

MŰSZAKI ISKOLA ADA



## **Multifunkciós Robot Auto**

Tanuló-Osztály:

Nagy Richárd IV/3

Mentor:

Fekete Lajos

### 1. Bevezető

Napjainkban a mobil robotok egyre inkább jelen vannak életünk számos területén. Megtalálhatjuk őket az autóiparban, az egészségügyben, a modern hadviselésben, az úrkutatásban és a szórakoztató iparban egyaránt.

A szakterület fejlődését a jelenkorban is zajló intenzív kutatások biztosítják. A kutatások kiterjednek a mobil robotok megfelelő kinematikai struktúrájának kutatására, az alkalmazható érzékelők és beavatkozók kutatására, különböző elektronikai és szoftveres megoldásokra és nem utolsósorban a mobil robotoknál hatékonyan alkalmazható irányítástechnikai megoldások keresésére.

A verseny eredményeként a gyakorlatban megvalósult 4 kereken guruló mobil robot. A robot vezérlőt egy RP2040 mikrovezérlővel rendelkező lap Raspberry Pi Pico alkotja. A struktúra további elemei: négy egyenáramú motor, egy szervó motor, egy távolság érzékelő szenzor, kettő infra szenzor, három darab akkumulátor, egy H-hidas motor vezérlő, egy nyomógomb, egy Bluetooth modul, illetve egy gyroskot tartalmaz. A beavatkozók és az érzékelők kapcsolatát a vezérlővel, egy saját fejlesztésű illesztőkártya biztosítja.

A robot programjai közül nyomógomb segítségével változtathatunk, ezzel a megoldással nem kell mindig külön programot fel tölteni a mikrovezérlőre ha programot szeretnénk váltani. A robot négy darab programmal rendelkezik: Bluetooth vezérlés, önvezető mód, vonal lekövető mód, illetve hang vezérléssel is rendelkezik.

A dokumentáció be mutatja a robot működési elvét, a szoftver működési elvét, az illesztőkártya kapcsolási rajzát és nyomtatott áramköri rajzát, a beültetési rajzot és nem utolsósorban az alkatrészek működési elvét.



## Tartalom

1. Bevezető.....	2
2. Hardware felépítése.....	4
2.1 RP2040.....	4
2.2 Motor vezérlő és PWM.....	5
2.3 Távolság érzékelő szenzor.....	6
2.4 Infra Szenzor.....	6
2.5 Bluetooth Modul.....	7
2.6 Gyroscope.....	7
2.7 Illesztőkártya.....	8
3 Software.....	10
3.1 Arduino Platform.....	10
3.2 Drivererek.....	11
3.3 Software Felépítése.....	12
4 Robot működési elve.....	14
5 Irodalom Jegyzék.....	15

## 2. Hardware felépítése

### 2.1 RP2040

A munka megvalósításához alkalmazott, választott beágyazott rendszer a Raspberry Pi Pico. A Raspberry Pi Pico egy RP2040 épülő mikrovezérlő, softveres elektronikai fejlesztőplatform, arra tervezve hogy különböző projektekben könnyen hozzáférhetőek legyenek, kezelhetőek legyenek. Széles körben használják mivel olcsó, könnyen beszerezhető, nem kell hozzá külső programozó, debuggolható, illetve több külső eszköz is csatlakoztatható hozzá. A fejlesztői platform az integrált fejlesztői környezet (IDE) és a Raspberry Pi Pico-ból áll. Az elkészített programok könnyedén fel tölthetők a mikrovezérlőre USB keresztül. A softveres fejlesztői platformot a következő fejezetben taglalom részletesen.

A munkához Raspberry Pi Pico lapot használtam, mivel több erőforrással rendelkezik egy átlagos Arduino-laphoz képest. Előnye hogy egyszerre több kommunikációs portot is tud kezelni, három timerrel is rendelkezik, harminckettő bites ARM architektúra, és huszonegy multifunkciós láb is rendelkezik.

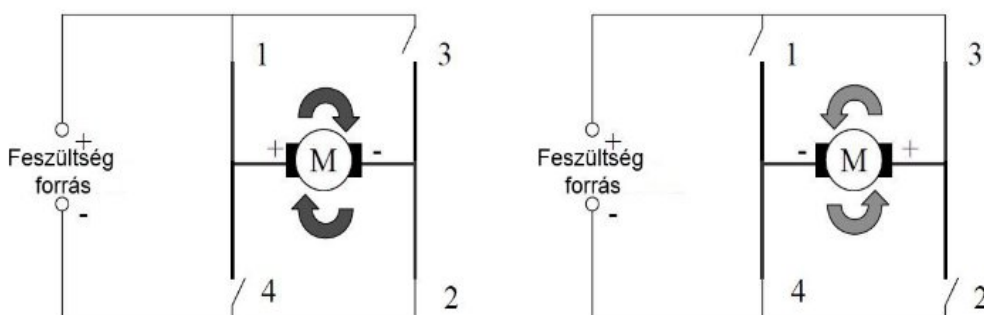
#### Technikai Paraméterei:

Mikrovezérlő	RP2040
Üzemi Feszültség	3.3V
Be meneti feszültség(ajánlott)	5V
Be meneti feszültség(határok)	4.5V-5.5V
Digitális ki/be menetek	28(Az összes képes PWM generálni)
Analóg be menetek	3
Max leadható egyenáram I/O Pinként	20 mA
Program memória	2MB
SRAM	264KB
Kommunikációs Portok	2x UART, 2x I <sup>2</sup> C, 2x SPI
Órajel sebessége	133MHz

### 2.2 Motor vezérlő és PWM

A PWM analóg tápkörök elkerülésére szolgáló módszer, mely kihasználja azt a tényt hogy a mechanikai rendszerek bizonyos késéssel működnek. Ez egy folyamatos ki meneti jel helyett teljes rendszerfeszültségen (pl. 3.3V) működő digitális pulzusokat generál egy fix frekvencián (pl. 20kHz). Az impulzus szélességét szoftveren belüli változtatásával tudjuk előállítani a ki meneti jelet (jel erősségét). A PWM által létrehozott jel kitöltött jel és nem-jel értéke határozza meg a kimenei jel nagyságát, ami a mi esetünkben a hajtott motor sebessége.

A legtöbb alkalmazásnál két dolgot akkarunk meg valósítani a motrokal: Előre és hátrafele lehesen meg hajtani illetve lehesen szabályozni a sebességét. H-híd kapcsolásra van szükségünk ahol hogy a motorot meg tudjuk hajtani előre-hátrafele.



Az ábra szemlélteti a H-híd felépítését, amiről kapta a nevét hogy hasonlít a “H” betűre. Láthatjuk hogy ha az 1-es és a 2-es kapcsoló van bekapcsolva, óramutató járásával megegyező irányba forog a motor, ha a 3-ast és a 4-est van bekapcsolva akkor óramutatóval ellentétes irányba fog forogni.

Mivel egy mikrovezérlő nem képes nagy áramot le adni ezért erre a célra motor vezérlő áramkört használunk a meghajtásra, vagy tranzisztors hajtást használunk. Erre a célra L298N motor driver modult használtam, mivel a PCB(Printed Circuit Board – Nyomtatott áramköt) minden rajta van ami a motor vezérléshez szükséges. Elektronikus kapcsolókat használ, a megfelelő lábaira adott feszültséggel lehet vezérelni őket. A sebesség vezérléshez pedig a kettő EN lábra kell adni a PWM jelet. Ezáltal könnyen vezérhető a H-híd a mikrovezérlőről.

### 2.3 Távolság érzékelő szenzor

Távolság érzékelő szenzornak HC-SR04 ultranangos szenzor modult használtam, mivel a szenzor könnyedén használható mikrovezérlővel, illetve nagy távolságban képes érzékelni a távolságot akár 4 métere is.

A mérés indításához a vezérlőnek kell adni minimum egy  $10\mu s$  magas jeletszintet (3.3V jelszint), ennek hatására az ultrahangos adó küld 8 darab 40KHz impulzust. Ha a jel vissza érkezik (ultrahangos vevőhöz) magas jelszinten, a magas szint ideje az az idő amely az ultrahang küldése és fogadása között eltelt. Ki számítani a következő képletel lehet centiméterben:

$$\text{Távolság} = (\text{elteltidő} * \text{hangsebbség}) / 2$$

eltelidő – amíg a be menet magas szinten van (az adó jel küldése, és a vevő jel érkezése közötti idő)

hangsebbség – értéke 0.034

2 – az ultrahang útja

### 2.4 Infra Szenzor

A munkához kettő darab infra szenzora is szükség van, mivel elengedhetetlen a vonal le követő programnál.

A szenzoron kettő darab led található ezek pedig a következők: egy IR led és egy fotódioda. Az infravörös adó folyamatosan bocsájt ki infravörös fényt, a vevő pedig folyamatosan ellenőrzi a vissza érkező fényt. Ha a vissza érkező fény valamilyen tárgyról érkezik vissza, akkor változik a szenzor ki menete. A modulon található egy potencióméter is amivel be lehet állítani a érzékenységét.



### 2.5 Bluetooth Modul

A Bluetooth RS232-es szabványnak rövidtávú vezeték nélküli alternatívája, amely az átvitelhez mikrohullámú rádióhullámokat használ. Sáv szélessége 2.4-től 2.485 GHz-ig terjed. Ez a protokoll full duplex kommunikációt is támogat.

A munkámhoz JY-MCU Bluetooth modult használtam, amit csatlakoztatok a mikrovezérlőhöz UART kommunikációs protokollt használva tud a modul kommunikálni a mikrovezérlővel. Ez a modul fontos lesz a Bluetooth vezérlés esetén.

#### UART Kommunikációs protokoll:

Előnye hogy full duplex kommunikáció. Az UART kommunikációban kettő egyenrangú eszköz vesz részt. Az információ csere két vezetéken történik. Mindkét eszközön van egy RX (receiver - fogadó), és egy TX(transceiver - küldő) láb. Az eszközök meghatározott sebességen kommunikál, ezt a sebességet nevezzük baud rate-nek ami azt fejezi ki, hogy hány bitet tudunk másodpercenként küldeni. Az adatcsere bitenként történik.

### 2.6 Gyroscope

A mai eszközökben rengeteg helyen meg található a gyroscope, autóiparban mobil iparban, és még sok máshol. A gyroscope a perdület megmaradás törvénye alapján működik.

A munkámhoz L3G4200D szenzort használtam, ez a szenzor ami egy háromtengelyes gyroscope. A szenzoral lehet SPI illetve I<sup>2</sup>C kommunikálni, ebben az esetben I<sup>2</sup>C kommunikációt használtam.

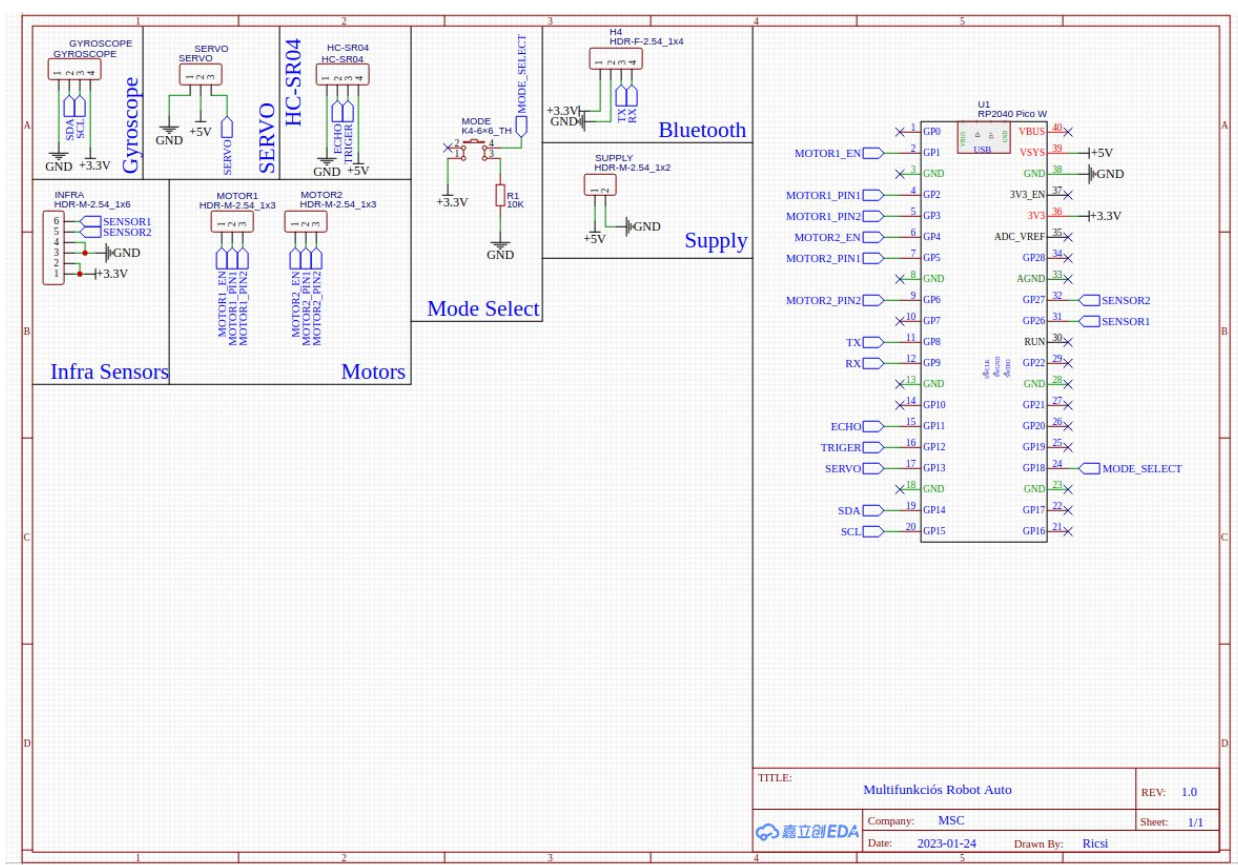
#### I<sup>2</sup>C kommunikációs protokoll:

Előnye hogy rengeteg eszköz között folyhat a kommunikáció(max 128 eszköz). Az I<sup>2</sup>C kommunikációban van egy úgynevezett maser(főnök) amely kommunikál a slave-el(szolga). Kettő vonalon folyik a kommunikáció SDA(Serial Data Line - Adatvonal) és a SCL(Serial Clock line - órajel) vonalon. A kettő vonalat fel kell húzni egy-egy ellenálláson(1k $\Omega$  - 10k $\Omega$ ) keresztül üzemi feszültségre(pl. 3.3V). A master kezdeményezi a kommunikációt, cím alapján(bele van építve az eszköz hardware-be) meg szólítja az eszközt, ezáltal tud küldeni vagy olvasni ki adatot. Hátránya hogy lassú, emellett half-duplex módban tud kommunikálni.

## 2.7 Illesztőkártya

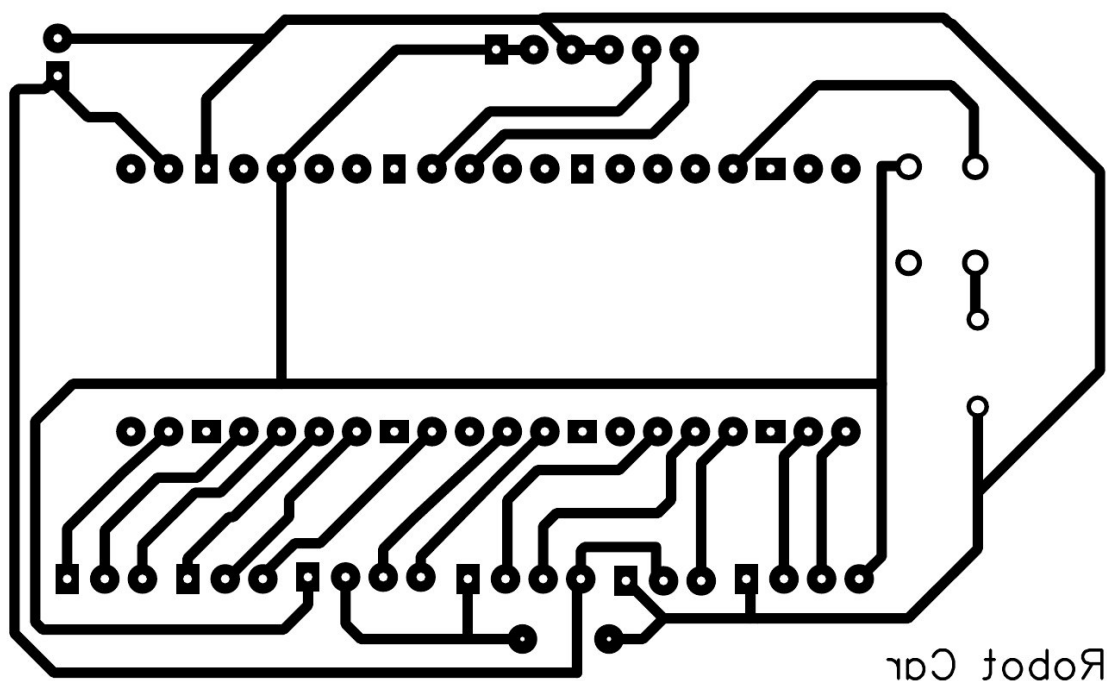
Illesztőkártyának az a feladat hogy, megteremtse a beavatkozók és a szenzorok közti kapcsolatot a mikrovezérlővel. A mód választó gomb is ezen a panelen kapott helyett. A mikrovezérlő és különböző perifériák tápellátása is ezen a panelen lett megvalósítva. Különböző csatlakozokon keresztül csatlakoznak a perifériák a mikrovezérlőhöz.

### Kapcsolási Rajz:

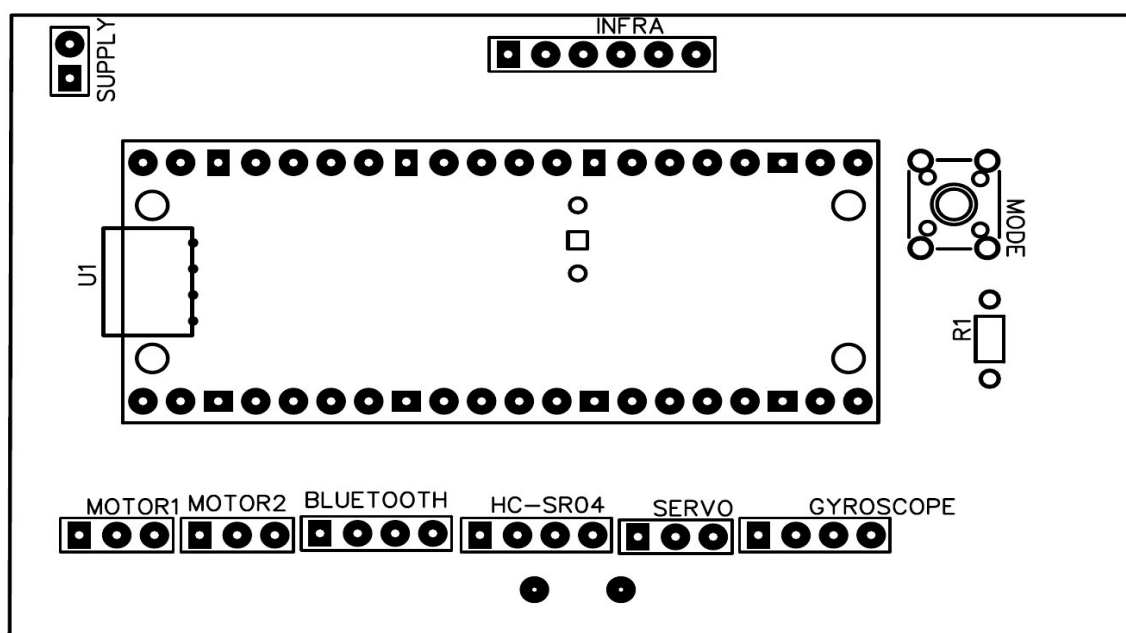




## Nyomtatott Áramköri Rajz:



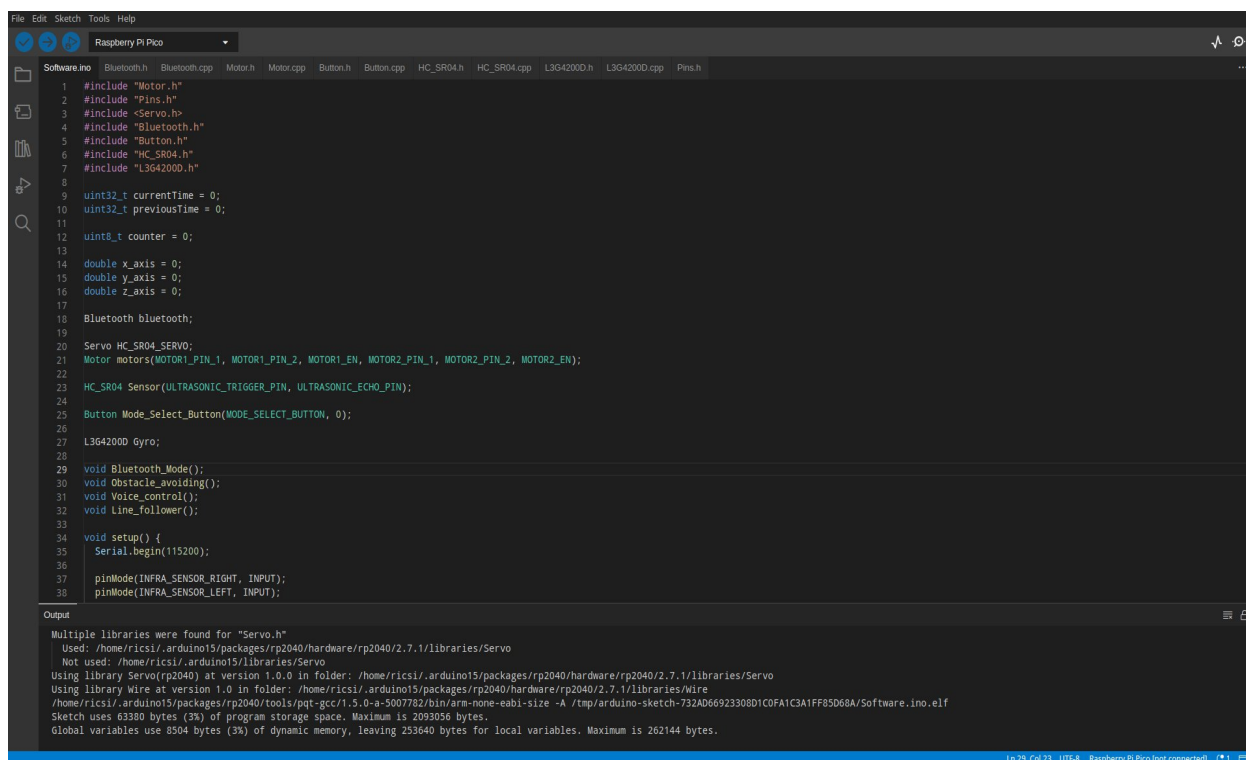
## Beültetése Rajz:



## 3 Software

### 3.1 Arduino Platform

Az Arduino egy szabad szoftveres elektronikai fejlesztőplatform, amely megkönnyebíti, felgyorsítja a fejlesztést, mert nem kell nekünk minden mikrovezérlőt natívan programozni (Assembly, vagy C-ben regiszter szinten) mivel kínál nekünk egy úgy nevezet hardware abstraction layer-et (HAL - hardveres absztrakciós réteg). Az Arduino rengeteg könyvtárral és be épített függvényekkel rendelkezik, ezért meg könnyíti a fejlesztést. Az Arduino IDE más lapokat is támogat (pl. ESP mikrovezérlők, Raspberry Pi Pico). A fejlesztő környezetben ha nem Arduino lapot szeretnénk használni akkor azt telepítenünk kell a Boardmanagerből (Alaplapkezelőből). Ha külső library-t (könyvtár) szeretnénk használni azt a librarymanager-ből (könyvtár kezelő) tudunk hozzá adni új library-t. Ezek mellett rengeteg be épített példa programmal is rendelkezik, aminek segítségével könnyen meg tanulhatunk Arduino keretrendszerben mikrovezérlőt programozni.



```
File Edit Sketch Tools Help
Raspberry Pi Pico

Software.ino Bluetooth.h Bluetooth.cpp Motor.h Motor.cpp Button.h Button.cpp HC_SR04.h HC_SR04.cpp L3G4200D.h L3G4200D.cpp Pins.h

1 #include "Motor.h"
2 #include "Pins.h"
3 #include <Servo.h>
4 #include "Bluetooth.h"
5 #include "Button.h"
6 #include "HC_SR04.h"
7 #include "L3G4200D.h"
8
9 uint32_t currentTime = 0;
10 uint32_t previousTime = 0;
11
12 uint_t counter = 0;
13
14 double x_axis = 0;
15 double y_axis = 0;
16 double z_axis = 0;
17
18 Bluetooth bluetooth;
19
20 Servo HC_SR04_SERVO;
21 Motor motors(MOTOR1_PIN_1, MOTOR1_PIN_2, MOTOR1_EN, MOTOR2_PIN_1, MOTOR2_PIN_2, MOTOR2_EN);
22
23 HC_SR04 Sensor(ULTRASONIC_TRIGGER_PIN, ULTRASONIC_ECHO_PIN);
24
25 Button Mode_Select_Button(MODE_SELECT_BUTTON, 0);
26
27 L3G4200D Gyro;
28
29 void Bluetooth_Mode();
30 void Obstacle_avoiding();
31 void Voice_control();
32 void Line_follower();
33
34 void setup() {
35   Serial.begin(115200);
36
37   pinMode(INFRA_SENSOR_RIGHT, INPUT);
38   pinMode(INFRA_SENSOR_LEFT, INPUT);
39 }
40
41 void loop() {
42   // Your code here
43 }
```

Output

Multiple libraries were found for "Servo.h"  
Used: /home/ricsi/.arduino15/packages/rp2040/hardware/rp2040/2.7.1/libraries/Servo  
Not used: /home/ricsi/.arduino15/libraries/Servo  
Using library Servo(rp2040) at version 1.0.0 in folder: /home/ricsi/.arduino15/packages/rp2040/hardware/rp2040/2.7.1/libraries/Servo  
Using library Wire at version 1.0 in folder: /home/ricsi/.arduino15/packages/rp2040/hardware/rp2040/2.7.1/libraries/Wire  
Sketch uses 63880 bytes (3%) of program storage space. Maximum is 2093056 bytes.  
Global variables use 8504 bytes (3%) of dynamic memory, leaving 253640 bytes for local variables. Maximum is 262144 bytes.

Ln 29, Col 23 UTF-8 Raspberry Pi Pico [not connected]

### 3.1 Drivereik

A szoftverben az Arduino HAL rétegét használtam a különböző szenzorokhoz és a beavatkozókhoz a drivereik elkészítéséhez.

A saját driver fejlesztés sokkal előnyösebb mint ha külső könyvtárakat használnánk a különböző perifériákhoz. A külső könyvtárak össze akadtatnak más könyvtárakkal, illetve ha nem lássuk át a hardver működését, akkor a későbbiekben nehezebb lesz hibát kereseni a programban és a hardverben egyaránt. Illetve a külső könyvtárak tartalmazhatnak hibákat is, ez által még nehezebb lehet meg találni a végső termékben a hibát.

A külső könyvtár használat akkor lehet előnyös, ha csak tesztelni szeretnénk az adott perifériát(pl. jó hardvert terveztünk-e a perifériához), illetve meg könnyebíti és felgyorsíthatja a fejlesztést mivel nem kell nekünk kézzel meg írni az összes drivert, nem kell ismernünk az adott hardvert mélyen.

Egy darab periférián(Szervó motor) kívül az összehez saját fejlesztésű drivert használtam. A következő perifériákhoz írtam drivert:

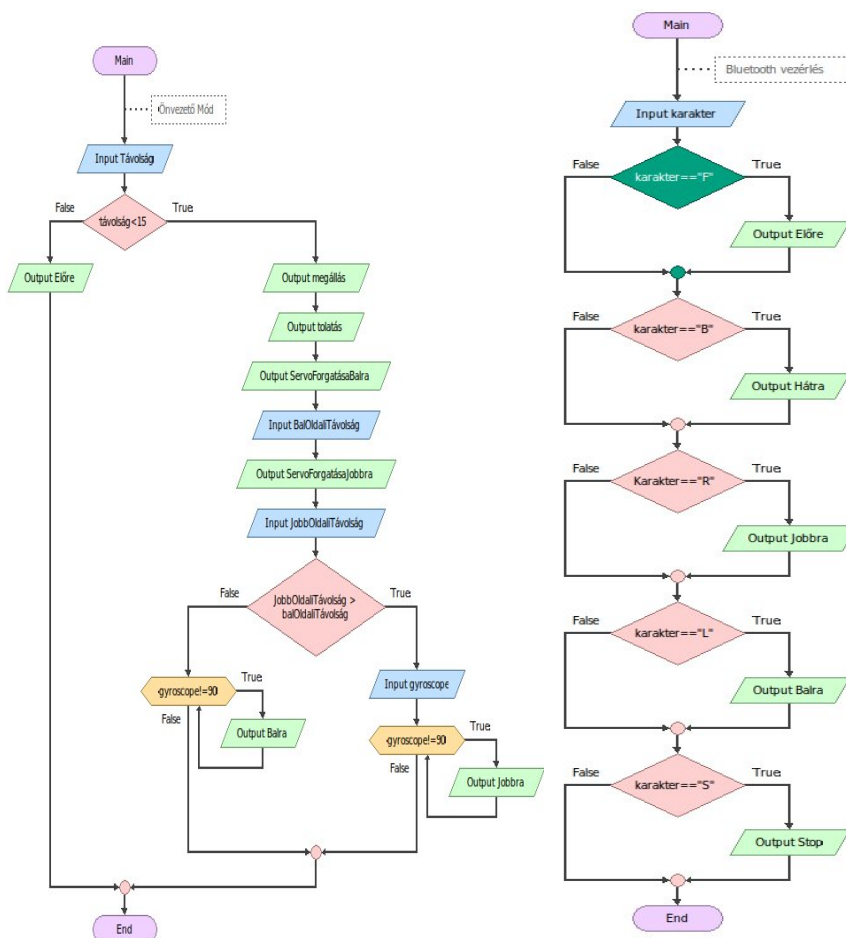
- L298N Motor Driver
- Bluetooth module
- Gomb
- HC-SR04 szenzor
- L3G4200 Szenzor
- Infra Szenzor

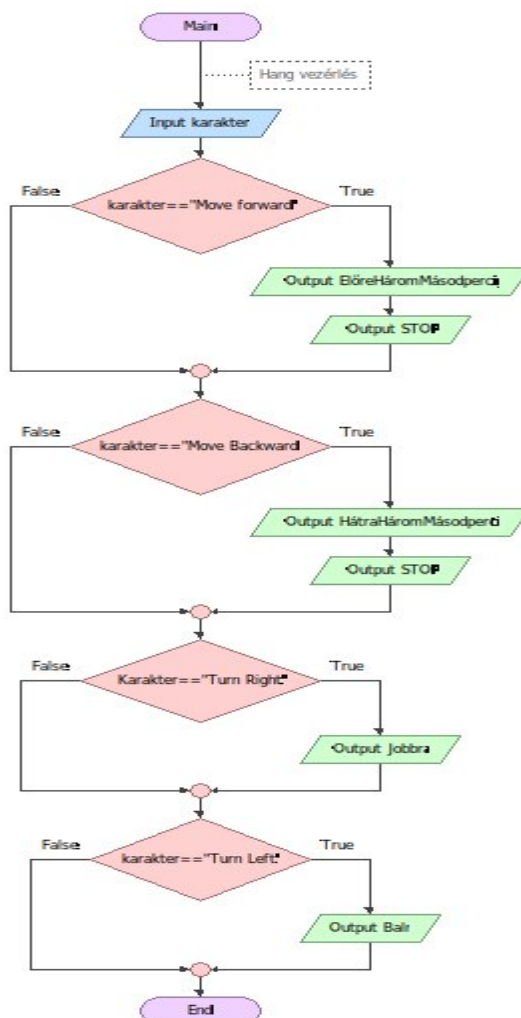
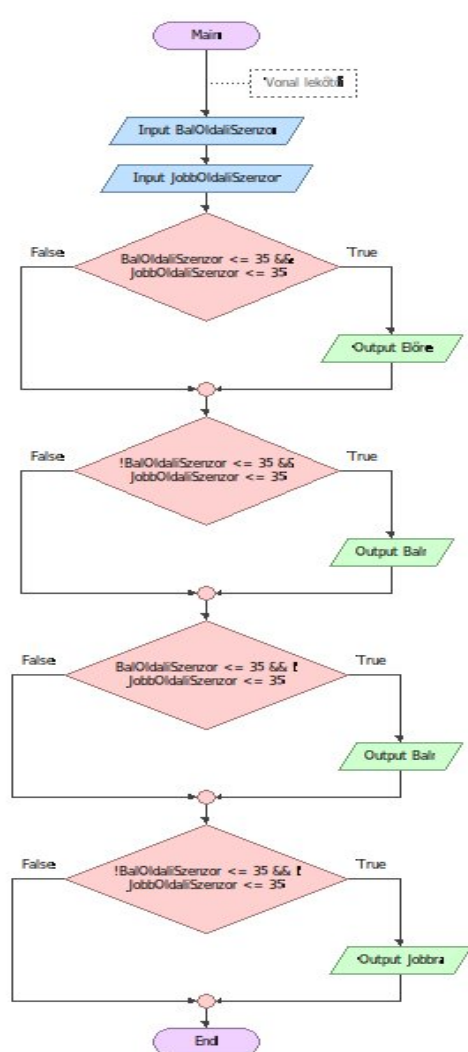
## 3.1 Software Felépítése

A softver a következő négy részből áll:

- Bluetooth vezérlés
- Önvezető Mód
- Vonal le követés
- Hang vezeérlés

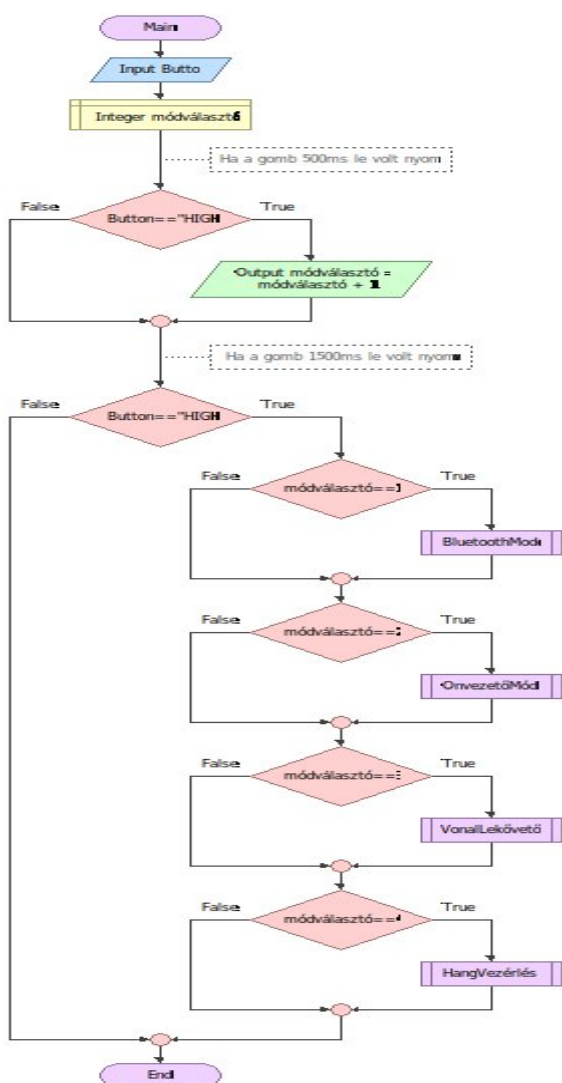
A funkciók a működését folyamat ábrán keresztül fogom szemléltetni működésüket. A software forráskódja a következő link elérhető: [https://drive.google.com/drive/folders/1-aY4UsCHaUzxW4le7QQCBdGUezGxioES?usp=share\\_link](https://drive.google.com/drive/folders/1-aY4UsCHaUzxW4le7QQCBdGUezGxioES?usp=share_link)





## 4. Robot működési elve

A Mobil roboton egy nyomógomb segítségével tudunk módot választani. A mód választás a pedig annak függvényében működik hogy hányszor nyomtuk le a nyomógombot(maximum négyszer egymás után), és mi után meg van a ki választott mód egy hosszanti nyomás segítségével be léphetünk az adott program módba. Ha a felhasználó szeretne ki lépni az adott módból azt nagyon könnyen meg teheti, ha le nyomja még egyszer a mód választó modott akkor sikeresen ki lépett az adott módból a robot. Ezt mutatja be az alábbi folyamatábra:





MŰSZAKI ISKOLA ADA

---

## 5 Irodalom Jegyzék