НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

МИЭМ НИУ ВШЭ

Разработка программно-аппаратного комплекса для удаленного доступа к отладочным платам Arduino

Руководитель направления
к.т.н., доцент Романов Александр Юрьевич

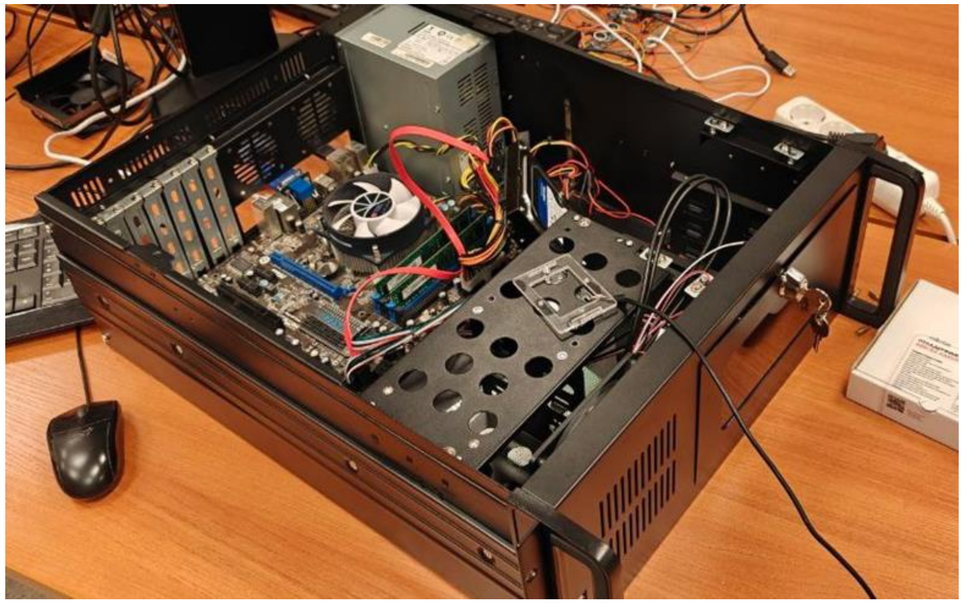
Руководитель НУГ:
к.т.н., доцент Американов Александр Александрович

Команда:
Апьюк Валерия Романовна (группа БИВ214)
Воркова Василиса Сергеевна (группа БИВ214)
Мещеряков Никита Константинович (группа МИВ231)
Падалица Кирилл Андреевич (группа МКС232)
Гопка Лидия Вячеславовна (БИТ222)

Москва, 2024

ЦЕЛИ И ЗАДАЧИ

Цель:
Разработка реконфигурируемого программно-аппаратного комплекса, обеспечивающего удаленный доступ к оборудованию лаборатории УЛ САПР МИЭМ НИУ ВШЭ учащимся и сотрудникам НИУ ВШЭ, а также внешним лицам;



Задачи:
- Перенести разработки, уже созданные на основе предыдущих научно-исследовательских проектов, на новую базу удаленных стендов;
- Продумать и реализовать аппаратную часть стенда, где будет возможность задавать и менять схему подключения элементов к отладочной плате Arduino удаленно, без какого-либо физического вмешательства;
- Расширить набор подключаемых датчиков;
- Разработать веб-приложение, которое позволит дистанционно взаимодействовать со стендом: прошивать платы и выводить изображения с камеры.

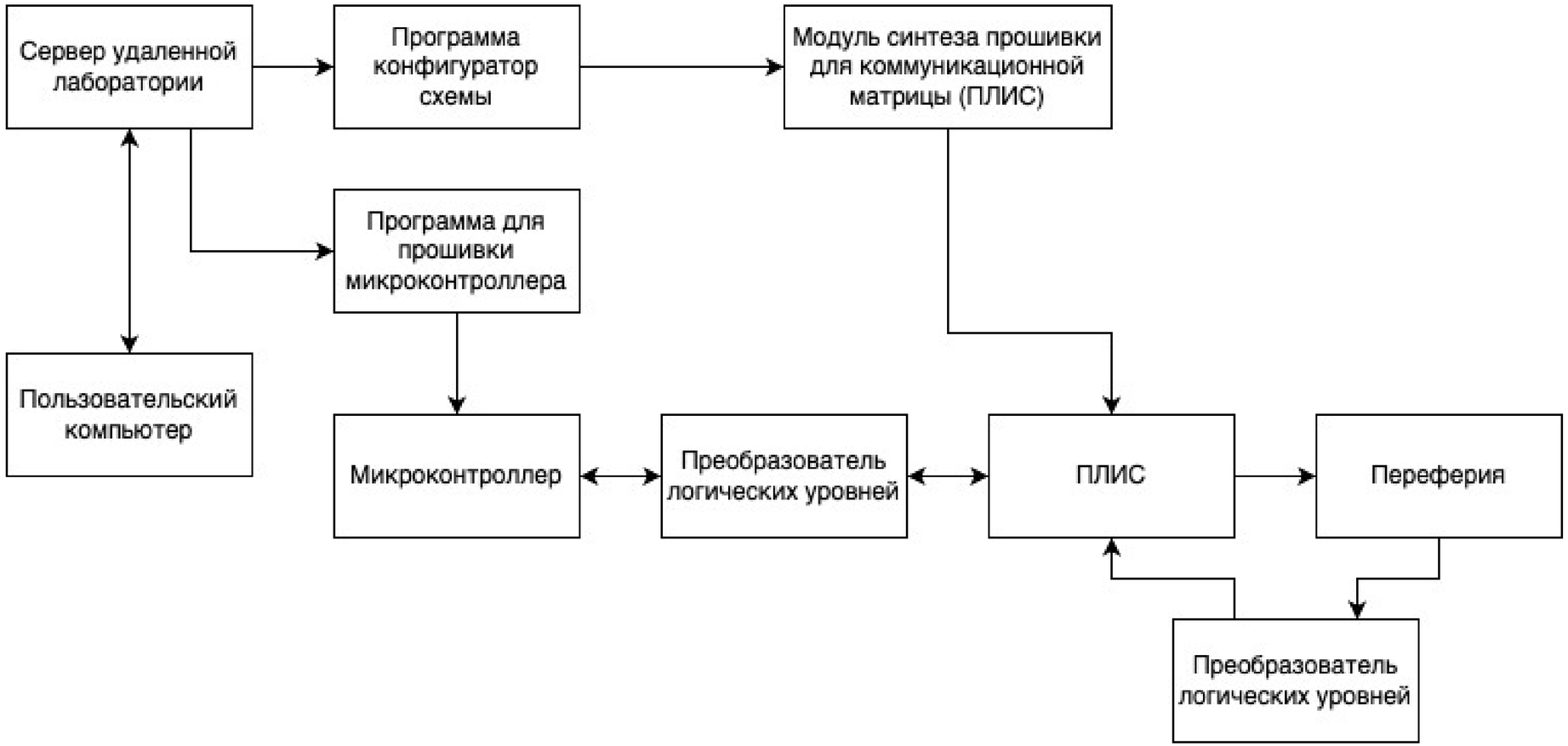
Заказчик:
Лаборатория САПР МИЭМ, реализуется в рамках НУГ.

ЭТАПЫ

1. Изучение теоретических вопросов.
2. Постановка ТЗ, составление сметы, закупка комплектующих для конструирования аппаратного комплекса.
3. Проектирование архитектуры для оптимального взаимодействия клиентского приложения с серверной и аппаратной частями.
4. Разработка клиентского приложения на базе Telegram Apps Mini.
5. Проектирование структуры серверной ячейки, где будет располагаться все необходимое для работы оборудование;
6. Подготовка серверной ячейки (стенда) к подключению аппаратной части стенда;
7. Разработка аппаратной части проекта: установка основных комплектующих (блок питания, материнская плата и тд.) на каркасы; реализация прототипа единой платы, где соединены все необходимые элементы (датчики и светодиоды); соединение получившегося прототипа платы с платой ПЛИС и платой Arduino; демонтаж серверного шкафа; подготовка периферии в серверном шкафу; загрузка ОС и приложений для выполнения проектов.
8. Проверка работоспособности сконструированной аппаратной части совместно с веб-приложением и серверной частью.
9. Тестирование разработанного аппаратно-программного комплекса.
10. Составление и оформление документации проекта.

ПРИНЦИП РАБОТЫ

Схема аппаратной реализации стенда:



ПРИНЦИП РАБОТЫ

Основные компоненты стенда:

1. Компьютер, расположенный в серверной ячейке, к которому подключено все необходимое оборудование.
2. Плата ПЛИС, которая выступает в роли коммутатора между платой Arduino, платой со всеми элементами и компьютером в ячейке.
3. Плата Arduino. В нашем случае Arduino Mega 2520.
4. Макетная плата с различными индикаторами и датчиками.

Архитектура программно-аппаратного комплекса:

СХЕМОТЕХНИКА

Распиновка:

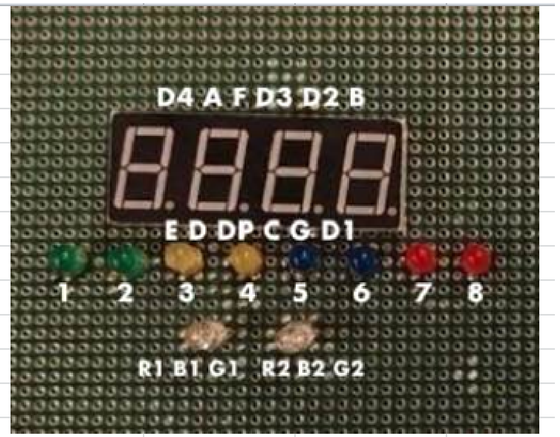
Семисегментник	Cyclone IV IN	Cyclone IV OUT
D1	PIN_D4	PIN_N6
A	PIN_J1	PIN_R8
F	PIN_K2	PIN_T9
D2	PIN_L2	PIN_T8
D3	PIN_K1	PIN_T10
B	PIN_L1	PIN_R9
D4	PIN_J2	PIN_R10
G	PIN_C2	PIN_T11
C	PIN_E5	PIN_M11
DP	PIN_F2	PIN_N12
D	PIN_D5	PIN_P8
E	PIN_F1	PIN_N8

Датчик дальности	Cyclone IV IN	Cyclone IV OUT
Trig	PIN_T6	PIN_A7
Echo	PIN_N5	PIN_B8

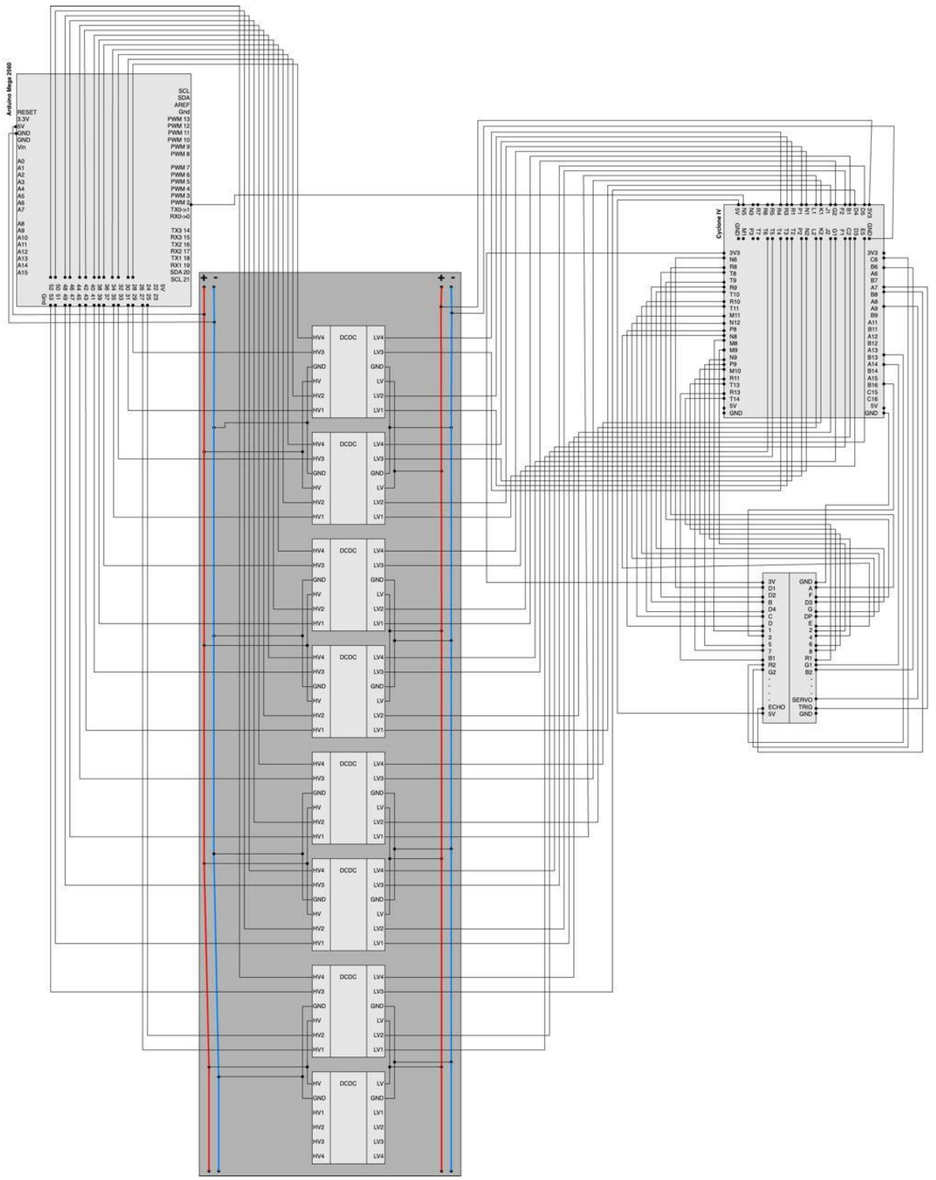
Сервопривод	Cyclone IV IN	Cyclone IV OUT
Servo	PIN_T5	PIN_A9

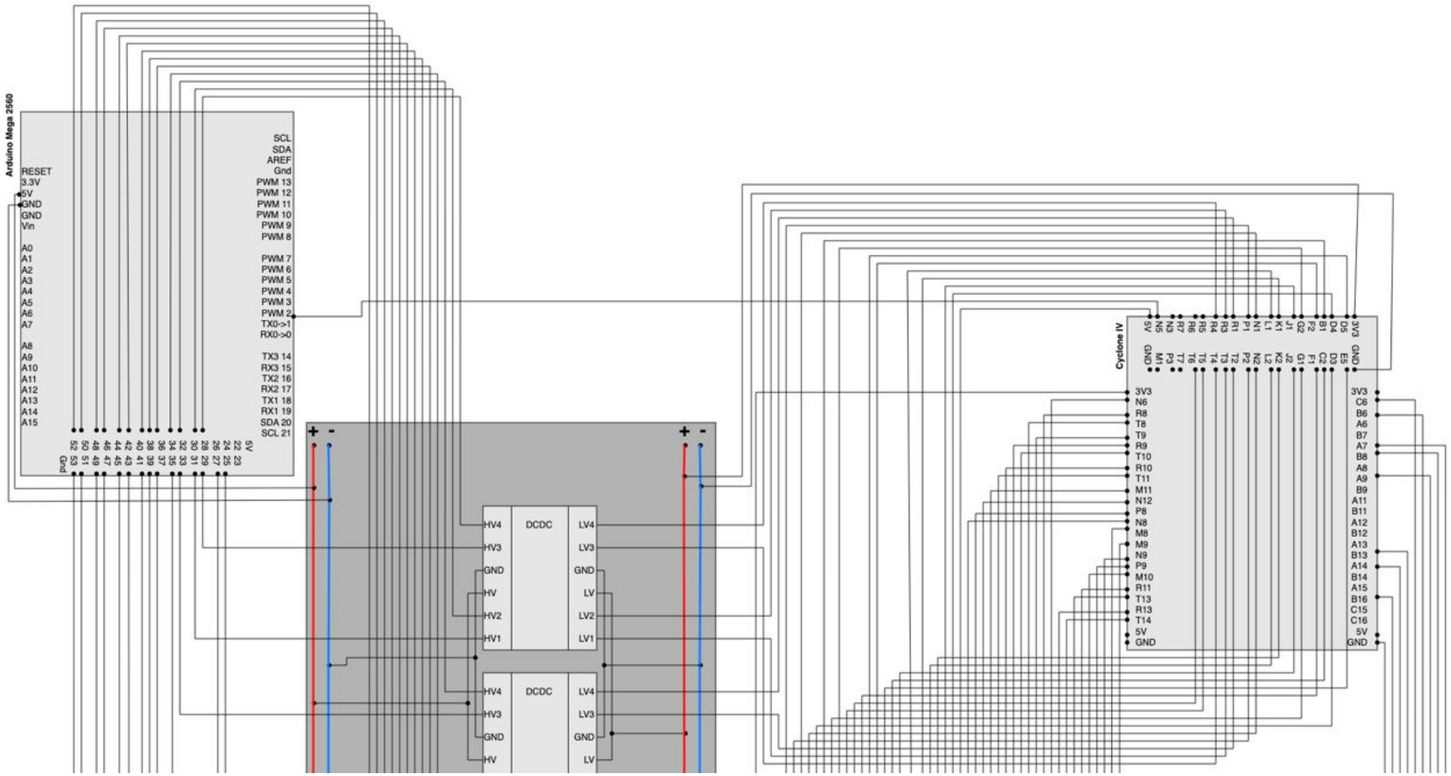
Arduino UNO	Светодиоды	Cyclone IV IN	Cyclone IV OUT
41	1	PIN_G2	PIN_M8
40	2	PIN_G1	PIN_M9
39	3	PIN_B1	PIN_A8
38	4	PIN_D3	PIN_N9
37	5	PIN_N2	PIN_P9
36	6	PIN_N1	PIN_M10
35	7	PIN_P2	PIN_R11
34	8	PIN_P1	PIN_T13
33	R1	PIN_T2	PIN_T14
32	G1	PIN_R1	PIN_B7
31	B1	PIN_T3	PIN_R13
30	R2	PIN_R3	PIN_A6
29	G2	PIN_T4	PIN_C6
28	B2	PIN_R4	PIN_B6

Левый столбец	Правый столбец
3V	GND
D1	A
D2	F
B	D3
D4	G
C	DP
D	E
1	2
3	4
5	6
7	8
B1	R1
R2	G1
G2	B2
	SERVO
ECHO	TRIG
5V	GND

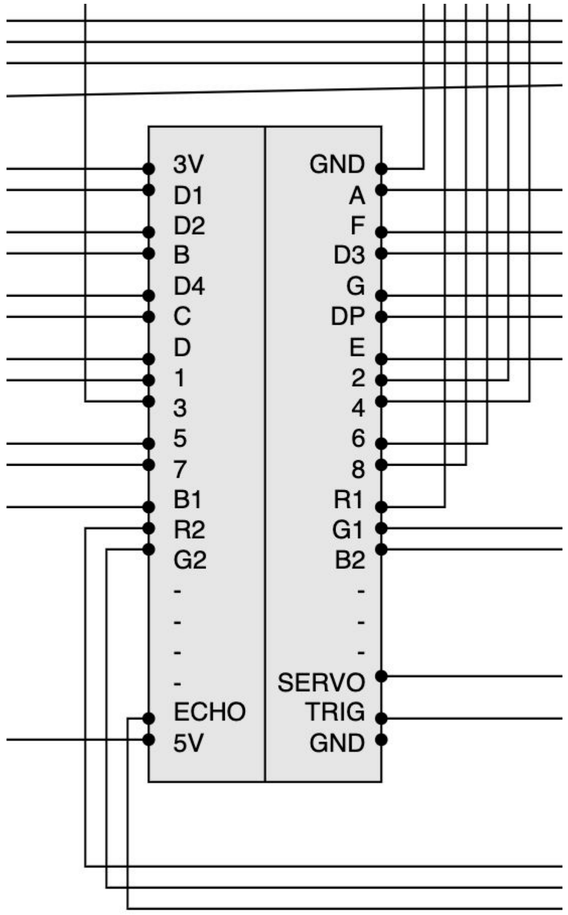


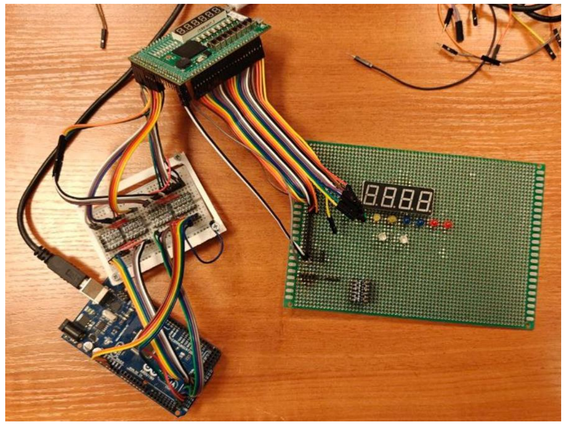
СХЕМОТЕХНИКА

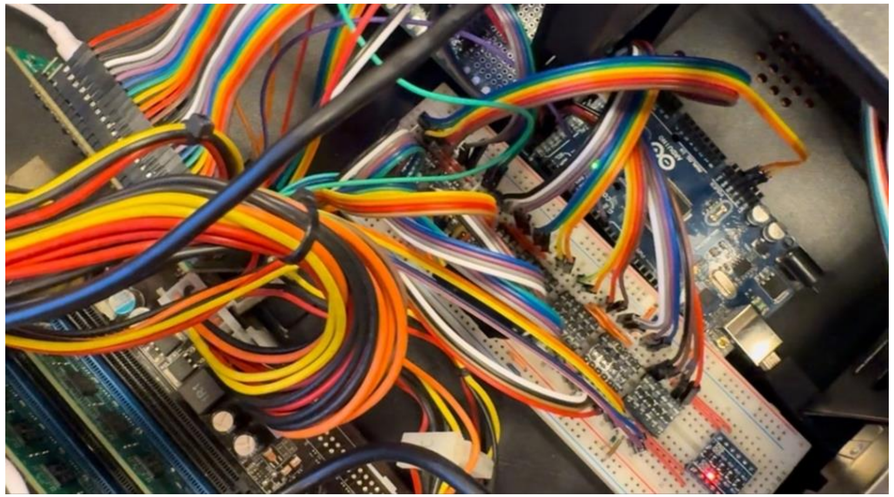


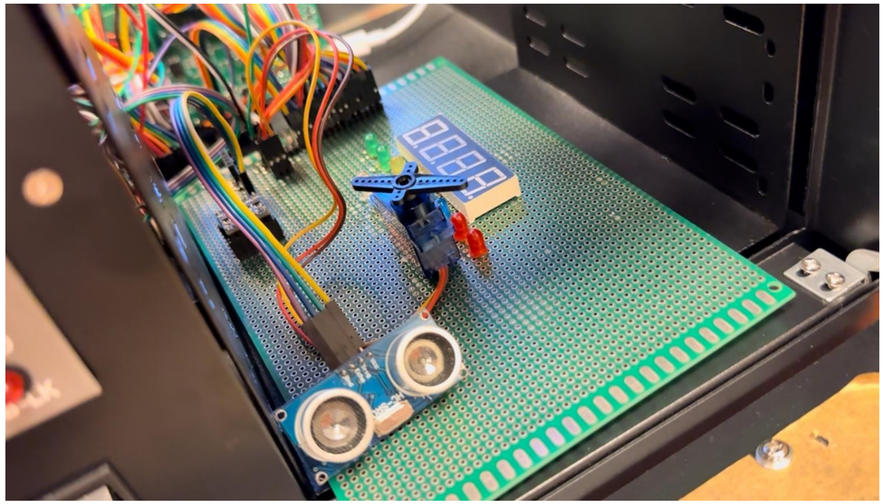


АППАРАТНАЯ ЧАСТЬ

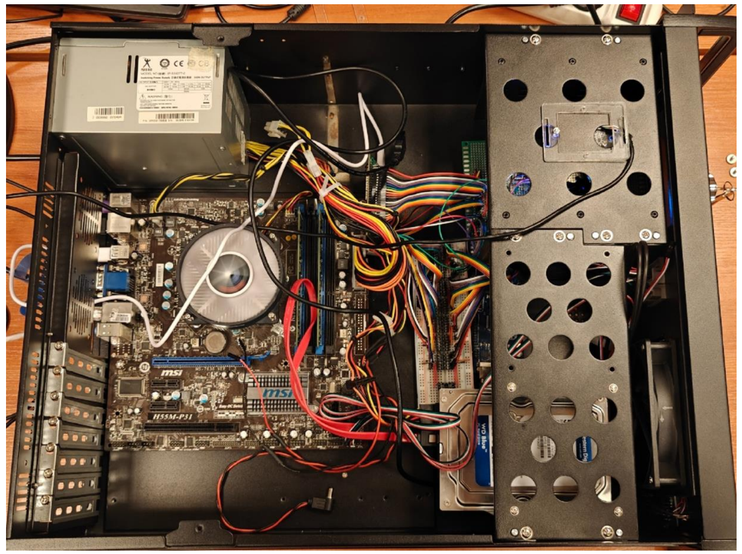


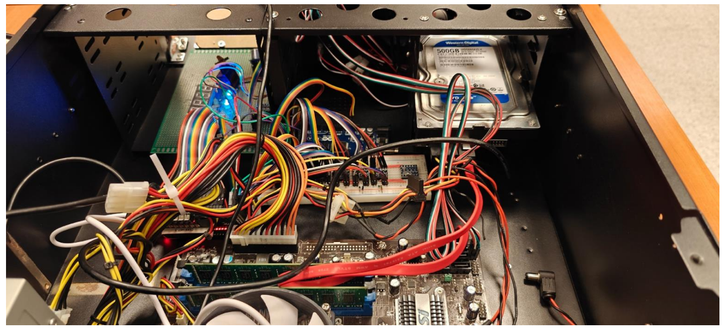


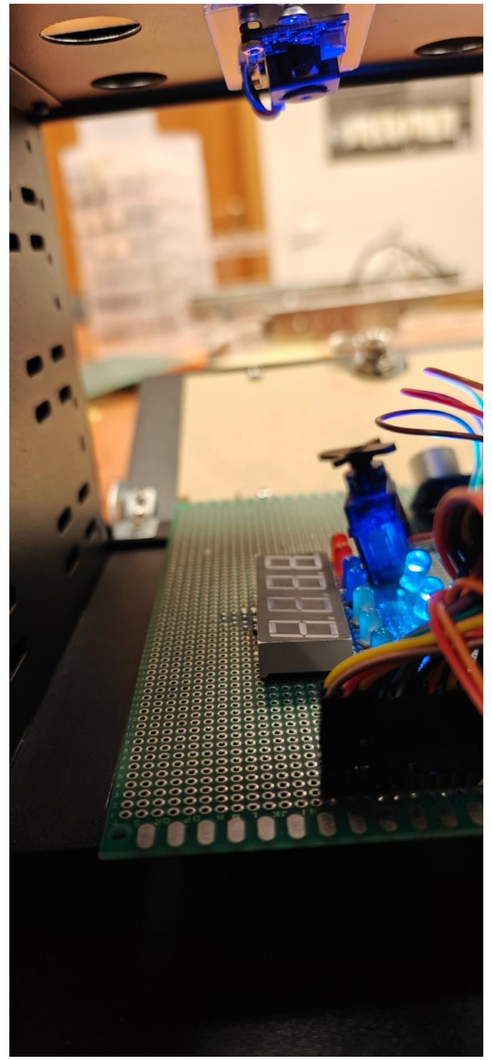


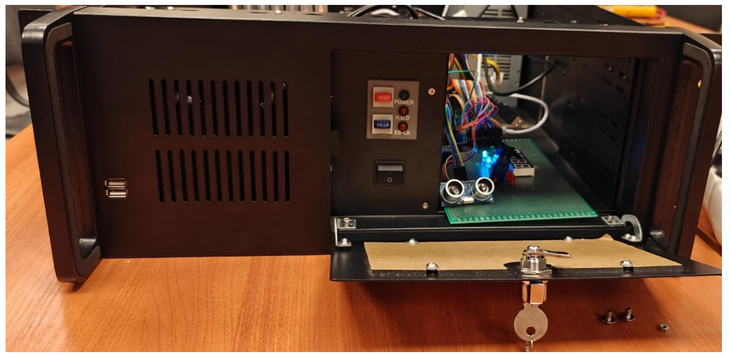


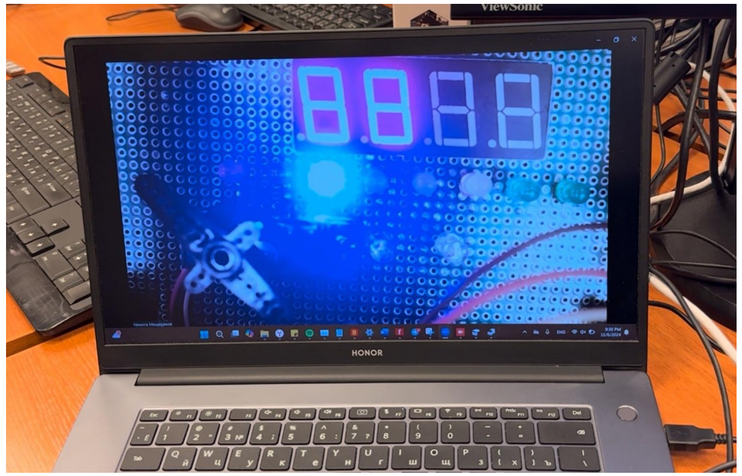
АППАРАТНАЯ ЧАСТЬ

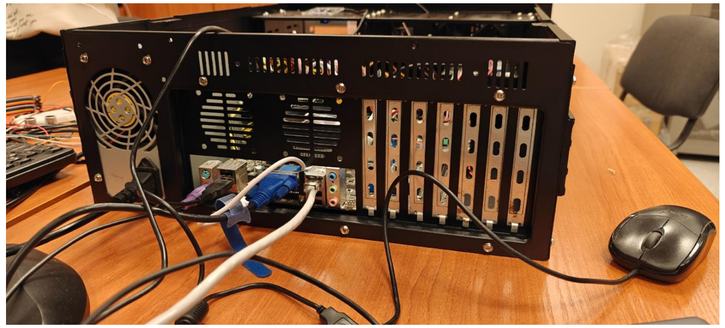












ПРОГРАММНАЯ ЧАСТЬ VERILOG

project_connection.v*

```
1  module project_connection(
2      input wire arduino53_in,
3      input wire arduino52_in,
4      input wire arduino51_in,
5      input wire arduino50_in,
6      input wire arduino49_in,
7      input wire arduino48_in,
8      input wire arduino47_in,
9      input wire arduino46_in,
10     input wire arduino45_in,
11     input wire arduino44_in,
12     input wire arduino43_in,
13     input wire arduino42_in,
14     input wire arduino41_in,
15     input wire arduino40_in,
16     input wire arduino39_in,
17     input wire arduino38_in,
18     input wire arduino37_in,
19     input wire arduino36_in,
20     input wire arduino35_in,
21     input wire arduino34_in,
22     input wire arduino33_in,
23     input wire arduino32_in,
24     input wire arduino31_in,
25     input wire arduino30_in,
26     input wire arduino29_in,
27     input wire arduino28_in,
28     input wire arduino27_in,
29     input wire arduino26_in,
30     input wire arduino25_in,
31     input wire fpgaEcho_in,
32     output wire fpgaD1_out,
33     output wire fpgaA_out,
34     output wire fpgaF_out,
35     output wire fpgaD2_out,
36     output wire fpgaD3_out,
37     output wire fpgaB_out,
38     output wire fpgaE_out,
39     output wire fpgaD_out,
40     output wire fpgaDP_out,
41     output wire fpgaC_out,
42     output wire fpgaG_out,
43     output wire fpgaD4_out,
44     output wire fpgaLED1_out,
45     output wire fpgaLED2_out,
46     output wire fpgaLED3_out,
47     output wire fpgaLED4_out,
48     output wire fpgaLED5_out,
49     output wire fpgaLED6_out,
50     output wire fpgaLED7_out,
51     output wire fpgaLED8_out,
52     output wire fpgaR1_out,
53     output wire fpgaR2_out,
54     output wire fpgaB1_out,
55     output wire fpgaB2_out,
56     output wire fpgaG1_out,
57     output wire fpgaG2_out,
58     output wire fpgaServo_out,
59     output wire fpgaTrig_out,
60     output wire arduino2_out
61 );
63 assign fpgaD1_out = arduino53_in;
64 assign fpgaA_out = arduino43_in;
65 assign fpgaF_out = arduino44_in;
66 assign fpgaD2_out = arduino46_in;
67 assign fpgaD3_out = arduino45_in;
68 assign fpgaB_out = arduino47_in;
69 assign fpgaD4_out = arduino42_in;
70 assign fpgaG_out = arduino52_in;
71 assign fpgaC_out = arduino51_in;
72 assign fpgaDP_out = arduino50_in;
73 assign fpgaD_out = arduino49_in;
74 assign fpgaE_out = arduino48_in;
75 assign fpgaLED1_out = arduino41_in;
76 assign fpgaLED2_out = arduino40_in;
77 assign fpgaLED3_out = arduino39_in;
78 assign fpgaLED4_out = arduino38_in;
79 assign fpgaLED5_out = arduino37_in;
80 assign fpgaLED6_out = arduino36_in;
81 assign fpgaLED7_out = arduino35_in;
82 assign fpgaLED8_out = arduino34_in;
83 assign fpgaR1_out = arduino33_in;
84 assign fpgaG1_out = arduino32_in;
85 assign fpgaB1_out = arduino31_in;
86 assign fpgaR2_out = arduino30_in;
87 assign fpgaG2_out = arduino29_in;
88 assign fpgaB2_out = arduino28_in;
89 assign fpgaServo_out = arduino27_in;
90 assign fpgaTrig_out = arduino25_in;
91 assign arduino2_out = fpgaEcho_in;
92
93
94 endmodule
```

ПРОГРАММНАЯ ЧАСТЬ ARDUINO


```
#include <Servo.h>

// Константы для индикаторов
const int a = 43;
const int b = 47;
const int c = 51;
const int d = 49;
const int e = 48;
const int f = 44;
const int g = 52;
const int p = 50; // Пин для точки

// Выводы для четырех индикаторов
const int D3 = 45;
const int D4 = 42;

// Константы для ультразвукового датчика
#define PIN_TRIG 25
#define PIN_ECHO 2

// Константы для сервопривода
const int servoPin = 27;
const int delayTime = 2;

// Константы для светодиодов
const int ledPins[] = {41, 40, 39, 38, 37, 36, 35, 34};
const int rgb1Pins[] = {33, 32, 31}; // Красный, Зеленый, Синий
const int rgb2Pins[] = {30, 29, 28}; // Красный, Зеленый, Синий
const int ledDelayTime = 50;

// Объект сервопривода
Servo myServo;

// Переменные для ультразвукового датчика
long duration, cm;
```

```
// Функция для включения сегментов, чтобы отобразить 8 с точкой
void display8() {
  digitalWrite(a, HIGH);
  digitalWrite(b, HIGH);
  digitalWrite(c, HIGH);
  digitalWrite(d, HIGH);
  digitalWrite(e, HIGH);
  digitalWrite(f, HIGH);
  digitalWrite(g, HIGH);
  digitalWrite(p, LOW);
}

void setup() {
  // Инициализация всех пинов как OUTPUT для индикаторов
  pinMode(a, OUTPUT);
  pinMode(b, OUTPUT);
  pinMode(c, OUTPUT);
  pinMode(d, OUTPUT);
  pinMode(e, OUTPUT);
  pinMode(f, OUTPUT);
  pinMode(g, OUTPUT);
  pinMode(p, OUTPUT);
  pinMode(D3, OUTPUT);
  pinMode(D4, OUTPUT);

  // Инициализируем взаимодействие по последовательному порту
  Serial.begin(9600);

  // Определяем вводы и выводы для ультразвукового датчика
  pinMode(PIN_TRIG, OUTPUT);
  pinMode(PIN_ECHO, INPUT);

  // Присоединение сервопривода к пину
  myServo.attach(servoPin);

  // Инициализация пинов для светодиодов
  for (int i = 0; i < 8; i++) {
    pinMode(ledPins[i], OUTPUT);
  }

  // Инициализация пинов для RGB светодиодов
  for (int i = 0; i < 3; i++) {
    pinMode(rgb1Pins[i], OUTPUT);
    pinMode(rgb2Pins[i], OUTPUT);
  }
```

ПРОГРАММНАЯ ЧАСТЬ ARDUINO

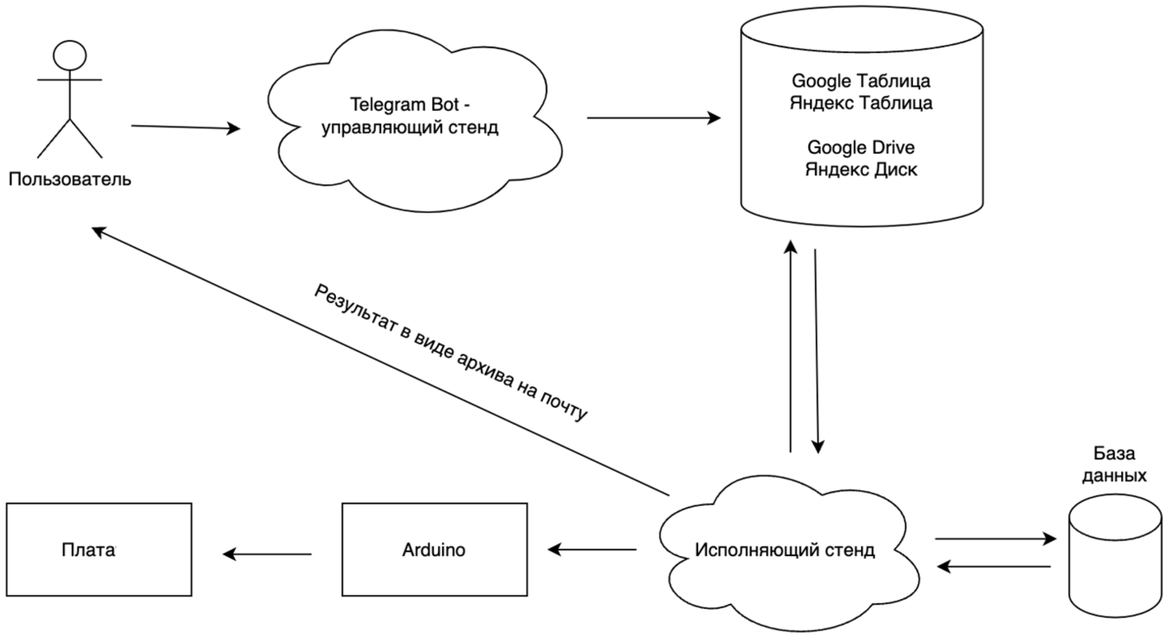

```
void loop() {

  digitalWrite(D4, HIGH);
  display8();
  delay(5);
  digitalWrite(D4, LOW);
  digitalWrite(D3, HIGH);
  display8();
  delay(5);
  digitalWrite(D3, LOW);

  // Управление ультразвуковым датчиком
  digitalWrite(PIN_TRIG, LOW);
  delayMicroseconds(5);
  digitalWrite(PIN_TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(PIN_TRIG, LOW);
  duration = pulseIn(PIN_ECHO, HIGH);
  cm = (duration / 2) / 29.1;
  Serial.print("Расстояние до объекта: ");
  Serial.print(cm);
  Serial.println(" см.");

  // Управление RGB светодиодами в зависимости от расстояния
  if (cm <= 10) {
    digitalWrite(rgb1Pins[1], HIGH);
    digitalWrite(rgb2Pins[1], HIGH);
    digitalWrite(rgb1Pins[0], LOW);
    digitalWrite(rgb2Pins[0], LOW);
    digitalWrite(rgb1Pins[2], LOW);
    digitalWrite(rgb2Pins[2], LOW);
  } else if (cm <= 20) {
    digitalWrite(rgb1Pins[1], LOW);
    digitalWrite(rgb2Pins[1], LOW);
    digitalWrite(rgb1Pins[0], LOW);
    digitalWrite(rgb2Pins[0], LOW);
    digitalWrite(rgb1Pins[2], HIGH);
    digitalWrite(rgb2Pins[2], HIGH);
  } else {
    digitalWrite(rgb1Pins[1], LOW);
    digitalWrite(rgb2Pins[1], LOW);
    digitalWrite(rgb1Pins[2], LOW);
    digitalWrite(rgb2Pins[2], LOW);
    digitalWrite(rgb1Pins[0], HIGH);
    digitalWrite(rgb2Pins[0], HIGH);
  }

  // Управление сервоприводом (пример: вращение от 0 до 180 градусов)
  static int angle = 0;
  static bool increasing = true;
  if (increasing) {
    myServo.write(angle);
    angle++;
    if (angle > 270) {
      increasing = false;
    }
  } else {
    myServo.write(angle);
    angle--;
    if (angle < 0) {
      increasing = true;
    }
  }
  delay(delayTime / 180); // Задержка для плавного вращения

  // Мигание обычных светодиодов
  static int ledIndex = 0;
  static bool ledState = false;
  digitalWrite(ledPins[ledIndex], ledState);
  ledState =!ledState;
  delay(ledDelayTime);
  if (ledState) {
    ledIndex = (ledIndex + 1) % 8;
  }
}
```

ПРОГРАММНАЯ ЧАСТЬ ARDUINO

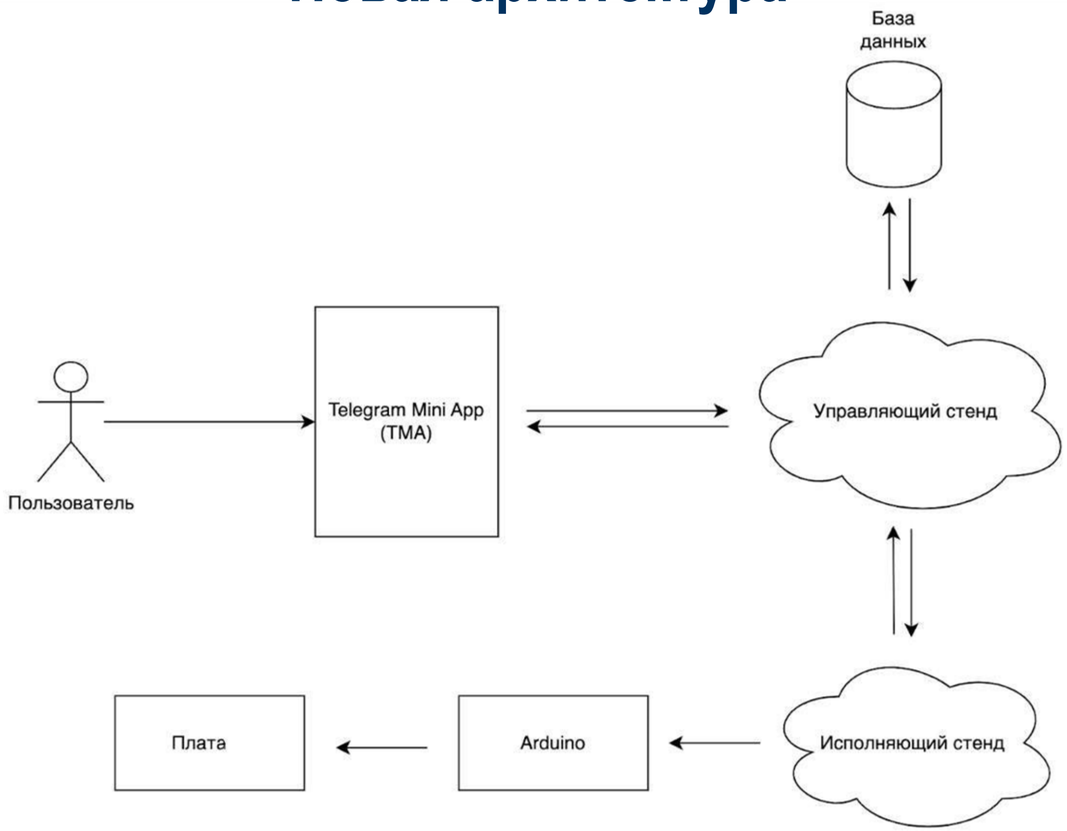

```
void loop() {

  digitalWrite(D4, HIGH);
  display8();
  delay(5);
  digitalWrite(D4, LOW);
  digitalWrite(D3, HIGH);
  display8();
  delay(5);
  digitalWrite(D3, LOW);

  // Управление ультразвуковым датчиком
  digitalWrite(PIN_TRIG, LOW);
  delayMicroseconds(5);
  digitalWrite(PIN_TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(PIN_TRIG, LOW);
  duration = pulseIn(PIN_ECHO, HIGH);
  cm = (duration / 2) / 29.1;
  Serial.print("Расстояние до объекта: ");
  Serial.print(cm);
  Serial.println(" см.");

  // Управление RGB светодиодами в зависимости от расстояния
  if (cm <= 10) {
    digitalWrite(rgb1Pins[1], HIGH);
    digitalWrite(rgb2Pins[1], HIGH);
    digitalWrite(rgb1Pins[0], LOW);
    digitalWrite(rgb2Pins[0], LOW);
    digitalWrite(rgb1Pins[2], LOW);
    digitalWrite(rgb2Pins[2], LOW);
  } else if (cm <= 20) {
    digitalWrite(rgb1Pins[1], LOW);
    digitalWrite(rgb2Pins[1], LOW);
    digitalWrite(rgb1Pins[0], LOW);
    digitalWrite(rgb2Pins[0], LOW);
    digitalWrite(rgb1Pins[2], HIGH);
    digitalWrite(rgb2Pins[2], HIGH);
  } else {
    digitalWrite(rgb1Pins[1], LOW);
    digitalWrite(rgb2Pins[1], LOW);
    digitalWrite(rgb1Pins[2], LOW);
    digitalWrite(rgb2Pins[2], LOW);
    digitalWrite(rgb1Pins[0], HIGH);
    digitalWrite(rgb2Pins[0], HIGH);
  }

  // Управление сервоприводом (пример: вращение от 0 до 180 градусов)
  static int angle = 0;
  static bool increasing = true;
  if (increasing) {
    myServo.write(angle);
    angle++;
    if (angle > 270) {
      increasing = false;
    }
  } else {
    myServo.write(angle);
    angle--;
    if (angle < 0) {
      increasing = true;
    }
  }
  delay(delayTime / 180); // Задержка для плавного вращения

  // Мигание обычных светодиодов
  static int ledIndex = 0;
  static bool ledState = false;
  digitalWrite(ledPins[ledIndex], ledState);
  ledState =!ledState;
  delay(ledDelayTime);
  if (ledState) {
    ledIndex = (ledIndex + 1) % 8;
  }
}
```

АРХИТЕКТУРА ПРИЛОЖЕНИЯ

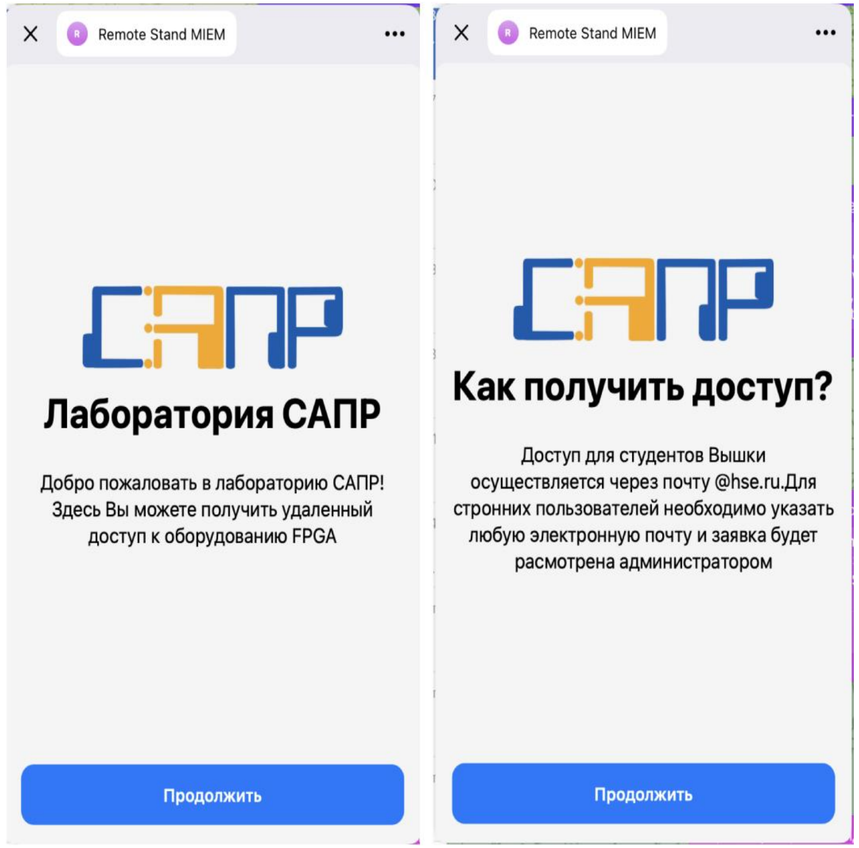
Старая архитектура



Новая архитектура



TELEGRAM MINI APP (TMA)



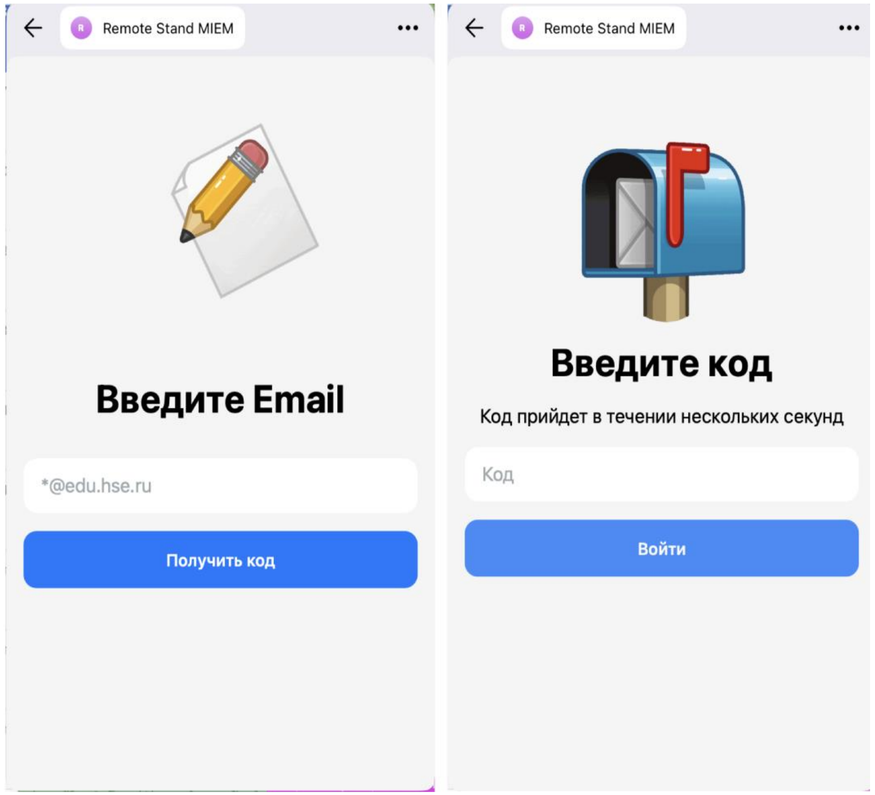
Онбординг

Отображается для неавторизованных пользователей, знакомит пользователя с сервисом.

Авторизация

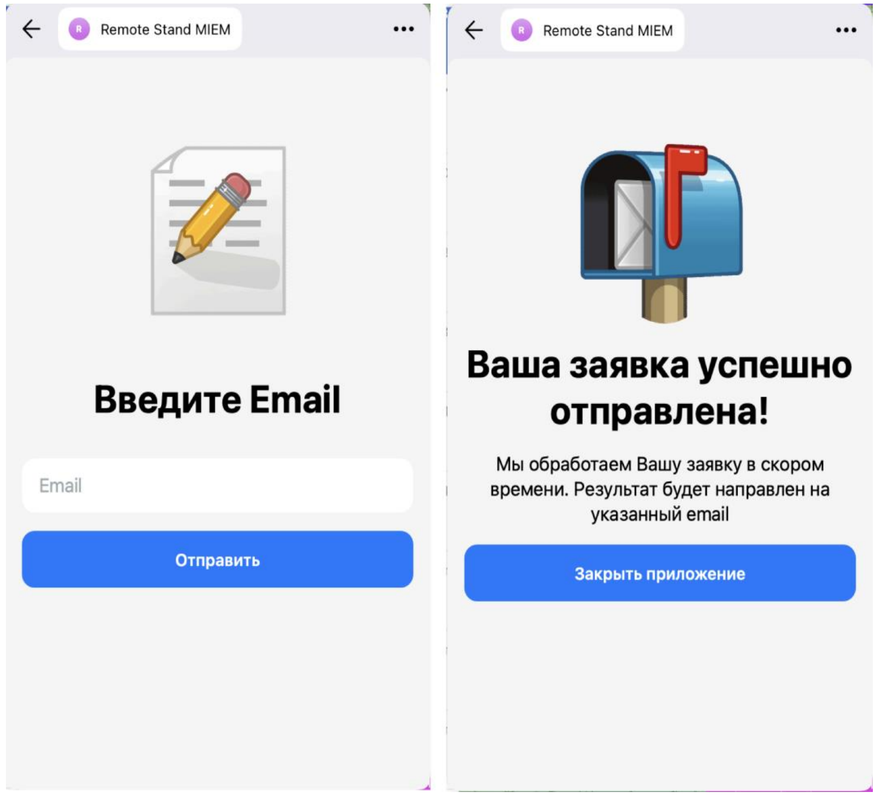
Возможность авторизоваться как студенту НИУ ВШЭ, так и стороннему пользователю.

TELEGRAM MINI APP (TMA)



Авторизация пользователя ВШЭ

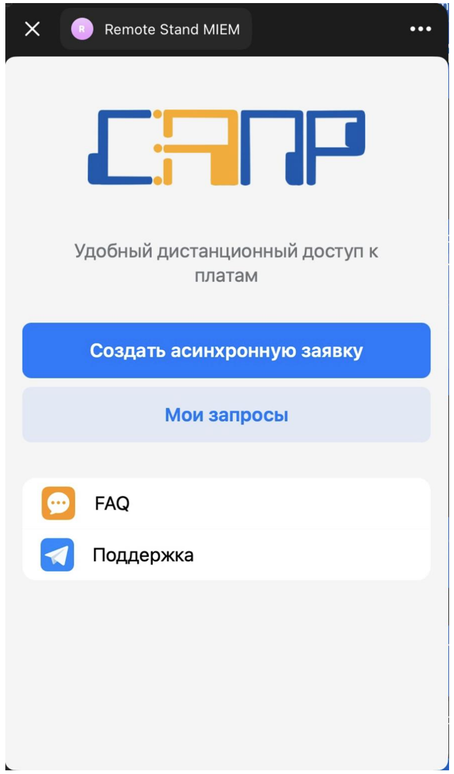
Возможность авторизоваться моментально путем получения кода авторизации на почту с доменом @edu.hse и @hse.ru.



Авторизация стороннего пользователя

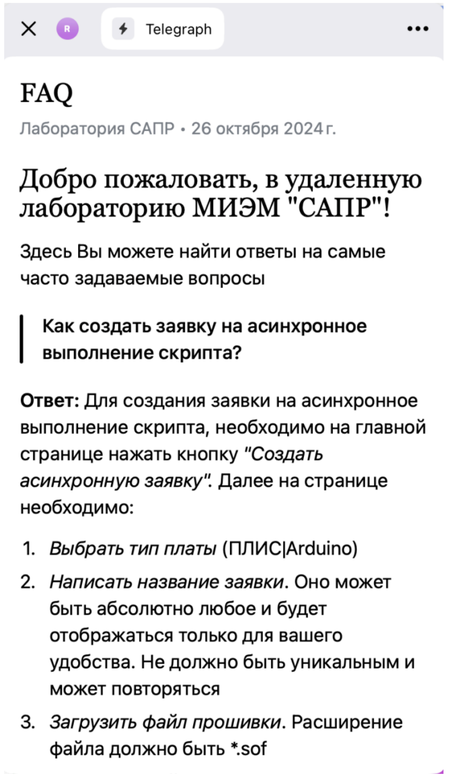
Возможность авторизоваться с помощью любой почты. В таком случае администратор в ручном режиме должен подтвердить регистрацию.

TELEGRAM MINI APP (TMA)



Главный экран

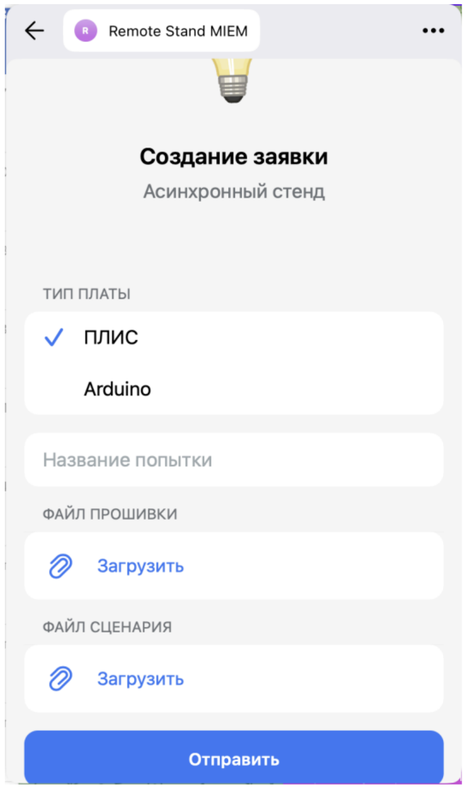
Возможность перейти к странице для создания заявки, перейти к странице для просмотра отправленных заявок, написать в поддержку и посмотреть ответы на “Часто задаваемые” вопросы.



FAQ

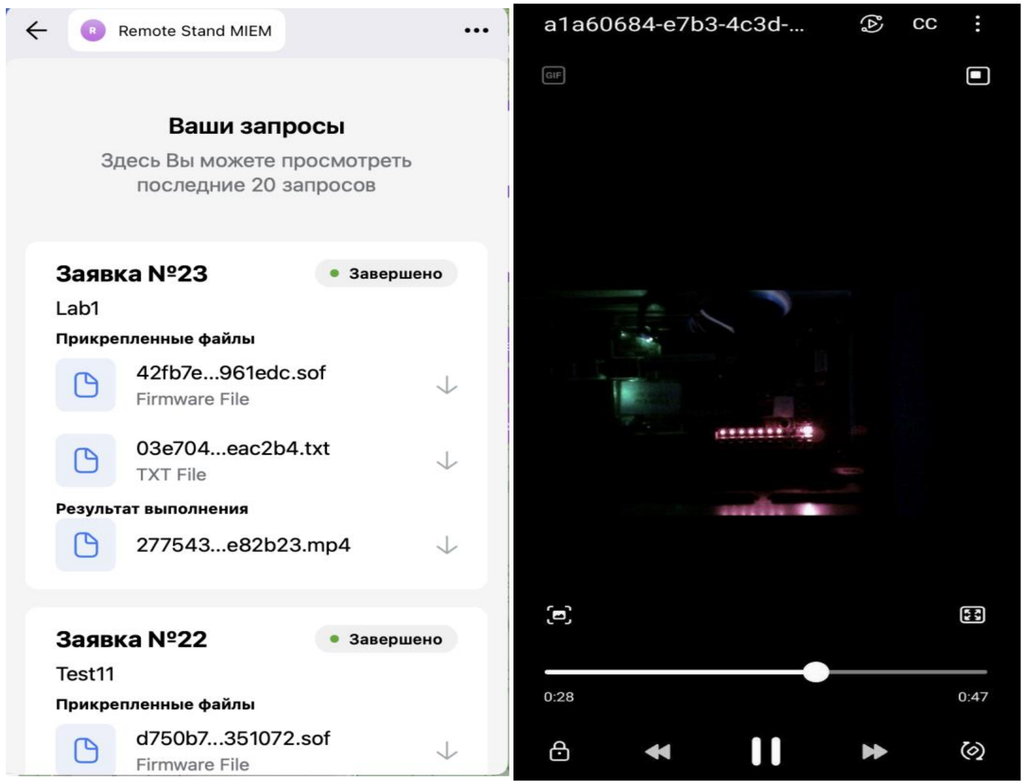
Открывается в отдельном TMA; статья telegraph.

TELEGRAM MINI APP (TMA)



Создание заявки

Загрузка файла прошивки и файла сценария, указание названия заявки, выбор платы (на данный момент недоступно).

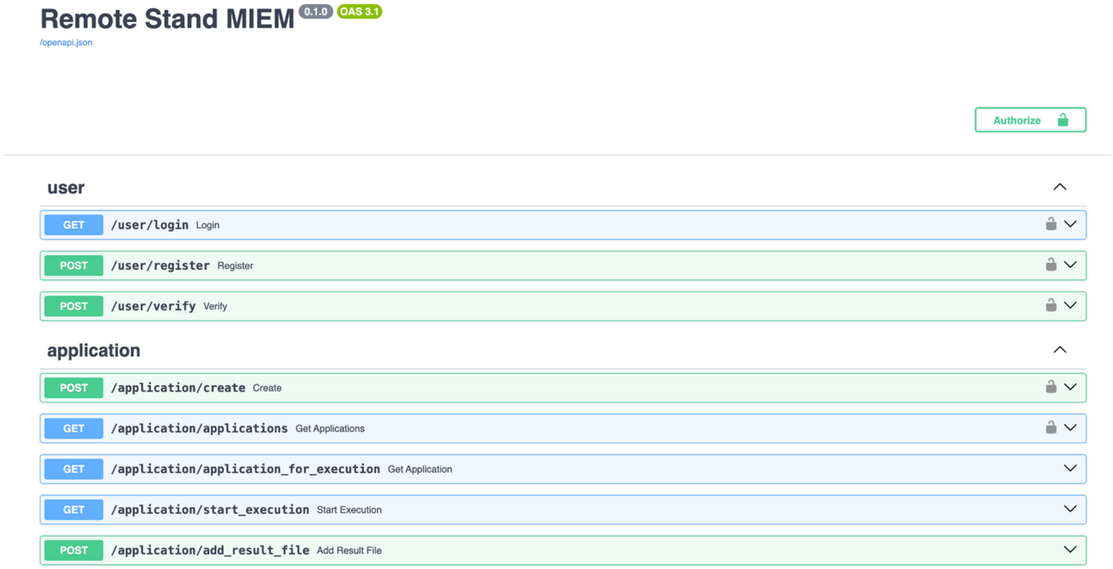


Просмотр заявок

Статус заявки, прикрепленные файлы и файл с результатом.

СТРУКТУРА

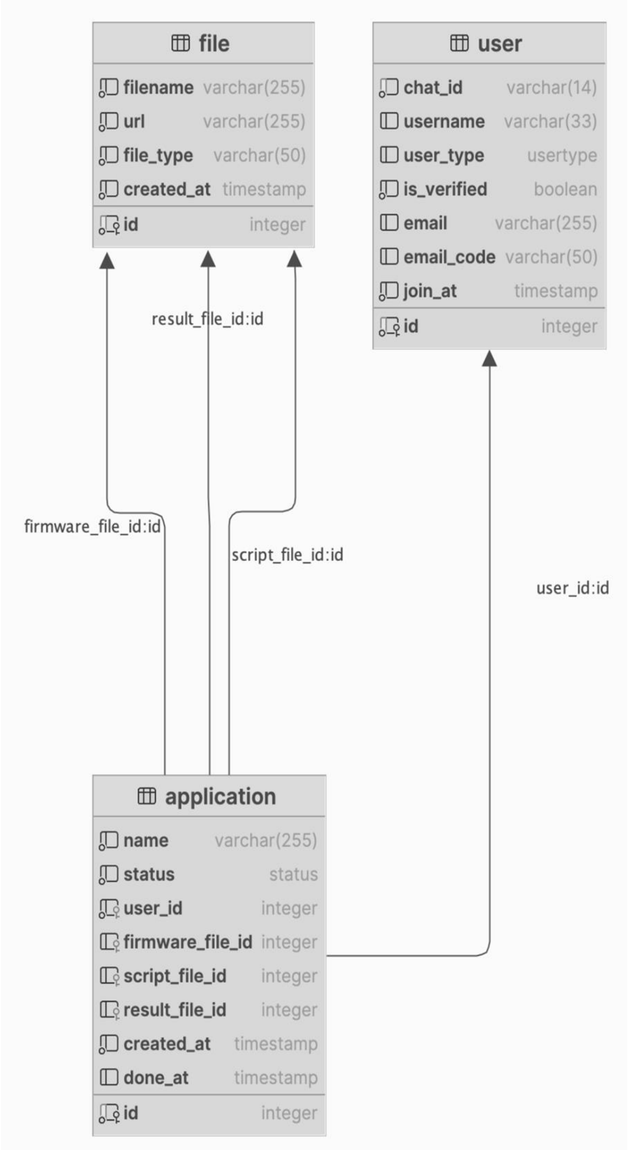
Управляющий стенд:



Подчиненный стенд:

1. проверяет наличие прошивки каждую минуту, когда не исполняет другую заявку;
2. скачивает файлы с управляющего стенда и загружает их туда;
3. благодаря архитектуре, легко масштабируется;
4. не влияет на основную работу сервиса.

ХРАНЕНИЕ ДАННЫХ



Файлы прошивки, сценария и результата хранятся на сервере управляющего стенда.

ССЫЛКИ

Telegram Mini App

https://t.me/remote_stand_miem_bot/app

Архив с тестовыми данными

https://dropmefiles.com/Kjhqp

ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ

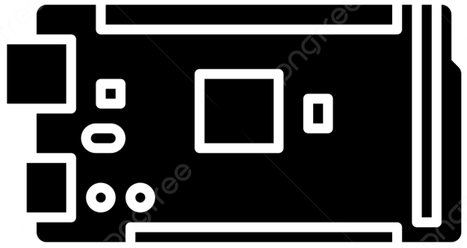
Проектные:

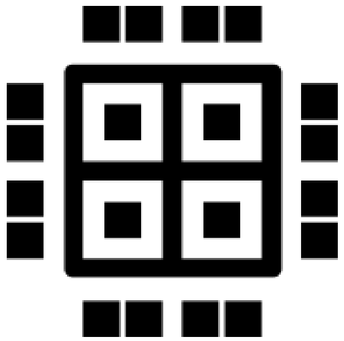
- собранный стенд с сопутствующей документацией;
- спроектирован программно-аппаратный комплекс с новой архитектурой;
- расширен набор датчиков на макетной плате;
- написанное ПО, с помощью которого у пользователя есть возможность реализовать необходимую для его задач схему;
- разработано приложение, благодаря которому получить асинхронный доступ к стенду может любой желающий;

Образовательные:

- улучшение навыков программирования на языках высокого уровня;
- расширение знания по разработке схемотехники;
- улучшение навыков реверс инжиниринга чужого кода;
- получение навыков работы с системами контроля версий;
- получение опыта командной работы над проектами.

ПРЕИМУЩЕСТВА РАЗРАБОТКИ ПРОЕКТА





- Удобство доступа для широкого круга пользователей.
- Расширение возможностей обучения.
- Экономическая выгода для пользователей, так как нет необходимости в покупке дорогостоящего оборудования.

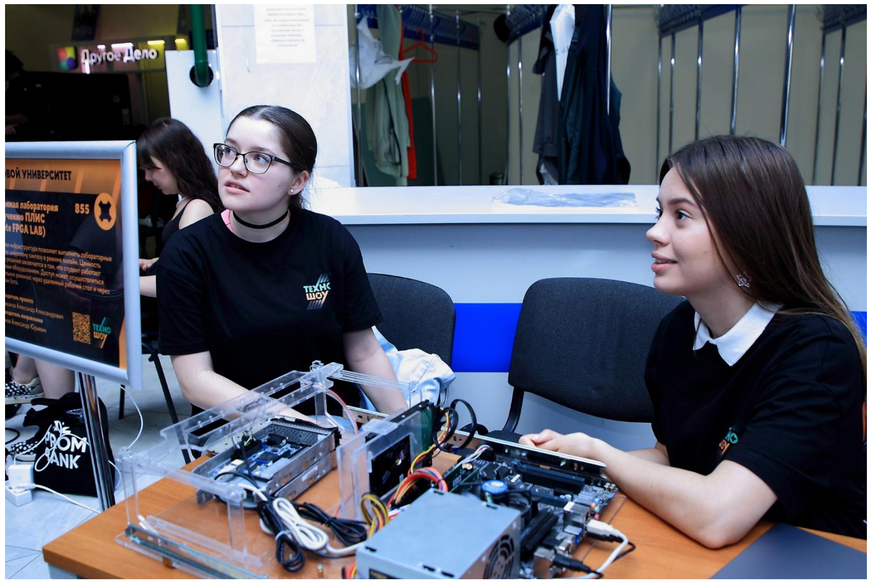
ДАЛЬНЕЙШИЕ ПЛАНЫ

- Добавление поддержки новых типов плат.
- Замена прототипа платы с элементами и других частей реконфигурируемого стенда на полноценное конструкторское решение.
- Разработка решения для проверки работоспособности ультразвукового датчика расстояния.
- Расширение пользовательских возможностей для удаленной работы.
- Регулярное обновление программной части проекта.

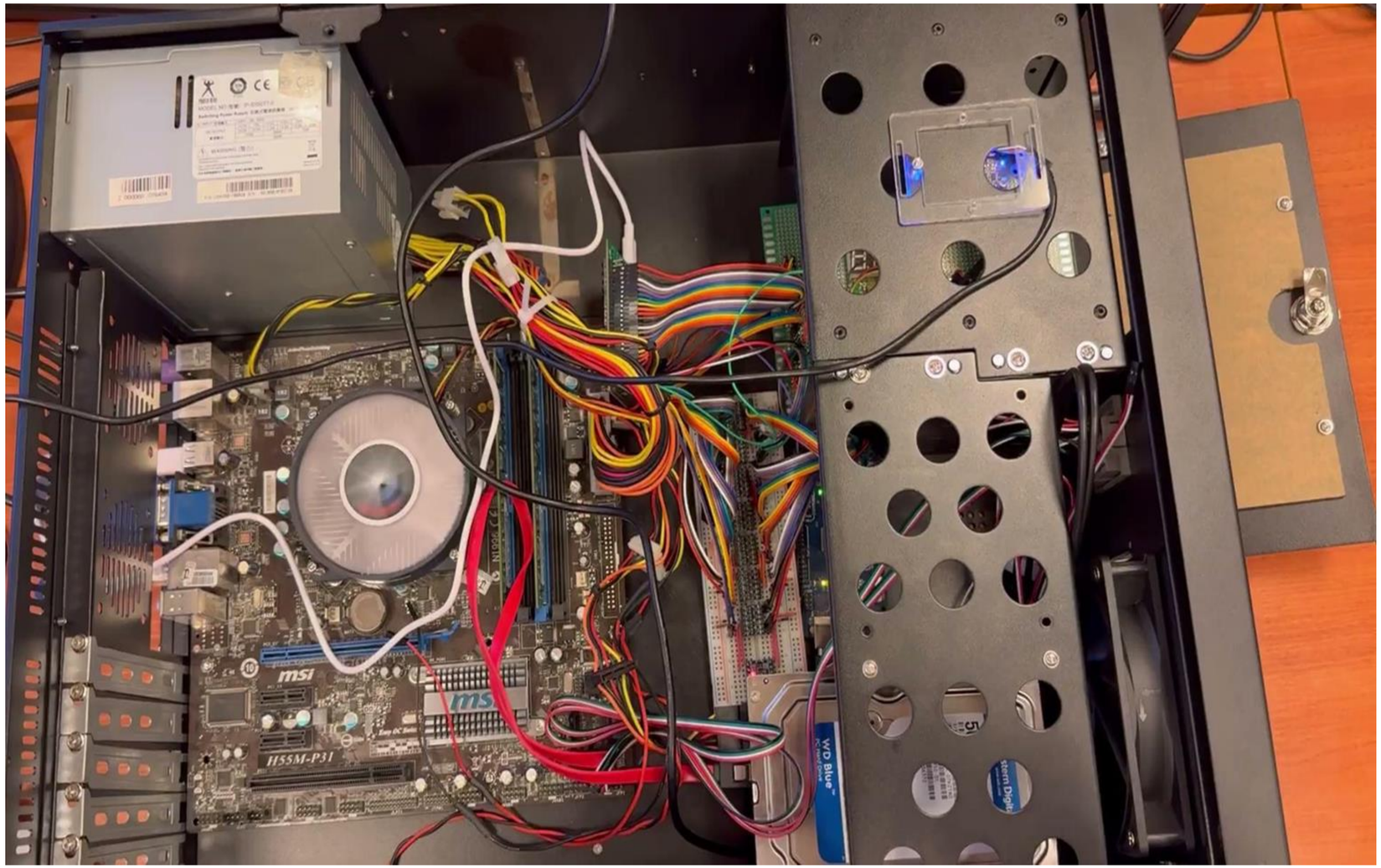
ПОТЕНЦИАЛ ДАЛЬНЕЙШЕЙ РАЗРАБОТКИ

- Использование разработанного стенда в лаборатории УЛ САПР МИЭМ НИУ ВШЭ.
- Представление прототипа устройства на различных мероприятиях НИУ ВШЭ.
- Предоставление стенда для использования другим учебным заведениям и предприятиям.
- Выступление на Ежегодной межвузовской научно-технической конференции студентов, аспирантов и молодых специалистов имени Е.В. Арменского.

САПР



РАБОТА СТЕНДА



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ