Systemy Wbudowane i Mikroprocesory
2024/2025
Raport z budowy robota
Paula Grzebyk 21236

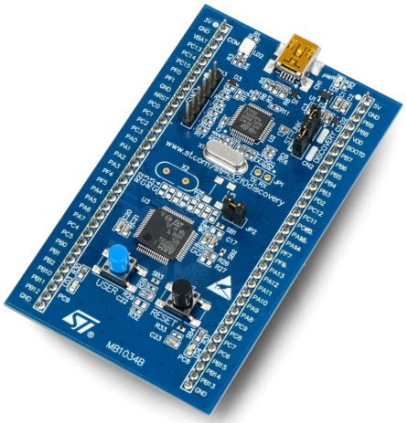
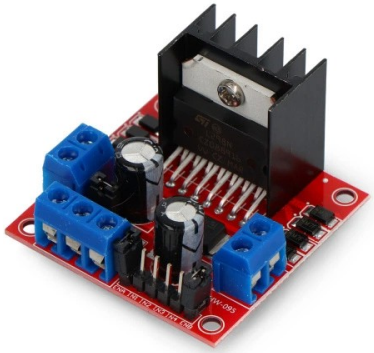
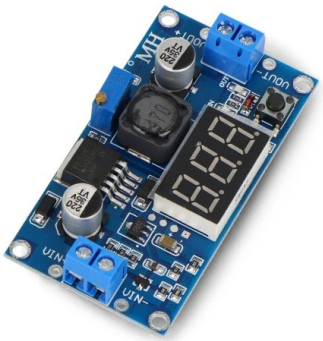
Sprawozdanie z milestone 2

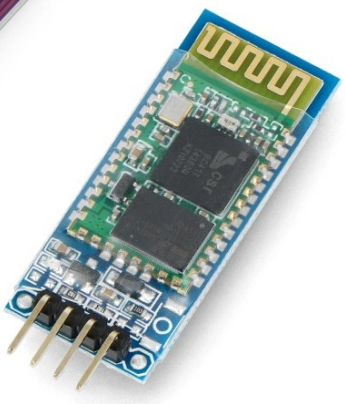
1. Opis robota

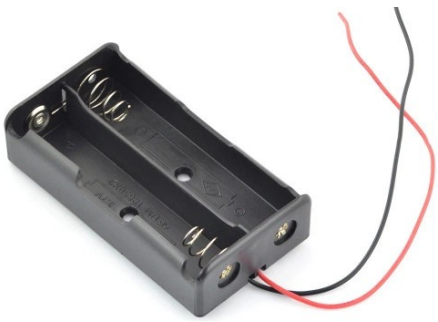
Robot trójkołowy o wymiarach ok. 24x20x15cm o napędzie kołowym za pomocą dwóch silników DC 5V, sterowany przez mikrokontroler STM32F051R8T6. Realizacja skrętu to obrót jednego koła według wskazówek zegara, a drugiego odwrotnie. Robot jest zasilany z koszyka na dwa ogniwa Li-On 18650 (łącznie 7,4V). Dodatkowo robot jest wyposażony w moduł Bluetooth HC-06 do komunikacji i sterowania z komputera za pomocą UART.

2. Elementy wybrane do budowy robota

- STM32F051R8T6
- L298N - dwukanałowy sterownik silników - moduł 12V/2A
- 2x Silnik DC 5V z oponą 65x26mm
- Kółko obrotowe
- Przetwornica step-down LM2596 3,2V-35V 3A z wyświetlaczem
- Moduł Bluetooth HC-06 ZS-040
- Koszyk na dwa ogniwa Li-On 18650
- 2x Cytron Maker Reflect - Analogowy czujnik odbiciowy podczerwieni IR
- Przewody

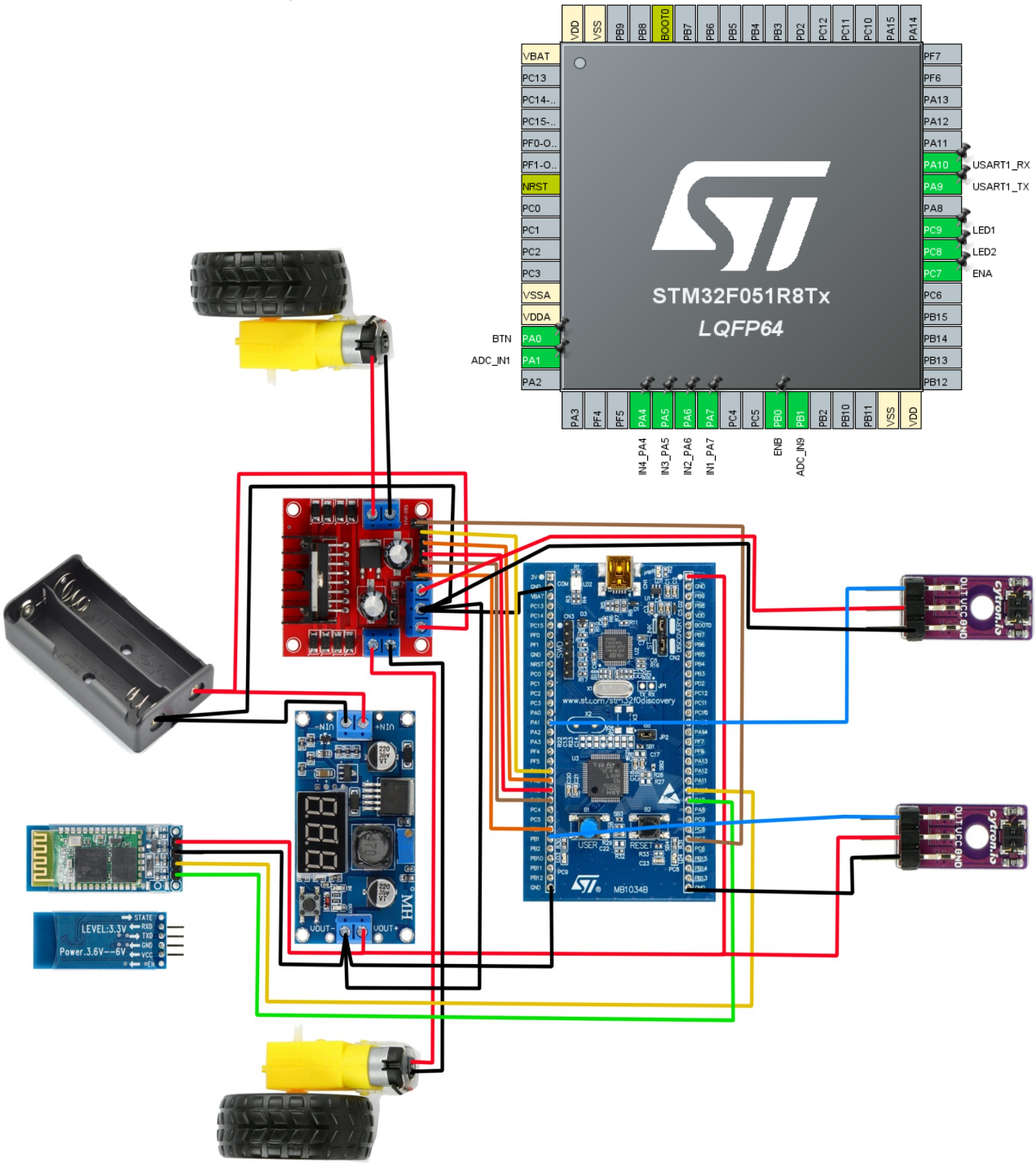






3. Mechanika robota

Podwozie zostało wykonane ze sklejki, na której umieszczono wszystkie elementy. Koła z silnikami umieszczono z tyłu a koło obrotowe pośrodku na przodzie pojazdu, i przyklejono gorącym klejem. Koszyk na akumulatory zamontowano z tyłu dla stabilizacji. L298N, przetwornicę, Bluetooth HC-06 oraz STM32F0 zamontowano przy użyciu gorącego kleju i taśmy dwustronnej. Czujniki IR zostały zamontowane na przodzie pojazdu w odstępach. Końcówki wybranych przewodów zostały cynowane aby zapobiec strzępieniu się żył miedzianych.

4. Schemat elektroniczny robota



5. Oprogramowanie sterujące

- Funkcje sterujące pojazdem – funkcje wysyłają odpowiednie sygnały do kół poprzez piny IN1-4 oraz ENA, ENB (PWM).

- (1,0) – koło kręci się według wskazówek zegara

- (0,1) – koło kręci się przeciwnie do wskazówek zegara

- (0,0) – koło nie kręci się

```
void Set_Speed(uint16_t speed){
    drivingspeed = speed; // zmienna ustalajaca predkosc (PWM)
    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_2, drivingspeed);
    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_3, drivingspeed);
}


void Go_Forward(void){
    Set_Speed(drivingspeed);
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 1); // oba kola do przodu
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 0);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 1);
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 0);
}
void Go_Left(void){
    Set_Speed(drivingspeed);
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 1); // prawe do przodu
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 0);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 0); // lewe do tylu
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 1);
}
void Go_Right(void){
    Set_Speed(drivingspeed);
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 0); // prawe do tylu
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 1);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 1); // lewe do przodu
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 0);
}
void Go_Backward(void){
    Set_Speed(drivingspeed);
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 0); // oba kola do tylu
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 1);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 0);
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 1);
}
void Stop(void){
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 0); // oba kola stop
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 0);
    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_2, 0);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 0);
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 0);
    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_3, 0);
```

- **Milestone 1:** Zaprogramowana sekwencja jazdy – po wciśnięciu przycisku na mikrokontroletrze pojazd porusza się po sekwencji wykonanej z 10 rozkazów.

```
 while (1)
 {
     if(HAL_GPIO_ReadPin(BTN_GPIO_Port, BTN_Pin)==1){
         Drive_Sequence();
     }
void Drive_Sequence(void){
    HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 1);
    Set_Speed(100);
    HAL_Delay(2000);

    Go_Forward(); //1
    HAL_Delay(1000);
    Stop();
    HAL_Delay(500);

    Go_Left(); //2
    HAL_Delay(350);
    Stop();
    HAL_Delay(500);

    Go_Forward(); //3
    HAL_Delay(1000);
    Stop();
    HAL_Delay(500);
```

- **Milestone 2:** Odczyt wartości z czujników przy użyciu ADC + DMA oraz przerwań i zapis do zmiennej uint16_t adc_values[2]

```
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
{
    HAL_ADC_Start_DMA(hadc, (uint32_t*)adc_values, 2);
}
```

Po wciśnięciu przycisku program robi 20 pomiarów i liczy średnią wartość obu czujników dla białego podłoża w danym pomieszczeniu (zmieniającą się ze względu na oświetlenie i typ podłoża). Następnie aktualne wartości czujników są porównywane do sumy wartości średniej i błędu – jeśli aktualna wartość jest większa, pojazd wykrywa czarną linię i odpowiednio skręca.

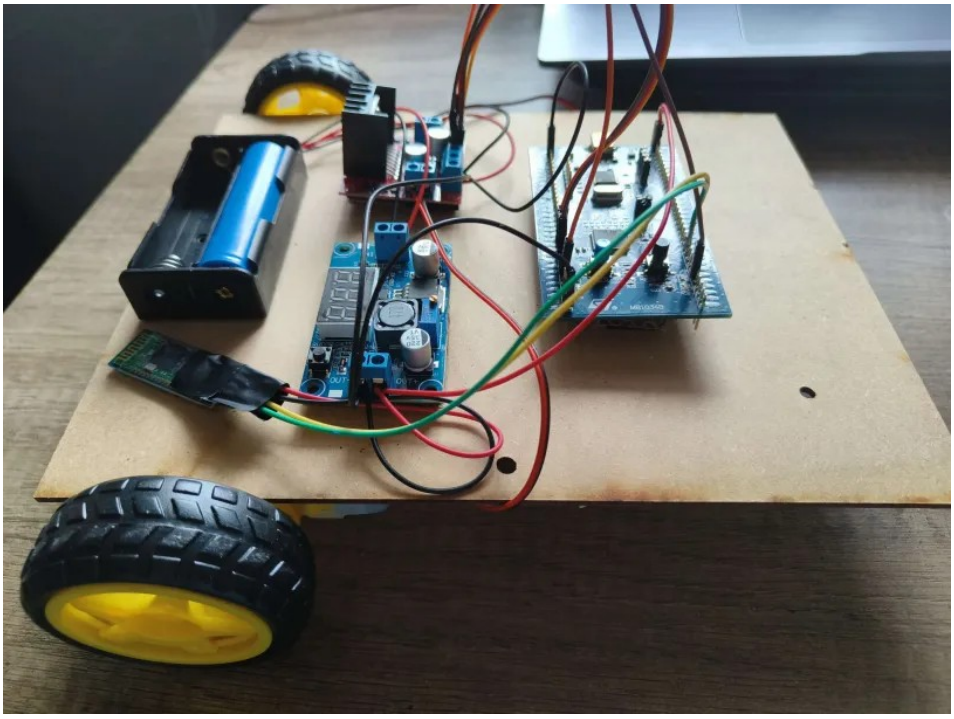
```
void Follow_Line(void){
    uint16_t ir_sum1=0, ir_sum2=0, samples=20;
    HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 1); // LED2

    //Zapisuje "lokalne" wartości dla białego podłoża w danym miejscu -> mierzy 20 razy i oblicza średnią
    for(uint16_t i=0; i<samples; i++){
        HAL_GPIO_TogglePin(LED1_GPIO_Port, LED1_Pin);
        ir_sum1 += adc_values[0];
        ir_sum2 += adc_values[1];
        HAL_Delay(20);
    }
    ir_value1_benchmark = ir_sum1/samples; // średnie
    ir_value2_benchmark = ir_sum2/samples;

    HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 0);
    HAL_Delay(500);

    char msg[64];
    sprintf(msg, "IR1 AVG: %lu, IR2 AVG: %lu\r\n", ir_value1_benchmark, ir_value2_benchmark);
    HAL_UART_Transmit(&huart1, (uint8_t*)msg, strlen(msg), 100);
    HAL_Delay(2000);

    while(1){
        ir_value1 = adc_values[0];
        ir_value2 = adc_values[1];

         // jeśli aktualny pomiar jest większy od 'lokalnego białego podłoża' -> skręt
        if(ir_value1 >= (ir_value1_benchmark + 500)){
            Go_Right();
        }
        if(ir_value2 >= (ir_value2_benchmark + 25)){
            Go_Left();
        }
        else{
            Go_Forward();
        }
        HAL_Delay(50);
    }
}
```

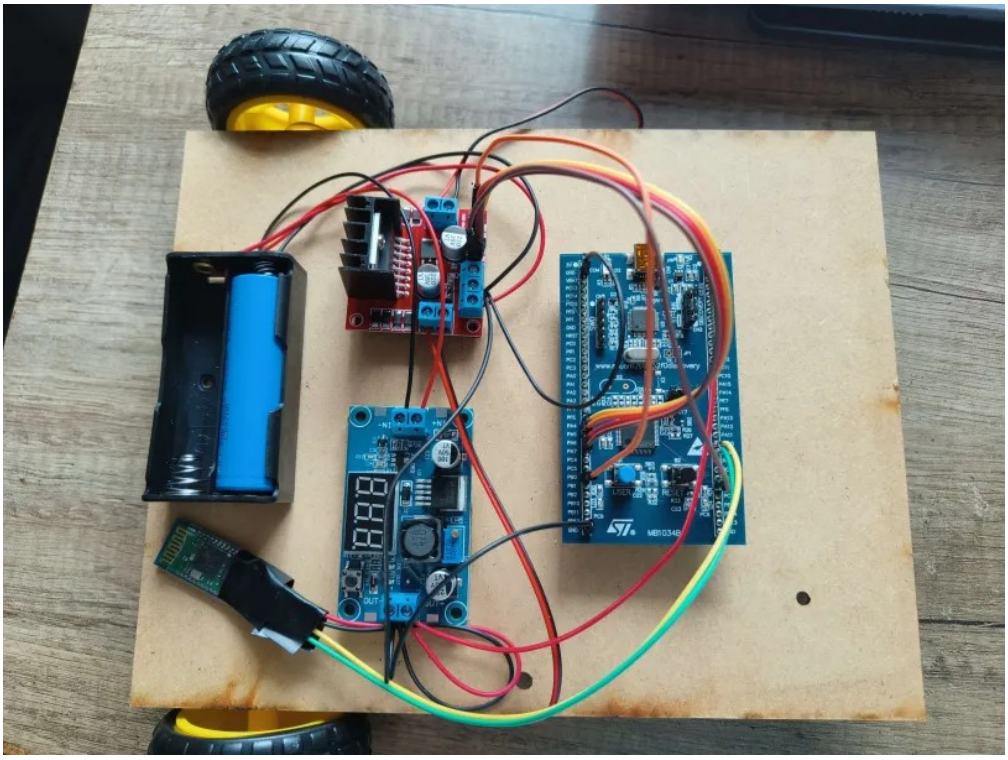
```
IR1: 1124, IR2: 85
IR1 AVG: 1119, IR2 AVG: 84
IR1: 1670, IR2: 94
```

6. Zdjęcia, filmy, kod robota

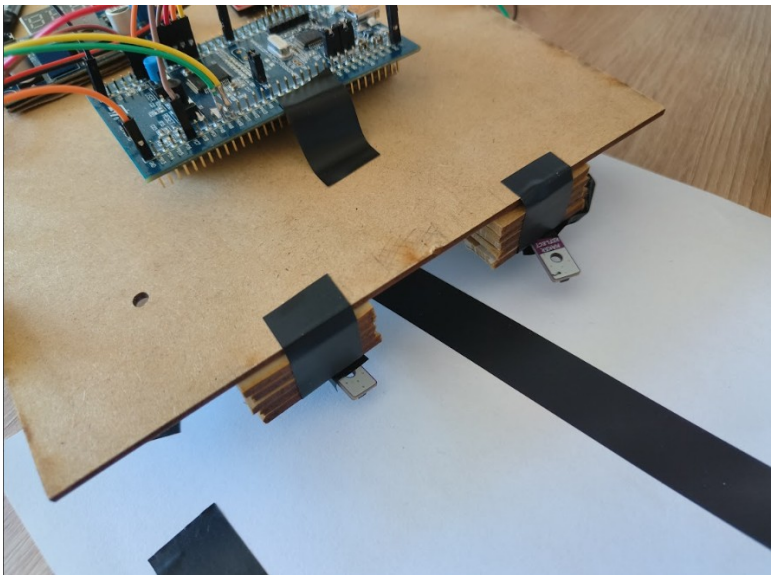
- Kod robota: https://github.com/BialaStrzala/SWIM_LineFollower
- Zdjęcia i filmy: https://drive.google.com/drive/folders/1Ak4zJQ64U5y1St-GSTiknMUM3oySruRG?usp=drive_link

- **Milestone 1:**





- **Milestone 2:**



Odczyt wartości czujników poprzez Bluetooth HC-06 oraz UART (IR1: ~240 = biała kartka, ~4095 = czarna linia, IR2: ~2000 = biała kartka, ~3500+ = czarna linia)

```
IR1: 2158, IR2: 240
IR1: 2114, IR2: 241
IR1: 1958, IR2: 247
IR1: 1895, IR2: 4095
IR1: 1846, IR2: 4095
IR1: 1849, IR2: 4095
IR1: 1851, IR2: 4095
IR1: 1856, IR2: 4095
```

```
IR1: 1969, IR2: 247
IR1: 2015, IR2: 240
IR1: 2071, IR2: 238
IR1: 1960, IR2: 242
IR1: 2126, IR2: 280
IR1: 3658, IR2: 258
IR1: 3692, IR2: 258
IR1: 3703, IR2: 248
```