ANS Elbląg

Instytut Informatyki Stosowanej im. Krzysztofa Brzeskiego

Systemy Wbudowanie i Mikroprocesory – Projekt 2023

Studium Stacjonarne, sem. 4, 2023

Sprawozdanie nr : 3,

nr grupy: 2 ,

dzień: Poniedziałek, godz.

10:45.

Data zadania projektu: 01.03

Data oddania sprawozdania: 04.06

Nazwiska i imiona: Kuczawski Kacper, Gładki Patryk, Kordalski Kacper

Nry albumów: 20195, 20225, 20192

Nazwa pliku : Sprawozdanie_projekt_3

Sprawozdanie 3 + finalna prezentacja (11.06.2023)

Sprawozdanie powinno szczegółowy raport z budowy pojazdu od samego początku z uwzględnieniem dodatkowej funkcjonalności.

Sprawozdanie należy przesłać w formie pliku *.PDF oraz plików *.zip zawierających projekt pojazdu i oprogramowania.

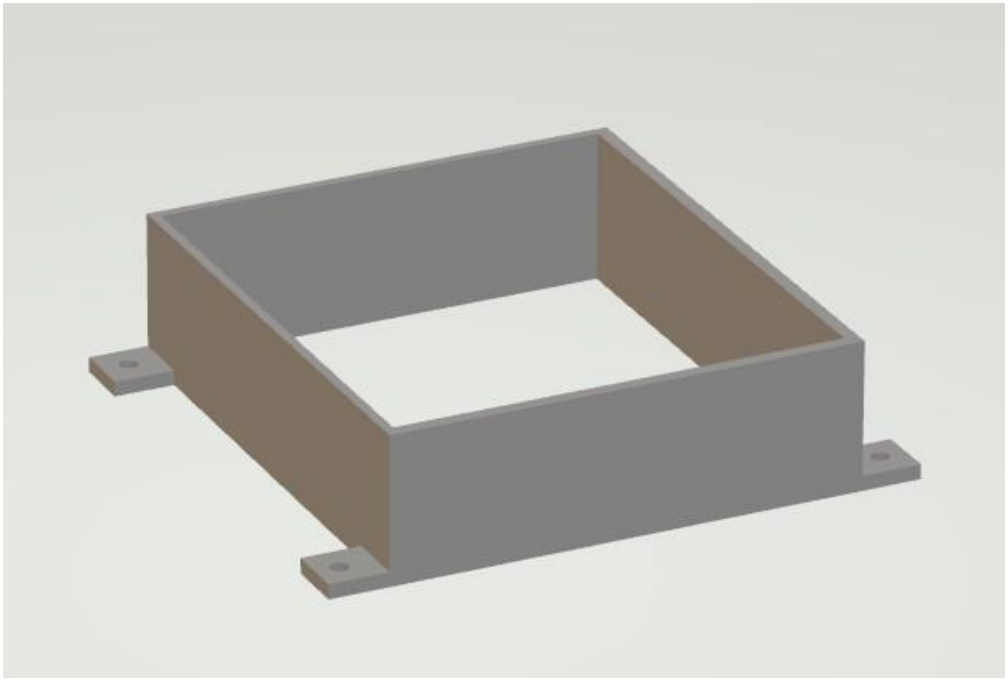
Prezentację należy przesłać w formie plików .mp4 + dodatkowe pliki.

Przebieg pracy

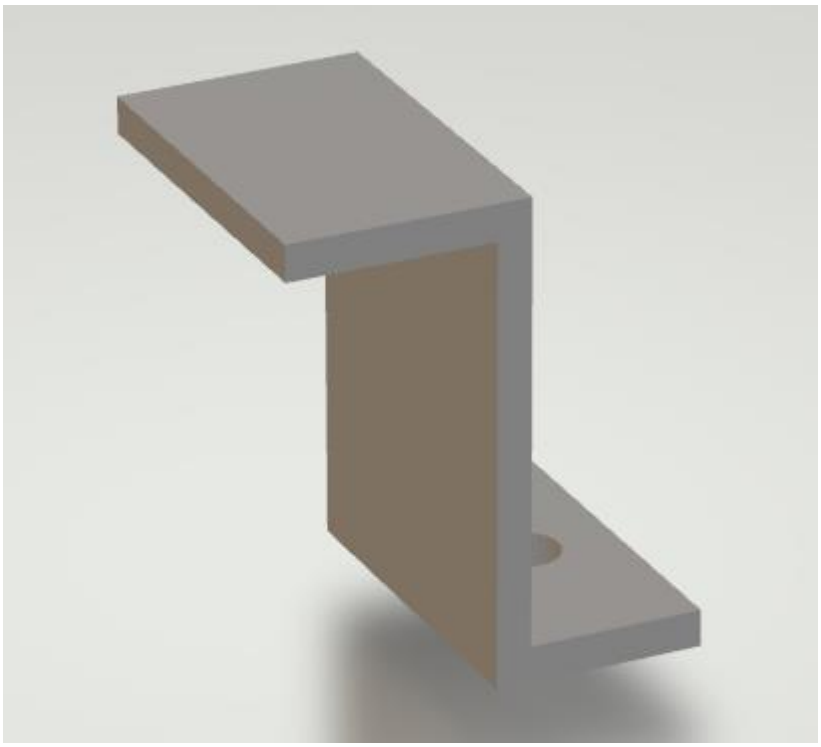
Pracę nad naszą autorską maszynę zaczęliśmy dość nieszablonowo, ponieważ od pomysłów na funkcjonalność. Koncepcji nam nie brakowało i każda była ciekawsza od poprzedniej. Musieliśmy w pewnym momencie jednak zejść na ziemię, ze względu na to, że myślenie o dodatkach przysłaniało nam nasz priorytetowy cel do pierwszego terminu, a mianowicie opracowanie wstępnego szkieletu łazika.

Zdecydowaliśmy się na coś uniwersalnego i łatwego do ewentualnej zmiany koncepcji tudzież rozbudowania. Kacper Kordalski zaproponował kształt robota, oraz rozmieszczenie na nim komponentów. Projekt po uzgodnieniu z grupą i drobnych, zaproponowanych poprawkach trafił do realizacji. Na jednych z zajęć, całą grupą usiedliśmy do opracowywania modeli do druku komponentów potrzebnych do zamontowania elementów pojazdu. W tym miejscu pomógł nam Doktor Kowalski udzielając cennych wskazówek jak pracować przy modelowaniu i jak posługiwać się programem do tworzenia plików, które potem trafiały do drukarek, z których to przeszkolenie również zostało nam udzielone. Pracą w programie Freecad również zajął się Kacper, jako że opanował to najlepiej, ale cały czas mógł liczyć na resztę grupy. W tym programie utworzyliśmy następujące części:

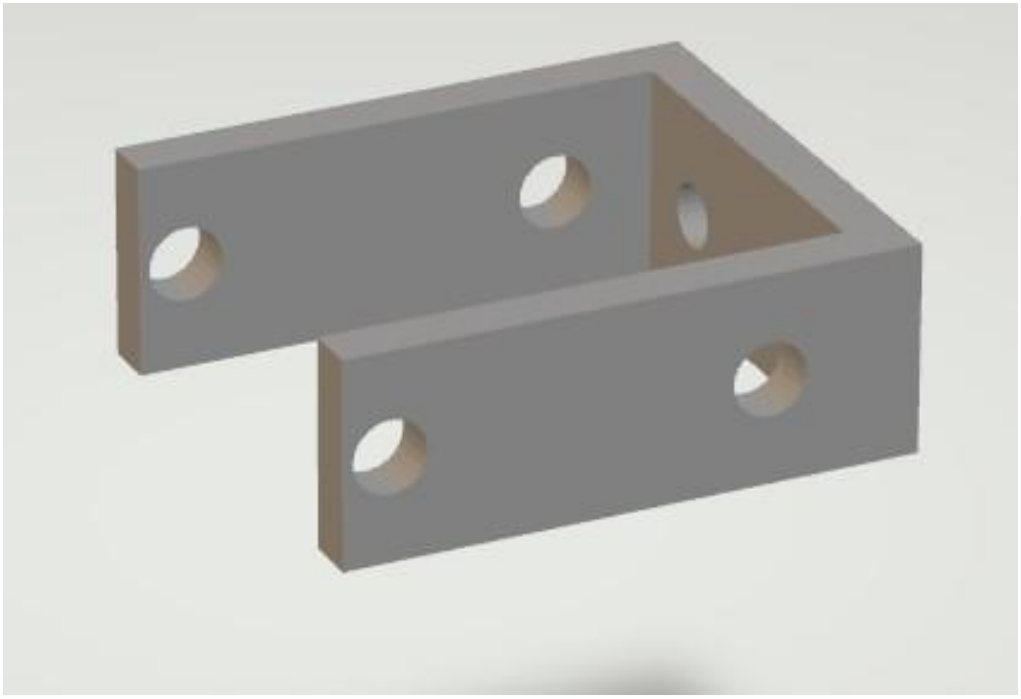
- **Koszyk na STM32** został wykonany w druku 3D, zapewnia bezpieczną pozycje oraz unieruchamia płytkę by uniknąć uszkodzeń



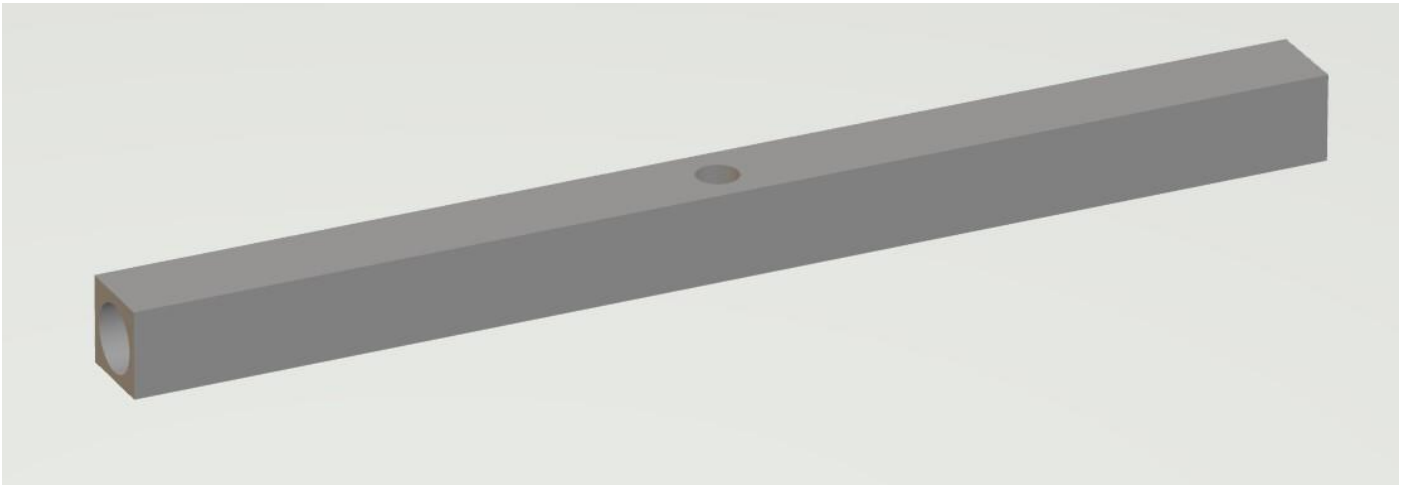
- **Uchwyty na baterie** zostały stworzone zmyślą o zamocowaniu 2 baterii potrzebnych do zasilenia serwa oraz silników



- **Uchwyty do silników** w celu przymocowania wypożyczonych silników od spodu



- **Belka pod servo** w celu zrealizowania skrętności pojazdu



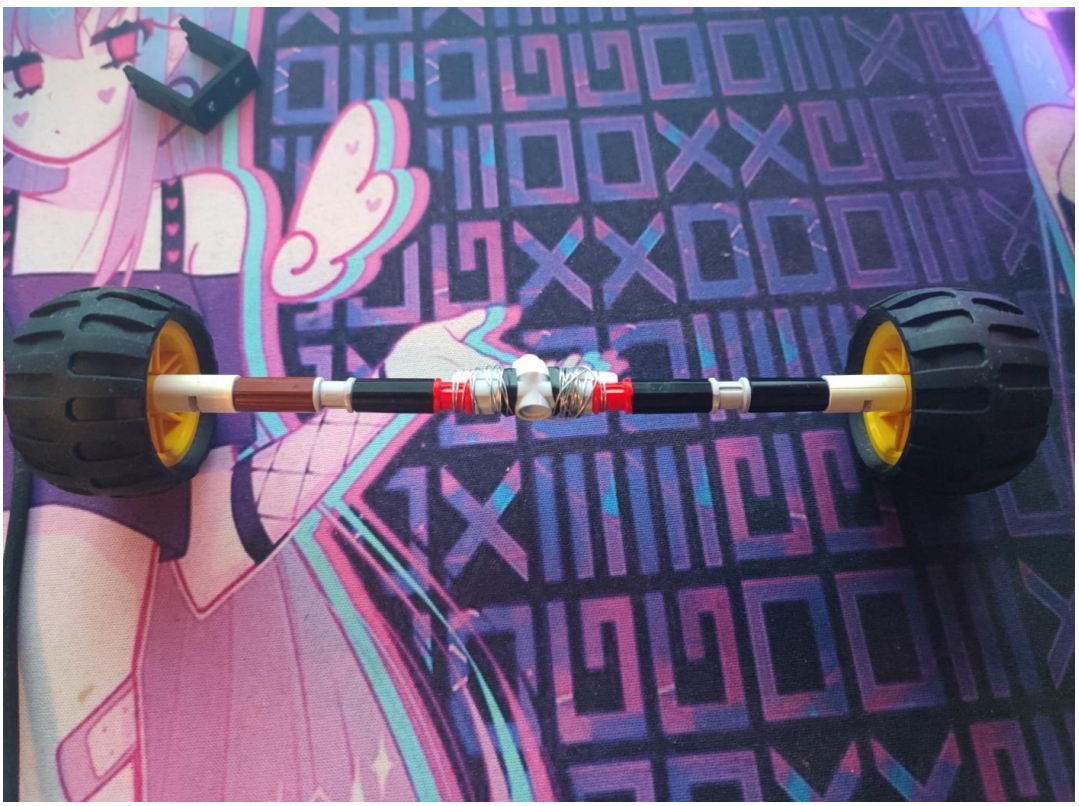
!!!Ostatecznie z niej zrezygnowaliśmy na rzecz komponentów z LEGO!!!

Po zaprojektowaniu wszystkich potrzebnych nam części przyszła pora na ich wydruki. Poświęciliśmy kila dni wolnych na regularne spotkania w laboratorium, by drukować, przymierzać, poprawiać i ponownie projektować części. Oczywiście jak w każdym tego typu większym przedsięwzięciu bywa, nie obeszło się bez komplikacji i utrudnień, z którymi musieliśmy sobie poradzić i oczywiście to zrobiliśmy. Problem wystąpił z przednią osią, której projekt był problematyczny przy naszym założeniu jej działania.

- **Przykładowe nie udane wydruki**

Na szczęście Kacper Kuczawski zaproponował wykorzystanie posiadanych przez niego materiałów mogących idealnie dopasować się do całokształtu projektu, w postaci części mechanicznych z serii Lego.

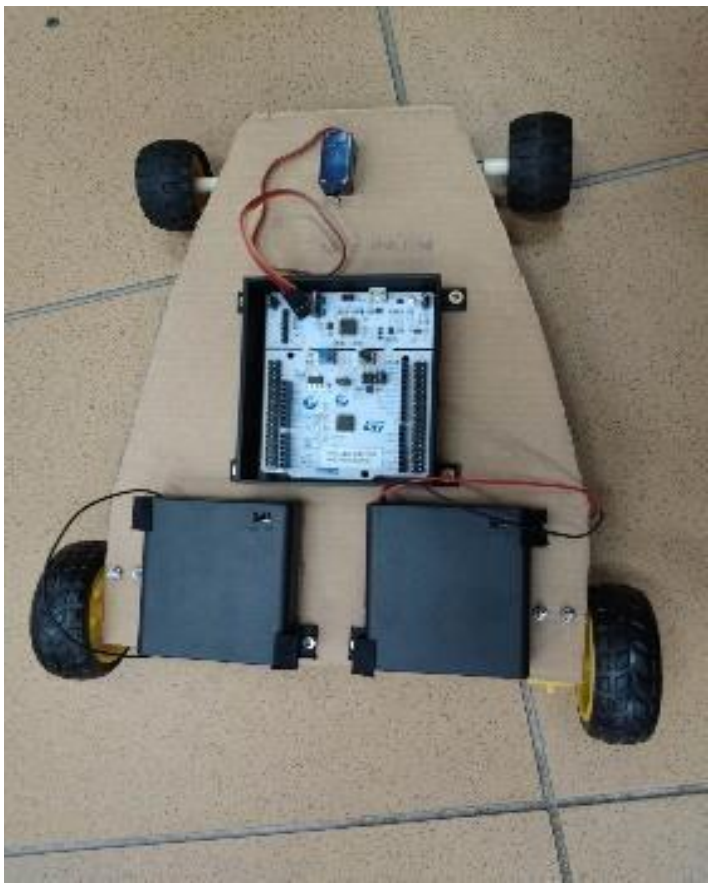
- **Użyte komponenty z lego**

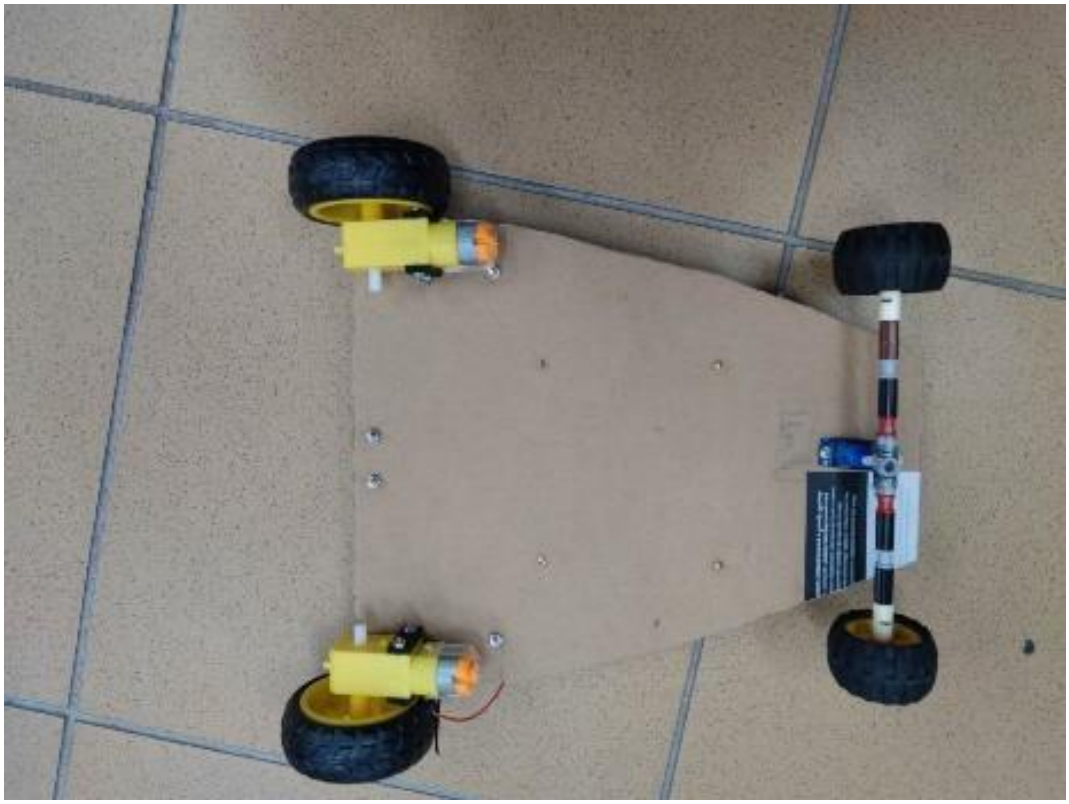


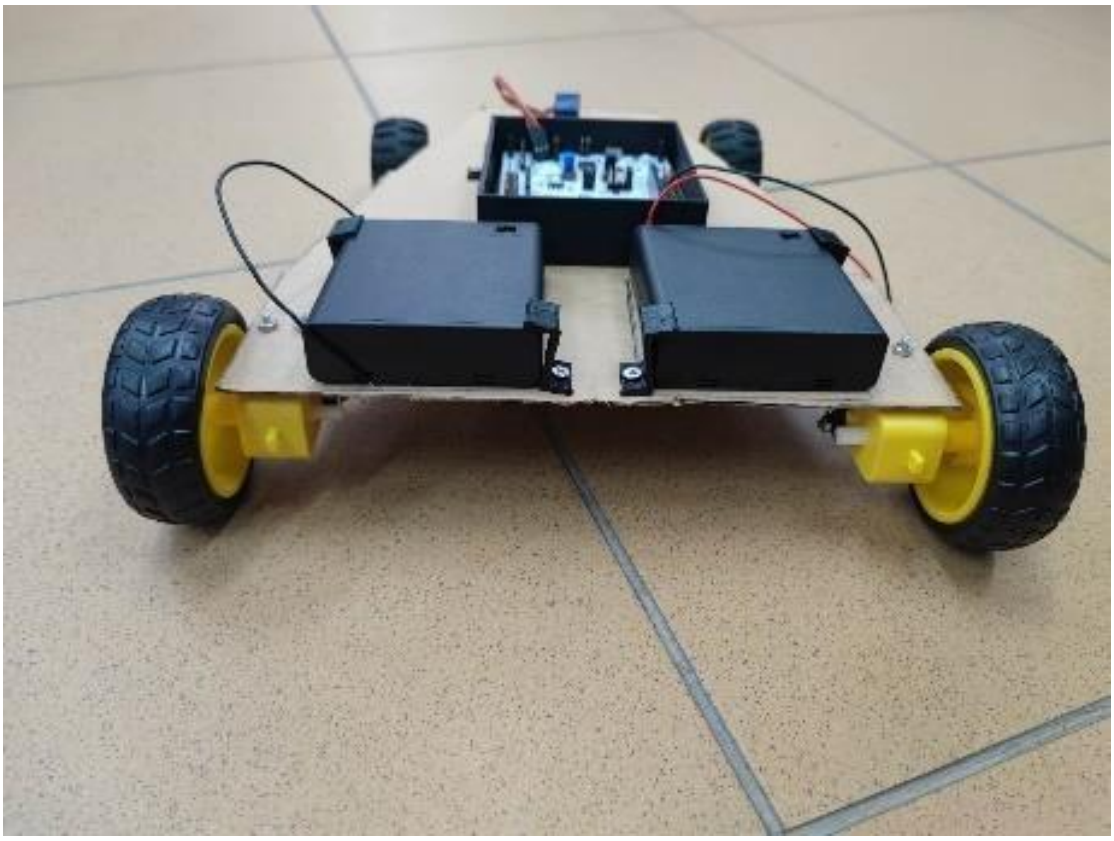
Był to strzał w dziesiątkę i pozwolił nam na wykończenie początkowego szkieletu. Do dokończenia i montażu zostały nam tylko śrubki i wkręty. Użyliśmy do tego śrub minimetrycznych o średnicy 3mm oraz wkrętów również o średnicy 3mm.

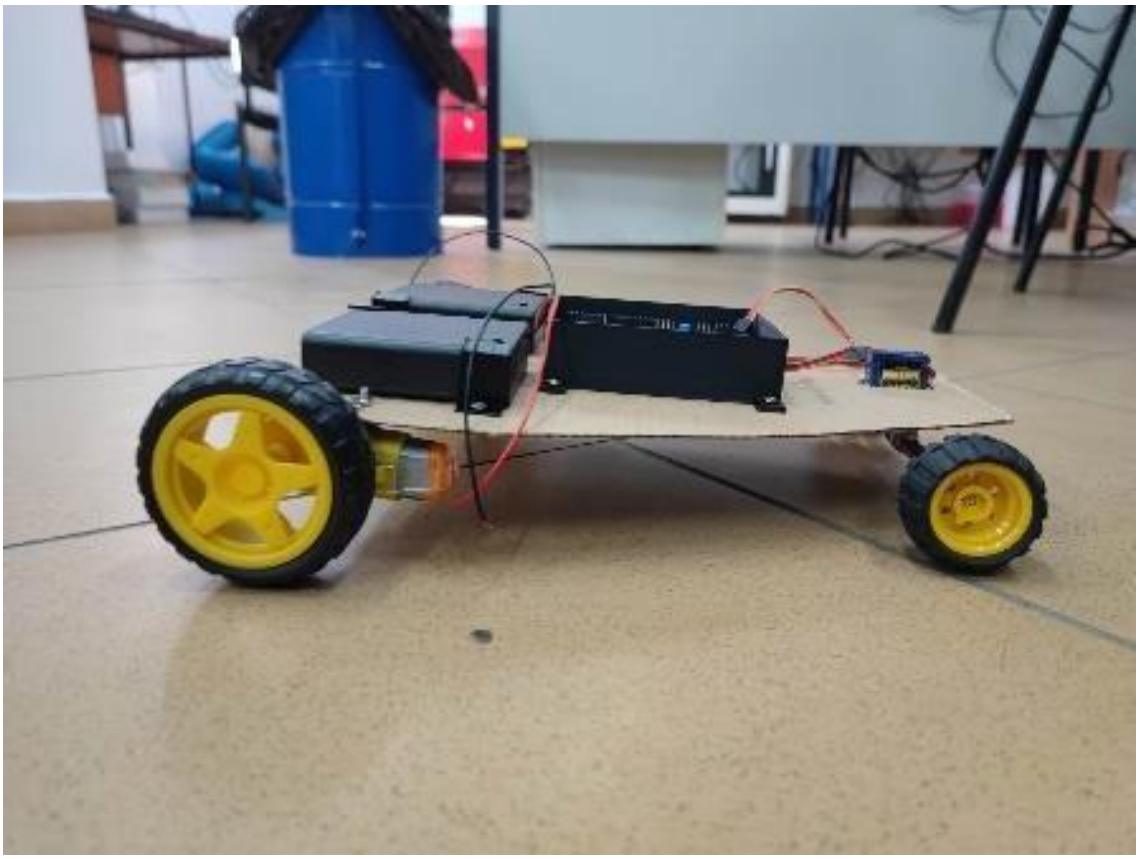
Cała konstrukcja po wykorzystaniu tego wszystkiego była stabilna oraz gotowa do jazdy. Efektem naszych prac jest przedstawiony w tym sprawozdaniu, wstępny wygląd naszej maszyny. Póki co jest to jednak prototypowa wersja która zostanie poprawiona w przyszłych tygodniach oraz rozszerzona o dodatkową funkcjonalność

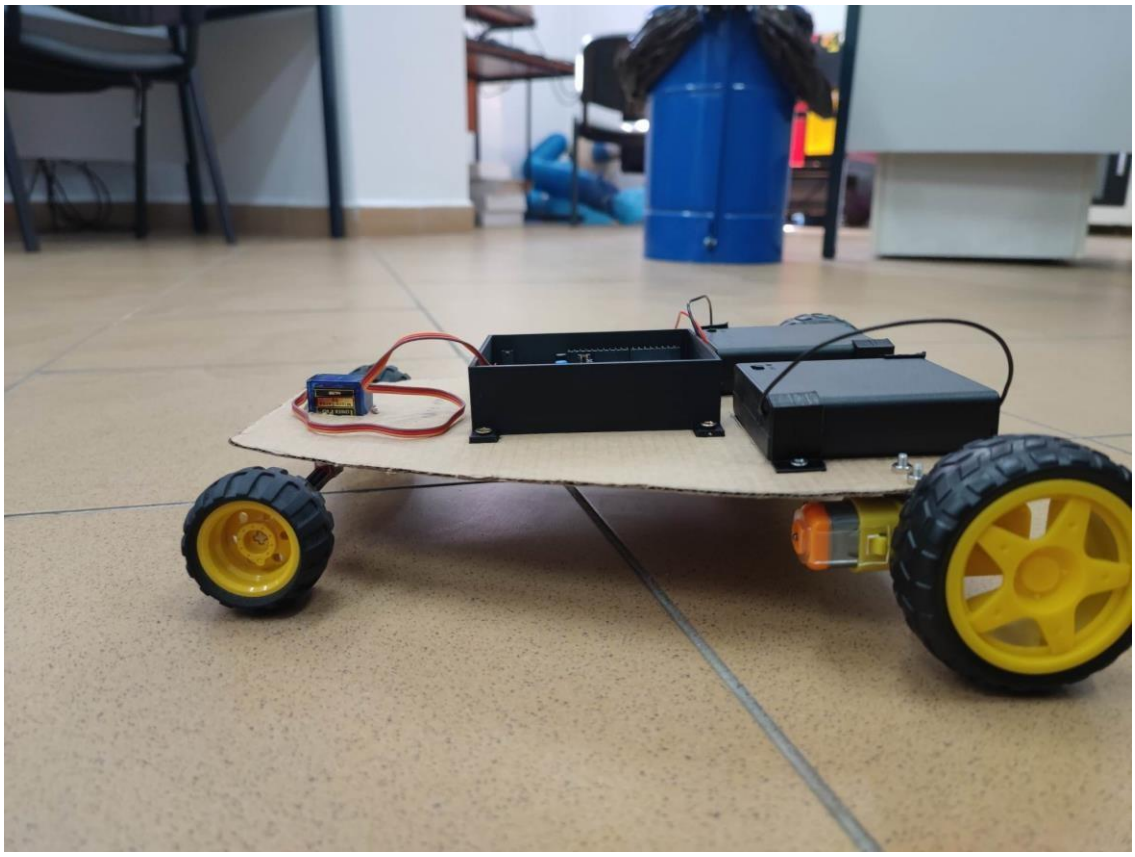
Zdjęcia pojazdu







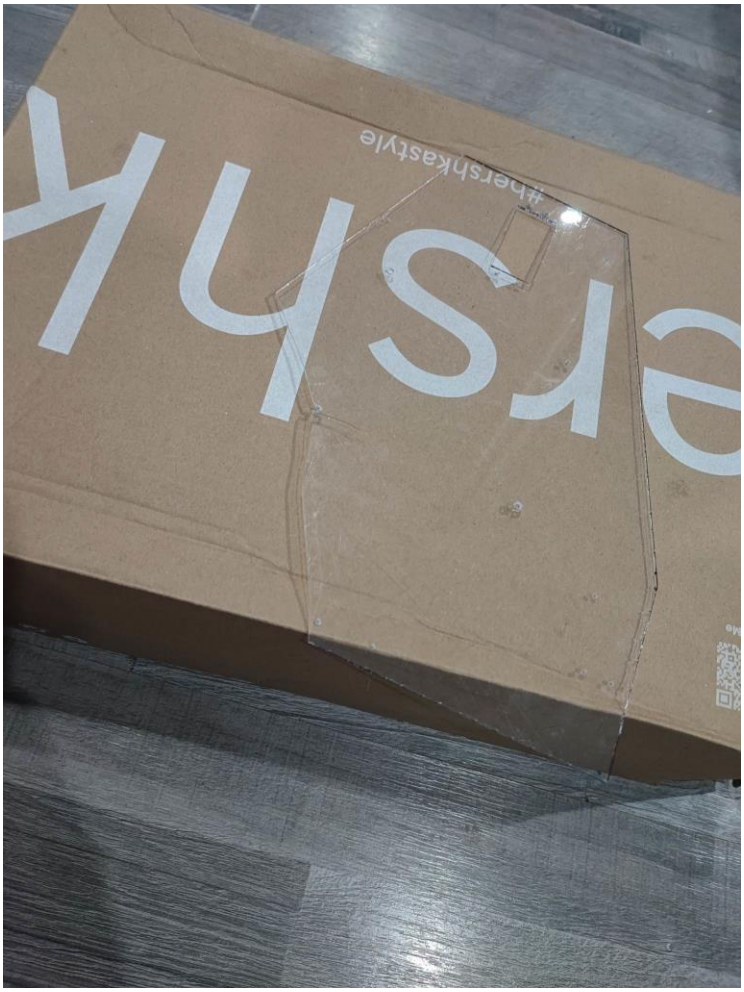




Przebieg pracy – 2 kamień milowy

Praca nad tą częścią projektu zaczęła się od wprowadzenia poprawek mechanicznych których nie udało się wprowadzić jeszcze w poprzedniej części. Na początku postanowiliśmy zmienić podstawę ponieważ karton nie był zbyt trwały. Pierwszy wybór padł na pleksi. Przy pomocy narzędzia stworzyliśmy projekt naszej podstawy i wycięliśmy go przy pomocy lasera. Jednak podstawka ta nie zdała próby dodatkowych modyfikacji….

I o to jak skończyła:

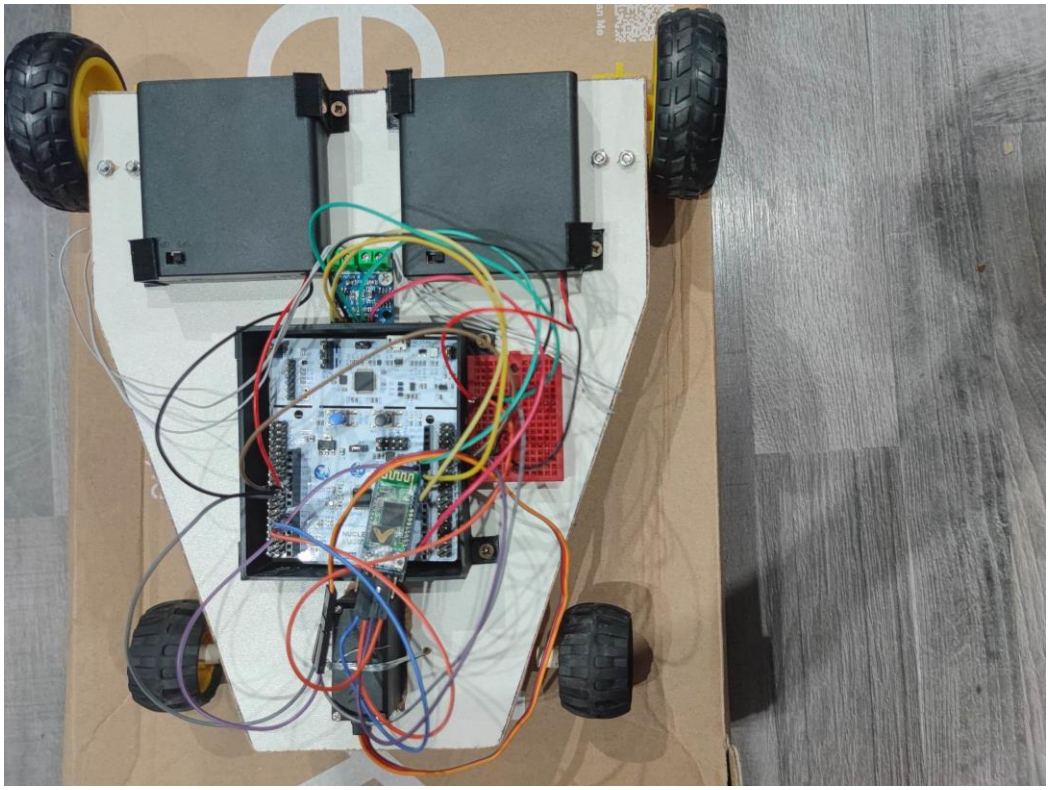


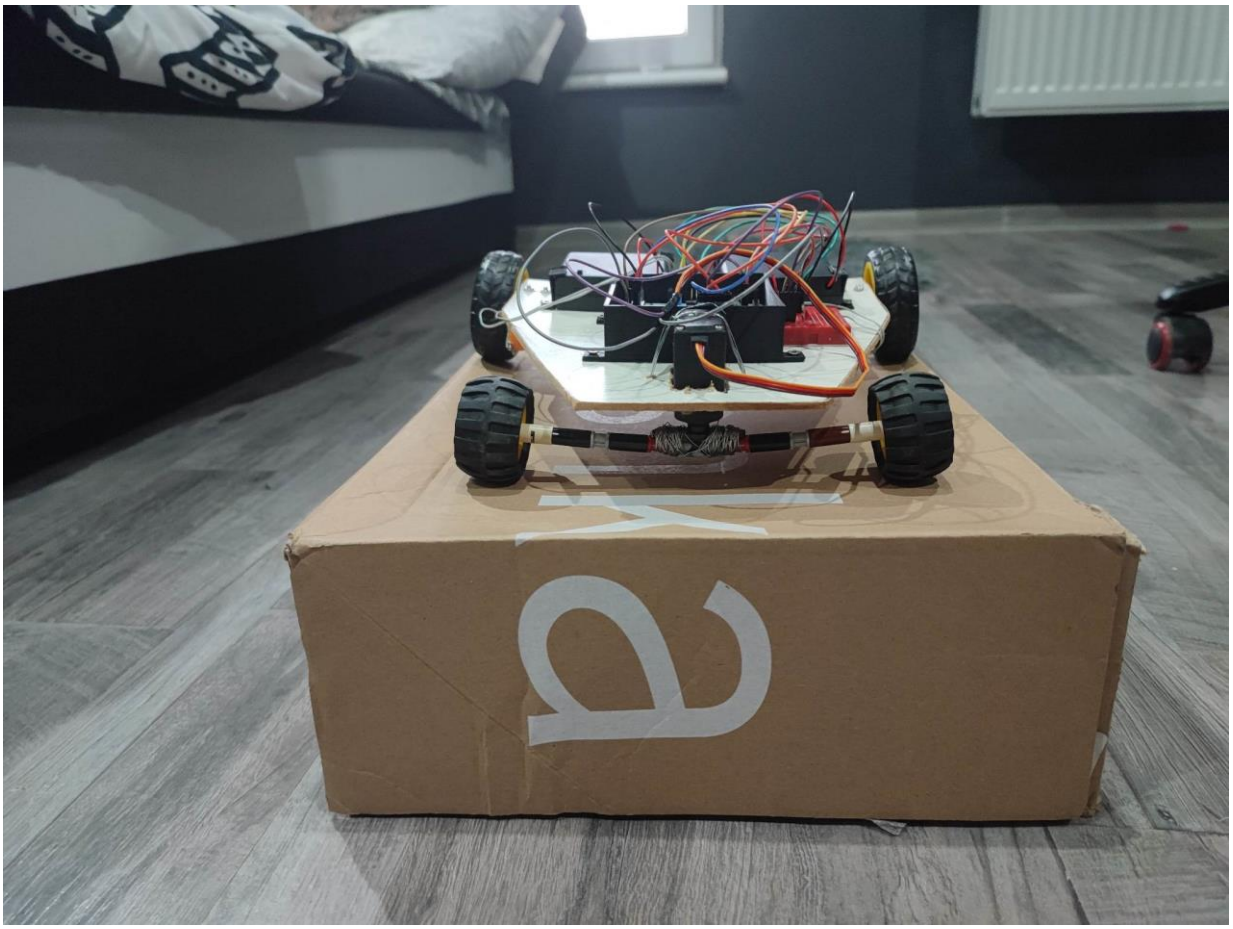
Ze względu na częste modyfikacje podstawy potrzebowaliśmy bardziej odpornego materiału. I wybór padł na sklejkę w postaci plecków od starej szafki które samodzielnie docięliśmy na wymiar. Dodatkowo poszerzyliśmy wyposażenie o mostek sterujący silnikami.

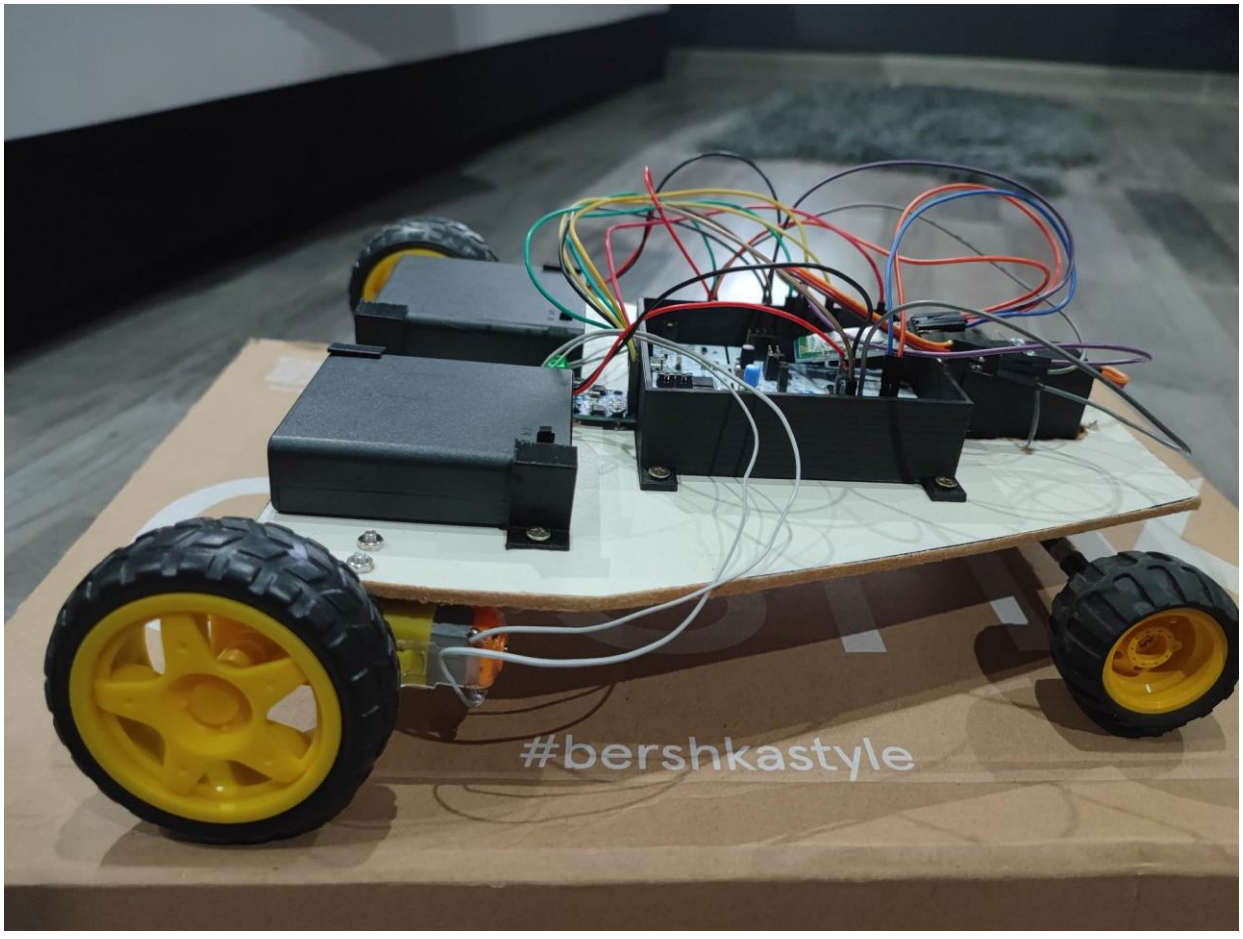
Moduł bluetooh w celu połączenia zdalnego a także breadboard w celu rozdzielenia masy i zasilania z obu źródeł zasilania. Wszystkie komponenty zostały połączone kablami lub zostały im dolutowane kable

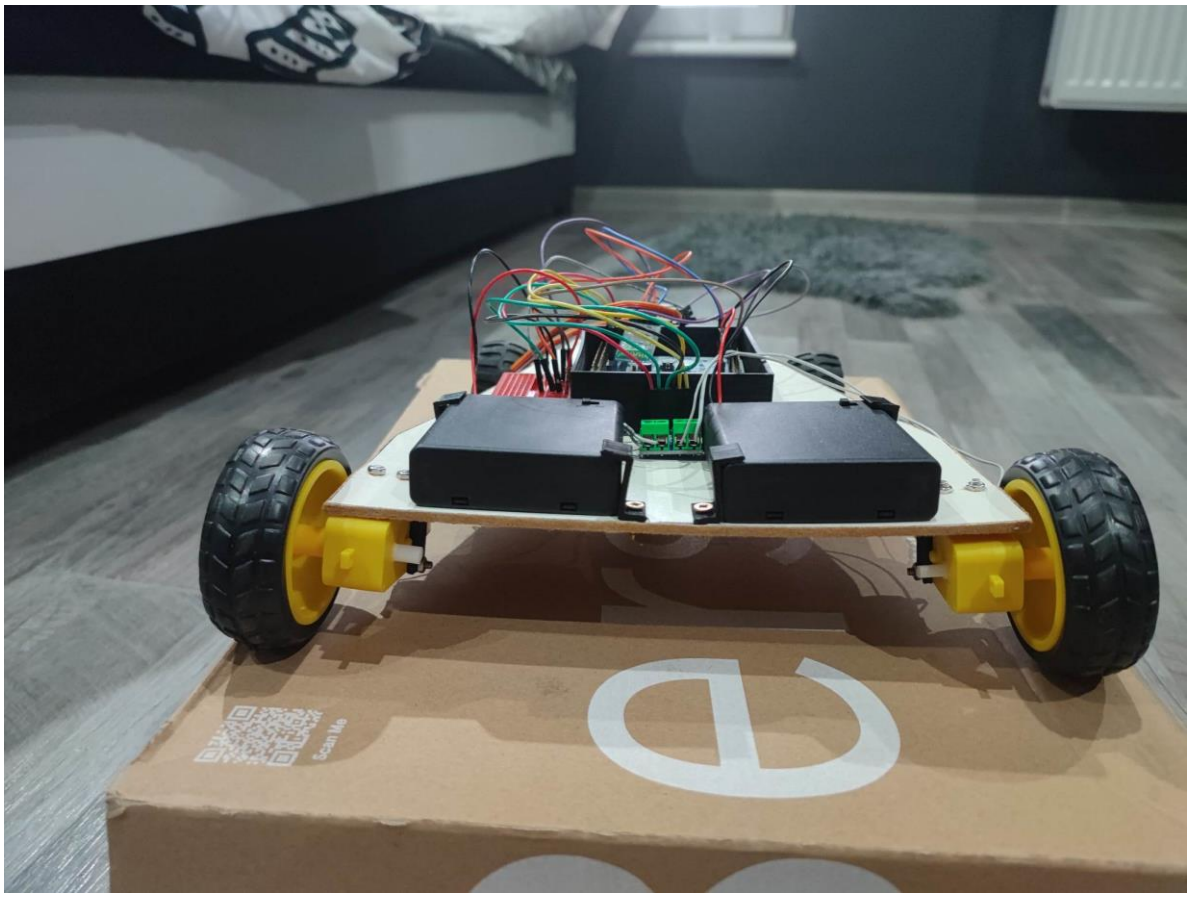
Zdecydowaliśmy się płytkę oraz moduł zasilać z jednego zestawu baterii a Serwo z osią skrętną oraz silniki drugim. Po sprawdzeniu potrzebnych nam pinów spięliśmy wszystko razem i pojazd był gotowy do programowania.

Zdjęcia pojazdu






#bershkastyle



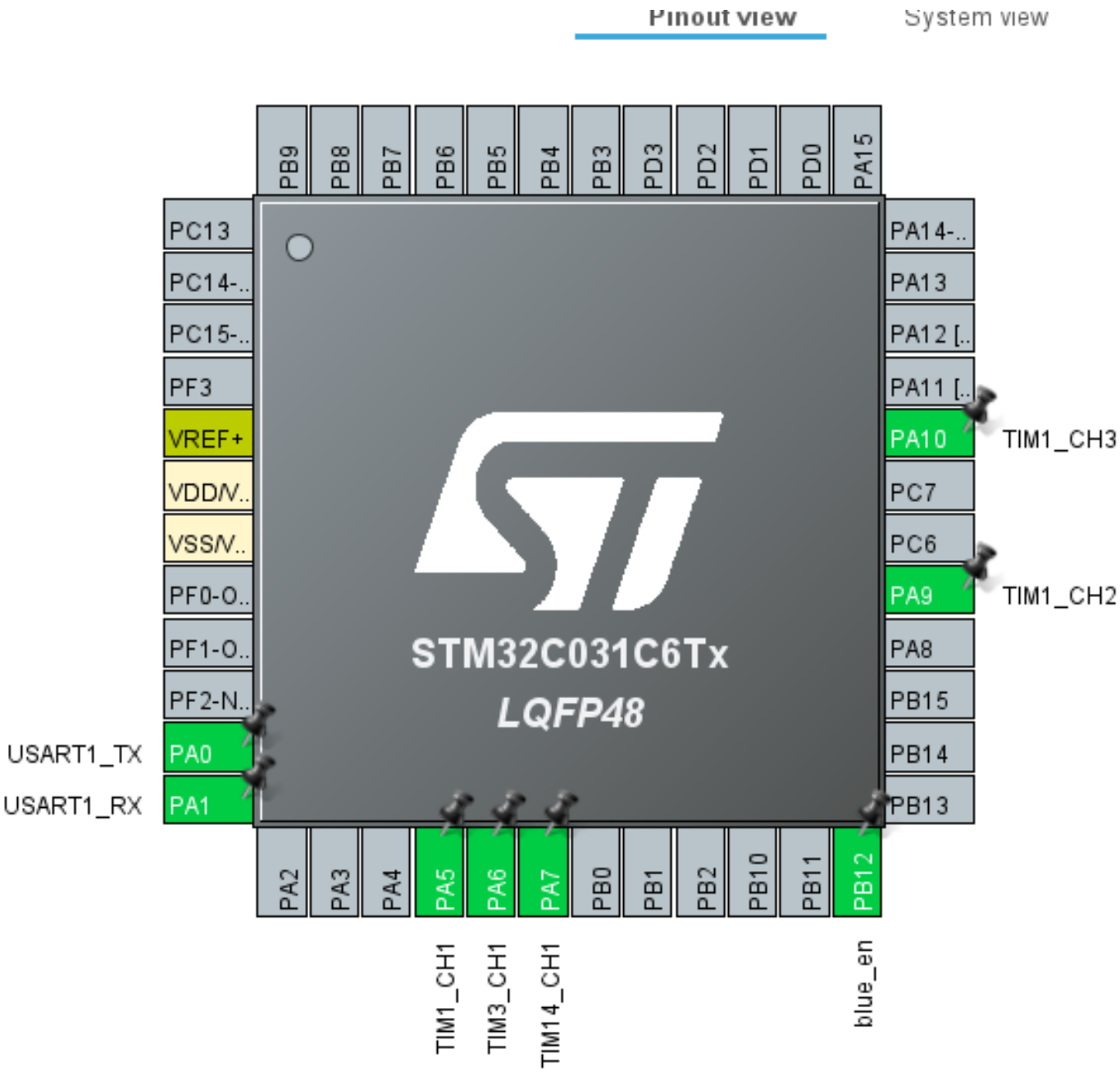
Stworzone oprogramowanie

Przyszła pora żeby zabrać się za oprogramowanie płytki. W tym wypadku postawiliśmy na póki co sterowanie przy pomocy uarta oraz modułu bluetooth. Silniki oraz serwo działają przy pomocy pwm i rozpędzają/ustawia się krokowo. Poniżej aplikacja oraz ustawienie pinów:

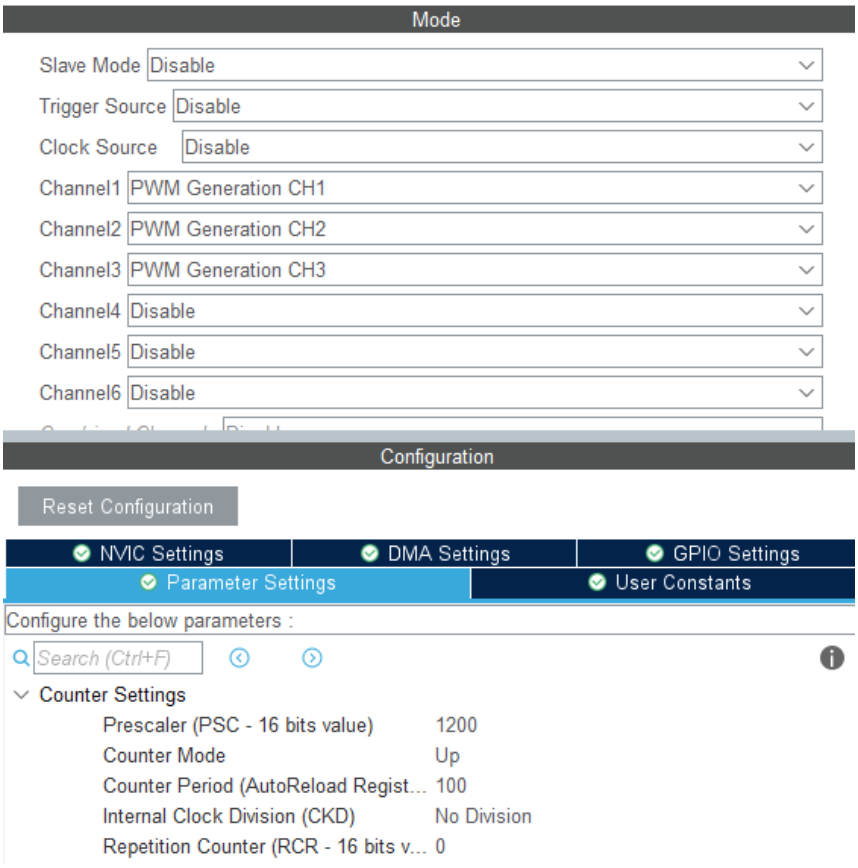
```
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM3_Init();
  MX_TIM1_Init();
  MX_TIM14_Init();
  MX_USART1_UART_Init();
  /* USER CODE BEGIN 2 */
  uint8_t buffer[1];


        HAL_TIM_PWM_Start(&htim3,TIM_CHANNEL_1);
        HAL_TIM_PWM_Start(&htim1,TIM_CHANNEL_3);
        HAL_TIM_PWM_Start(&htim1,TIM_CHANNEL_2);
        HAL_TIM_PWM_Start(&htim1,TIM_CHANNEL_1);
        HAL_TIM_PWM_Start(&htim14,TIM_CHANNEL_1);

        int i=62;
        int j=0;
        __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
       MX_USART1_UART_Init();
       buffer[0]=0;
        HAL_UART_Receive(&huart1, buffer, 1, 150);
                    if(buffer[0]=='w'){
                           HAL_UART_Transmit(&huart1,buffer,1,100);
                           __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,0);
                           __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,0);
```

```
        for(j=0;j<51;j++){
                __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_2,j);
                __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_3,j);
                HAL_Delay(10);
        }
 }

if(buffer[0]=='s'){
        HAL_UART_Transmit(&huart1,buffer,1,100);

        __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_2,0);
        __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_3,0);

        for(j=0;j<51;j++){
                __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,j);
                __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,j);
                HAL_Delay(10);
        }
 }

if(buffer[0]=='d'){
        HAL_UART_Transmit(&huart1,buffer,1,100);

        for(i; i<68; i++){
        __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
        HAL_Delay(50);
        }
 }

if(buffer[0]=='a'){
        HAL_UART_Transmit(&huart1,buffer,1,100);
        for(i; i>56; i--){
                        __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                        HAL_Delay(50);
                }
 }

if(buffer[0]=='r'){
        HAL_UART_Transmit(&huart1,buffer,1,100);

        if(i>60){
                for(i; i>62; i--){
                                __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                                HAL_Delay(50);
                        }
        }
        else{
                for(i; i<62; i++){
                                __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                                HAL_Delay(50);
                        }
        }
 }

 if(buffer[0]=='n'){
        HAL_UART_Transmit(&huart1,buffer,1,100);
                __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_2,0);
                __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_3,0);
                __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,0);
                __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,0);
        }
```

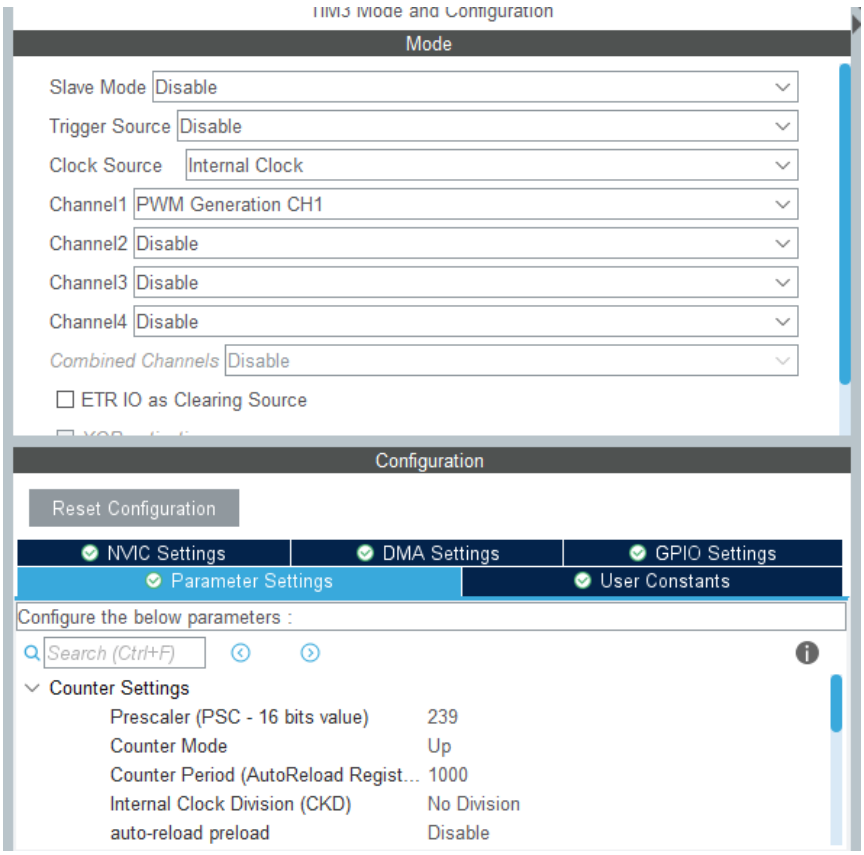
```
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```



Silniki

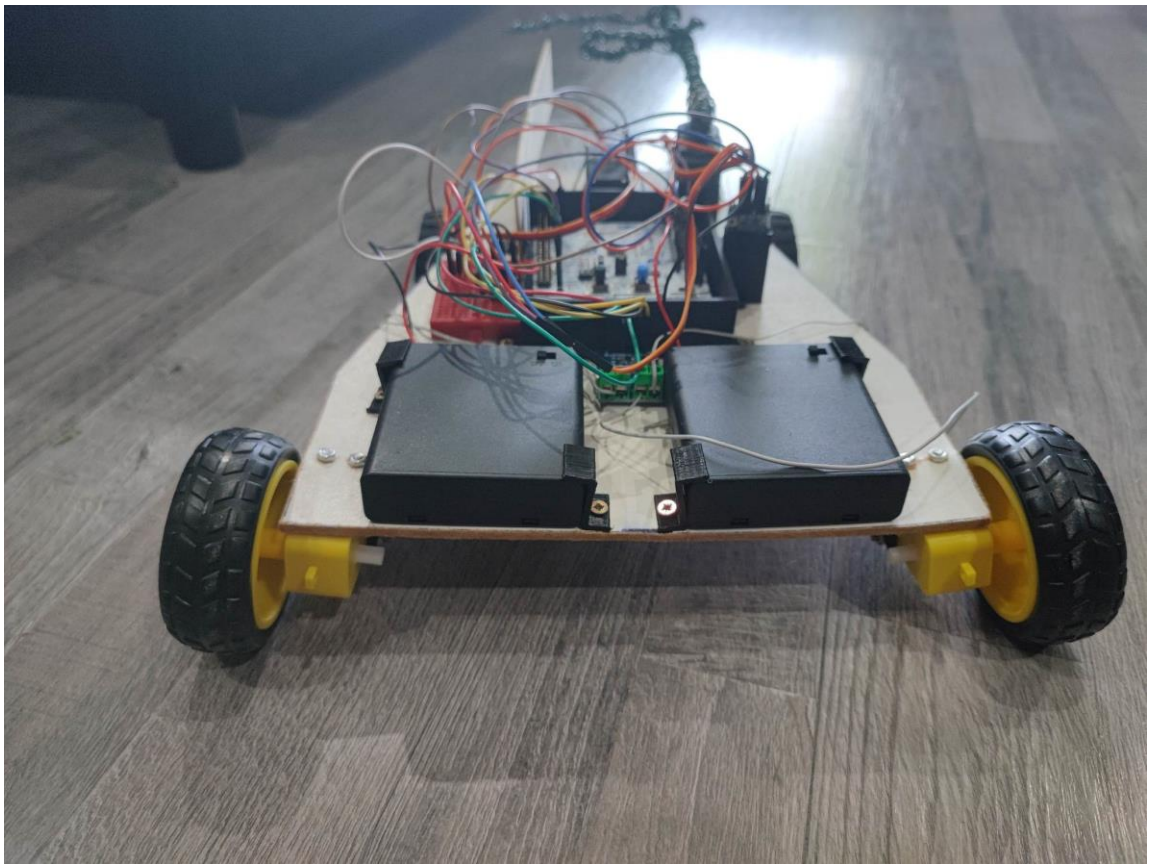


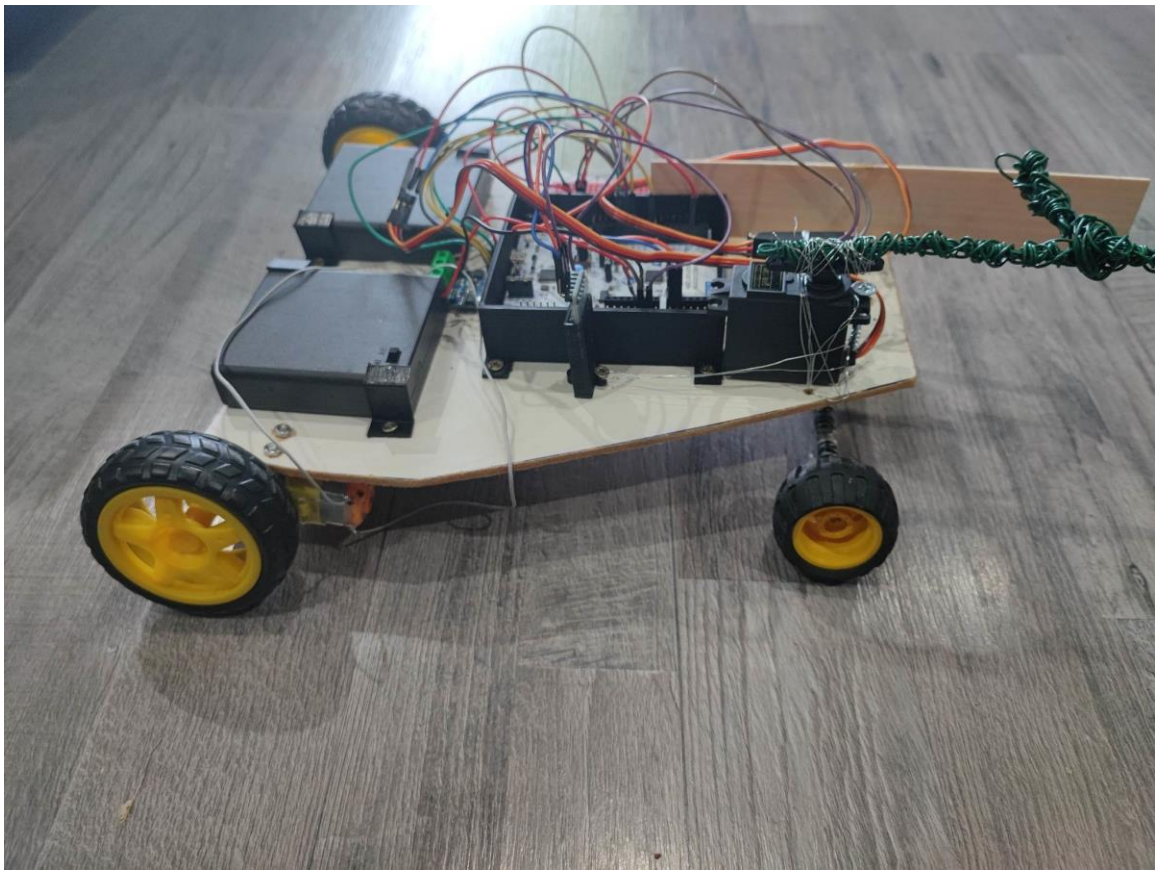
Serwo

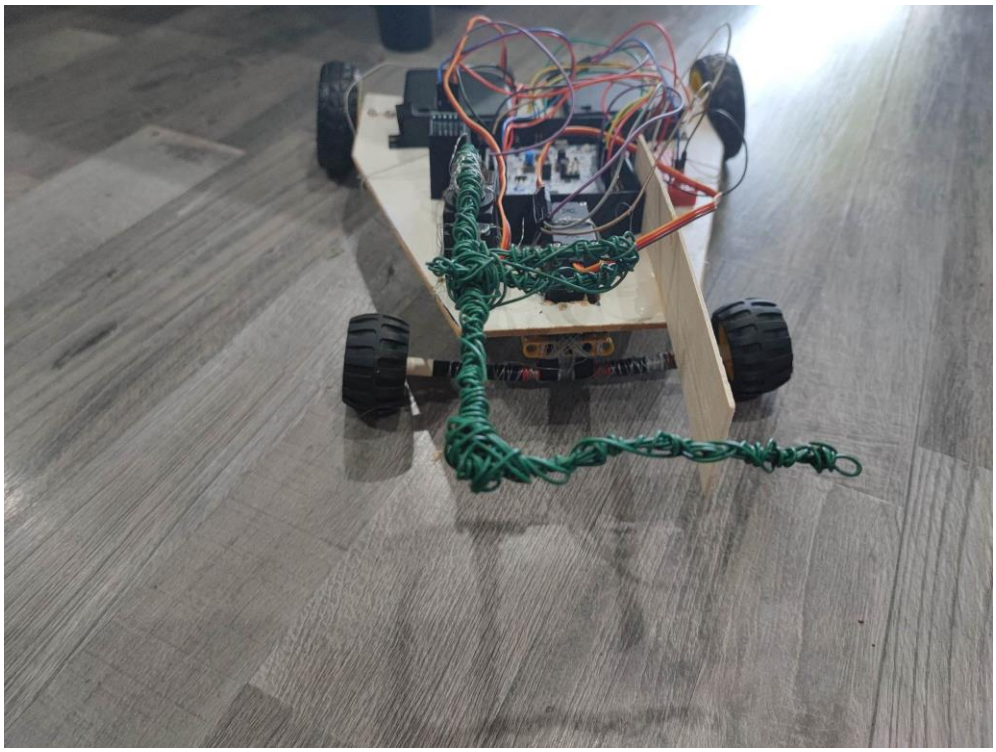


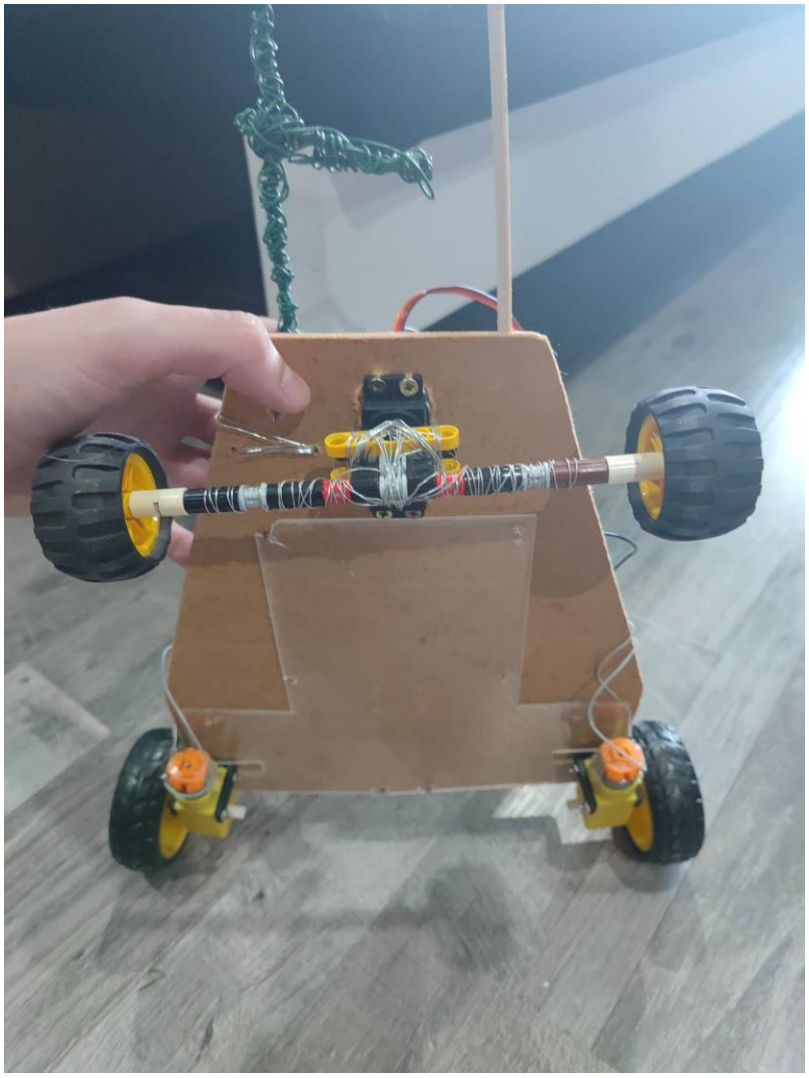
Przebieg pracy – 3 kamień milowy

Praca nad tą częścią projektu zaczęła się od zastanowienia nad kolejną funkcjonalnością dla naszego robota, mieliśmy już oś skrętną ale chcieliśmy go udoskonalić żeby potrafił robić jeszcze coś. Wtedy wpadliśmy na pomysł chwytaka ale wcześniej musieliśmy udoskonalić sam pojazd. To znaczy wesprzeć podstawkę oraz podnieść przednie zawieszenie oto jak prezentują się wprowadzone modyfikacje :









Jak widać pojazd również został wyposażony w dodatkowe serwo oraz 2 ramiona jedno dynamiczne do zatrzaskiwania oraz drugie statyczne od przytrzymywania

Cała funkcjonalność została pokazana na video chociaż dalej jest udoskonalana

W międzyczasie udoskonaliliśmy również sterowanie wykonując aplikacje na androida :

Aplikacja posiada przyciski do połączenia i rozłączenia, tryby prędkości, sterowanie chwytakiem, oraz sterowanie jazdą

Poniżej poprawiony kod stm oraz android studio :

```
package com.example.robot
import android.os.Handler
import android.bluetooth.BluetoothAdapter
import android.bluetooth.BluetoothDevice
import android.bluetooth.BluetoothSocket
import android.content.Intent
import android.content.pm.PackageManager
import android.os.Bundle
import android.widget.Button
import android.widget.Toast
import androidx.activity.ComponentActivity
import java.io.IOException
import java.io.OutputStream
import java.util.UUID
import android.Manifest
import android.view.MotionEvent


class MainActivity : ComponentActivity() {
    private val DEVICE_ADDRESS = "00:18:E4:40:00:06" // Zmień na rzeczywisty adres
urządzenia
    private val SERIAL_UUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB")

    private val LOCATION_PERMISSION_REQUEST_CODE = 1001

    private var predkosc: Int = 1;

    private var bluetoothAdapter: BluetoothAdapter? = null
    private var device: BluetoothDevice? = null
    private var socket: BluetoothSocket? = null
    private var outputStream: OutputStream? = null

    private var w: Button? = null
    private var s: Button? = null
    private var a: Button? = null
    private var d: Button? = null

    private var pre1: Button? = null
    private var pre2: Button? = null
    private var pre3: Button? = null
    private var pre4: Button? = null

    private var pusc: Button? = null
    private var chwyc: Button? = null

    private var isButtonpre1Pressed = false
    private var isButtonpre2Pressed = false
    private var isButtonpre3Pressed = false
    private var isButtonpre4Pressed = false

    private var isButtonWPressed = false
    private var isButtonSPressed = false
    private var isButtonAPressed = false
    private var isButtonDPressed = false

    private var isButtonPUSCPressed = false
    private var isButtonCHWYCPressed = false
```

```
    private var connectButton: Button? = null
    private var disconnectButton: Button? = null

    private var isSendingData = false

    private var sendWRunnable: Runnable? = null
    private var sendSRunnable: Runnable? = null
    private var sendDRunnable: Runnable? = null
    private var sendARunnable: Runnable? = null
    private var send5Runnable: Runnable? = null
    private var send6Runnable: Runnable? = null
    private var send7Runnable: Runnable? = null
    private var send8Runnable: Runnable? = null


    private val handler = Handler()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.main_activity)

        w = findViewById(R.id.w)
        s = findViewById(R.id.s)
        a = findViewById(R.id.a)
        d = findViewById(R.id.d)

        pre1 = findViewById(R.id.pre1)
        pre2 = findViewById(R.id.pre2)
        pre3 = findViewById(R.id.pre3)
        pre4 = findViewById(R.id.pre4)

        pusc = findViewById(R.id.pusc)
        chwyc = findViewById(R.id.chwyc)

        connectButton = findViewById(R.id.connectButton)
        disconnectButton = findViewById(R.id.disconnectButton)

        fun startSendingWRepeatedly() {
            sendWRunnable = object : Runnable {
                override fun run() {
                    if (isButtonWPressed) {
                        sendData("w")
                        handler.postDelayed(this, 111) // Ustaw opóźnienie pomiędzy
kolejnymi sygnałami "s"
                    }
                }
            }
            handler.postDelayed(sendWRunnable as Runnable, 111) // Pierwszy sygnał "s"
zostanie wysłany po 100 ms
        }

        fun stopSendingWRepeatedly() {
            sendWRunnable?.let {
                handler.removeCallbacks(it)
                sendWRunnable = null
            }
        }

        fun startSendingSRepeatedly() {
            sendSRunnable = object : Runnable {
                override fun run() {
                    if (isButtonSPressed) {
                        sendData("s")
                        handler.postDelayed(this, 111) // Ustaw opóźnienie pomiędzy
kolejnymi sygnałami "s"
                    }
                }
```

```
            }
            handler.postDelayed(sendSRunnable as Runnable, 111) // Pierwszy sygnał "s"
zostanie wysłany po 100 ms
        }

        fun stopSendingSRepeatedly() {
            sendSRunnable?.let {
                handler.removeCallbacks(it)
                sendSRunnable = null
            }
        }

        fun startSendingDRepeatedly() {
            sendDRunnable = object : Runnable {
                override fun run() {
                    if (isButtonDPressed) {
                        sendData("d")
                        handler.postDelayed(this, 111) // Ustaw opóźnienie pomiędzy
kolejnymi sygnałami "s"
                    }
                }
            }
            handler.postDelayed(sendDRunnable as Runnable, 111) // Pierwszy sygnał "s"
zostanie wysłany po 100 ms
        }

        fun stopSendingDRepeatedly() {
            sendDRunnable?.let {
                handler.removeCallbacks(it)
                sendDRunnable = null
            }
        }

        fun startSendingARepeatedly() {
            sendARunnable = object : Runnable {
                override fun run() {
                    if (isButtonAPressed) {
                        sendData("a")
                        handler.postDelayed(this, 111) // Ustaw opóźnienie pomiędzy
kolejnymi sygnałami "s"
                    }
                }
            }
            handler.postDelayed(sendARunnable as Runnable, 111) // Pierwszy sygnał "s"
zostanie wysłany po 100 ms
        }

        fun stopSendingARepeatedly() {
            sendARunnable?.let {
                handler.removeCallbacks(it)
                sendARunnable = null
            }
        }


        fun startSending5Repeatedly() {
            send5Runnable = object : Runnable {
                override fun run() {
                        sendData("5")
                        handler.postDelayed(this, 111) // Ustaw opóźnienie pomiędzy
kolejnymi sygnałami "s"
                }
            }
            handler.postDelayed(send5Runnable as Runnable, 111) // Pierwszy sygnał "s"
zostanie wysłany po 100 ms
        }

        fun stopSending5Repeatedly() {
```

```
            send5Runnable?.let {
                handler.removeCallbacks(it)
                send5Runnable = null
            }
        }

        fun startSending6Repeatedly() {
            send6Runnable = object : Runnable {
                override fun run() {
                    sendData("6")
                    handler.postDelayed(this, 111) // Ustaw opóźnienie pomiędzy
kolejnymi sygnałami "s"
                }
            }
            handler.postDelayed(send6Runnable as Runnable, 111) // Pierwszy sygnał "s"
zostanie wysłany po 100 ms
        }

        fun stopSending6Repeatedly() {
            send6Runnable?.let {
                handler.removeCallbacks(it)
                send6Runnable = null
            }
        }

        fun startSending7Repeatedly() {
            send7Runnable = object : Runnable {
                override fun run() {
                    sendData("7")
                    handler.postDelayed(this, 111) // Ustaw opóźnienie pomiędzy
kolejnymi sygnałami "s"
                }
            }
            handler.postDelayed(send7Runnable as Runnable, 111) // Pierwszy sygnał "s"
zostanie wysłany po 100 ms
        }

        fun stopSending7Repeatedly() {
            send7Runnable?.let {
                handler.removeCallbacks(it)
                send7Runnable = null
            }
        }

        fun startSending8Repeatedly() {
            send8Runnable = object : Runnable {
                override fun run() {
                    sendData("8")
                    handler.postDelayed(this, 111) // Ustaw opóźnienie pomiędzy
kolejnymi sygnałami "s"
                }
            }
            handler.postDelayed(send8Runnable as Runnable, 111) // Pierwszy sygnał "s"
zostanie wysłany po 100 ms
        }

        fun stopSending8Repeatedly() {
            send8Runnable?.let {
                handler.removeCallbacks(it)
                send8Runnable = null
            }
        }
        fun check() {
            if(isButtonWPressed && isButtonAPressed){
                stopSendingARepeatedly()
                stopSendingSRepeatedly()
                stopSendingWRepeatedly()
                stopSendingDRepeatedly()
```

```
            stopSending6Repeatedly()
            stopSending7Repeatedly()
            stopSending8Repeatedly()
            sendData("5")
            startSending5Repeatedly()
        }
        else if(isButtonWPressed && isButtonDPressed){
            stopSendingARepeatedly()
            stopSendingSRepeatedly()
            stopSendingWRepeatedly()
            stopSendingDRepeatedly()
            stopSending5Repeatedly()
            stopSending7Repeatedly()
            stopSending8Repeatedly()
            sendData("6")
            startSending6Repeatedly()
        }
        else if(isButtonSPressed && isButtonAPressed){
            stopSendingARepeatedly()
            stopSendingSRepeatedly()
            stopSendingWRepeatedly()
            stopSendingDRepeatedly()
            stopSending5Repeatedly()
            stopSending6Repeatedly()
            stopSending8Repeatedly()
            sendData("7")
            startSending7Repeatedly()
        }
        else if(isButtonSPressed && isButtonDPressed){

            stopSendingARepeatedly()
            stopSendingSRepeatedly()
            stopSendingWRepeatedly()
            stopSendingDRepeatedly()
            stopSending5Repeatedly()
            stopSending6Repeatedly()
            stopSending7Repeatedly()
            sendData("8")
            startSending8Repeatedly()
        }
        else if(isButtonSPressed){
            stopSendingARepeatedly()
            stopSendingWRepeatedly()
            stopSendingDRepeatedly()
            stopSending5Repeatedly()
            stopSending6Repeatedly()
            stopSending7Repeatedly()
            stopSending8Repeatedly()
            sendData("s")
            startSendingSRepeatedly()

        }
        else if(isButtonWPressed){
            stopSendingARepeatedly()
            stopSendingSRepeatedly()
            stopSendingDRepeatedly()
            stopSending5Repeatedly()
            stopSending6Repeatedly()
            stopSending7Repeatedly()
            stopSending8Repeatedly()
            sendData("w")
            startSendingWRepeatedly()
        }
        else if(isButtonDPressed){
            startSendingDRepeatedly()
            stopSendingSRepeatedly()
            stopSendingWRepeatedly()
            stopSending5Repeatedly()
```

```
                stopSending6Repeatedly()
                stopSending7Repeatedly()
                stopSending8Repeatedly()
                sendData("d")
                stopSendingARepeatedly()
            }
            else if(isButtonAPressed){
                stopSendingSRepeatedly()
                stopSendingWRepeatedly()
                stopSendingDRepeatedly()
                stopSending5Repeatedly()
                stopSending6Repeatedly()
                stopSending7Repeatedly()
                stopSending8Repeatedly()
                sendData("a")
                startSendingARepeatedly()
            }
            else{
                stopSendingARepeatedly()
                stopSendingSRepeatedly()
                stopSendingWRepeatedly()
                stopSendingDRepeatedly()
                stopSending5Repeatedly()
                stopSending6Repeatedly()
                stopSending7Repeatedly()
                stopSending8Repeatedly()
            }
        }

        w?.setOnTouchListener { _, event ->
            when (event.action) {
                MotionEvent.ACTION_DOWN -> {
                    isButtonWPressed = true
                    check();
                }
                MotionEvent.ACTION_UP -> {
                    if (isButtonWPressed) {
                        isButtonWPressed = false
                    }
                    check();
                }
            }
            true
        }

        s?.setOnTouchListener { _, event ->
            when (event.action) {
                MotionEvent.ACTION_DOWN -> {
                    isButtonSPressed = true
                    check();
                }
                MotionEvent.ACTION_UP -> {
                    if (isButtonSPressed) {
                        isButtonSPressed = false
                    }
                    check();
                }
            }
            true
        }

        a?.setOnTouchListener { _, event ->
            when (event.action) {
                MotionEvent.ACTION_DOWN -> {
                    isButtonAPressed = true
                    check();
                }
                MotionEvent.ACTION_UP -> {
```

```
                if (isButtonAPressed) {
                    isButtonAPressed = false
                }
                check();
            }
        }
        true
    }

    d?.setOnTouchListener { _, event ->
        when (event.action) {
            MotionEvent.ACTION_DOWN -> {
                isButtonDPressed = true
                check();
            }
            MotionEvent.ACTION_UP -> {
                if (isButtonDPressed) {
                    isButtonDPressed = false
                    check();
                }
            }
        }
        true
    }


    pre1?.setOnClickListener {
        if (!isButtonpre1Pressed) {
            predkosc=1;
            sendData("1")
            isButtonpre1Pressed = true
            pre1?.isEnabled = false
            pre2?.isEnabled = true
            pre3?.isEnabled = true
            pre4?.isEnabled = true


            isButtonpre2Pressed = false
            isButtonpre3Pressed = false
            isButtonpre4Pressed = false
        }
    }

    pre2?.setOnClickListener {
        if (!isButtonpre2Pressed) {
            predkosc=2;
            sendData("2")
            isButtonpre2Pressed = true
            pre1?.isEnabled = true
            pre2?.isEnabled = false
            pre3?.isEnabled = true
            pre4?.isEnabled = true

            isButtonpre1Pressed = false
            isButtonpre3Pressed = false
            isButtonpre4Pressed = false
        }
    }

    pre3?.setOnClickListener {
        if (!isButtonpre3Pressed) {
            predkosc=3;
            sendData("3")
            isButtonpre3Pressed = true
            pre1?.isEnabled = true
            pre2?.isEnabled = true
            pre3?.isEnabled = false
            pre4?.isEnabled = true
```

```
                isButtonpre1Pressed = false
                isButtonpre2Pressed = false
                isButtonpre4Pressed = false
            }
        }

        pre4?.setOnClickListener {
            if (!isButtonpre4Pressed) {
                predkosc=4;
                sendData("4")
                isButtonpre4Pressed = true
                pre1?.isEnabled = true
                pre2?.isEnabled = true
                pre3?.isEnabled = true
                pre4?.isEnabled = false

                isButtonpre2Pressed = false
                isButtonpre3Pressed = false
                isButtonpre1Pressed = false
            }
        }

        chwyc?.setOnClickListener{
            if(!isButtonCHWYCPressed){
                sendData("h")
                isButtonCHWYCPressed = true
                chwyc?.isEnabled = false
                pusc?.isEnabled = true
                isButtonPUSCPressed = false
            }
        }

        pusc?.setOnClickListener{
            if(!isButtonPUSCPressed){
                sendData("p")
                isButtonCHWYCPressed = false
                chwyc?.isEnabled = true
                pusc?.isEnabled = false
                isButtonPUSCPressed = true
            }
        }


        connectButton?.setOnClickListener {
            connectToDevice()
        }

        disconnectButton?.setOnClickListener {
            disconnectFromDevice()
        }

        bluetoothAdapter = BluetoothAdapter.getDefaultAdapter()
        if (bluetoothAdapter == null) {
            // Sprawdź, czy urządzenie obsługuje Bluetooth
            Toast.makeText(this, "Bluetooth nie jest obsługiwane na tym urządzeniu",
Toast.LENGTH_SHORT).show()
            finish()
        }


    }

    private fun sendData(data: String) {
        val bluetoothSocket = socket as? BluetoothSocket
        if (bluetoothSocket != null && bluetoothSocket.isConnected) {
            try {
                outputStream?.write(data.toByteArray())
```

```
                Toast.makeText(this, "Dane wysłane: $data", Toast.LENGTH_SHORT).show()
            } catch (e: IOException) {
                Toast.makeText(this, "Błąd podczas wysyłania danych: ${e.message}",
Toast.LENGTH_SHORT).show()
            }
        } else {
            Toast.makeText(this, "Brak połączenia z urządzeniem",
Toast.LENGTH_SHORT).show()
        }
    }

    override fun onResume() {
        super.onResume()
        if (!bluetoothAdapter!!.isEnabled) {
            // Sprawdź, czy Bluetooth jest włączony
            val enableBtIntent = Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE)
            startActivityForResult(enableBtIntent, 1)
        } else {
            if (checkSelfPermission(Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {
                // Masz już uprawnienia do lokalizacji, kontynuuj
                connectToDevice()
            } else {
                // Poproś użytkownika o uprawnienia do lokalizacji
                requestLocationPermission()
            }
        }
    }

    private fun connectToDevice() {
        val remoteDevice = bluetoothAdapter?.getRemoteDevice(DEVICE_ADDRESS)
        if (remoteDevice != null) {
            try {
                socket = remoteDevice.createRfcommSocketToServiceRecord(SERIAL_UUID)
                val bluetoothSocket = socket as? BluetoothSocket
                if (bluetoothSocket != null) {
                    if (checkSelfPermission(Manifest.permission.ACCESS_FINE_LOCATION)
== PackageManager.PERMISSION_GRANTED) {
                        bluetoothSocket.connect()
                        outputStream = bluetoothSocket.outputStream
                        Toast.makeText(this, "Połączono z urządzeniem:
${remoteDevice.name}", Toast.LENGTH_SHORT).show()
                        updateButtonsState(true)
                    } else {
                        if
(shouldShowRequestPermissionRationale(Manifest.permission.ACCESS_FINE_LOCATION)) {
                            // Wyświetl dodatkowe informacje dla użytkownika dotyczące
uprawnienia lokalizacji
                            Toast.makeText(this, "Potrzebujemy uprawnienia do
lokalizacji w celu nawiązania połączenia Bluetooth.", Toast.LENGTH_SHORT).show()
                        }

requestPermissions(arrayOf(Manifest.permission.ACCESS_FINE_LOCATION),
LOCATION_PERMISSION_REQUEST_CODE)
                    }
                }
            } catch (e: IOException) {
                Toast.makeText(this, "Błąd podczas nawiązywania połączenia:
${e.message}", Toast.LENGTH_SHORT).show()
            } catch (e: SecurityException) {
                Toast.makeText(this, "Brak uprawnień do lokalizacji: ${e.message}",
Toast.LENGTH_SHORT).show()
            } catch (e: Exception) {
                Toast.makeText(this, "Wystąpił błąd: ${e.message}",
Toast.LENGTH_SHORT).show()
            }
        }
    }
```

```
    private fun disconnectFromDevice() {
        try {
            if (socket != null && socket!!.isConnected) {
                outputStream!!.close()
                socket!!.close()
                Toast.makeText(this, "Rozłączono z urządzeniem",
Toast.LENGTH_SHORT).show()
                updateButtonsState(false)
            }
        } catch (e: IOException) {
            e.printStackTrace()
        }
    }

    private fun requestLocationPermission() {
        requestPermissions(arrayOf(Manifest.permission.ACCESS_FINE_LOCATION),
LOCATION_PERMISSION_REQUEST_CODE)
    }

    override fun onRequestPermissionsResult(
        requestCode: Int,
        permissions: Array<out String>,
        grantResults: IntArray
    ) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults)
        if (requestCode == LOCATION_PERMISSION_REQUEST_CODE) {
            if (grantResults.isNotEmpty() && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                // Uprawnienia do lokalizacji zostały udzielone, kontynuuj
                connectToDevice()
            } else {
                // Uprawnienia do lokalizacji zostały odrzucone, obsłuż to odpowiednio
(np. poinformuj użytkownika)
                Toast.makeText(this, "Aplikacja wymaga uprawnień do lokalizacji",
Toast.LENGTH_SHORT).show()
            }
        }
    }

    private fun updateButtonsState(connected: Boolean) {
        connectButton?.isEnabled = !connected
        disconnectButton?.isEnabled = connected
    }
}
```

```c
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2023 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-IS.
  *
  ******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */

/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */
/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/

TIM_HandleTypeDef htim1;
TIM_HandleTypeDef htim3;
TIM_HandleTypeDef htim14;
TIM_HandleTypeDef htim16;

UART_HandleTypeDef huart1;

/* USER CODE BEGIN PV */

/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM3_Init(void);
static void MX_TIM1_Init(void);
static void MX_TIM14_Init(void);
static void MX_USART1_UART_Init(void);
static void MX_TIM16_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
```

```c
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM3_Init();
  MX_TIM1_Init();
  MX_TIM14_Init();
  MX_USART1_UART_Init();
  MX_TIM16_Init();
  /* USER CODE BEGIN 2 */
  uint8_t buffer[1];


        HAL_TIM_PWM_Start(&htim3,TIM_CHANNEL_1);
        HAL_TIM_PWM_Start(&htim16,TIM_CHANNEL_1);
        HAL_TIM_PWM_Start(&htim1,TIM_CHANNEL_3);
        HAL_TIM_PWM_Start(&htim1,TIM_CHANNEL_2);
        HAL_TIM_PWM_Start(&htim1,TIM_CHANNEL_1);
        HAL_TIM_PWM_Start(&htim14,TIM_CHANNEL_1);

        int i=62;
        int j=0;
        int u=96;
        int tryb=50;
        __HAL_TIM_SET_COMPARE(&htim16,TIM_CHANNEL_1,u);
        __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
       MX_USART1_UART_Init();
       buffer[0]=0;
       HAL_UART_Receive(&huart1, buffer, 1, 150);


        if(buffer[0]=='1'){
```

```
            HAL_UART_Transmit(&huart1,buffer,1,100);
            tryb=50;
 }
 else if(buffer[0]=='2'){
            HAL_UART_Transmit(&huart1,buffer,1,100);
            tryb=70;
 }
 else if(buffer[0]=='3'){
              HAL_UART_Transmit(&huart1,buffer,1,100);
              tryb=80;
       }
 else if(buffer[0]=='4'){
              HAL_UART_Transmit(&huart1,buffer,1,100);
              tryb=100;
       }
 else if(buffer[0]=='5'){
       HAL_UART_Transmit(&huart1,buffer,1,100);

       __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,0);
       __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,0);

       if(j<tryb){
             for(j;j<tryb;j++){
                   __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_2,j);
                   __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_3,j);
                   HAL_Delay(1);
             }
       }

       if(i!=44){
       for(i; i>44; i--){
                  __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                  HAL_Delay(5);
             }
       }

     }
 else if(buffer[0]=='6'){
       HAL_UART_Transmit(&huart1,buffer,1,100);

       __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,0);
       __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,0);

       if(j<tryb){
             for(j;j<tryb;j++){
                   __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_2,j);
                   __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_3,j);
                   HAL_Delay(1);
             }
       }

       if(i!=80){
       for(i; i<80; i++){
             __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
             HAL_Delay(5);
             }
       }
}
 else if(buffer[0]=='7'){
                   HAL_UART_Transmit(&huart1,buffer,1,100);

                   if(j<tryb){
                   for(j;j<tryb;j++){
                         __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,j);
                         __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,j);
                         HAL_Delay(1);
```

```
                    }
                    }

                    if(i!=44){
                    for(i; i>44; i--){
                                __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                                HAL_Delay(5);
                          }
                    }


              }
else if(buffer[0]=='8'){
                    HAL_UART_Transmit(&huart1,buffer,1,100);

                    if(j<tryb){
                    for(j;j<tryb;j++){
                          __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,j);
                          __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,j);
                          HAL_Delay(5);
                    }
                    }

                    if(i!=80){
                    for(i; i<80; i++){
                          __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                          HAL_Delay(5);
                          }
                    }
              }
else if(buffer[0]=='w'){
                    HAL_UART_Transmit(&huart1,buffer,1,100);
                    __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,0);
                    __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,0);

                    if(j<tryb){
                          for(j;j<tryb;j++){
                                __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_2,j);
                                __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_3,j);
                                HAL_Delay(1);
                          }
                    }
              }
else if(buffer[0]=='s'){
                  HAL_UART_Transmit(&huart1,buffer,1,100);

                    __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_2,0);
                    __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_3,0);

                    if(j<tryb){
                    for(j;j<tryb;j++){
                          __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,j);
                          __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,j);
                          HAL_Delay(1);
                    }
                    }
              }
             else if(buffer[0]=='d'){
                    HAL_UART_Transmit(&huart1,buffer,1,100);

                    for(i; i<80; i++){
                    __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                    HAL_Delay(5);
                    }
              }
              else if(buffer[0]=='a'){
```

```
                        HAL_UART_Transmit(&huart1,buffer,1,100);
                         for(i; i>44; i--){
                                    __HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                                    HAL_Delay(5);
                              }

                        }
                        else if(buffer[0]=='h'){
                                    HAL_UART_Transmit(&huart1,buffer,1,100);
                                     for(u; u<120; u++){
                                           __HAL_TIM_SET_COMPARE(&htim16,TIM_CHANNEL_1,u);
                                           HAL_Delay(5);
                                           }


                        }
                        else if(buffer[0]=='p'){
                                           HAL_UART_Transmit(&huart1,buffer,1,100);

                                                 for(u; u>96; u--){

__HAL_TIM_SET_COMPARE(&htim16,TIM_CHANNEL_1,u);
                                                                  HAL_Delay(5);
                                                                  }

                                           }
                        else{
                              HAL_UART_Transmit(&huart1,buffer,1,100);

                                    __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_2,0);
                                    __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_3,0);
                                    __HAL_TIM_SET_COMPARE(&htim14,TIM_CHANNEL_1,0);
                                    __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,0);
                                    j=0;

                                         if(i>60){
                                                       for(i; i>62; i--){

__HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                                                                        HAL_Delay(5);
                                                              }
                                                  }
                                                  else{
                                                       for(i; i<62; i++){

__HAL_TIM_SET_COMPARE(&htim3,TIM_CHANNEL_1,i);
                                                                        HAL_Delay(5);
                                                              }
                                                  }


                         }




    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
```

```
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSIDiv = RCC_HSI_DIV4;
  RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_HSI;
  RCC_ClkInitStruct.SYSCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_APB1_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
  {
    Error_Handler();
  }
}

/**
  * @brief TIM1 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM1_Init(void)
{

  /* USER CODE BEGIN TIM1_Init 0 */

  /* USER CODE END TIM1_Init 0 */

  TIM_MasterConfigTypeDef sMasterConfig = {0};
  TIM_OC_InitTypeDef sConfigOC = {0};
  TIM_BreakDeadTimeConfigTypeDef sBreakDeadTimeConfig = {0};

  /* USER CODE BEGIN TIM1_Init 1 */

  /* USER CODE END TIM1_Init 1 */
  htim1.Instance = TIM1;
  htim1.Init.Prescaler = 1200;
  htim1.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim1.Init.Period = 100;
  htim1.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim1.Init.RepetitionCounter = 0;
  htim1.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_PWM_Init(&htim1) != HAL_OK)
  {
    Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterOutputTrigger2 = TIM_TRGO2_RESET;
```

```
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim1, &sMasterConfig) != HAL_OK)
  {
    Error_Handler();
  }
  sConfigOC.OCMode = TIM_OCMODE_PWM1;
  sConfigOC.Pulse = 0;
  sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
  sConfigOC.OCNPolarity = TIM_OCNPOLARITY_HIGH;
  sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
  sConfigOC.OCIdleState = TIM_OCIDLESTATE_RESET;
  sConfigOC.OCNIdleState = TIM_OCNIDLESTATE_RESET;
  if (HAL_TIM_PWM_ConfigChannel(&htim1, &sConfigOC, TIM_CHANNEL_1) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_TIM_PWM_ConfigChannel(&htim1, &sConfigOC, TIM_CHANNEL_2) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_TIM_PWM_ConfigChannel(&htim1, &sConfigOC, TIM_CHANNEL_3) != HAL_OK)
  {
    Error_Handler();
  }
  sBreakDeadTimeConfig.OffStateRunMode = TIM_OSSR_DISABLE;
  sBreakDeadTimeConfig.OffStateIDLEMode = TIM_OSSI_DISABLE;
  sBreakDeadTimeConfig.LockLevel = TIM_LOCKLEVEL_OFF;
  sBreakDeadTimeConfig.DeadTime = 0;
  sBreakDeadTimeConfig.BreakState = TIM_BREAK_DISABLE;
  sBreakDeadTimeConfig.BreakPolarity = TIM_BREAKPOLARITY_HIGH;
  sBreakDeadTimeConfig.BreakFilter = 0;
  sBreakDeadTimeConfig.BreakAFMode = TIM_BREAK_AFMODE_INPUT;
  sBreakDeadTimeConfig.Break2State = TIM_BREAK2_DISABLE;
  sBreakDeadTimeConfig.Break2Polarity = TIM_BREAK2POLARITY_HIGH;
  sBreakDeadTimeConfig.Break2Filter = 0;
  sBreakDeadTimeConfig.Break2AFMode = TIM_BREAK_AFMODE_INPUT;
  sBreakDeadTimeConfig.AutomaticOutput = TIM_AUTOMATICOUTPUT_DISABLE;
  if (HAL_TIMEx_ConfigBreakDeadTime(&htim1, &sBreakDeadTimeConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN TIM1_Init 2 */

  /* USER CODE END TIM1_Init 2 */
  HAL_TIM_MspPostInit(&htim1);

}

/**
  * @brief TIM3 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM3_Init(void)
{

  /* USER CODE BEGIN TIM3_Init 0 */

  /* USER CODE END TIM3_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};
  TIM_OC_InitTypeDef sConfigOC = {0};

  /* USER CODE BEGIN TIM3_Init 1 */
```

```
  /* USER CODE END TIM3_Init 1 */
  htim3.Instance = TIM3;
  htim3.Init.Prescaler = 239;
  htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim3.Init.Period = 1000;
  htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim3.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim3) != HAL_OK)
  {
    Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim3, &sClockSourceConfig) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_TIM_PWM_Init(&htim3) != HAL_OK)
  {
    Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim3, &sMasterConfig) != HAL_OK)
  {
    Error_Handler();
  }
  sConfigOC.OCMode = TIM_OCMODE_PWM1;
  sConfigOC.Pulse = 0;
  sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
  sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
  if (HAL_TIM_PWM_ConfigChannel(&htim3, &sConfigOC, TIM_CHANNEL_1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN TIM3_Init 2 */

  /* USER CODE END TIM3_Init 2 */
  HAL_TIM_MspPostInit(&htim3);

}

/**
  * @brief TIM14 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM14_Init(void)
{

  /* USER CODE BEGIN TIM14_Init 0 */

  /* USER CODE END TIM14_Init 0 */

  TIM_OC_InitTypeDef sConfigOC = {0};

  /* USER CODE BEGIN TIM14_Init 1 */

  /* USER CODE END TIM14_Init 1 */
  htim14.Instance = TIM14;
  htim14.Init.Prescaler = 1200;
  htim14.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim14.Init.Period = 100;
  htim14.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim14.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim14) != HAL_OK)
  {
```

```
    Error_Handler();
  }
  if (HAL_TIM_PWM_Init(&htim14) != HAL_OK)
  {
    Error_Handler();
  }
  sConfigOC.OCMode = TIM_OCMODE_PWM1;
  sConfigOC.Pulse = 0;
  sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
  sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
  if (HAL_TIM_PWM_ConfigChannel(&htim14, &sConfigOC, TIM_CHANNEL_1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN TIM14_Init 2 */

  /* USER CODE END TIM14_Init 2 */
  HAL_TIM_MspPostInit(&htim14);

}

/**
  * @brief TIM16 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM16_Init(void)
{

  /* USER CODE BEGIN TIM16_Init 0 */

  /* USER CODE END TIM16_Init 0 */

  TIM_OC_InitTypeDef sConfigOC = {0};
  TIM_BreakDeadTimeConfigTypeDef sBreakDeadTimeConfig = {0};

  /* USER CODE BEGIN TIM16_Init 1 */

  /* USER CODE END TIM16_Init 1 */
  htim16.Instance = TIM16;
  htim16.Init.Prescaler = 239;
  htim16.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim16.Init.Period = 1000;
  htim16.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim16.Init.RepetitionCounter = 0;
  htim16.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim16) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_TIM_PWM_Init(&htim16) != HAL_OK)
  {
    Error_Handler();
  }
  sConfigOC.OCMode = TIM_OCMODE_PWM1;
  sConfigOC.Pulse = 0;
  sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
  sConfigOC.OCNPolarity = TIM_OCNPOLARITY_HIGH;
  sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
  sConfigOC.OCIdleState = TIM_OCIDLESTATE_RESET;
  sConfigOC.OCNIdleState = TIM_OCNIDLESTATE_RESET;
  if (HAL_TIM_PWM_ConfigChannel(&htim16, &sConfigOC, TIM_CHANNEL_1) != HAL_OK)
  {
    Error_Handler();
  }
  sBreakDeadTimeConfig.OffStateRunMode = TIM_OSSR_DISABLE;
```

```
  sBreakDeadTimeConfig.OffStateIDLEMode = TIM_OSSI_DISABLE;
  sBreakDeadTimeConfig.LockLevel = TIM_LOCKLEVEL_OFF;
  sBreakDeadTimeConfig.DeadTime = 0;
  sBreakDeadTimeConfig.BreakState = TIM_BREAK_DISABLE;
  sBreakDeadTimeConfig.BreakPolarity = TIM_BREAKPOLARITY_HIGH;
  sBreakDeadTimeConfig.BreakFilter = 0;
  sBreakDeadTimeConfig.AutomaticOutput = TIM_AUTOMATICOUTPUT_DISABLE;
  if (HAL_TIMEx_ConfigBreakDeadTime(&htim16, &sBreakDeadTimeConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN TIM16_Init 2 */

  /* USER CODE END TIM16_Init 2 */
  HAL_TIM_MspPostInit(&htim16);

}

/**
  * @brief USART1 Initialization Function
  * @param None
  * @retval None
  */
static void MX_USART1_UART_Init(void)
{

  /* USER CODE BEGIN USART1_Init 0 */

  /* USER CODE END USART1_Init 0 */

  /* USER CODE BEGIN USART1_Init 1 */

  /* USER CODE END USART1_Init 1 */
  huart1.Instance = USART1;
  huart1.Init.BaudRate = 9600;
  huart1.Init.WordLength = UART_WORDLENGTH_8B;
  huart1.Init.StopBits = UART_STOPBITS_1;
  huart1.Init.Parity = UART_PARITY_NONE;
  huart1.Init.Mode = UART_MODE_TX_RX;
  huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
  huart1.Init.OverSampling = UART_OVERSAMPLING_16;
  huart1.Init.OneBitSampling = UART_ONE_BIT_SAMPLE_DISABLE;
  huart1.Init.ClockPrescaler = UART_PRESCALER_DIV1;
  huart1.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
  if (HAL_UART_Init(&huart1) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_UARTEx_SetTxFifoThreshold(&huart1, UART_TXFIFO_THRESHOLD_1_8) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_UARTEx_SetRxFifoThreshold(&huart1, UART_RXFIFO_THRESHOLD_1_8) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_UARTEx_DisableFifoMode(&huart1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN USART1_Init 2 */

  /* USER CODE END USART1_Init 2 */

}
```

```c
/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();
  __HAL_RCC_GPIOD_CLK_ENABLE();

  /*Configure GPIO pin : blue_en_Pin */
  GPIO_InitStruct.Pin = blue_en_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(blue_en_GPIO_Port, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */

/* USER CODE END 4 */

/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}

#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```

Wnioski

W trakcie pracy na projektem napotkaliśmy wiele wyzwań, zobaczyliśmy jak bardzo taki projekt jest złożony i na ilu płaszczyznach trzeba pracować aby go wykonać. Każdy z nas znalazł coś dla siebie i mógł wykonać powierzone sobie zadania co poskutkował w wersji ostatecznej naszego robota.