

MŰSZAKI ISKOLA ADA



Multifunkciós robot autó

Tanuló:

Nagy Richárd

IV/3 osztály – Számítógépek Elektrotechnikusa

Mentor:

Fekete Lajos



Tartalom

1. Bevezető.....	1
2. A hardver felépítése.....	2
2.1 RP2040.....	2
2.2 Motorvezérlő és PWM.....	3
2.3 Távolságérzékelő szenzor.....	4
2.4 Infraszensor.....	4
2.5 Bluetooth modul.....	5
2.6 Giroszkóp.....	5
2.7 Illesztőkártya.....	6
3. Szoftver.....	9
3.1 Arduino platform.....	9
3.2 Driverekek.....	10
3.3 A szoftver felépítése.....	11
4. A robot működési elve.....	13
5. Irodalomjegyzék.....	14

1. Bevezető

Napjainkban a mobil robotok egyre inkább jelen vannak életünk számos területén. Megtalálhatjuk őket az autópárhban, az egészségügyben, a modern hadviselésben, az úrkutatásban és a szórakoztatópárhban 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 utolsó sorban a mobil robotoknál hatékonyan alkalmazható irányítástechnikai megoldások keresésére.

A munka eredményeként a gyakorlatban megvalósult egy négy keréken guruló mobil robot. A robotvezérlőt egy RP2040 mikrovezérlővel rendelkező lap, Raspberry Pi Pico alkotja. A struktúra további elemei: négy egyenáramú motor, egy szervomotor, egy távolságérzékelő szenzor, két infraszzenzor, három darab akkumulátor, egy H-hidas motorvezérlő, egy nyomógomb, egy Bluetooth modul, illetve egy giroszkóp. 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 feltölteni a mikrovezérlőre, ha programot szeretnénk váltani. A robot négy darab programmal rendelkezik: Bluetooth vezérlés, önvezető mód, vonalkövető mód, illetve hangvezérléssel is rendelkezik.

A dokumentáció bemutatja a robot működési elvét, a szoftver működési elvét, az illesztőkártya kapcsolási rajzát és nyomtatott áramköri rajzát, a beültetési rajzot és nem utolsó sorban az alkatrészek működési elvét.

2. A hardver felépítése

2.1 RP2040

A munka megvalósításához alkalmazott, választott beágyazott rendszer a Raspberry Pi Pico. A Raspberry Pi Pico egy RP2040-re épülő mikrovezérlő, szoftveres elektronikai fejlesztőplatform, arra tervezve, hogy különböző projektekben könnyebben hozzáférhetőek legyenek, kezelhetőek legyenek a perifériák. 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 környezetből (IDE) és a Raspberry Pi Pico-ból áll. Az elkészített programok könnyedén feltölthetők a mikrovezérlőre USB-n keresztül. A szoftveres 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, harminckét bites ARM architektúrával és huszonnyolc multifunkciós (GPIO) lábbal is rendelkezik.

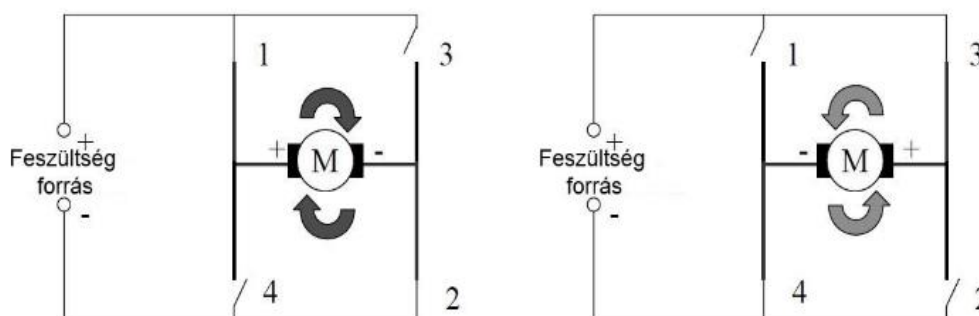
Technikai paraméterei:

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

2.2 Motorvezérlő és PWM

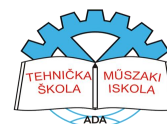
A PWM analóg tápkörök elkerülésére szolgáló módszer, mely kihasználja azt a tényt, hogy a mechanikai rendszerek bizonyos késéssel működnek. Ez egy folyamatos kimeneti jel helyett teljes rendszerfeszültségen (pl. 3.3V) működő digitális pulzusokat generál egy fix frekvencián (pl. 20kHz). Az impulzus szélességének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

A legtöbb alkalmazásnál két dolgot akarunk megvalósítani a motorokkal: előre- és hátrafelé lehessen meghajtani, illetve lehessen szabályozni a sebességét. H-híd kapcsolásra van szükségünk ahhoz, hogy a motort meg tudjuk hajtani előre- és hátrafelé.



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

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



2.3 Távolságérzékelő szenzor

Távolságérzékelő szenzornak HC-SR04 ultrahangos szenzor modult használtam, mivel a szenzor könnyedén használható mikrovezérlővel, illetve nagy távolságban képes érzékelni a távolságot, akár 4 méterre 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 visszaérkezett (ultrahangos vevőhöz) magas jelszinten, a magas szint ideje az az idő, amely az ultrahang küldése és fogadása között eltelt. Kiszámítani a következő képlettel lehet centiméterben:

$$\text{Távolság} = (\text{eltelt idő} * \text{hangsebesség}) / 2$$

eltelt idő – amíg a bemenet magas szinten van (az adó jel küldése és a vevő jel érkezése közötti idő)

hangsebesség – értéke 0.034

2 – ultrahang útja

2.4 Infraszenzor

A munkához két darab infraszenzorra is szükség van, mivel elengedhetetlen a vonalkövető programnál.

A szenzoron két darab led található, ezek pedig a következők: egy IR led és egy fotodióda. Az infravörös adó folyamatosan bocsájt ki infravörös fényt, a vevő pedig folyamatosan ellenőrzi a visszaérkező fényt. Ha a visszaérkező fény valamilyen tárgyról érkezik vissza, akkor változik a szenzor kimenete. A modulon található egy potenciométer is, amivel be lehet állítani az érzékenységet.

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 két egyenrangú eszköz vesz részt. Az információcsere két vezetéken történik. Mindkét eszközön van egy RX (receiver - fogadó), és egy TX (transceiver - küldő) láb. Az eszközök meghatározott sebességen kommunikálnak, ezt a sebességet nevezzük baud rate-nek, ami azt fejezi ki, hogy hány bitet tudunk másodpercenként küldeni. Az adatcsere bitenként történik.

2.6 Giroszkóp

A mai eszközökben rengeteg helyen megtalálható a giroszkóp, autóiparban, mobiliparban és még sok más területen. A giroszkóp a perdületmegmaradás törvénye alapján működik.

A munkámhoz L3G4200D szenzort használtam, ami egy háromtengelyes giroszkóp. A szenzorral lehet SPI illetve I²C kommunikálni, ebben az esetben I²C kommunikációt használtam. A szenzor a pontos szögelfordulás miatt került felhasználásra.

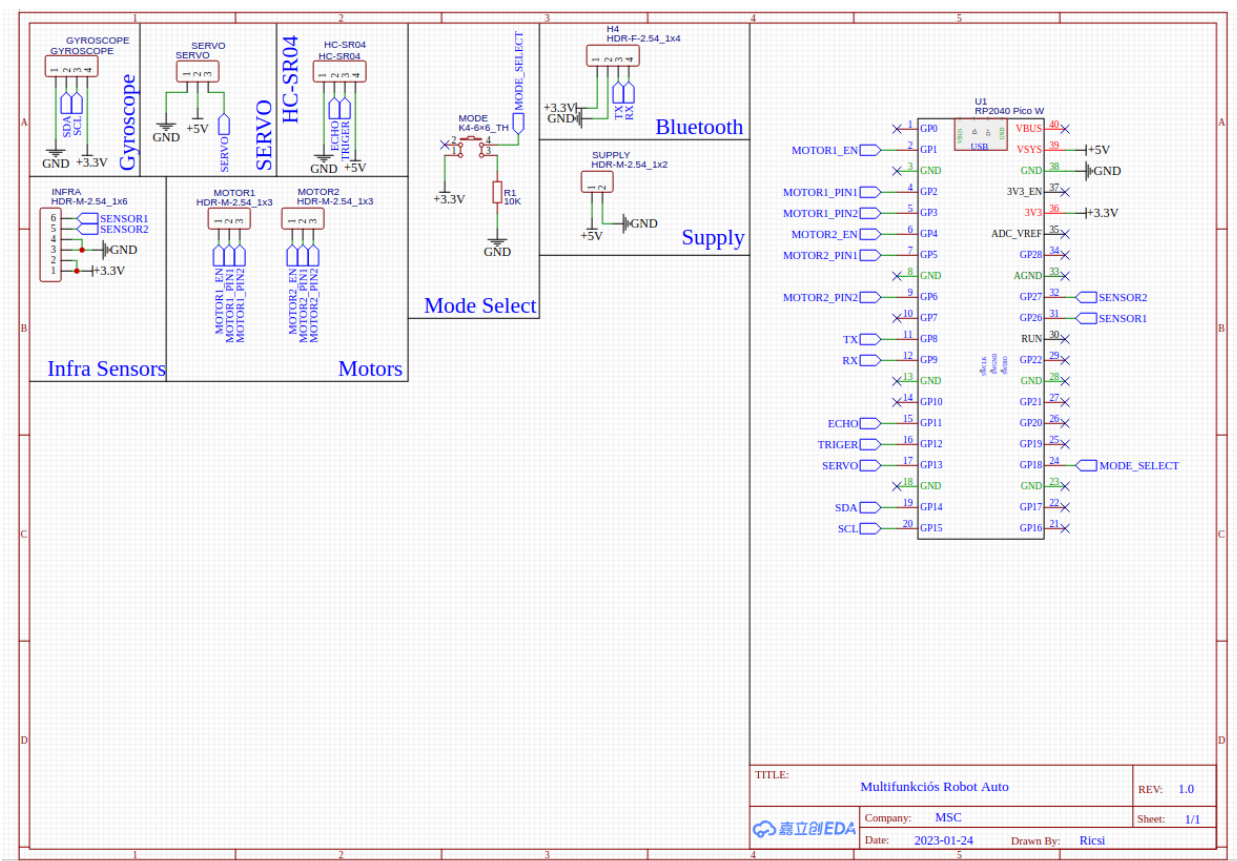
I²C kommunikációs protokoll:

Előnye, hogy rengeteg eszköz között folyhat a kommunikáció (max. 128 eszköz). Az I²C kommunikációban van egy úgynevezett master (főnök), amely kommunikál a slave-vel (szolga). Két vonalon folyik a kommunikáció: SDA (Serial Data Line - adatvonal) és a SCL (Serial Clock Line - órajel) vonalon. A két vonalat fel kell húzni egy-egy ellenálláson (1k Ω - 10k Ω) keresztül üzemi feszültségre (pl. 3.3V). A master kezdeményezi a kommunikációt, cím alapján (bele van égetve az eszköz hardverébe) megszólítja az eszközt, ezáltal tud küldeni vagy kiolvasni adatot. Hátránya, hogy lassú, emellett half-duplex módban tud kommunikálni.

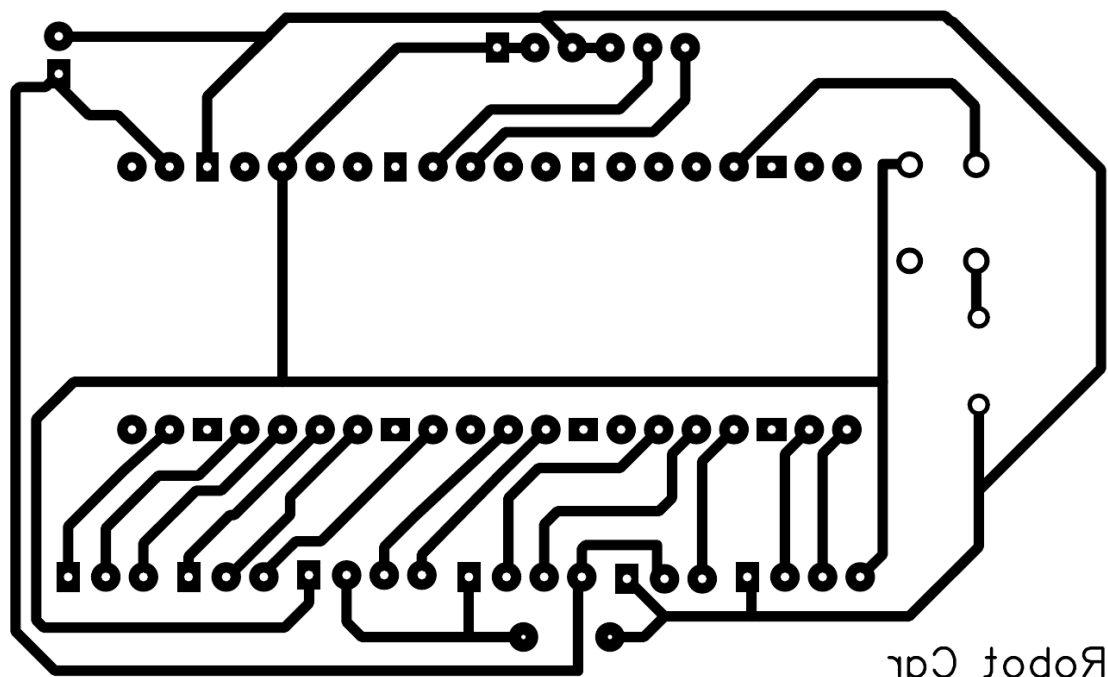
2.7 Illesztőkártya

Az illesztőkártyának az a feladata, hogy megteremtse a beavatkozók és a szenzorok közti kapcsolatot a mikrovezérlővel. A módválasztó gomb is ezen a panelen kapott helyet. A mikrovezérlő és különböző perifériák tápellátása is ezen a panelen lett megvalósítva. Különböző csatlakozókon keresztül csatlakoznak a perifériák a mikrovezérlőhöz, illetve látható, hogyan csatlakoznak a perifériák az illesztőkártyához.

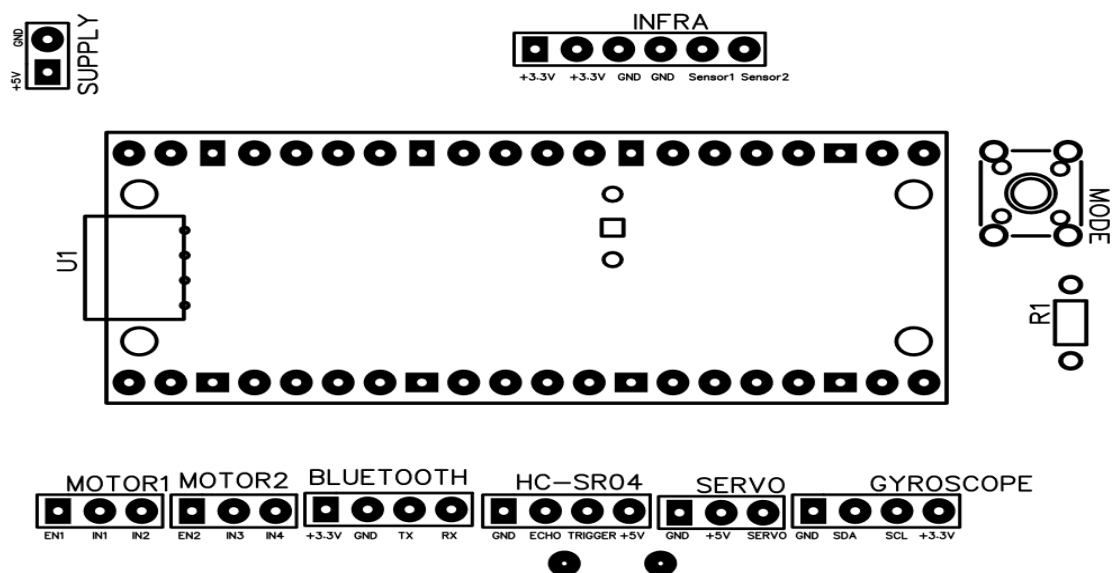
Kapcsolási rajz:



Nyomtatott áramköri rajz:

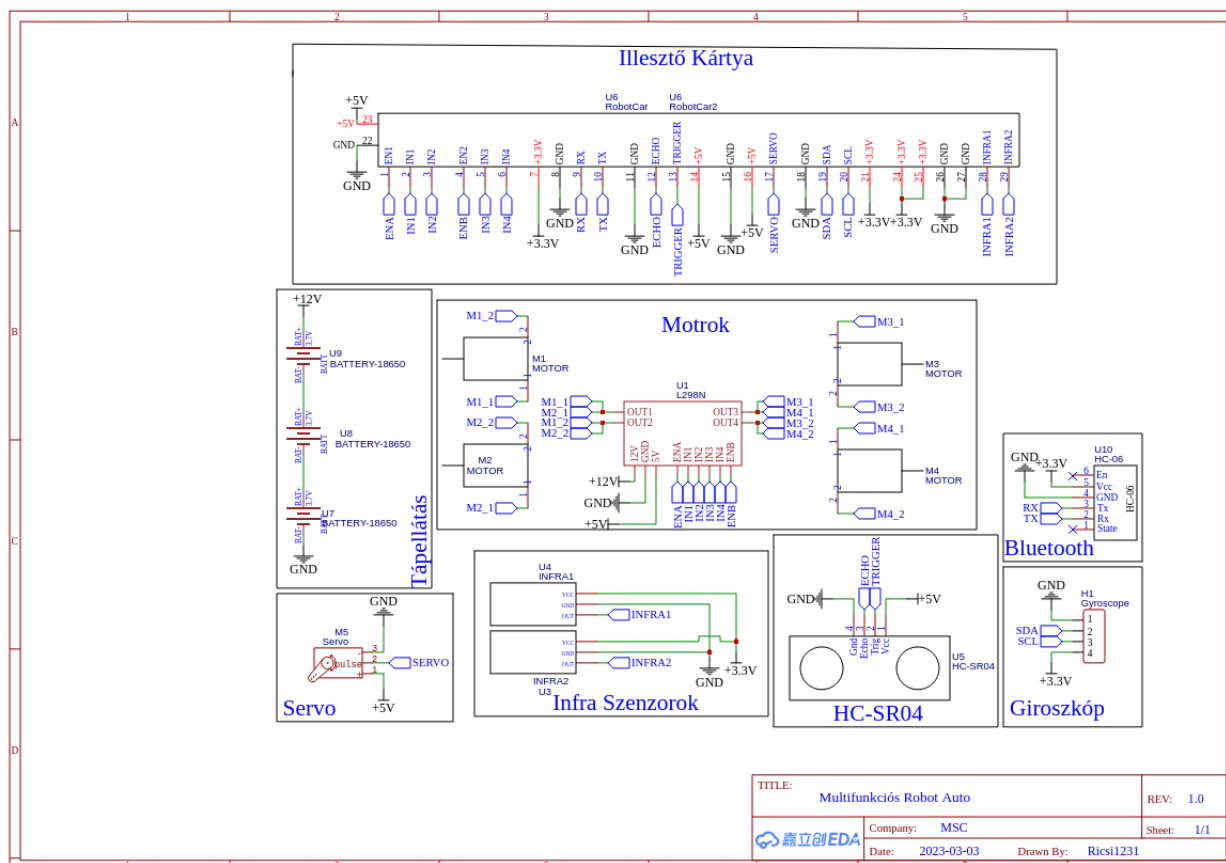


Beültetési rajz:



MÚSZAKI ISKOLA ADA

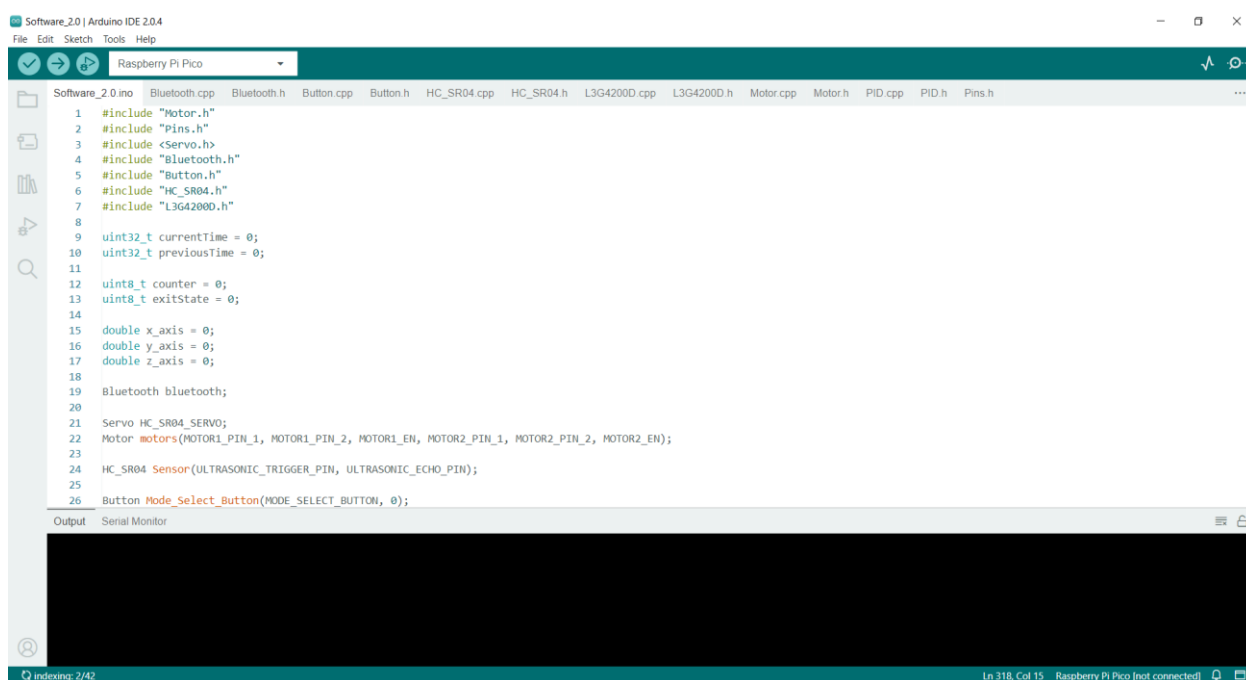
Illesztőkártya - perifériák csatlakozása:



3. Szoftver

3.1 Arduino platform

Az Arduino egy szabad szoftveres elektronikai fejlesztőplatform, amely megkönnyíti, felgyorsítja a fejlesztést, mert nem kell minden mikrovezérlőt natívan programoznunk (Assembly, vagy C-ben regiszter szinten), mivel kínál nekünk egy úgynevezett hardware abstraction layer-t (HAL - hardveres absztrakciós réteg). Az Arduino rengeteg könyvtárral és beépített függvényekkel rendelkezik, ezért megkönnyíti 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, akkor a library managerből (könyvtárkezelő) tudunk hozzáadni új library-t. Ezek mellett rengeteg beépített példaprogrammal is rendelkezik, amelyek segítségével könnyen megtanulhatunk Arduino keretrendszerben mikrovezérlőt programozni.



```
Software_2.0.ino | Bluetooth.cpp | Bluetooth.h | Button.cpp | Button.h | HC_SR04.cpp | HC_SR04.h | L3G4200D.cpp | L3G4200D.h | Motor.cpp | Motor.h | PID.cpp | PID.h | 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 uint8_t counter = 0;
13 uint8_t exitState = 0;
14
15 double x_axis = 0;
16 double y_axis = 0;
17 double z_axis = 0;
18
19 Bluetooth bluetooth;
20
21 Servo HC_SR04_SERVO;
22 Motor motors(MOTOR1_PIN_1, MOTOR1_PIN_2, MOTOR1_EN, MOTOR2_PIN_1, MOTOR2_PIN_2, MOTOR2_EN);
23
24 HC_SR04 Sensor(ULTRASONIC_TRIGGER_PIN, ULTRASONIC_ECHO_PIN);
25
26 Button Mode_Select_Button(MODE_SELECT_BUTTON, 0);
```

3.1 Driverrek

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 összeakadhatnak más könyvtárakkal, illetve, ha nem látjuk át a hardver működését, akkor a későbbiekben nehezebb lesz hibát keresni a programban és a hardverben egyaránt. Illetve a külső könyvtárak tartalmazhatnak hibákat is, ezáltal még nehezebb lehet megtalálni a végső termékben a hibát.

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

Egy darab periférián (szervomotor) kívül az összeshez saját fejlesztésű drivert használtam. A következő perifériákhoz írtam drivert:

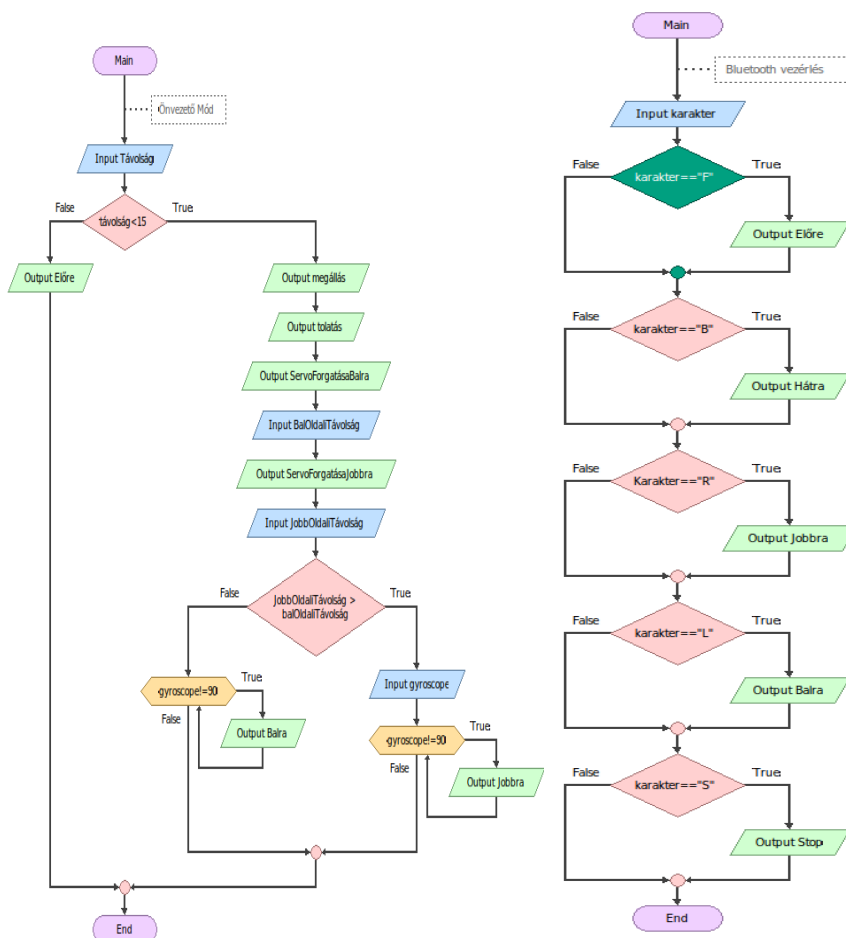
- L298N Motor Driver
- Bluetooth module
- Gomb
- HC-SR04 szenzor
- L3G4200 szenzor
- Infraszennzor

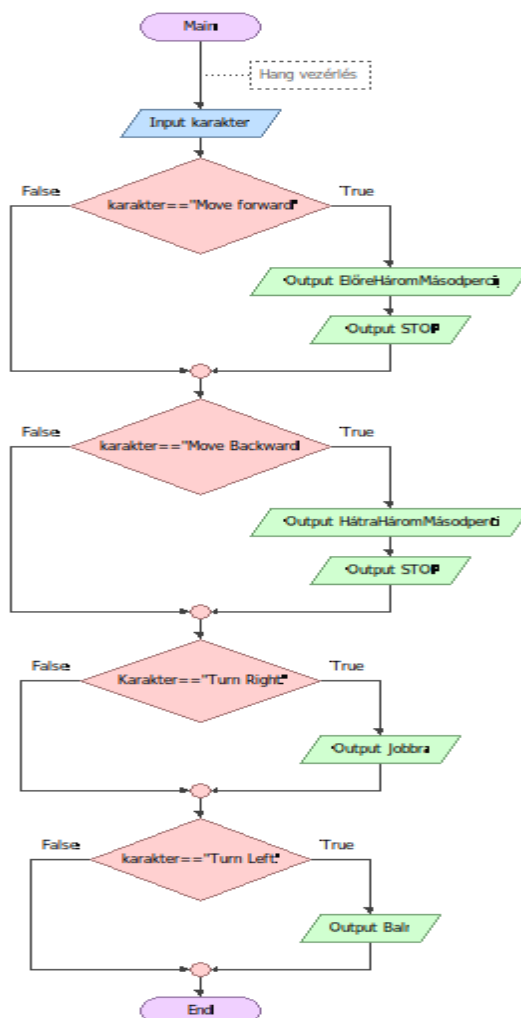
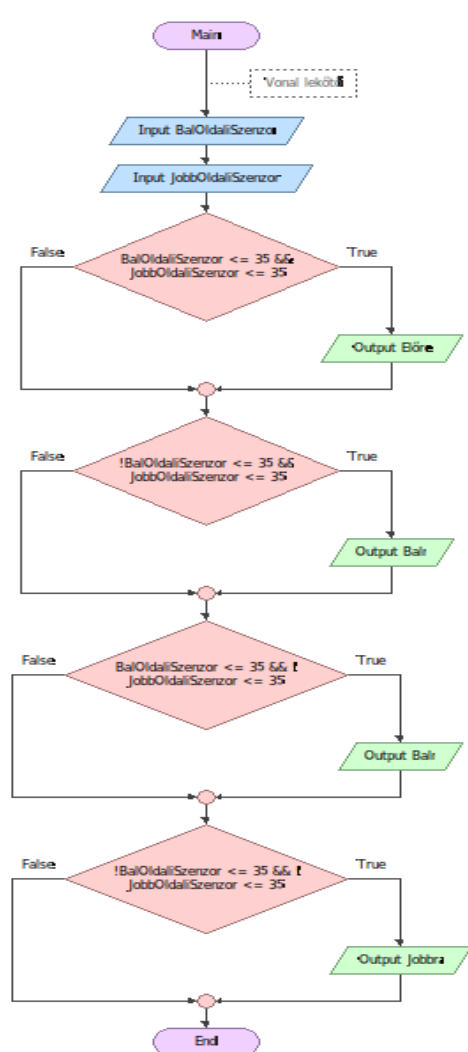
3.1 A szoftver felépítése

A szoftver a következő négy részből áll:

- Bluetooth vezérlés
- Önvezető mód
- Vonalkövetés
- Hangvezérlés

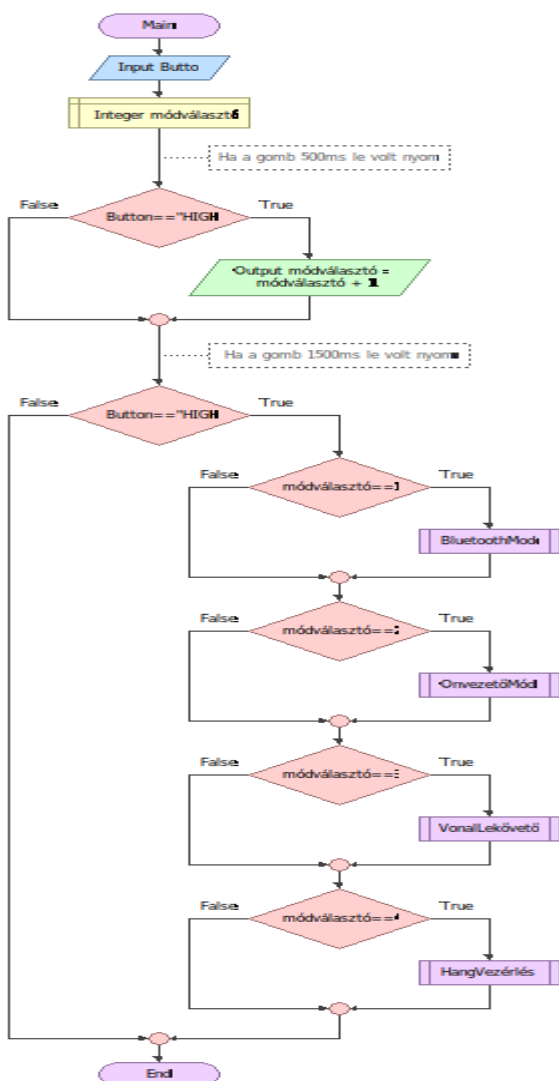
A funkciók a működését folyamatábrán keresztül fogom szemléltetni. A szoftver forráskódja a következő linken érhető el: https://drive.google.com/drive/folders/1-aY4UsCHaUzxW4le7QQCBdGUezGxioES?usp=share_link





4. A robot működési elve

A mobil roboton egy nyomógomb segítségével tudunk módot választani. A mód választása pedig annak függvényében működik, hogy hányszor nyomtuk le a nyomógombot (maximum négyszer egymás után). Miután megvan a kiválasztott mód, egy hosszanti nyomás segítségével beléphetünk az adott programmódba. Ha a felhasználó szeretne kilépni az adott módból, azt nagyon könnyen megteheti, ha lenyomja még egyszer a módválasztó gombot, akkor sikeresen kilépett az adott módból a robot. Ezt mutatja be az alábbi folyamatábra:



5. Irodalomjegyzék

<https://www.arduino.cc/>

https://www.sparkfun.com/datasheets/Robotics/L298_H_Bridge.pdf

https://components101.com/sites/default/files/component_datasheet/HC-05%20Datasheet.pdf

<https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf>

https://www.elecrow.com/download/L3G4200_AN3393.pdf

https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwj5wd3gjNb9AhWxhP0HHSsiBj0QFnoECAoQAQ&url=https%3A%2F%2F5.imimg.com%2Fdata5%2FYT%2FKV%2FMY-1833510%2Farduino-ir-infrared-obstacle-avoidance-sensor-module.pdf&usq=AOvVaw03lxw3_Hvfr8v83ly3RxX8

<https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf>

<https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf>