YTU YILDIZ TEKNİK ÜNİVERSİTESİ

2WD Mobile Robot (with Bluetooth)

EMİN EMİRHAN ŞENER – 20067040
ENES EMEKSİZ – 19067015
SHOHRUKH DADAKHON KHASAN – 19067905

Microprocessors and Programming | MKT3811 | 25.12.2022

Video footage of the project:

https://youtu.be/FfXRndWmehA

I. CODE

```
#pragma config FOSC = XT      // Oscillator Selection bits (HS oscillator)
#pragma config WDTE = OFF      // Watchdog Timer Enable bit (WDT disabled)
#pragma config PWRTE = ON      // Power-up Timer Enable bit (PWRT enabled)
#pragma config BOREN = OFF      // Brown-out Reset Enable bit (BOR enabled)
#pragma config LVP = ON      // Low-Voltage (Single-Supply) In-Circuit Serial Programming Enable bit (RB3 is digital I/O,
HV on MCLR must be used for programming)
#pragma config CPD = OFF      // Data EEPROM Memory Code Protection bit (Data EEPROM code protection off)
#pragma config WRT = OFF      // Flash Program Memory Write Enable bits (Write protection off; all program memory
may be written to by EECON control)
#pragma config CP = OFF      // Flash Program Memory Code Protection bit (Code protection off)

#define _XTAL_FREQ 4000000
#define Baud_rate 9600

#include<xc.h>

int duty1 = 0;
int duty2 = 0;

void Initialize_Bluetooth(void)

{
  TRISC6 = 0; // TX -> OUTPUT
  TRISC7 = 1; // rx -> INPUT

  SPBRG = ((_XTAL_FREQ/16)/Baud_rate) - 1; //Period of a free running timer.
  BRGH  = 1;  // High speed baud rate

  SYNC  = 0;    // Asynchronous communication is set
  SPEN  = 1;    // Enable serial port pins

  TXEN  = 1;    // enable transmission
  CREN  = 1;    // enable reception

  // 8-bit communication is set
  TX9   = 0;
  RX9   = 0;
}

void BT_load_char(char byte)
{
  TXREG = byte;
  while(!TXIF);
  while(!TRMT);
}

void BT_load_string(char* string)
{
  while(*string)
  BT_load_char(*string++);
}

void broadcast_BT()
{
 TXREG = 13;
 __delay_ms(20);
}
```

```
char BT_get_char(void)
{
   if(OERR) // check for over run error
   {
      CREN = 0;
      CREN = 1; //Reset CREN
   }

   if(RCIF==1) // returns the value which user sends in ASCII value.
   {
      while(!RCIF);
      return RCREG;
   }

   else
      return 0;
}

//****** Functions which are named pwm_set_dutyx, to alter the duty cycle easier are defined. *****//
void pwm_set_duty1(int duty1)
{
   CCPR1L = duty1>>2; //8 highest bits of duty1 value is written on CCPR1L register. (>>2))
   CCP1X = duty1&1; //First bit of duty1 is written on CCP1X bit.
   CCP1Y = duty1&2; //Second bit of duty1 is written on CCP1Y bit.
}

void pwm_set_duty2(int duty2)
{
   CCPR2L = duty2>>2; //8 highest bits of duty2 value is written on CCPR1L register. (>>2))
   CCP2X = duty2&1; //First bit of duty2 is written on CCP1X bit.
   CCP2Y = duty2&2; //Second bit of duty2 is written on CCP1Y bit.
}
//****** Functions which are named pwm_set_dutyx, to alter the duty cycle easier are defined. *****//


//******* Speed Configuration Functions *******//
void speed0()
{
   duty1 = 0;
   duty2 = 0;
}

void speed1()
{
   duty1 = 250;
   duty2 = 250;
}

void speed2()
{
   duty1 = 275;
   duty2 = 275;
}

void speed3()
{
   duty1 = 300;
   duty2 = 300;
}

void speed4()
{
   duty1 = 325;
   duty2 = 325;
}
```

```
void speed5()
{
   duty1 = 345;
   duty2 = 345;
}

void speed6()
{
   duty1 = 365;
   duty2 = 365;
}

void speed7()
{
   duty1 = 375;
   duty2 = 375;
}

void speed8()
{
   duty1 = 385;
   duty2 = 385;
}

void speed9()
{
   duty1 = 395;
   duty2 = 395;
}

void speed10()
{
   duty1 = 400;
   duty2 = 400;
}

//******* Speed Configuration Functions *******//

// function for driving straight
void forward()
{
   RB0 = 0;
   RB1 = 1;
   pwm_set_duty1(duty1);

   RB5 = 0;
   RB4 = 1;
   pwm_set_duty2(duty2);
}

// function for driving backward
void backward()
{
   RB0 = 1;
   RB1 = 0;
   pwm_set_duty1(duty1);

   RB5 = 1;
   RB4 = 0;
   pwm_set_duty2(duty2);
}
```

```
// function for driving left
void left()
{
   RB0 = 0;
   RB1 = 0;
   pwm_set_duty1(0);

   RB5 = 0;
   RB4 = 1;
   pwm_set_duty2(duty2);
}

// function for driving right
void right()
{
   RB0 = 0;
   RB1 = 1;
   pwm_set_duty1(duty1);

   RB5 = 0;
   RB4 = 0;
   pwm_set_duty2(0);
}

// function for driving forward left
void forward_left()
{
   RB0 = 0;
   RB1 = 1;
   pwm_set_duty1(duty1-50);

   RB5 = 0;
   RB4 = 1;
   pwm_set_duty2(duty2);
}

// function for driving forward right
void forward_right()
{
   RB0 = 0;
   RB1 = 1;
   pwm_set_duty1(duty1);

   RB5 = 0;
   RB4 = 1;
   pwm_set_duty2(duty2-50);
}

// function for driving backward left
void backward_left()
{
   RB0 = 1;
   RB1 = 0;
   pwm_set_duty1(duty1-50);

   RB5 = 1;
   RB4 = 0;
   pwm_set_duty2(duty2);
}
```

```
// function for driving backward right
void backward_right()
{
   RB0 = 1;
   RB1 = 0;
   pwm_set_duty1(duty1);

   RB5 = 1;
   RB4 = 0;
   pwm_set_duty2(duty2-50);
}
// function to stop
void stop()
{
   RB0 = 0;
   RB1 = 0;
   pwm_set_duty1(0);

   RB5 = 0;
   RB4 = 0;
   pwm_set_duty2(0);
}

/*
// buzzer on
void buzzer_on()
{
   RB2 = 1;
}

// buzzer off
void buzzer_off()
{
   RB2 = 0;
}
*/

void main(void)

{
   char command;

   TRISB = 0x00; // PORTB -> OUTPUT
   TRISC = 0X00; // PORTC -> OUTPUT

   PORTB = 0X00; // PORTB -> LOW
   PORTC = 0X00; // PORTC -> LOW

  Initialize_Bluetooth(); //call initialize function

  __delay_ms(50);

  BT_load_string("Bluetooth Initialized and Ready");
  broadcast_BT();

  //*********************** Initialize PWM ***********************//

  CCP1CON = 0B00001111;
                // CCP1CON register is configured to use pwm.
                // Duty cycle is set to 0 (Default).
  CCP2CON = 0B00001111;
  T2CON = 0B00000101; // T2 enable, prescale 4
  PR2 = 99;
  //Max duty is 400.
```

```
//************************ Initialize PWM ************************//

while(1) //The infinite loop
{
command = BT_get_char(); //Read the char which comes via BT.

switch(command){

 case '0':
 speed0();
 break;

 case '1':
 speed1();
 break;

 case '2':
 speed2();
 break;

 case '3':
 speed3();
 break;

 case '4':
 speed4();
 break;

 case '5':
 speed5();
 break;

 case '6':
 speed6();
 break;

 case '7':
 speed7();
 break;

 case '8':
 speed8();
 break;

 case '9':
 speed9();
 break;

 case 'q':
 speed10();
 break;

 case 'F':
 forward();
 break;

 case 'B':
 backward();
 break;

 case 'L':
 left();
 break;

 case 'R':
 right();
 break;
```

```
     case 'G':
     forward_left();
     break;

     case 'I':
     forward_right();
     break;


     case 'H':
     backward_left();
     break;

     case 'J':
     backward_right();
     break;

     case 'S':
     stop();
     break;

     /*
     case 'V':
     buzzer_on();
     break;

     case 'v':
     buzzer_off();
     break;
     */
    }
    }
  }
```

II. CIRCUIT SKETCH

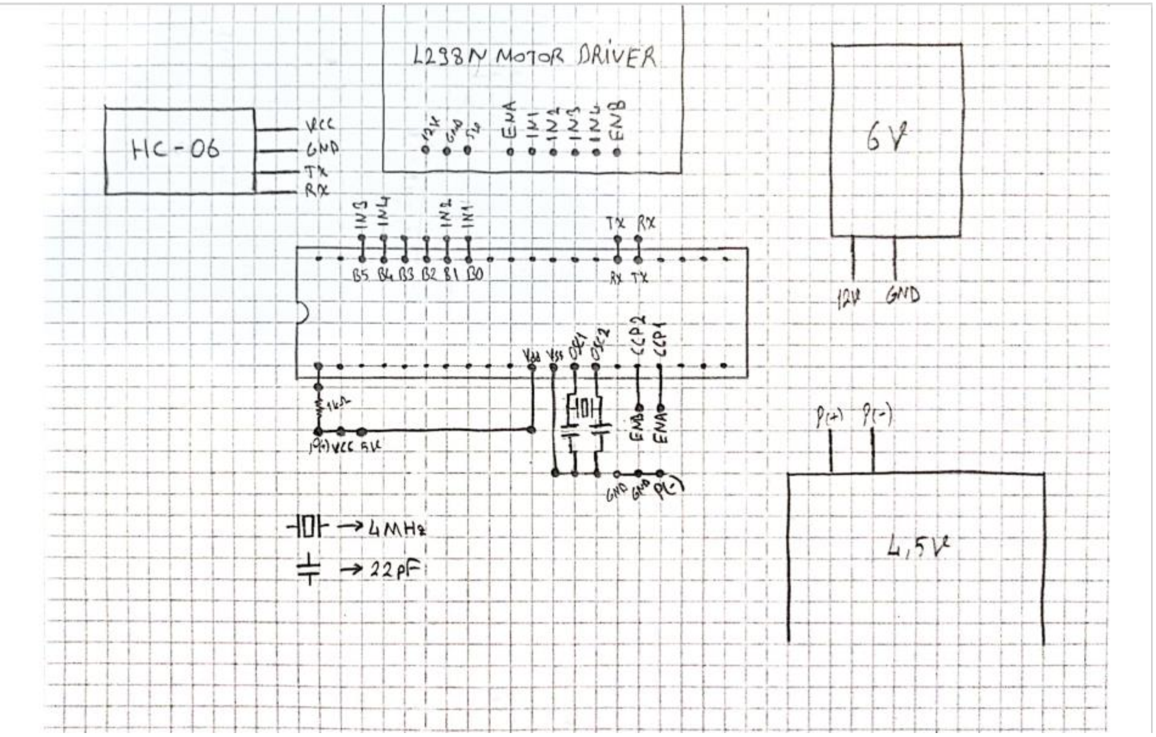


Figure 1: Sketch of the circuit.

III. CIRCUIT ON PROTEUS

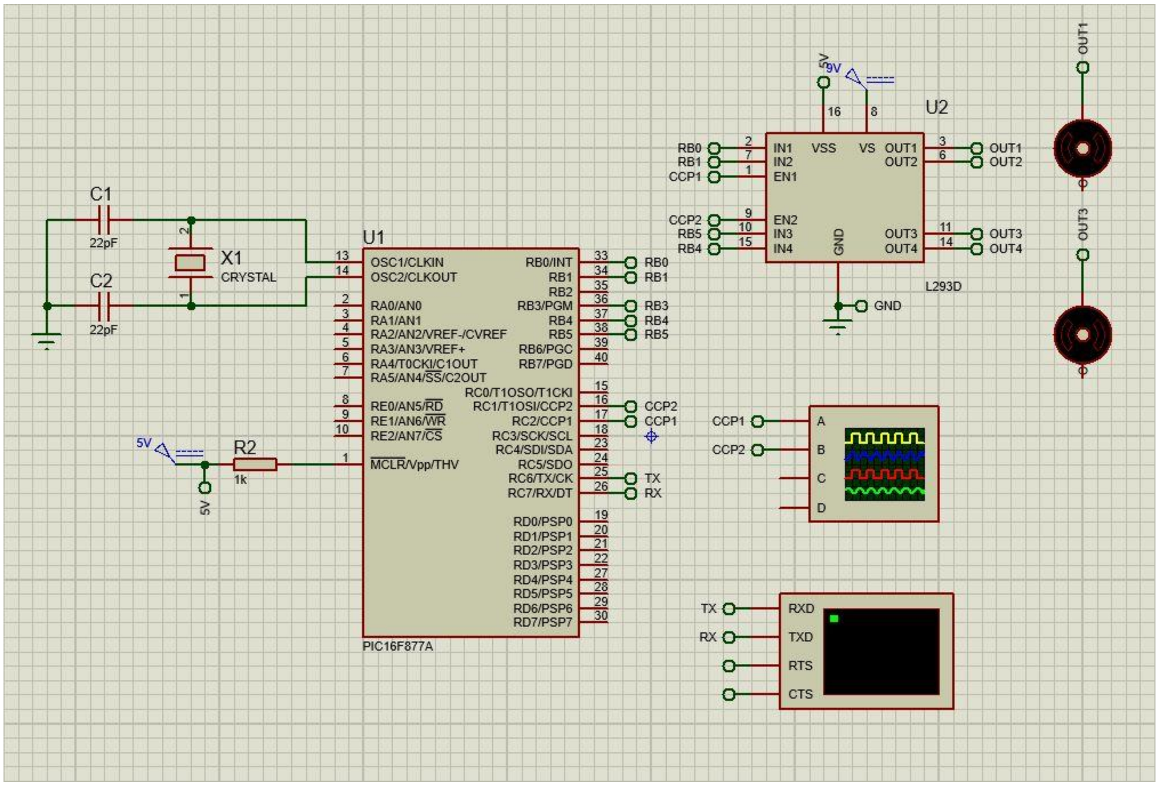
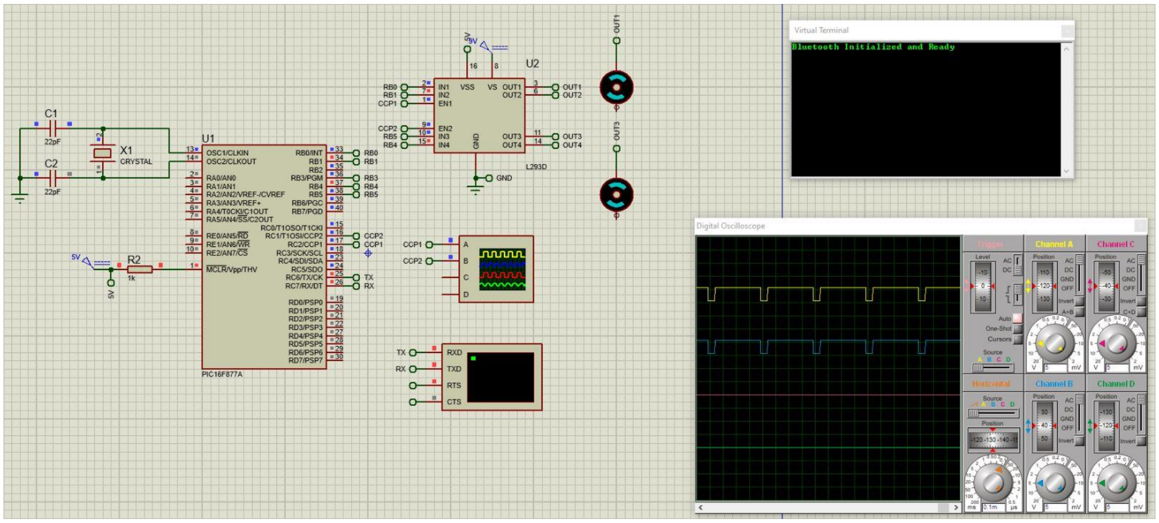


Figure 2: Sketch of the circuit redrawn on Proteus Software

IV. SIMULATION



```
Bluetooth Initialized and Ready
```

Figure 3: Simulation on Proteus Software.

V. 3D DRAWING OF THE VEHICLE FRAMEWORK

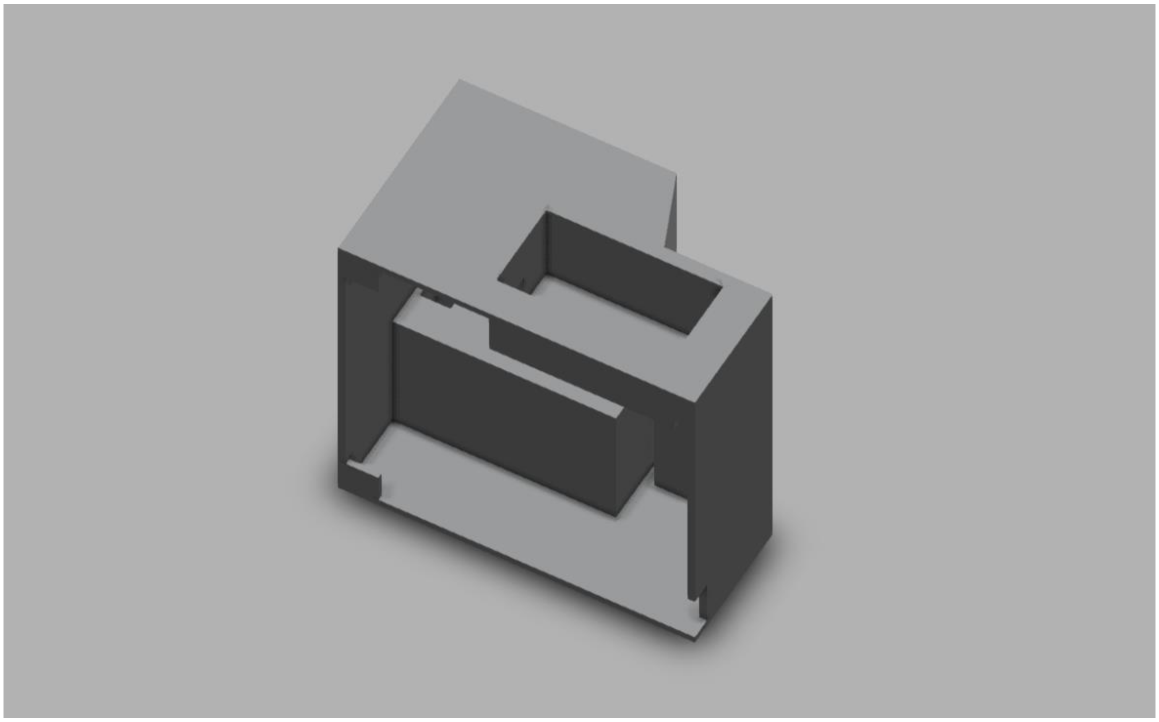
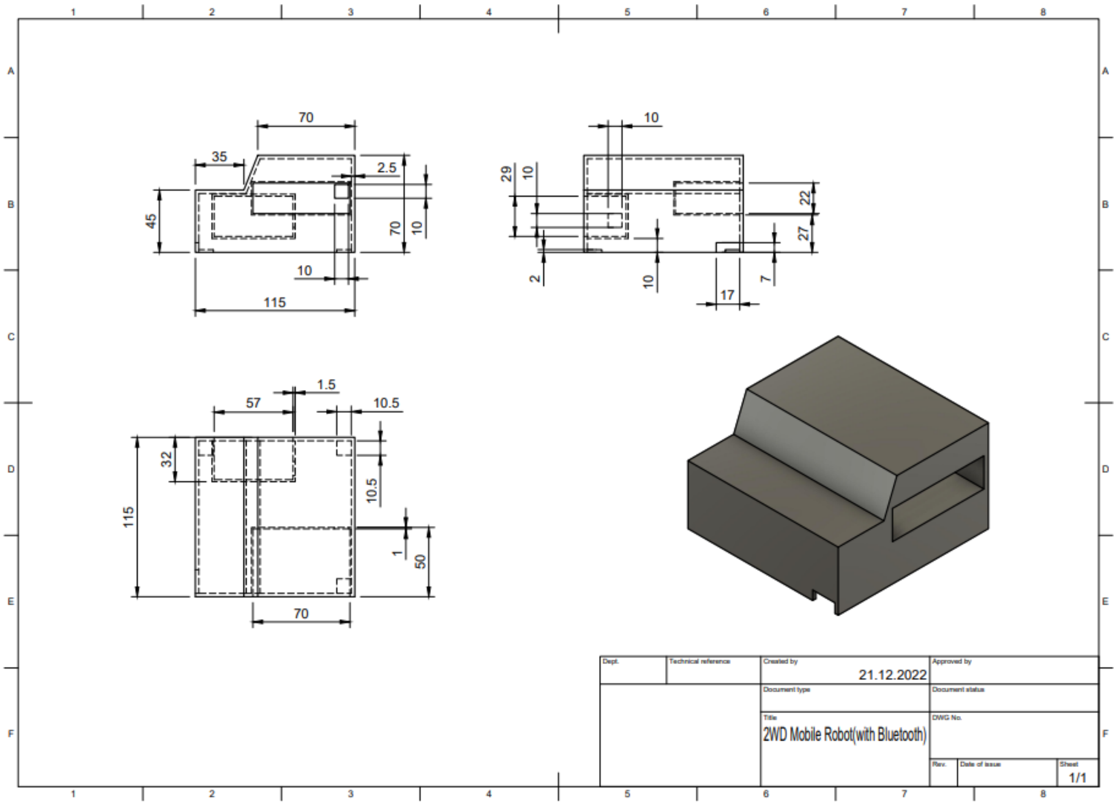


Figure 4: 3D drawing of the vehicle framework, drawn using Fusion 360 software.

VI. TECHNICAL DRAWING OF THE VEHICLE FRAMEWORK



Dept.	Technical reference	Created by 21.12.2022	Approved by
		Document type	Document status
		Title 2WD Mobile Robot(with Bluetooth)	DWG No.
			Rev. | Date of issue | Sheet 1/1

Figure 5: Technical drawing of the vehicle outer framework.

VII. PHOTOS

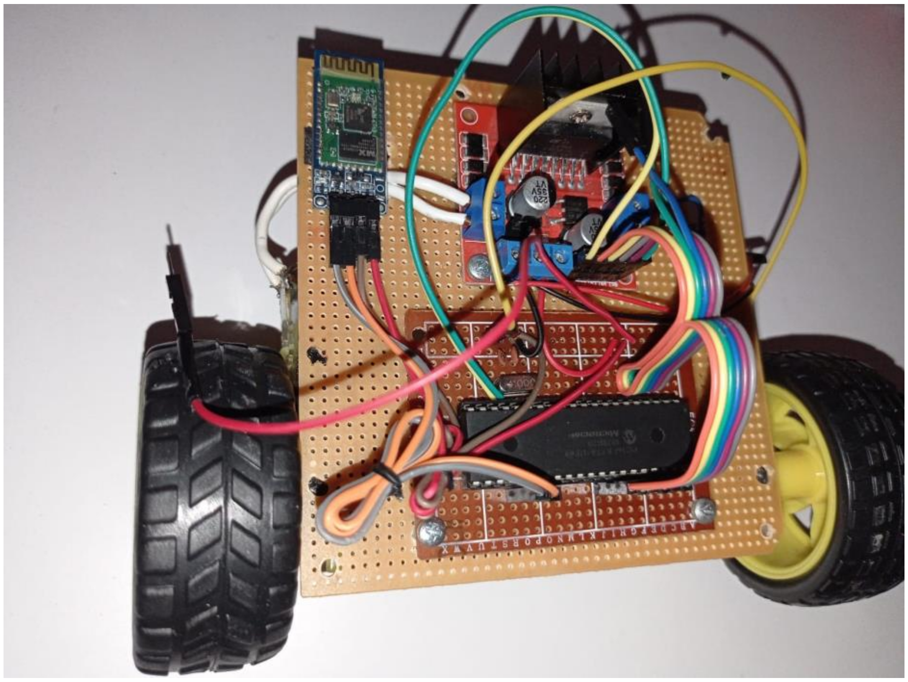


Figure 6: Real life circuit of the project. Shot from back.

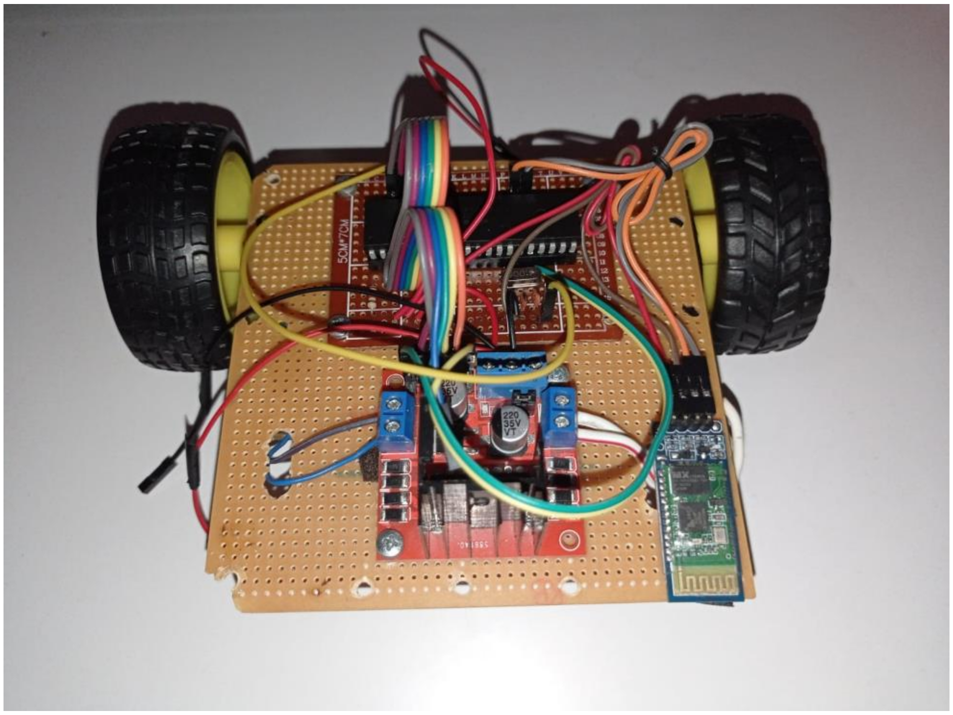


Figure 7: Real life circuit of the project. Shot from front.

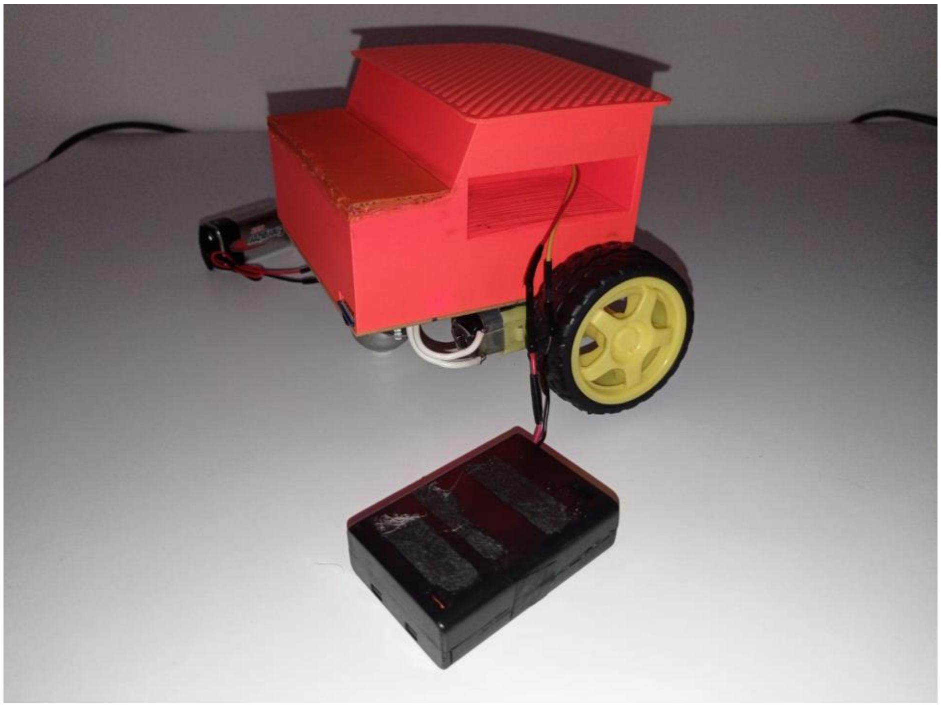


Figure 8: Finished version of the project. Shot from right.

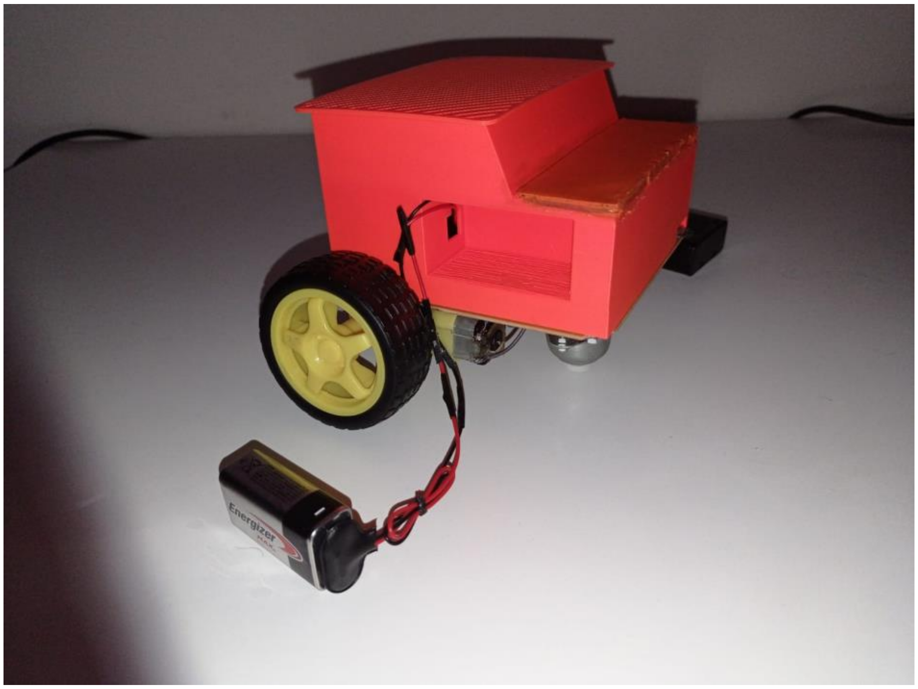


Figure 9: Finished version of the project. Shot from left.

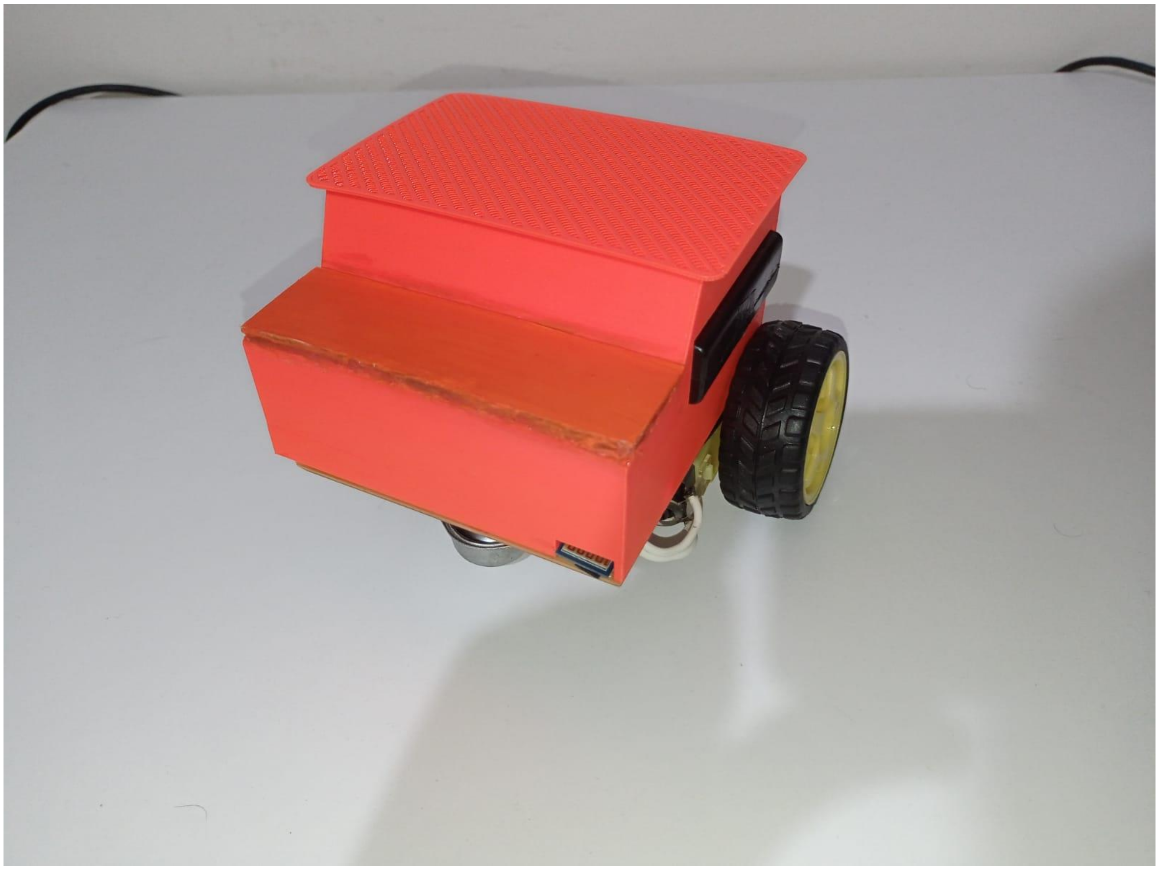


Figure 10: Finished version of the project. Shot from front.

VIII. Materials and Resources

Materials and Resources used in this project:

- Two 5V DC Motors
- L298N Motor Driver
- PIC16F877A Microcontroller
- Two 22 pF Capacitors
- 1 kilohm Resistor
- 9 V Battery
- 4 MHz Crystal
- Three 1.5 V Batteries
- HC-06 Bluetooth module
- 3 AA Battery holder
- PCB Layout
- 40 Pin PIC Socket
- Cables
- Ball Caster
- Toy Car Wheels
- Android Mobile Phone
- Bluetooth RC Controller Mobile Application
- Fusion 360 Software
- Proteus Software
- Microchip Softwares
- PICKit 3 Programmer
- Computer (for programming)
- Soldering iron