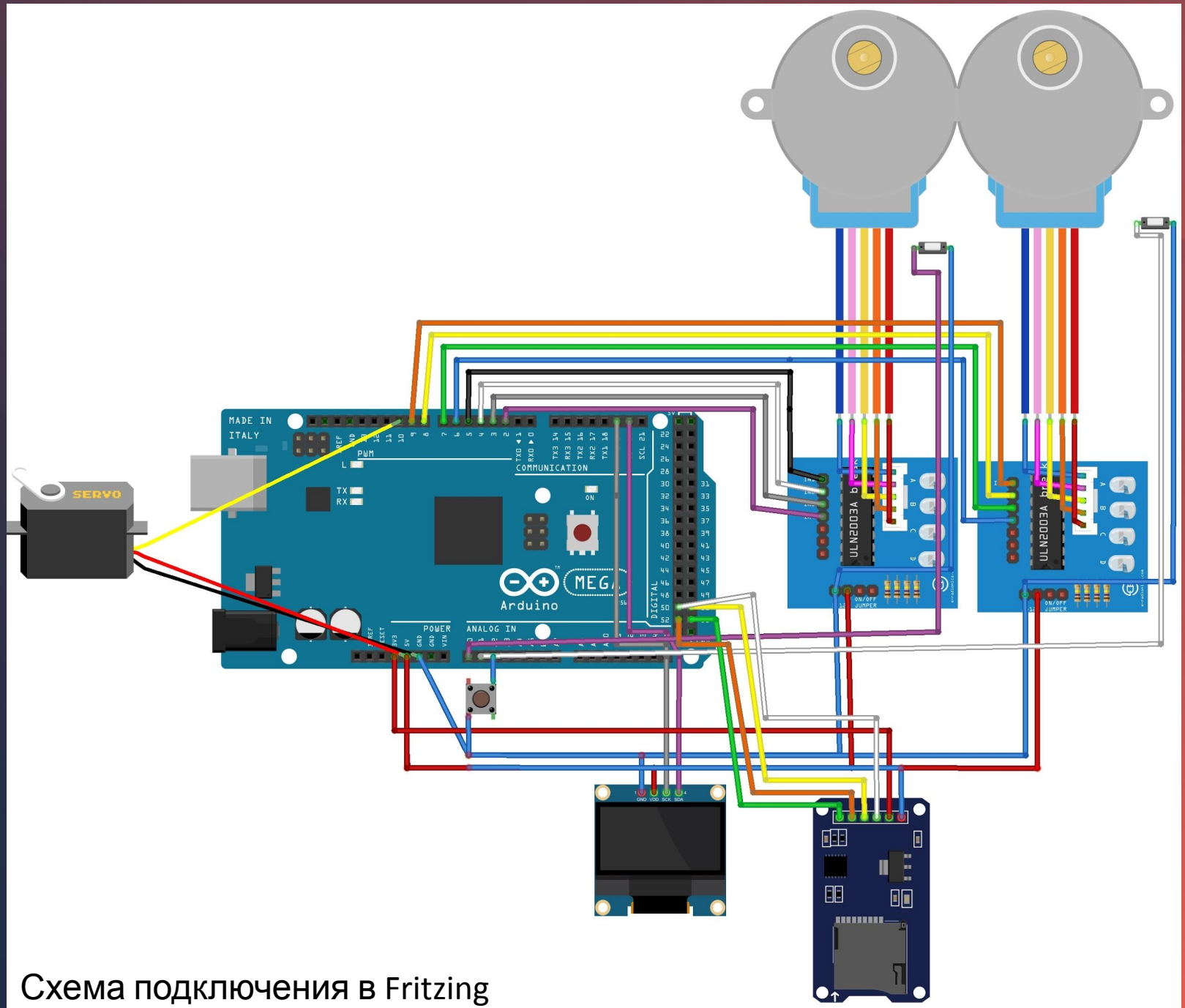
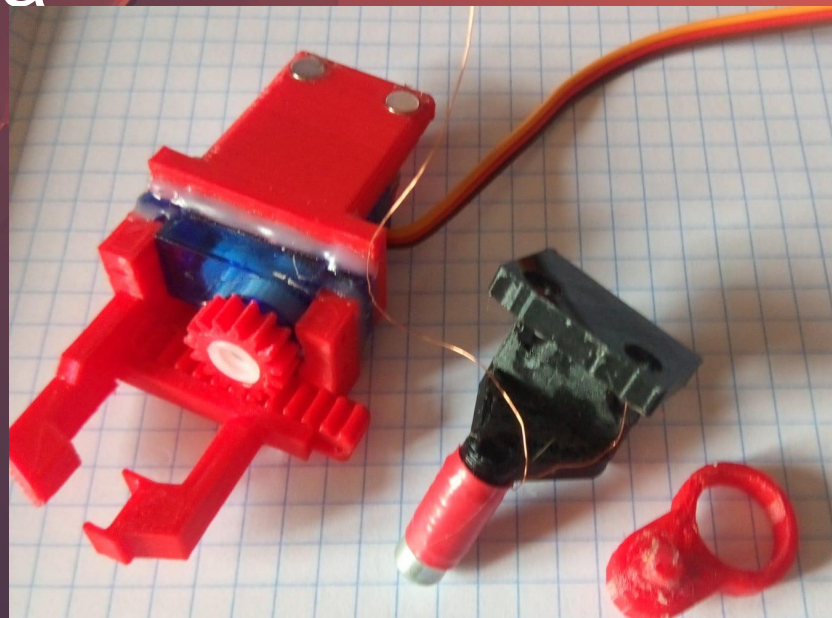


Схема подключения в Fritzing

Многофункциональность

На конце звеньев распологается магнитный захват, благодаря ему можно быстро менять необходимые модули:

1. Модуль горизонтального захвата
2. Модуль вертикального захвата
3. Модуль электромагнита
4. Модуль для ручки



Программирование

Программировал я довольно много, начал с маленьких функций , а потом с каждой новой программой добавлял новые и улучшал старые. В данный момент у меня около 10 различных по функциям и сложности программ. Самая последняя версия может считывать координаты и угол захвата с sd карты, приблизительно двигаться по этим координатам, парковаться по нажатию кнопки и выводить немного информации на экран.

mega3g | Arduino 1.8.19

Файл Правка Скетч Инструменты Помощь

mega3g | OLED_update | SD_read | SD_read2 | xy_degrees

```
1
2  /* ПИНЫ ПОДКЛЮЧЕНИЯ */
3  #define LIMSW_L A1    //пин левого концевика
4  #define LIMSW_R A0    //пин правого концевика
5  #define BTN_HOME A2   //пин кнопки home
6  #define chipSelect 53 //пин sd карты
7  #define SERVO_PIN 10
8  #define SERVO 1
9
10 /*ПЕРЕМЕННЫЕ ДЛЯ ВЫЧИСЛЕНИЙ*/
11 bool btn; //переменная для значений home
12 bool home = 0; //переменная парковки
13 int left_position;   //позиция левого мотора
14 int right_position;  //позиция правого мотора
15 // переменные углов
16 float a;
17 float b;
18 float BDH;
19 float d;
20
21 byte l = 95; //длинна рычагов
22 int i = 0; // переменная для посчёта строк
23 int positions[7][2]; //массив для координат
24 char* names[7][2]; //массив для координат с карты
25 int gripper[7][2]; //массив для координат
26 char* names2[7][2]; //массив для координат с карты
27 int h = 5; //количетво строк массива считая от 0
28 int x = 0; //начальный x
29 int y = 0; //начальный y
30 bool motor ;
```

```
/*ПОДКЛЮЧАЕМ БИБЛИОТЕКИ*/
#include <Servo.h>
#include <SPI.h>
#include <SD.h>
#include <EncButton.h>
#include "GyverStepper2.h"
#include <GyverOLED.h>
#include <GyverWDT.h>

Servo myservo;  //создаём серву
File myFile; //файл с карты
EncButton<EB_TICK, BTN_HOME > btnl(INPUT_PULLUP);//создаём кнопку
GStepper2< STEPPER4WIRE> stepper_left(2038, 2, 4, 3, 5);//создаём левый мотор
GStepper2< STEPPER4WIRE> stepper_right(2038, 6, 8, 7, 9);//создаём правый мотор
GyverOLED<SSD1306_128x32, OLED_BUFFER> oled;//создаём дисплей


void setup() {
  Serial.begin(9600);
  Serial.println("left_m,right_m");
  if (SERVO == 1) {
    myservo.attach(SERVO_PIN);
  }

  Watchdog.enable(RESET_MODE, WDT_PRESCALER_512); // Режим сторжевого сброса , таймаут ~4с
  pinMode(LIMSW_L, INPUT_PULLUP);
  pinMode(LIMSW_R, INPUT_PULLUP);

  oled.init();        // инициализация
  oled.clear();       // очистка
```

```
65  if (!SD.begin(chipSelect)) {
66    //  Serial.println("initialization failed!");
67    oled.home();
68    oled.setScale(4);
69    oled.print("ERROR");
70    oled.update();
71    while (1);
72  }
73  // Serial.println("initialization done.");
74  oled.home();
75  oled.setScale(4);
76  oled.print("SD_OK");
77  oled.update();
78  delay(500);
79  oled.clear();
80  config_pSD();//чтение координат с карты
81  config_gSD();
82  Watchdog.reset(); // Переодический сброс watchdog, означающий, что устройство не зависло
83  Watchdog.disable();
84  oled.setScale(1);   // масштаб текста (1..4)
85  oled.home();        // курсор в 0,0
86  oled.print("Left:");
87  oled.setCursor(0, 1);
88  oled.print("Right:");
89  oled.update();
90  homing_left();//паркуем левый мотор
91
92  oled.setCursor(29, 0);
93  oled.print("home");
94  oled.update();
95  homing_right(); //паркуем правый мотор
```

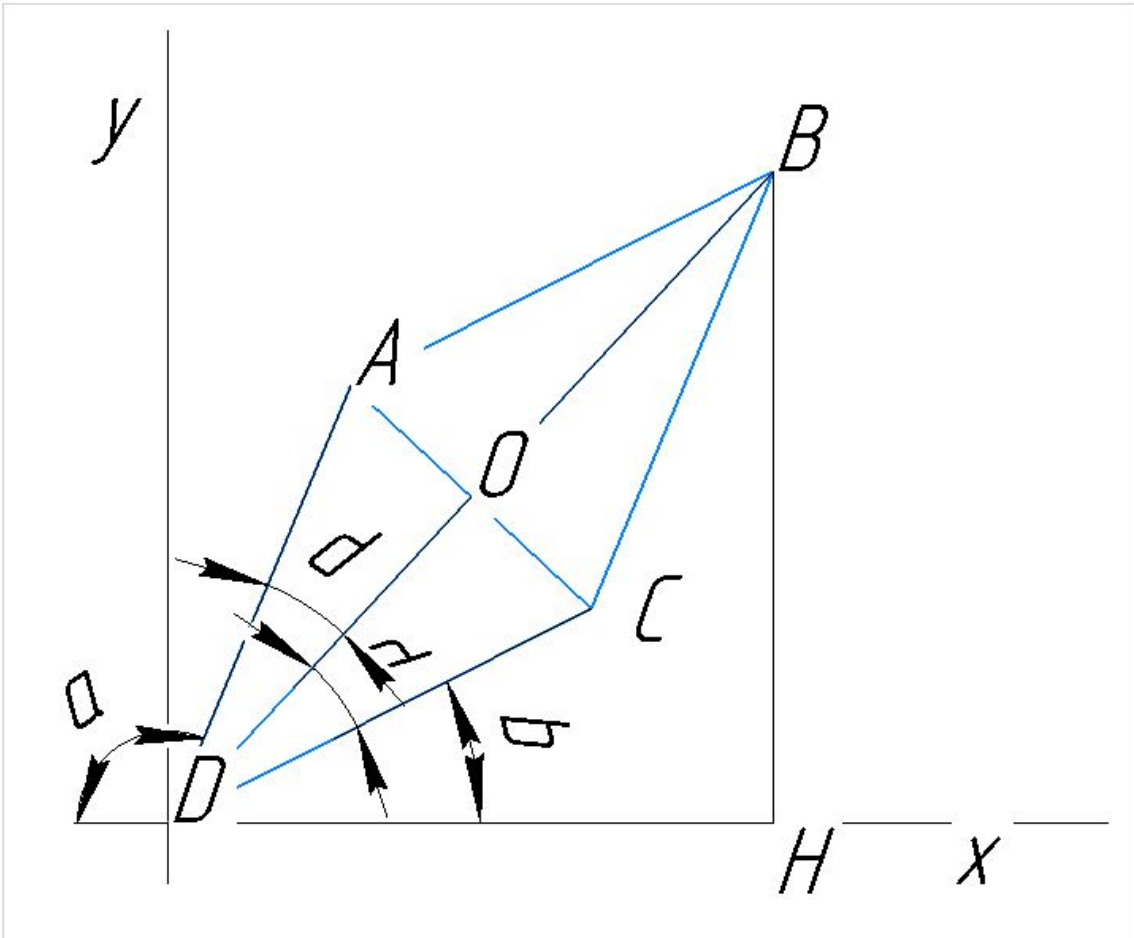
```
  oled.setCursor(39, 1);
  oled.print("home");
  oled.update();
  //получаем первые координаты из массива
  x = positions[i][0];
  y = positions[i][1];
  if (SERVO == 1) {
    myservo.write(gripper[i][1]);
  }
  //находим первые углы по координатам
  degrees2();
  stepper_left.setAcceleration(500);
  stepper_left.setMaxSpeed(400);
  stepper_right.setAcceleration(500);
  stepper_right.setMaxSpeed(400);


  //даём моторам нужные координаты
  stepper_left.setTargetDeg(a);
  stepper_right.setTargetDeg(b);
  Watchdog.enable(RESET_MODE, WDT_PRESCALER_512); // Режим сторжевого сброса , таймаут ~4с

}

```

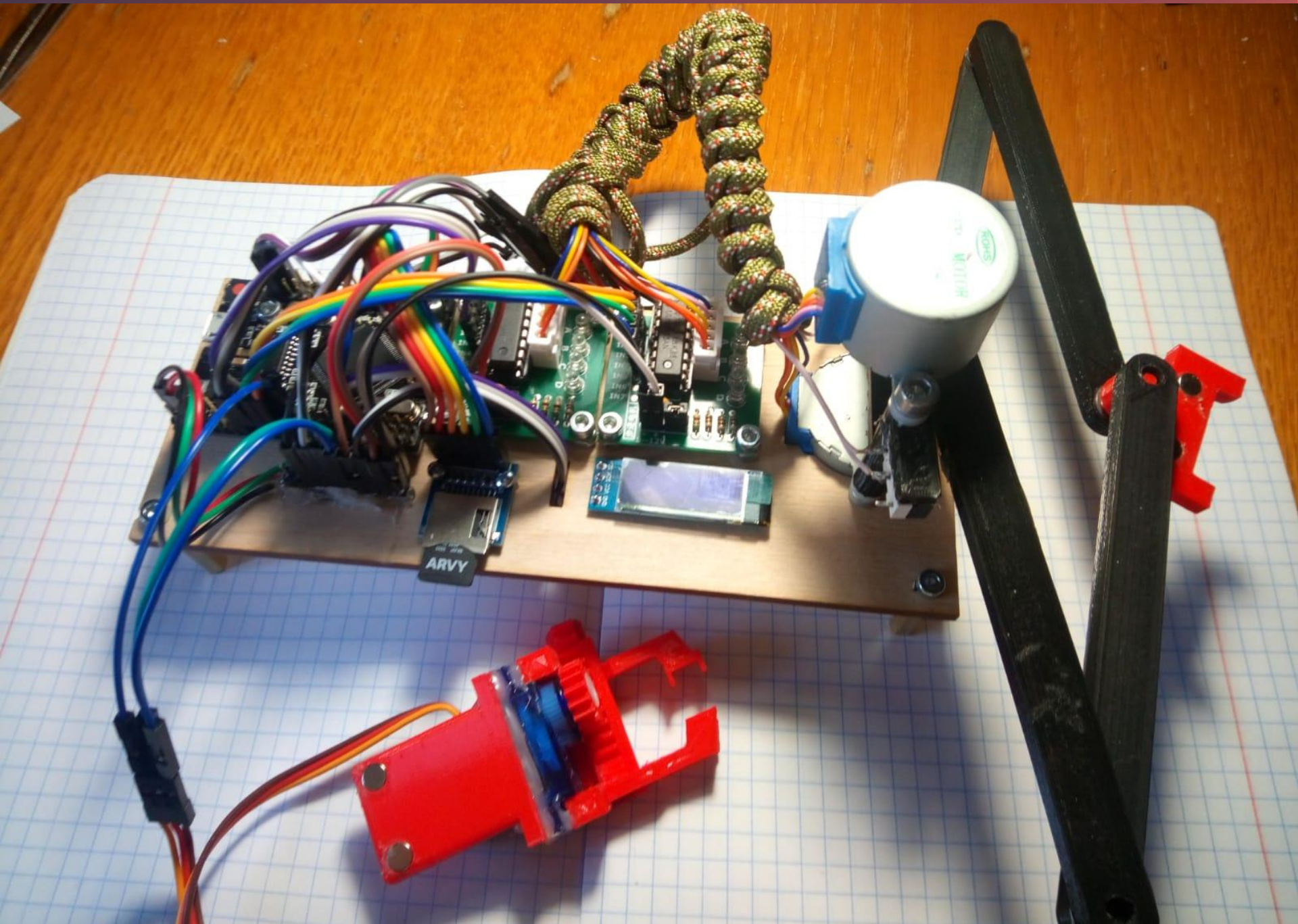
```
120 void loop() {
121   //опрос кнопки и моторов
122   btn1.tick();
123   stepper_left.tick();
124   stepper_right.tick();
125   stepper_left.setAcceleration(500);
126   stepper_left.setMaxSpeed(400);
127   stepper_right.setAcceleration(500);
128   stepper_right.setMaxSpeed(400);
129   dispUpd();//обновляем дисплей
130
131   //val = analogRead(POT_PIN);
132   // val = map(val, 0, 1023, 0, 180);
133   // myservo.write(val);
134
135   if (home == 0) {
136     //если оба мотора дошли до конечной позиции
137     if (stepper_left.ready() == true && stepper_right.ready() == true && i < h) {
138       delay(1000);
139       i++;//прибавляем 1 к i,
140       x = positions[i][0]; //следующий x из строки i столбца 0
141       y = positions[i][1]; //следующий y из строки i столбца 1
142       if (SERVO == 1) {
143         myservo.write(gripper[i][1]);
144       }
145
146       degrees2();//высчитываем углы
147       stepper_left.setTargetDeg(a);//цель левого мотора угол a
148       stepper_right.setTargetDeg(b);//цель правого мотора угол b
149     }
150   }
```

```
  if (btnl.click()) home = !home; //если нажали на кнопку HOME ,то инвертируем значение home

  //если home=1, то идём в 0,0
  if (home == 1) {
    stepper_left.brake();//останавливаем левый мотор
    stepper_right.brake();//останавливаем правый мотор
    stepper_left.setTargetDeg(0);//левый мотор в 0
    stepper_right.setTargetDeg(0);//правый мотор в 0
  }
  Watchdog.reset(); // Переодический сброс watchdog, означающий, что устройство не зависло
}
//парковка левого мотора
void homing_left() {
  if (digitalRead(LIMSW_L)) {       // если концевик X не нажат
    stepper_left.setSpeed(-10);       // ось X, -10 шаг/сек
    while (digitalRead(LIMSW_L)) {  // пока кнопка не нажата
      stepper_left.tick();              // крутим
    }
    // кнопка нажалась - покидаем цикл
    stepper_left.brake();               // тормозим, приехали
  }
  stepper_left.reset();    // сбрасываем координаты в 0

}
//парковка правого мотора
void homing_right() {
  if (digitalRead(LIMSW_R)) {       // если концевик X не нажат
    stepper_right.setSpeed(-10);       // ось X, -10 шаг/сек
    while (digitalRead(LIMSW_R)) {  // пока кнопка не нажата
      stepper_right.tick();              // крутим
    }
    // кнопка нажалась - покидаем цикл
    stepper_right.brake();               // тормозим, приехали
  }
  stepper_right.reset();    // сбрасываем координаты в 0

```

Программа для расчёта углов из координат

```
void degrees2() {  //находим углы по координатам
if (x == 0) {//если x=0, то оба угла будут равны
    float DO = y / 2;
    float OC = sqrtf(sq(l) - sq(DO));
    d = atan((float)OC / DO);
    d = degrees(d);
    b = 90 - d;
    a = b;
    a = round(a);
    b = round(b);
  } else {
    long z;
    if (x > 0) {
      z = x;
    } else {
      z = abs(x);
    }
    BDH = atan((float)y / z);
    BDH = degrees(BDH);
    float DO = hypot(z, y) / 2;
    float OC = sqrtf(sq(l) - sq(DO));
    d = atan((float)OC / DO);
    d = degrees(d);

    if (x > 0) {
      b = BDH - d;
      a = 180 - d - d - b;
    } else if (x < 0) {
      a = BDH - d;
      b = 180 - d - d - a;
    }
  }
  a = round(a);
  b = round(b);
}
```




ARVY

Плюсы и минусы

Плюсы:

1. Получилось собрать рабочую модель
2. Есть возможность улучшений
3. Красивый внешний вид

Минусы:

1. Конструкция оказалась непрочной
2. Иногда программа зависает

Вывод

- Я создал рабочий прототип манипулятора
- Научился программировать сложные проекты и составлять схемы
- Решил поставленные задачи
- Понял что надо улучшить
- Создал новые цели улучшения

Спасибо за внимание!