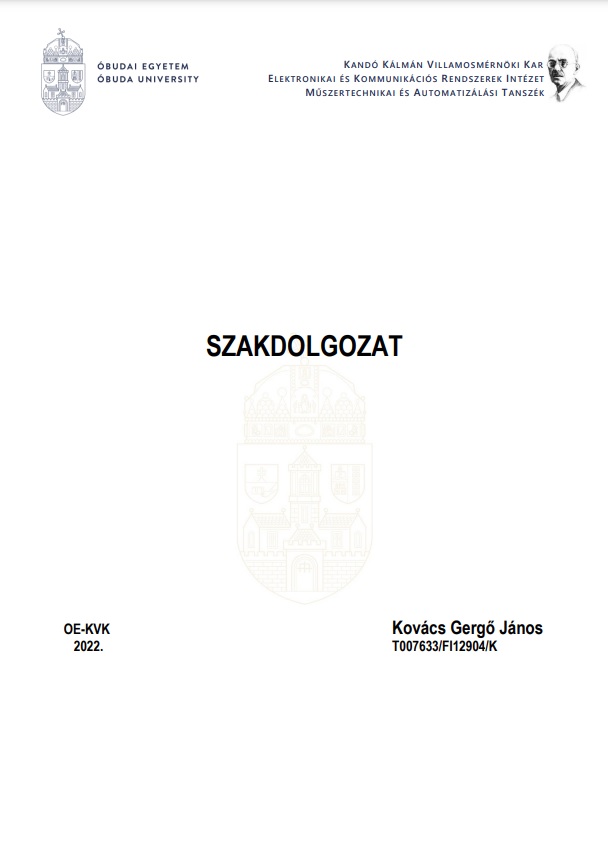
****

**Köszönetnyilvánítás**

*Szeretném megköszönni az Óbudai Egyetem Kandó Kálmán Villamosmérnöki Karának, hogy rendelkezésünkre bocsátotta a delta robotot, ezzel is támogatva a projektünk megvalósulását.*

*Köszönöm Sándor Tamás Tanár Úrnak és Borsos Döníz Tanárnőnek azt a rengeteg segítséget és tanácsot, amit a projekt és az egyetemi évek alatt kaptam tőlük.*

*Köszönöm Schmidt Péternek a projekten való közös munkát. Peti, egy élmény veled dolgozni!*

*Köszönöm szüleimnek, hogy mindig mellettem vannak, és minden formában támogatnak, hogy elérjem álmaim és céljaim.*

**Tartalomjegyzék**

[1 Bevezetés 7](#_Toc119954003)

[2 Az ipari robotok 8](#_Toc119954004)

[2.1 Történelmi háttér 8](#_Toc119954005)

[2.2 Az ipari robotok fogalma, csoportosításuk 11](#_Toc119954006)

[2.3 A delta robot 14](#_Toc119954007)

[3 Specifikáció 17](#_Toc119954008)

[4 Logikai rendszerterv 19](#_Toc119954009)

[5 Fizikai rendszerterv 21](#_Toc119954010)

[5.1 Az első verzió 21](#_Toc119954011)

[5.2 A módosított, végleges verzió 24](#_Toc119954012)

[6 Motorvezérlő modul 27](#_Toc119954013)

[6.1 A léptető motorok 27](#_Toc119954014)

[6.2 A motorvezérlők 28](#_Toc119954015)

[6.3 Visszacsatolás a karok pozíciójáról 30](#_Toc119954016)

[7 Kommunikációk 34](#_Toc119954017)

[7.1 SPI kommunikáció 34](#_Toc119954018)

[7.2 Wi-fi kommunikáció 36](#_Toc119954019)

[8 Motorvezérlő program 38](#_Toc119954020)

[8.1 Makrók és változók definiálása 38](#_Toc119954021)

[8.2 Motorok beállításához szükséges lépésszám számítása 40](#_Toc119954022)

[8.3 A főprogram 42](#_Toc119954023)

[9 Tesztelés 46](#_Toc119954024)

[10 Összefoglaló 49](#_Toc119954025)

[11 Summary 50](#_Toc119954026)

[Irodalomjegyzék 51](#_Toc119954027)

[Mellékletek 57](#_Toc119954028)

[Ábrajegyzék 58](#_Toc119954029)

# Bevezetés

A szakdolgozatomban részletezett feladattal Projektmunka II. tárgy keretein belül kezdtem el dolgozni szaktársammal. Egy mondhatni nagyobb projekten dolgoztunk ketten, így a feladat során több kihívással kellett szembe néznünk. Azonban így több tématerületbe is betekintést nyerhettünk, amellyel sok új ismeretre tettünk szert.

Projektünk az alábbi modulokra bontható fel:

* Motorvezérlő modul
* Kommunikáció modul
* Képfeldolgozó modul
* Sakk modul

Ugyan a projekt során a modulokon közösen dolgoztunk, segítettük egymás munkáját, én főként a motorvezérlő modullal, illetve a kommunikáció modullal foglalkoztam. A másik két modulról Schmidt Péter szakdolgozatában lehet olvasni.

Először is a hazai és nemzetközi szakirodalmak feldolgozásának segítségével röviden szeretném bemutatni az ipari robotok fejlődésének történelmét (főbb mérföldköveit), az ipari robotok fogalmát, csoportosításukat, részletezni magáról a delta robotról fontos információkat. Ezután térnék rá feladatunk specifikációjára, majd a specifikáció alapján elkészült logikai, illetve fizikai rendszertervre. Ezek ismertetése után következik a korábban említett két modul bemutatása, illetve ezek tesztelése. Végül egy rövid összefoglalóban írom le a végeredményt, beszélek a tovább fejlesztés lehetőségéről és a projekthez köthető élményekről.

# Az ipari robotok

## Történelmi háttér

Az ipari robotok szerepe az elmúlt évtizedek alatt drasztikusan megnövekedett a gyártásban. Segítségükkel egy termék legyártása sokkal gyorsabb, olyan feladatok ellátására alkalmasak, mely az ember számára már rövidebb idő után is fárasztó, monoton lehet, illetve a robotok veszélyes munkakörnyezetben is probléma nélkül működnek. Ezen felül biztosítják, hogy a gyártósorról szinte mindig ugyan olyan minőségű termék kerüljön ki egymás után, ami mind a gyártónak, mind a megrendelőnek egy fontos tényező.

Ezek a főbb szempontok indították el azt a gondolatmenetet, hogy gyártás során vannak folyamatok, amelyek automatizálhatók. Viszont az egyes feladatok végrehajtásához lehet, hogy olyan eszköz szükséges, mely négy, öt, vagy akár hat szabadsági fokkal rendelkezik, vagy a folyamat végrehajtása során elengedhetetlen a nagy precizitás. A fejlesztések így ebbe az irányba indultak el, hogy olyan robotokat hozzanak létre, melyek ezeket a feltételeket teljesítik, kialakításukkal alkalmasak az ipari környezetben való munkára.

A robotika rohamos fejlődése a 20. században indult el, mivel ekkor jelentek meg olyan technológiai újítások (tranzisztor, integrált áramkörök), melyekkel nagyobb és bonyolultabb működésű robotokat lehetett elkészíteni. A robottechnológia fejlődését 4 generációra lehet osztani [1], mindegyik generációnak megvannak a maga sajátosságai, illetve mérföldkövei.

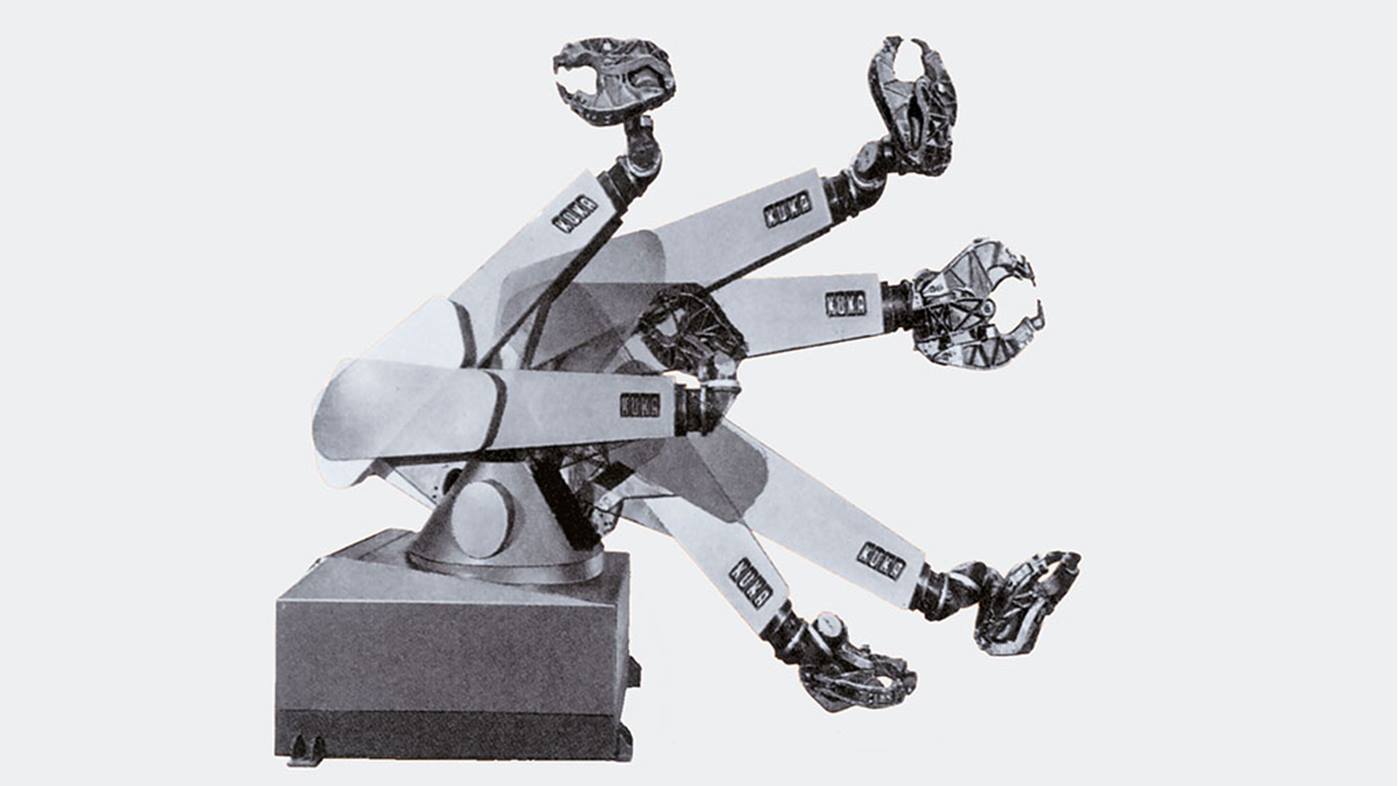
Az első generációban (1950-1967) alkalmazták az első manipulátorokat, melyeknek még nem volt tudomásuk az őket körülvevő környezetről. Ezek a manipulátorok numerikus vezérléssel működtek. Az első ipari robotot 1961-ben szabadalmaztatták, mely a Unimate (2.1. ábra) nevet kapta. Ez a robot már ipari használatra alkalmas volt, főként pozícionálási feladatokat látott el, tárgyakat tudott egyik pontból a másikba mozgatni. [1]



2.. ábra Unimate, az első szabadalmaztatott ipari robot [2]

A fejlesztések a második generációban (1968-1977) a felé irányultak, hogy a robotoknak némi ismeretük legyen környezetükről, azzal részben kapcsolatot tartsanak, és az aktuális helyzet szerint működjenek. Így jelentek meg az első olyan ipari robotok, melyeket érzékelőkkel láttak el. Képesek voltak kézi betanulásra, mely univerzálisabb működést tett lehetővé. Az első szenzoros mobil robotot, a Shakeyt (2.2. ábra), 1966 és 1972 között fejlesztették. [1] A következő nagy mérföldkő 1973-ban volt, amikor a KUKA megépítette Famulust (2.2. ábra), az első ipari robotot, mely hat elektromechanikus meghajtású tengellyel rendelkezett [4].

2.2. ábra A Shakey (bal oldalt) [3] és a Famulus (jobb oldalt) [4]



A harmadik generációs (1978-1999) ipari robotoknak már saját vezérlőjük volt, ezek legfőképpen valamilyen számítógépes egységek voltak. Új programozási nyelvek jelentek meg, melyek az ipari robotok egyszerűbb és gyorsabb programozására voltak hivatottak. Lehetőség nyílt arra, hogy az ipari robotokat újra lehessen programozni, így nem csak egy feladatot képesek ellátni egész működési idejük alatt. 1978-ban építették meg az első SCARA robotot (2.3. ábra), mely tárgyak egyik pontból a másikba való pontos és gyors áthelyezésére volt ideális. Ebben az időben indult meg a computer vision fejlesztése is, mely alkalmazhatónak bizonyult az ipari robotok működésében is.



2.. ábra Első SCARA prototípus [5]

Napjainkban (21. század) a legnagyobb fejlesztések az intelligens robotok felé irányulnak, melyek sokkal bonyolultabb számításokat végeznek el, mint ezelőtt. Ezek a robotok logikus gondolkodásra, önálló tanulásra képesek, a fejlettebb szenzoroknak köszönhetően szinte teljes információjuk van a környezetükről, az onnan begyűjtött adatok segítségével tudják a feladatokat saját döntéseik alapján megoldani. Megjelentek az olyan robotok, melyek képesek más robottal kooperatívan dolgozni, egymás feladatát nem akadályozzák. [1]

## Az ipari robotok fogalma, csoportosításuk

Az ISO 8373:2021-es szabvány megfogalmazása szerint az ipari robot:

*„Automatikusan vezérelt, újraprogramozható, univerzális manipulátor, mely 3 vagy több tengelyen képes mozogni, fix területen vagy mobil platformon van rögzítve, ipari környezetben való automatizált feladatok ellátására szolgálnak. [6]”*

A fogalom is tükrözi, hogy egy olyan komplex szerkezetről beszélünk, mellyel az iparban számos feladatot képesek vagyunk megoldani. Az, hogy a feladatot miként oldja meg, a vezérlőprogramtól függ, aminek megírása az egyre bővülő programozási nyelveknek köszönhetően gyorsabb és egyszerűbb, mint ezelőtt. Adott feladatok végrehajtására számos típusú robotot fejlesztettek ki, melyek a megfelelő felszereltséggel vannak ellátva, és kialakításukkal a feladat jellegéhez illeszkednek. Kialakítás alapján az ipari robotokat több csoportba lehet osztani.

Mozgásuk kinematikája alapján a robotok négy típusba sorolhatók. Descartes koordinátás robotok esetében a mozgás főként egyenes vonalú. Egy henger koordinátás robot két egyenes vonalú és egy forgó mozgást végez. Ha a robot egy egyenes vonalú és két forgó mozgást végez, akkor beszélünk gömbkoordinátás robotról. Humanoid típusú robotok esetében viszont nem történik egyenes vonalú mozgás. Ma főként a humanoid megoldást használják, ugyan maga az irányítása bonyolult, de mechanikai szempontból jóval egyszerűbb megoldás, mert az egyenes vonalú mozgás megvalósítása nehéz a mechanika korlátozó tényezői miatt. Descartes koordinátás robotok esetében pedig az irányítás egyszerűbb, viszont a mechanikai felépítés lényegesen bonyolultabb. [7]

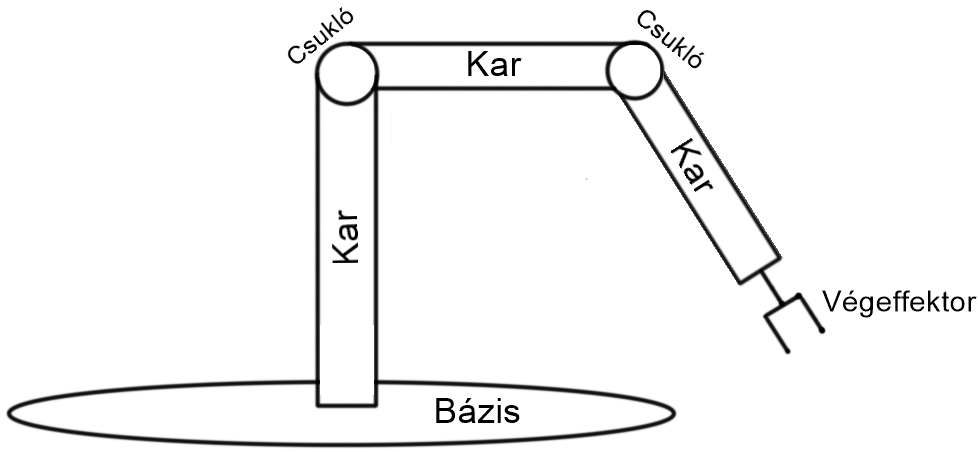
Pályakövető képesség alapján Point To Point, illetve Continous Path robotokat különböztetünk meg. Point To Point esetben a robot egyik pontból a másik pontba képes eljutni, viszont a két pont közötti pálya bizonytalan, azt nem teljesen ismerjük, és nem tudjuk pontosan megjósolni sem. Continous Path esetén egy adott pontot meghatározott pályán képes elérni a robot, pályakövetésre alkalmas.

Csoportosíthatjuk a robotokat hajtásuk szerint is. A legelterjedtebb a villamos hajtás, mivel az ilyen robotokat könnyebb irányítani. A hajtás lehet hidraulikus is, ami nagyobb tömegek mozgatására alkalmas. Azonban az ilyen típusú robotok rendszeres karbantartást igényelnek, illetve mechanikai kialakításuk miatt pontatlanabb a pozícionálásuk. [7]

Az ipari robotokat lehet csoportosítani megoldandó feladat szerint is. Megkülönböztetünk anyagmozgató robotokat, amik csak azért felelősek, hogy egy munkadarabot egyik pontból a másikba mozgassák. Léteznek heggesztő robotok, melyek méretben és terhelhetőségben is nagyobbak. Ezen belül megkülönböztetünk pontheggesztő és ívheggesztő robotokat. Az anyagmozgató és heggesztő robotokon felül vannak a szerelő robotok, ezek munkadarabokat munkálnak meg, esetleg mozgatják is őket.

A fentebb felsorolt csoportokon kívül még számos módon lehet kategorizálni az ipari robotokat. Alaptípus szerint azonban megkülönböztetünk soros, illetve párhuzamos robotokat.

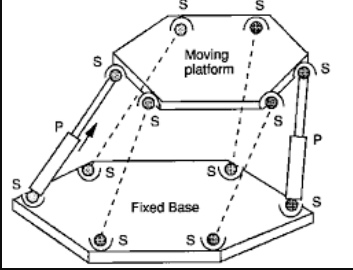
Soros manipulátorok esetében (2.4. ábra) az egyes merevtest elemek (karok) csuklókkal vannak összekötve [8]. Tehát a robot bázisától kiindulva először egy kar, majd csukló, ezután ismét egy kar következik. A maximális hossz, amelyet a robot képes elérni, növelhető további csukló-kar elemek hozzáadásával. A csukló-kar lánc végén helyezkedik el a végeffektor, mely segítségével történik az adott anyag felvétele, illetve azon való szállítása.



2.. ábra Soros manipulátor vázlata

Egy soros robot tervezésekor figyelembe kell venni, hogy az egyes csuklókra nem ugyan olyan nagyságú terhelés fog hatni. A végeffektorhoz legközelebb eső csuklóra fog a legkisebb terhelés, míg a végeffektortól legtávolabb (a robot bázisának tengelyére eső) csuklójára pedig a legnagyobb terhelés fog hatni, így a robot terhelhetőségét ezek alapján is meg kell fontolni.

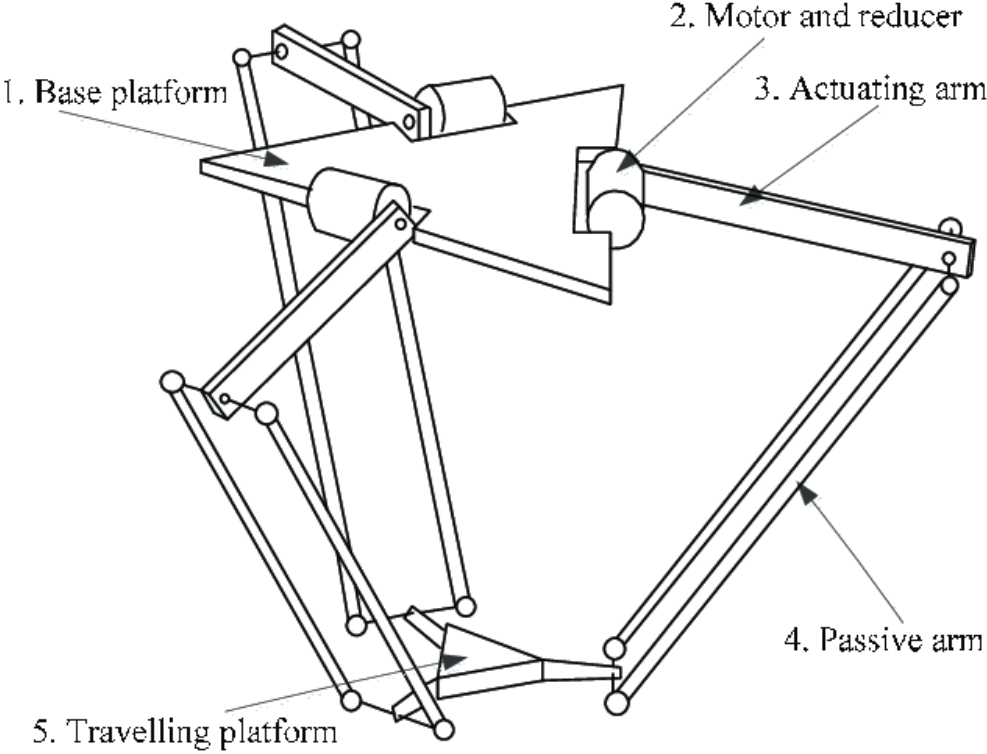
Párhuzamos robotok esetében (2.5. ábra) a végeffektorra nem egy, hanem több csukló csatlakozik. Ezen megoldás segítségével pontosabb pozícionálás, nagyobb teherbírás és stabilabb végeffektor mozgatás valósítható meg. A csuklókra ugyan olyan hosszú karok csatlakoznak, ezek határozzák meg a munkatér nagyságát. Mivel a párhuzamos robotoknál több kar csatlakozik a végeffektorhoz, ezért az azonos csuklószámú és karhosszúságú soros robotokhoz képest kisebb munkaterületet képesek bejárni. Ugyan úgy bázissal, illetve végeffektorral rendelkeznek, viszont itt az egyes csuklók a mozgás során nem egymást támogatják, hanem kizárólag a végeffektor mozgását direkt módon. A végeffektor a bázis felett, vagy alatta helyezkedik el. Ha a végeffektor a bázis alatt helyezkedik el, illetve három (vagy négy) kar támogatja annak mozgását, akkor beszélünk delta robotról.



2.. ábra Párhuzamos manipulátor vázlata [9]

## A delta robot

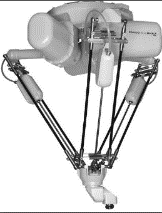
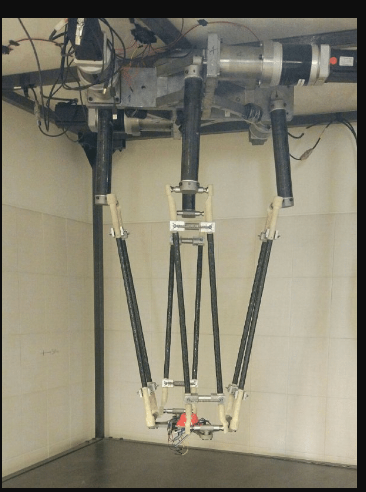
A delta robot tehát olyan párhuzamos manipulátor, amely három karral rendelkezik, és ezen karok irányítása a bázison található 3 motor segítségével történik (2.6. ábra). A motorok egymástól 120°-ra vannak elhelyezve egymástól. Az egyes motorokra egy-egy kar csatlakozik rá, ezek lesznek a felső karok (aktuátor). A felső karokra csuklók segítségével kapcsolódnak a passzív karok, melyek mozgását a felső karok segítik. A passzív karok végén található csuklók segítségével kapcsolódnak az egyes karok a végeffektorhoz, mely segítségével az anyagmozgatás történik.



2.. ábra A delta robot vázlata [10]

A passzív karok kialakításuk által egy paralelogrammát zárnak be, mely a robot mozgási mechanizmusában jelentős szerepet játszik. Ennek a kialakításnak köszönhetően a végeffektor stabil pozícióban marad, mozgása során végig párhuzamosan van a bázissal [11]. Emellett viszont a kialakítás korlátozza a robot munkaterének nagyságát, így ezek kis munkatérben elvégzendő feladatokat látnak el. A munkatér nagysága függ a robot karjainak hosszától, így a megfelelő méretek megválasztásával minimálisan lehet növelni a robot által maximálisan elérhető pont távolságát.

Mozgás kinematikáját nézve a végeffektor csak az X, Y, és Z koordináták mentén tud mozgást végezni. Alapesetben a robot szabadságfoka 3, azaz egyszerre 3 tengely mentén képes mozogni. Azonban már léteznek olyan delta robotok, melyek négy (2.7. ábra), vagy hat (2.7. ábra) szabadsági fokkal rendelkeznek, itt már a működés során megjelenik adott síkban forgó mozgás is.



2.. ábra Négy (bal oldalt) [12], illetve hat (jobb oldalt) szabadságfokú [13] delta robot

Ha egy adott pozícióba szeretnénk állítani a végeffektort (mely az X, Y, és Z koordináták mentén mozog), akkor az aktuátorokon található motorok segítségével tudjuk pozícionálni a karokat, és így a végeffektort is. Azonban a motorok forgó mozgást végeznek, így kapcsolatot kell teremteni a motorok aktuális szögállása és a végeffektor aktuális Descartes koordináta-beli pozíciója között. Ennek megoldása egy külön matematikai probléma, mely megoldásában a direkt és inverz kinematika ad támogatást.

A karokat könnyű anyagból állítják elő, így a szerkezet kis tömegének köszönhetően gyors mozgás valósítható meg vele. A delta robot karjainak könnyű szerkezete miatt azonban kisebb tömegű tárgyak (kb. 3 kg) mozgatására alkalmasak. Főként Pick and Place feladatok ellátására alkalmazzák, például valamilyen alkatrész áthelyezésére egyik pontból a másikba, illetve csomagoló feladatokat is delta robotokkal valósítanak meg.

# Specifikáció

Feladatunk során egy olyan rendszer felépítése volt a cél, mely segítségével a delta robot képes az aktuális tábla állapotának ismeretében, a sakk szabályainak megfelelően játszani. A sakk általában két személyes játék, így el kellett dönteni, hogy pontosan a robotnak ki ellen is kell játszania. Itt két választási lehetőség volt: ember ellen, vagy egy másik robot ellen. Mivel emberek számára mutatjuk be az ipari robotok alkalmazását, így egyértelmű volt, hogy a robotnak ember ellen kell játszania. Így a nézők egy interaktív és szórakoztató módon ismerhetik meg a robot működését.

Ennek megfelelően a játék menete a következő:

* A felhasználó megteszi lépését a sakk szabályoknak megfelelően. Ha végzett lépésével, azt a roboton rögzített gombbal tudja jelezni.
* A gomb lenyomásával a rendszer egy képet készít a sakktábláról, ezzel olvassa be annak aktuális állását.
* Ezután egy ellenőrzés következik, hogy a felhasználó lépése tényleg szabályos volt. Ellenkező esetben visszajelzés kerül kiküldésre ezzel kapcsolatban, és a játékosnak meg kell ismételnie lépését.
* Szabályos lépés esetén egy válaszlépés kerül kiszámításra.
* A számítást követően történik a lépés megtétele. A lépés befejeztével a rendszer visszajelzést küld a felhasználónak. Visszajelzés lehet arról, hogy ő következik, vagy pedig arról, hogy a játék véget ért és ki lett a győztes.

A rendszer tervezése során vannak fontos tényezők, melyeket figyelembe kell venni. Mivel a delta robot villamos hajtású, így a működése során folyamatosan biztosítani kell a megfelelő tápellátást. Ez a rendszert felépítő áramköri egységekre is vonatkozik, megfelelő tápfeszültség ellátás hiányában működési hibák léphetnek fel. Meg kell oldani, hogy a tábla beolvasása során ne legyen zavaró objektum a tábla környezetében, vagy azokat a megfelelő módon ki kell szűrni. Az ilyen elemek megnehezíthetik a tábla beolvasását, rosszabb esetben hamis eredménye lehet a műveletnek. Biztonsági szempontból fontos, hogy a robotok mozgása során munkaterükben nem tartózkodhat ember. A delta robot esetében ez azt jelenti, hogy mozgása közben a játékos nem helyezheti be a kezét a munkatérbe. Ha ez mégis megtörténik, annak súlyos emberi sérülés lehet a következménye, illetve maga a robot is sérülhet, tönkre mehet. Ennek elkerülése érdekében a rendszerbe egy védelmi funkcióra is szükség van. Védelmi funkció lehet például a robot vázán kialakított fényfüggöny. Ha a felhasználó a robot mozgása során a munkatérbe akarja a kezét helyezni, azt az érzékelő időben detektálja és a rendszer egy vészleállást hajt végre.

# Logikai rendszerterv

A 4.1.-es ábrán látható a rendszer logikai felépítése. Maga a delta robot mechanikája adott volt, az ahhoz tartozó elemeket egy vékonyabb kerettel jelöltem.

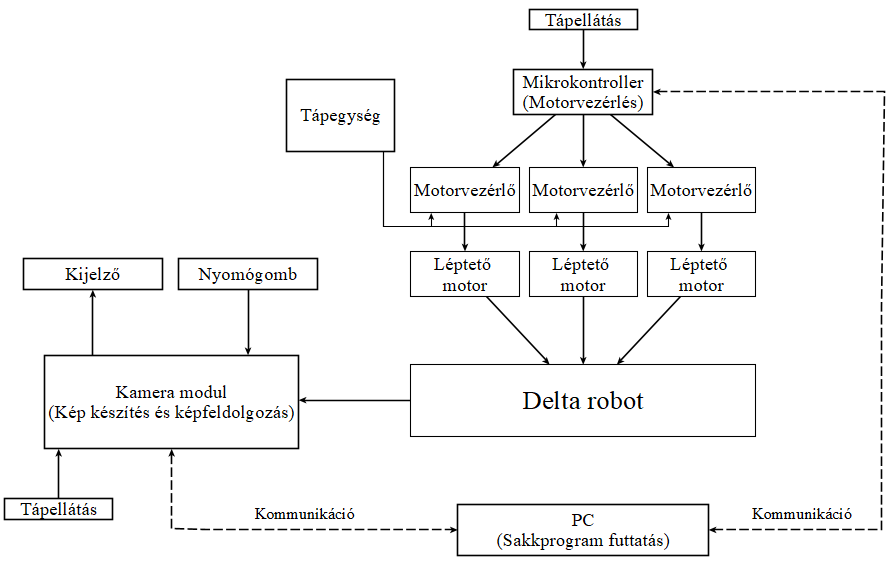
A karok mozgatásához léptető motorok lettek felszerelve a robot vázára. Ezek megfelelő irányításának érdekében külön motorvezérlők is felhelyezésre kerültek, melyek a léptető motorokhoz közel találhatók meg. A karok adott pozícióba való beállításához a motorvezérlőknek kell a jelet kiadni, így szükség van egy olyan mikrokontrollerre, ami a beállítandó pozíció ismeretében képes kiszámolni, hogy az egyes motoroknak milyen szögbe kell állniuk ehhez. Ha a számítás megtörtént, akkor az adott motorvezérlőre kell a megfelelő jelet kiadnia. A mikrokontroller megválasztása során az egyik kritérium, hogy kimeneti feszültsége legalább akkora legyen, mint amit a motorvezérlő bemeneti feszültsége megkövetel, különben a motorok vezérlése nem lesz megfelelő.

Szükség van egy kamera modulra, mely egy feldolgozó egységgel is rendelkezik. A felhasználó a nyomógomb lenyomásával jelzi, hogy lépését megtette. Ezt a bemeneti jelet kell érzékelnie a modulnak, és a kamera segítségével egy képet készítenie a tábláról. Viszonylag jó minőségűnek kell a képnek lennie, így akkor a tábla detektálása és a bábuk felismerése jóval egyszerűbb és biztosabb lesz. A feldolgozó egységen fog futni maga a képfeldolgozás. Egy képnek a feldolgozása sok és bonyolult műveletből épül fel. Ebből következőleg az egységnek olyan műveletvégzési sebességgel kell rendelkeznie, hogy ezeket a műveleteket képes legyen végrehajtani megfelelő feldolgozási idő alatt.

A kijelző segítségével információ jeleníthető meg a felhasználó számára. Ilyen információ lehet a játék állása, éppen ki következik lépésben, vagy a rendszer működése során fellépő hibaüzenet. A kijelzőre való üzenetek kiírásáért a kamera modul vezérlője legyen felelős.

A sakkjáték lebonyolításáért, és a lépés megtételéhez szükséges adatok kiküldéséért egy PC fog felelni. A PC-nek kommunikációs kapcsolatban kell állnia a motorvezérlésért felelős mikrokontrollerrel, illetve a képfeldolgozásért felelős modullal. A PC a kamera modultól fogadja a képfeldolgozás eredményét, azaz pontosan mi a sakktábla aktuális állása. Ennek megfelelően a PC egy válaszlépést számol ki, és az annak megtételéhez szükséges információkat továbbítja a motorvezérlésnek. Miután a mikrokontroller a megfelelő pozícióba állította a karokat, a PC-nek egy visszajelzést kell küldenie arról, hogy végzett a beállítással. Ezen felül a kijelzőre szánt információkat a PC fogja továbbítani a kamera modul feldolgozó egységének, mely a kapott információt megjeleníti a kijelzőn.

Megfelelő tápfeszültség ellátást kell biztosítani a motorvezérlést végző mikrokontrollernek, a kamera modulnak, illetve a motorvezérlőknek. A léptető motorok a motorvezérlőn keresztül kapják a tápfeszültséget. A 4.1.-es ábrán is jól látható, hogy az egyes egységek szoros kapcsolatban állnak egymással. Ha valamelyik egység tápfeszültség ellátása nem megfelelő (alacsony tápfeszültség, tápfeszültség ellátás időközönként kimarad), akkor a rendszer bizonytalan működést produkálhat, mely súlyos hibákhoz vezethet.

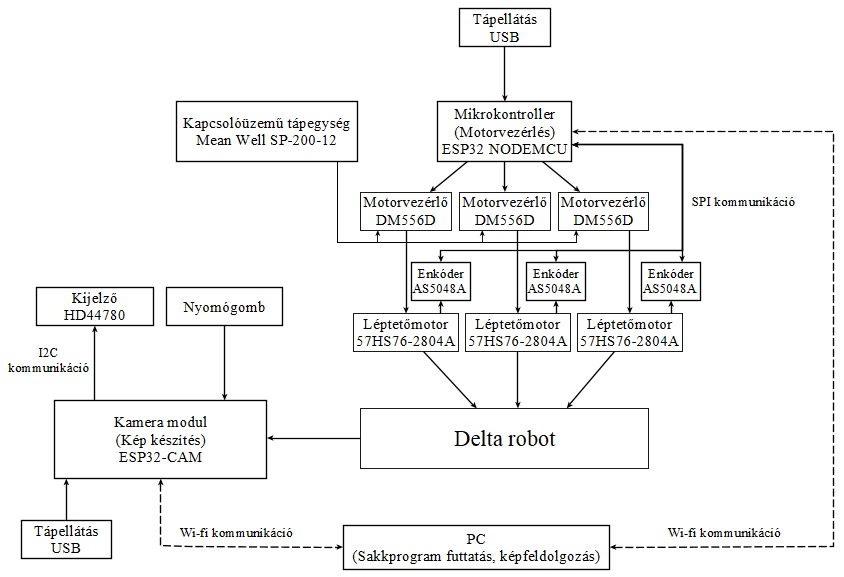


4.. ábra Logikai rendszerterv

# Fizikai rendszerterv

## Az első verzió

Az 5.1.-es ábrán látható a fizikai rendszerterv első változata. Ugyan úgy, mint a logikai rendszertervnél, mivel a delta robot adott volt, így az ahhoz tartozó elemeket egy vékonyabb kerettel jelöltem. Először a delta robothoz tartozó elemeket, majd a rendszerünk felépítéséhez kiválasztott elemeket ismertetném.



5.. ábra A fizikai rendszerterv első változata

A delta robot (5.2. ábra) mechanikai szerkezete egy előző projekt során készült el, és jelenleg az Óbudai Egyetem Kandó Kálmán Villamosmérnöki Kar tulajdonában áll. Így volt lehetőségünk ezzel a robottal dolgozni. A robot váza, illetve a karok alumíniumból készültek. Az aktuátor karok 15 cm, a passzív karok pedig 33 cm hosszúak. [27] Ezzel a kialakítással egy 24 cm x 24 cm x 24 cm méretű munkatér állt a rendelkezésünkre, így egy ekkora méretű sakktáblát lehetett készíteni a játékhoz. Kialakításából adódóan a robot mozgása lassabb, de ez a feladatot tekintve nem volt hátráltató tényező, a játék élményét nem befolyásolja. A végeffektorra egy elektromágnes került felhelyezésre, mely segítségével fémtárgyakat lehet felemelni, illetve letenni.

A képen beltéri látható

Automatikusan generált leírás

5.. ábra A delta robot

A karok mozgatásáért a robotra felszerelt 3 darab 57HS76-2804A típusú bipoláris léptetőmotor felel. Ez a típus egy 2 fázisú, 3V-os fázis feszültségű, 2.8A fázisáramú motor. Nyomatéka 1.9Nm, lépési szöge pedig 1.8° [14]. Az iparban jóval erősebb motorokat használnak a karok mozgatására, de ennek a robotnak egy könnyű korongot kell mozgatnia, így a működés bemutatására ez a típus a megfelelő paraméterekkel rendelkezik.

A delta robot motorjainak vezérlésére DM556D típusú motorvezérlő egységek állnak rendelkezésre. Ez a típus egy teljesen digitális DSP alapú modul, melyben a microstepping, és a kimeneti áram nagysága változtatható szoftveres módon, illetve DIP switch kapcsolók segítségével is [15].

A motorvezérlők tápfeszültség ellátása a Mean Well SP-200-12 kapcsolóüzemű tápegység segítségével történik. Ez a tápegység 230V AC feszültséggel működik, és kimenetén 12V DC feszültséget és 16.7A áramot biztosít [16]. A motorok a hozzájuk tartozó motorvezérlőn keresztül kapják a megfelelő tápfeszültség ellátást.

A végeffektor megfelelő pozícionálásához szükségünk van a motorok aktuális szögállására. Ennek meghatározására az AS5048A típusú forgójeladót választottuk, mind a három motorra ezt a típust szereltük fel. Ez a forgójeladó tulajdonképpen négy Hall szenzorból áll, amely a mágneses fluxus mértékét mérik, és annak feszültség értékét képzik le. DSP algoritmusok és 14 bites A/D átalakító segítségével teszi továbbíthatóvá az aktuális szögértéket. Az adatok továbbítása SPI kommunikáción keresztül történik. [17]

A motorvezérlésért egy ESP-32 NODEMCU modul fog felelni. Ez egy kétmagos ESP32-D0WDQ6 RISC architektúrájú processzorral lett ellátva. Vezeték nélküli- (Wi-fi 802.11 b/g/n, Bluetooth 4.2), illetve több vezetékes kommunikációra képes. A modul Wi-fi kommunikáció segítségével tartja a kapcsolatot a PC-vel, a vezetékes kommunikációk közül a feladat szempontjából pedig fontos az SPI kommunikáció, mivel ezen keresztül tudjuk lekérdezni az enkódereket. A modul 35 GPIO-val rendelkezik, 3.3V DC a kimeneti feszültsége [18]. Úgy gondoltuk, ez a fajta mikrokontroller modul alkalmas a motorvezérlés ellátására. Ugyan 35 GPIO ehhez a feladathoz több, mint kellene, viszont ez nem okoz problémát, sőt, esetleges további fejlesztéseknél ez kedvező is lehet. Így esett választásunk erre a modulra, ami méretének köszönhetően kényelmesen elhelyezhető a delta robot vázán. A modul a megfelelő tápfeszültséget USB porton keresztül kapja hálózati adapter segítségével.

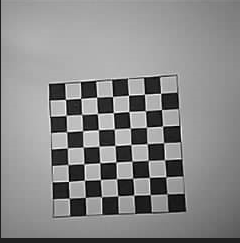
A sakktábláról való képkészítésért egy ESP32-CAM modul fog felelni. A modul alapja egy ESP32-S kétmagos processzor, ugyan úgy képes vezeték nélküli- (Wi-fi 802.11 b/g/n, Bluetooth 4.2), illetve több vezetékes kommunikáción keresztül adatot továbbítani, mint az előbb említett modul. A modulra egy 2 megapixeles OV2640 kamera lett beépítve, így e miatt a modulon csak kilenc GPIO lett kivezetve, melyek kimeneti feszültsége 3.3V DC. A kamera számos funkcióval rendelkezik, ilyen például az automata fényerő állítás, fénycsík-túlvezérlés kompenzáció [19]. Ahhoz, hogy a felhasználó jelezni tudja, hogy végzett a lépésével, a modulra egy nyomógomb kerül bekötésre. A gomb lenyomását követően készül el a kép a tábláról, melyet WiFi kommunikáción keresztül továbbít a modul a PC-nek. A megfelelő tápfeszültséget USB porton keresztül kapja hálózati adapter segítségével.

Információ megjelenítés céljára a HD44780 LCD kijelző került kiválasztásra, mely szintén a kamera modulra lesz bekötve. A modul kevés GPIO száma miatt a kijelzőre egy I2C interfész adapter modul [20] lett felhelyezve forrasztással.

A rendszerhez egy olyan PC-re van szükség, mely képes futtatni a sakkprogramot és a képfeldolgozást. WiFi kommunikáción keresztül küldi ki a motorvezérlőnek a beállítandó szögeket, a beállítás után pedig fogadja, hogy a pozícionálás megtörtént. Szintén WiFi kommunikáción keresztül történik az adatküldés és fogadás a kamera modullal. A PC a kamera modultól kapja meg az elkészített képet, a fogadás sikerességéről visszajelzés kerül kiküldésre a modulnak. Ezen felül a PC a kamera modul felé küldi meg az információt, amit az LCD kijelzőre kell kiíratni.

## A módosított, végleges verzió

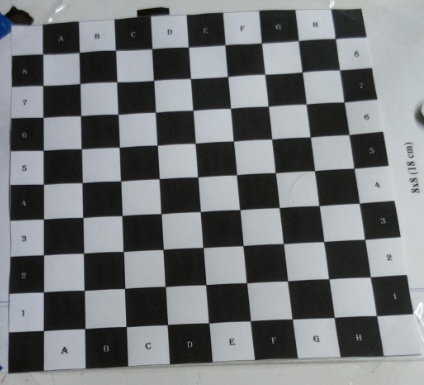
A kamera modul tesztelése során azt tapasztaltuk, hogy a kamera nem képes megfelelő minőségű képet készíteni a sakktábláról. Különböző beállítási módokat alkalmazva is azt láttuk, hogy a kamera képe kicsi, zavaros, enyhén torzított (5.3. ábra). Ezen felül problémát jelentett, hogy a kamerának csak a kivezetései lettek bekötve, másmilyen módon a kamera nem került rögzítésre a modulon. Ennek az volt a következménye, hogy a robot mozgása során a kamera szabadon tudott lengeni, így tiszta képet a tábláról készíteni lehetetlenség volt. Ez a feladatunk szempontjából nem volt elfogadható, mivel a kamera modul a rendszer egyik kulcspontja, nem megfelelő kép esetén a játék nem tud működni.



5.. ábra ESP32-CAM modullal készített kép

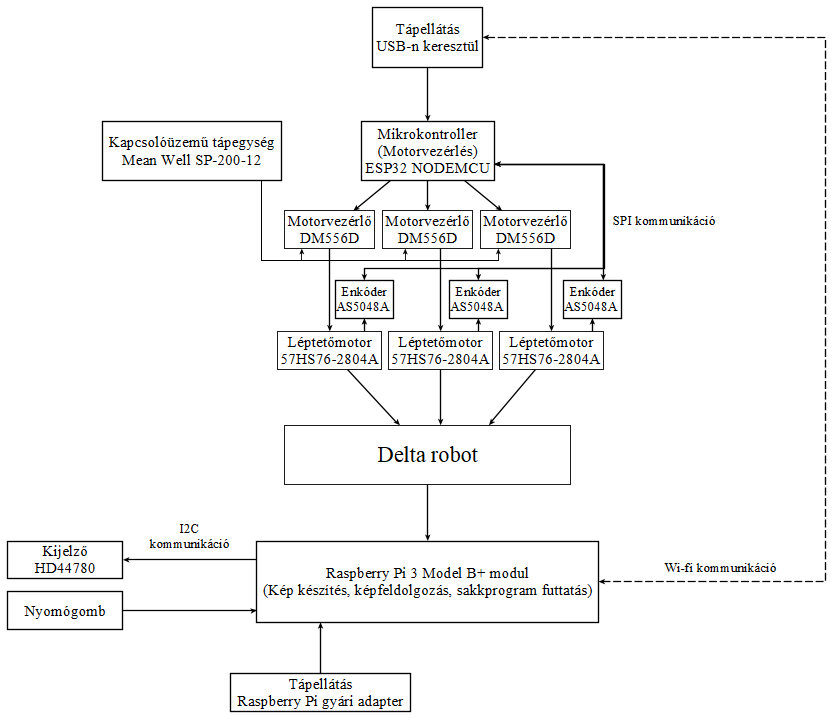
E miatt döntöttünk úgy, hogy az ESP32-CAM modul helyett a Raspberry Pi 3 Model B+ egykártyás számítógépet alkalmazzuk. A lapon egy Broadcom BCM2837B0 SoC található, melyben egy 64 bites Cortex-A53 processzor van. Ezzel, illetve az 1GB méretű LPDDR2 SDRAM-al sokkal nagyobb és bonyolultabb feladatokat képes ellátni. 40 GPIO kivezetése van (kimeneti feszültség 5V), ezen felül HDMI, 4 darab USB, és Ethernet csatlakozó is rendelkezésre áll. Képes WLAN (IEEE 802.11.b/g/n/) és Bluetooth (4.2, BLE) kommunikációra is. [21]

Ehhez a Raspberry modellhez tartozik egy Raspberry Pi kameramodul (v2), ami 8 megapixeles felbontású, és egy Sony IMX 219 szenzor található rajta [22], így sokkal jobb minőségű képek készíthetők vele (5.4. ábra).



5.. ábra A Raspberry Pi-al készített kép

A Raspberry Pi egy nagyobb számításkapacitású eszköz az ESP32-nél, több és bonyolultabb feladat is megoldható vele, illetve több programozási nyelv segítségével lehet felprogramozni. Éppen ezért úgy döntöttünk, hogy a képkészítésért, annak feldolgozásáért és a sakkprogram futtatásáért a Raspberry Pi modul fog felelni. Erre lesz bekötve a nyomógomb és az LCD kijelző, melyek ugyanazt a funkciót látják el, mint az előző verzióban. A tápfeszültség ellátást a hozzátartozó gyári hálózati adapteren keresztül fogja kapni. A motorvezérlést végző mikrokontrollerrel WiFi kommunikáción keresztül fogja a kapcsolatot tartani. A PC a végső fizikai rendszertervben nem szerepel, annak feladatát teljesen a Raspberry Pi veszi át (5.5. ábra).



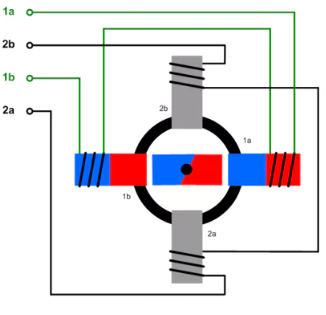
5.. ábra A fizikai rendszerterv végső változata

# Motorvezérlő modul

## A léptető motorok

A motorvezérlés megvalósításának tárgyalása előtt fontos ismertetni a roboton lévő motorok működését és azok vezérlésének módját. Ahogy az a logikai- és fizikai rendszertervben is említésre került, a robot mechanikája már készen a rendelkezésünkre állt, így adott motorokkal kellett dolgoznunk.

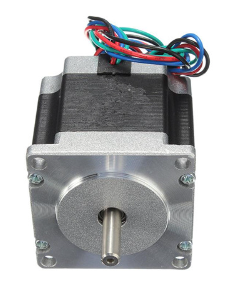
A delta robot karjainak mozgatása bipoláris léptetőmotorokkal lett megvalósítva. A léptetőmotorok olyan szénkefe nélküli DC motorok, melyek álló részén több elektromágnes található, amik a forgó részen lévő fogaskerék alakú vasat veszik körül. [23] Általában az elektromágnesek tekercsei párokban vannak összekötve (6.1. ábra). A tekercsre adott áram hatására a rotor egy adott pozícióba áll be. A rotor beállásának pozíciója attól függ, hogy éppen melyik elektromágnest kapcsoljuk be. A bipoláris jelző arra utal, hogy a motor két polaritással működik, azaz a tekercsben folyó áram iránya változó. Az, hogy most a rotor melyik irányba kezd el forogni, illetve mennyit fordul el, függ a vezérlés ütemétől és irányától.



6.. ábra Bipoláris léptetőmotor tekercseinek bekötése [24]

Ezekkel a motorokkal megvalósítható teljes léptetésű, féllépéses, illetve mikrolépéses mód is. Mikrolépéses módban az egy impulzusra megtett fordulási szög nagyon kicsi, ezzel nagyon pontos és finom pozícionálás érhető el. Azonban nagyfrekvenciás impulzus kiadás esetében a tekercsek impedanciája megnövekszik, ezáltal a tekercseken átfolyó áram nagysága csökken. A kisebb áram hatására pedig a motor nyomatéka csökkenhet le olyan mértékben, hogy a motor egy impulzus hatására nem tesz meg egy adott lépést. Ennek elkerülése érdekében a motorvezérlő adatlapjában leírtak szerint kell a vezérlő jelet az előírt sűrűségben kiküldeni a vezérlőnek.

A delta roboton található motorokat (6.2. ábra) microstep módban fogjuk használni, így nagyon pontosan lehet beállítani a végeffektort a kívánt pozícióra, ugyanakkor a beállítás viszonylag lassan történik. Mivel a motorok nyomatéka alacsonyabb sebességen nagyobb, így hibamentesen nem lehetséges ebben az üzemmódban nagy sebességű beállítás. [24] A feladat szempontjából fontosabbnak tartottuk a pontos beállást hiszen, ha a bábu lehelyezése pontatlanul történik, akkor a következő lépések folyamán ez a pontatlanság folyamatosan nőne.



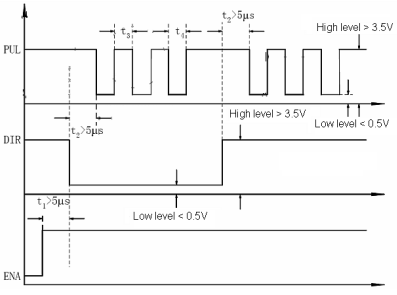
6.. ábra 57HS76-2804A típusú bipoláris léptetőmotor [25]

## A motorvezérlők

A motorvezérlőkön található DIP switch kapcsolókkal a feladathoz 1600-as osztót állítottunk, így microstepping módban fognak a motorok működni. Ez azt jelenti, hogy egy teljes körbeforduláshoz 1600 lépést kell megtennie a motornak, tehát egy impulzus hatására az 0.225°-ot fordul el. A maximális áramot, amit a meghajtó ad ki a motoroknak, 4A-re állítottuk.

Annak érdekében, hogy a meghajtó a megfelelő vezérlési jelet szolgáltassa, bizonyos szabályokat kell követni a motorvezérlő impulzus kiadása során, melyet a motorvezérlő adatlapja ír le. [15] A 6.3.-as ábrán lévő jelölések a következők:

* PUL (Pulse) jel segítségével történik a motor léptetése
* DIR (Direction): jellel lehet a forgási irányt változtatni
* ENA (Enable): jel szolgál a motorvezérlő engedélyezésére, illetve tiltására



6.. ábra A meghajtó vezérlési diagramja [15]

Az ENA jel kiadásának időzítésére nem térnék ki a dolgozatban részletesen, mivel a motorvezérlőknek a rendszer működése során folyamatosan engedélyezve kell lenniük. Ebben az esetben az adatlap [15] úgy rendelkezik, hogy a vezérlő azon portjára nem kell kötni semmilyen vezetéket.

A PUL jel kiadása előtt legalább 5 μs-al kell kiadni a DIR jelet. A DIR logikai 1 értéke esetén óramutató járásával megegyező fordulás, logikai 0 érték esetén óramutató járásával ellentétes forgás valósul meg.

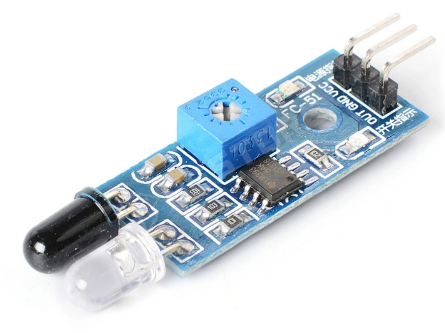
A PUL vonalnak alapállapotban magas szinten kell lennie, ekkor nem történik motorléptetés. Egy lépés kiadásához egy 0-1 impulzust kell kiadni. Az adott jelek kiadásánál figyelembe kell venni, hogy mind a magas, mind az alacsony szinteket legalább 2.5 μs-ig kell tartani.

Mindegyik jel esetében a magas jelszintnek nagyobbnak kell lennie, mint 3.5V-nak, az alacsony jelszintnek pedig kisebbnek, mint 0.5V-nak. Ha ez nem teljesül, akkor előfordulhat, hogy a motorvezérlő nem érzékeli a kiküldött jelet és hibás működés léphet fel.

## Visszacsatolás a karok pozíciójáról

Feladatunk megkezdése során a legelső probléma, amivel találkoztunk az volt, hogy a delta robot akkori kialakításából adódóan, a karok aktuális pozíciójáról nem volt semmiféle információnk. Nem tudtuk meghatározni, éppen milyen pozícióban van a kar, pontosan ráállt-e az adott pontra, vagy egyáltalán alaphelyzetben vannak-e a robot karok. A megfelelő pozícionáláshoz feltétlenül szükség van visszacsatolásra. Azon kívül, hogy ezáltal tudjuk a robot aktuális helyzetét, pontatlan beállás esetén képesek vagyunk korrigálni, így a bábu felvétel, illetve letétel hiba nélkül valósulhat meg.

A delta robot alap konstrukciója szerint a léptetőmotorok alaphelyzetbe való állításához OBS-IR típusú infravörös akadályérzékelő modulokat [26] alkalmaztak (6.4. ábra), melyeket az aktuátor karokhoz közel helyeztek el. Az alaphelyzetbe állítás úgy zajlott, hogy a motorokat egyesével léptették felfele. Mikor az aktuátor kar az emitter vonalába esett (ez kb. az aktuátor 90°-os állásánál volt), akkor az infravörös sugarat a karról visszaverődve a nyelő érzékelte, és magas jelet küldött. Ezt a léptetést mindhárom motoron végrehajtva történt meg az alaphelyzetbe állítás. [28]

****

6.. ábra OBS-IR infravörös akadályérzékelő modul [26]

A tesztelések során kiderült, hogy ez a fajta megoldás nem a legmegbízhatóbb működést biztosítja. Ugyanis a Nap is infravörös sugarakat bocsát ki, amelyeket az érzékelő detektál, ezzel hamis mérési eredmény kerül továbbításra. Ebben az esetben előfordult, hogy például egyes motorok el sem indultak, esetleg egy vagy több motornál nem állt meg a léptetés a helytelen detektálás miatt. Ugyan az érzékelőn található potenciométer segítségével lehet a szenzor érzékenységét állítani, azonban akkor az aktuátorról visszaverődő sugár érzékelése lett bizonytalan. Különböző tesztelési környezetet próbáltunk ki, először árnyékosabb helyen, végül szinte teljesen elsötétített helyiségben zajlott a tesztelés. Minden esetben a működés bizonytalan volt, így ez a megoldás nem volt alkalmas a robot megfelelő pozícionálására.

Végállás kapcsolókat mozgó alkatrészek esetében alkalmaznak abból a célból, hogy érzékeljék, ha az alkatrész egy adott végállásban van. Felhasználása sokrétű, a pneumatikus munkahengerek végállás érzékelésétől a 3D nyomtatók alaphelyzetbe állításáig sok funkció valósítható meg velük. Ez egy elektromechanikus eszköz, amely erőhatásra nagyon érzékeny gombbal van ellátva. Így gyorsan valósul meg az érintkezők zárása, azonnali visszajelzés történik az alkatrész állapotáról.

Elgondolásunk szerint az aktuátor karjai felé helyeztük volna a végálláskapcsolókat úgy, hogy a karok kb. 90°-os szögben álljanak, mikor a kapcsoló zárt állapotba kerül. Így az infravörös érzékelőnél ismertetett alaphelyzetbe állítás maradt volna meg, csak egy jóval megbízhatóbb eszköz segítségével. Azonban már a tervezés során kiderült, hogy a robot kialakításából adódóan nem tudjuk úgy elhelyezni a kapcsolókat, hogy az ne lenne zavaró a robot mozgásában.

Következő ötletünk az volt, hogy Hall szenzoros érzékelőket alkalmazzunk. A Hall szenzor segítségével adott távolságon belül lehet mágneses térerősséget mérni, de felhasználhatjuk, mint akadályérzékelőként is.

Tervünk szerint az aktuátor karokra egy-egy állandó mágnest rögzítettünk volna. A Hall szenzorokat olyan helyzetbe állítjuk, hogy érzékelés során a karok kb. 90°-ban állnak. Ekkor a motorok felfele léptetése során a mágneses teret a Hall szenzor egy ponton érzékeli, és magas jelet küld. A probléma ismét a fizikai elhelyezéssel volt, illetve a szenzor akkor is bejelezhet, ha a kar még nem állt be a megfelelő pozícióba, de mágneses teret érzékel. Az ilyen fajta beállítások befolyásolják a későbbi pozícionálásokat, így a pontatlanság mértéke folyamatosan halmozódik a játék során, ami nem megengedhető.

Ezen próbálkozások hatására rájöttünk, hogy tökéletes alaphelyzetbe állítást csak nagyon nehezen lehet megvalósítani. Ebben szerepet játszik az is, hogy a felfele léptetés befejezése során a motoroknak van egy kisebb tehetetlensége, melynek hatására a delta robot enyhén oszcillál és úgy áll meg. Így az alaphelyzet a kívánttól lejjebb kerülhet.

Végül arra jutottunk, hogy szükségünk van minden motor aktuális pozíciójára ahhoz, hogy ne csak az alaphelyzetbe állítás legyen pontos, de az összes mezőre is pontos ráállást tudjunk biztosítani. Ehhez kiváló megoldást jelentett minden motornál a forgójeladó (rotary encoder) használata, mely segítségével nagyon pontosan meg lehet határozni, hogy az egyes motorok milyen szögben állnak éppen. Az aktuális szög adatokkal és a megfelelő számítások elvégzésével pedig a robotot egyik pontból a másikba vinni sokkal egyszerűbb, illetve a beállás végén a visszacsatolás révén korrekciót is végre lehet hajtani.

A forgójeladók olyan eszközök, amelyek a forgómozgás hatására létrejövő változás alapján képesek a motor aktuális szögének mérésére, az eredményt analóg vagy digitális jelként továbbítani. A forgó jeladókat két fő csoportba lehet osztani: inkrementális, illetve abszolút forgójeladók.

Inkrementális enkóderek esetén mindig egy alappozícióhoz képest mérjük az elfordulás mértékét, ebből tudunk következtetni arra, hogy éppen milyen helyzetben van a motor. Hátrányuk, hogy a tápfeszültség megszűnte után a következő bekapcsoláskor nem képesek a motor aktuális helyzetét meghatározni.

Az abszolút enkóderek azonban az ismételt bekapcsolás után pontosan ugyan azt a szögértéket tudják visszaadni, mint kikapcsolás előtt. Azaz ez a típusú jeladó mindig képes az aktuális szögértékről információt biztosítani, ami a delta robot pozícionálásának szempontjából elengedhetetlen. Éppen ezért az abszolút enkóder mellett döntöttünk.

A jelenlegi konstrukció mellett vagy optikai, vagy mágneses abszolút forgójeladó közül lehetett választani. Az optikai jeladó hátránya, hogy a külső fényforrásra érzékeny lehet, illetve az érzékelőre tapadható por, szennyeződés hatására az érzékelés bizonytalan lehet. A mágneses forgójeladók esetében a szennyeződés nem tudja befolyásolni a mérés eredményét, hiszen mágneses térerősség mérése zajlik. Az egyetlen befolyásoló tényező lehet az érzékelő és a mágnes közötti távolság, de a motorok kialakításából adódóan azt láttuk, hogy ezeket megfelelő távolságra tudjuk elhelyezni egymástól.

Ezen megfontolások alapján döntöttünk az AS5048A típusú mágneses forgójeladó mellett. Az enkóderhez tartozó permanens mágnest a motor tengelyére rögzítettük, a mágnessel szemben pedig az enkódert helyeztük el egy magasságban.

# Kommunikációk

## SPI kommunikáció

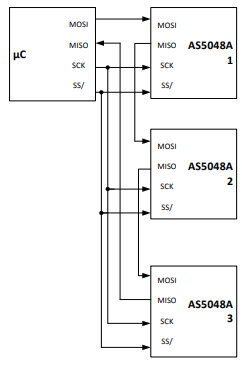
A motorok aktuális állását a forgójeladók segítségével tudjuk meg, melyek SPI kommunikáción keresztül képesek továbbítani a mért adatokat.

Az SPI kommunikációt beágyazott rendszerekben alkalmazzák rövid távú kommunikáció esetén, mely master-slave kialakítású, illetve full-duplex módú. Ez a fajta kommunikáció akkor előnyös, ha például több mérőeszközünk van, melyektől adatokat szeretnénk lekérdezni adott időközönként, vagy valamilyen esemény hatására.

Kommunikáció szempontjából a forgójeladók lesznek a Slavek, akik közül a Master (mikrokontroller) parancsára a megfelelő Slave Select bebillentésével kiválasztott eszköz fog adatot küldeni egy időben. Ebben az esetben a mérőeszközök egy közös adatvezetékre fűzhetők fel, melyen keresztül a Masternek továbbítanak adatot. A kommunikáció kialakításánál ügyelni kell arra, hogy egyszerre csak egy Slave küldjön adatot a közösített adatvezetéken, meg kell várni, míg a vonal felszabadul. Csak ezután lehet újabb Slavetől adatot lekérni, különben a küldött adatban zavar keletkezik, hibás értéket kapunk, legrosszabb esetben a kommunikációs csatorna tönkre mehet.

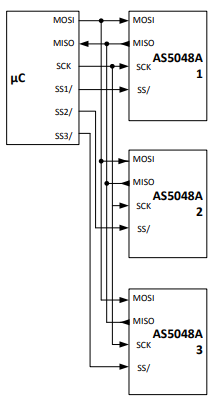
Két fajta kialakításból lehetett választanunk: az eszközöket többszörös Slave-szelekcióban fűzzük fel, vagy pedig Daisy Chain módban.

A Daisy Chain kialakítás (7.1. ábra) alkalmazásával egyszerűbb a vezetékezés megoldása, viszont ebben a kialakításban kötött a prioritás, így nem tudjuk szabadon lekérdezni egy kifejezett jeladónak az értékét. Ebben az esetben mindig 3 adatot fogunk kapni egyszerre, így azok feldolgozása nehezebb.



7.. ábra Daisy Chain kialakítás [17]

Többszörös Slave-szelekció esetében (7.2. ábra) ugyan több vezetéket kell alkalmazni, viszont itt nincs kötött prioritás, attól a jeladótól kérünk le adatot, amelyiket éppen kiválasztjuk. Az adatvezeték szintén közösítve van. Itt viszont fokozottan kell figyelni az időzítésre, hogy a következő jeladót csak akkor kérjük adattovábbításra, mikor a másik befejezte azt. Ebből következhet az, hogy több szenzor lekérdezéséhez több időre van szükség. A feladat szempontjából ez a lekérdezés nem kritikus feladat, mivel csak egy mozgás sorozat kezdetén kérdezzük le a motorok állását. Így ez a kialakítás bizonyult a legmegfelelőbbnek.



7.. ábra Többszörös Slave-szelekció [17]

## Wi-fi kommunikáció

A rendszer indítását követően az ESP32 mikrokontrollert access point-ként konfiguráljuk. Ezáltal az ESP32 egy saját WiFi hálózatot képes kiépíteni, melyre más eszközök képesek csatlakozni, ezen keresztül kommunikálni egymással, illetve magával a mikrokontrollerrel. Ez a kialakítás további fejlesztések esetén kihasználható, például ha új eszközt szeretnénk felfűzni a hálózatra, ami egy új funkciót láthat el.

A mikrokontrollerrel egy „ESP\_SERVER” nevű WiFi hálózatot hozunk létre. A játék megkezdése előtt a Raspberry-nek erre a hálózatra kell csatlakoznia, hogy a kommunikáció létrejöjjön a két eszköz között. A sikeres csatlakozást követően elindulhat az adatküldés és fogadás a Raspberry Pi és a mikrokontroller között.

A mikrokontroller a kliens csatlakozása után adatot vár a Raspberry-től. Ez az adatsor tartalmazza az információt arról, hogy az egyes motorokat milyen szögbe kell beállítani. A Raspberry-től kapott üzenet formátumát az alábbi példán keresztül mutatom be:

9000a9000b9000c1d0e

A motorra kiadott szögeket két tizedes pontossággal, tizedesvessző nélkül küldjük ki, melyet a mikrokontroller a fogadást követően a megfelelő formátumra alakítja át. Az egyes beállítandó szögeket betűk választják el. Az „a” jelzés jelenti, hogy az egyes motort kell az előtte lévő szögértékre beállítani. A „b” jelzés a második motort jelenti, a „c” jelzés pedig a harmadik motorra utal. Az üzenetben a „d” jelzés alatt értendő, hogy a mozgás során a mágnesnek bekapcsolt állapotba kell lennie (logikai 1), vagy pedig kikapcsolt állapotban (logikai 0). Az „e” jelzés előtt lévő adat szolgáltat arról, hogy a karok beállítása után szükséges-e korrekciót végezni. A tesztelések során azt láttuk, hogy a hosszabb mozgássorozatok során a robot karjai pontatlanabban állnak be. Így ezt a hibát úgy küszöböltük ki, hogy az egyes lépések végén a visszacsatolás által korrigáljuk a karok állását. Így, ha az „e” logikai 1 értékű, akkor a mozgás befejezte után addig léptetjük a megfelelő irányba a motorokat, míg az aktuális szög egy adott határon belül közelíti a beállítani kívánt szöget.

Tehát a példa üzenet szerint mind az 1-es, 2-es és 3-as motort 90.00°-ba kell állítani, a mágnest be kell kapcsolni, illetve a mozgás végén korrekció nem szükséges.

# Motorvezérlő program

## Makrók és változók definiálása

A motorvezérlés megkezdése előtt szükség van hét makró definiálására. Értékeik a mikrokontrollernek azon portjai, melyen keresztül a motorvezérlőnek és az elektromágnesnek küldjük ki a vezérlőjeleket. Ezeknek a definiálása megkönnyítették a fejlesztés folyamatát, ezen felül segítségükkel a kód könnyebben olvasható. A 8.1. táblázat foglalja össze a definiált makrókat, azok értékét és magyarázatukat.

|  |  |  |
| --- | --- | --- |
| **Makró neve** | **Értéke (Portszám)** | **Magyarázat** |
| MAGNET | 13 | Mágnes vezérlés |
| DIR1 | 32 | 1-es motor forgási irányának vezérlése |
| DIR2 | 33 | 2-es motor forgási irányának vezérlése |
| DIR3 | 25 | 3-es motor forgási irányának vezérlése |
| PUL1 | 27 | 1-es motor léptetése |
| PUL2 | 14 | 2-es motor léptetése |
| PUL3 | 12 | 3-as motor léptetése |

8.. táblázat A definiált makrók és azok magyarázata

A makrókon kívül változókra is szükségünk van, melyekben számadatot, karaktert vagy sztringet tárolunk, ezeket az 1. melléklet tartalmazza. Vannak olyan változók, melyek állapot flagként szolgálnak, melyeket a 8.2. táblázat összegez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Változó neve** | **Típusa** | **Értéke** | **Magyarázat** |
| setUp | uint8\_t | 0 vagy 1 | Jelzi, ha a működéshez minden adat beérkezett a klienstől |
| startStep | uint8\_t | 0 vagy 1 | Motorvezérlést indító flag |
| checkByCmd | uint8\_t | 0 vagy 1 | A kar beállítása utáni korrekció szükségességének jelzése |
| done1 | uint8\_t | 0 vagy 1 | Az 1-es motor beállítása befejeződött |
| done2 | uint8\_t | 0 vagy 1 | A 2-es motor beállítása befejeződött |
| done3 | uint8\_t | 0 vagy 1 | A 3-as motor beállítása befejeződött |
| pulse | uint8\_t | 0 vagy 1 | A motorvezérlőknek való vezérlőjel kiadását ütemezi |

8.. táblázat Az állapot flagek

## Motorok beállításához szükséges lépésszám számítása

Ahhoz, hogy egy motort a megfelelő pozícióba tudjuk állítani, meg kell határoznunk, hogy ezt hány léptetés segítségével lehet megvalósítani. Ez egyszerűen kiszámítható három ismert adat segítségével:

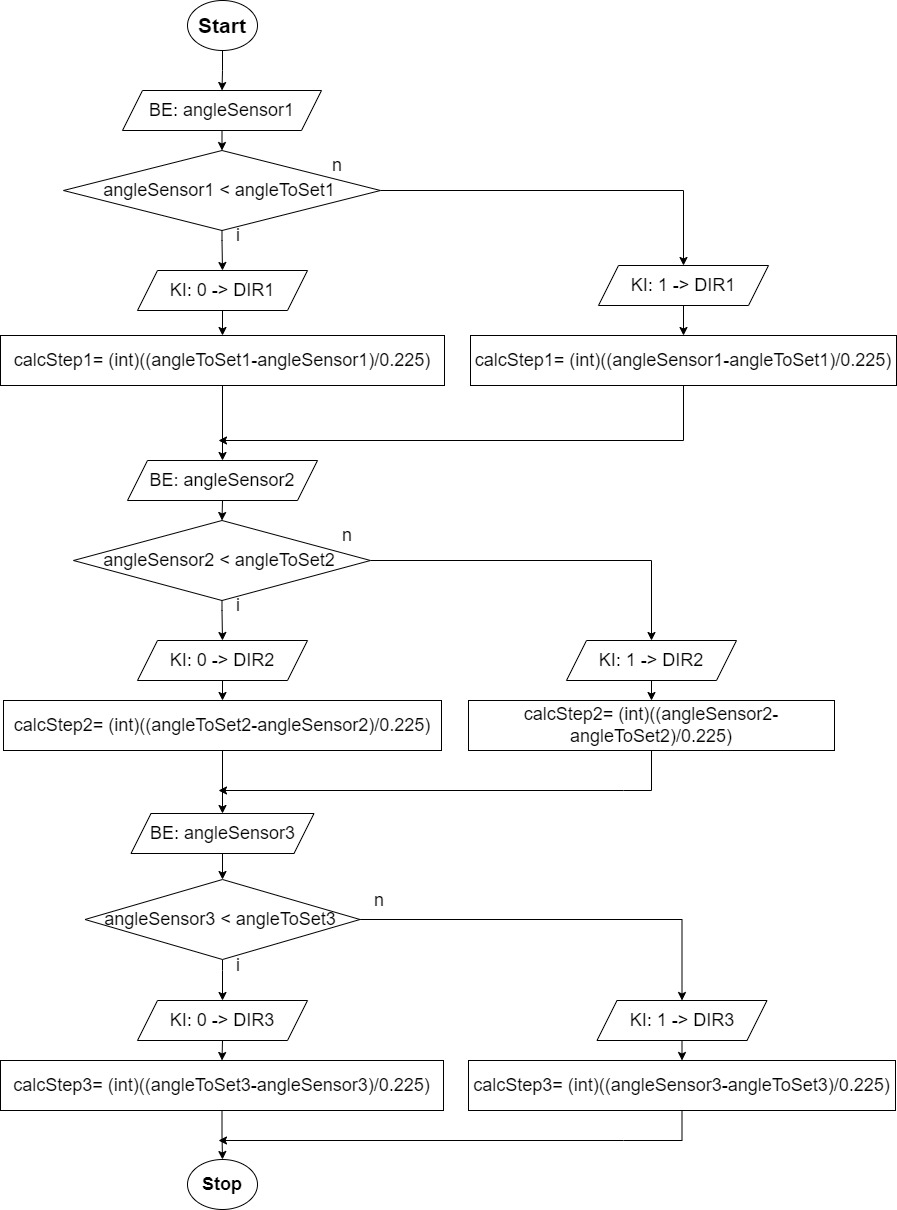
* Az enkóder által mért aktuális motorszög (angleSensorX)
* A beállítandó motorszög (angleToSetX)
* Egy step hatására megtett szög (0.225°)

Először azt kell eldönteni, hogy melyik irányba kell a motort forgatni. Ha az enkóder által mért szög kisebb, mint az elvárt szög, akkor óramutató járásával ellentétes irányú forgás, ellenkező esetben pedig óramutató járásával megegyező forgás szükséges.

Ezután következhet a lépésszám számolása. Az aktuális és az elvárt szög különbségét véve tudjuk, hogy mennyi hiányzik az elvárt érték eléréséhez. Ha ezt az eredményt osztjuk az egy impulzus hatására megtett szög értékével, akkor kapjuk meg, hogy még hányszor kell az adott motort léptetni:

A fenti képlet akkor alkalmazható, ha a mért szög nagyobb, mint az elvárt szög, különben negatív érték lenne az eredmény. Abban az esetben, ha az elvárt szög a nagyobb, akkor a képlet az alábbiban módosul:

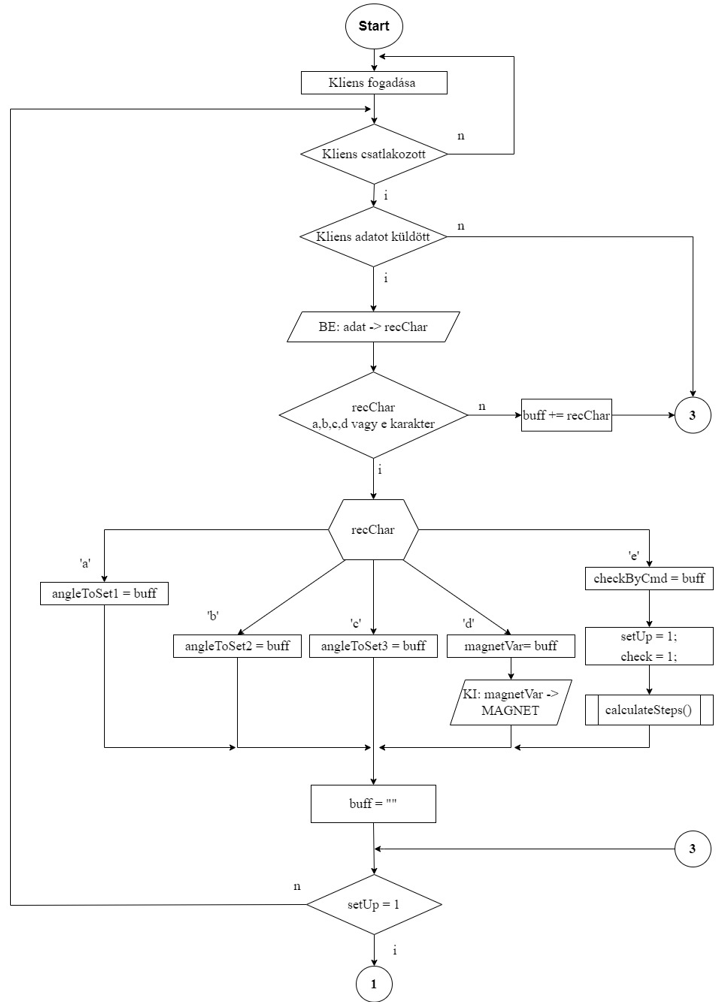
A művelet során előfordulhat, hogy az eredmény tizedes tört lesz. Azonban a léptetés során csak egész értéket tudunk léptetni, így az eredményt egész típusra kell átalakítani. Ez a számítási metódus mindhárom motorra érvényes, ezeket az értékeket a motorléptetés előtt kell kiszámolni. Mivel ezt a számítást a játék során többször kell végrehajtani, ezért erre egy függvényt hoztam létre calculateSteps() néven, melynek folyamatábrája a 8.1.-es ábrán látható.



8.. ábra calculateSteps() folyamatábrája

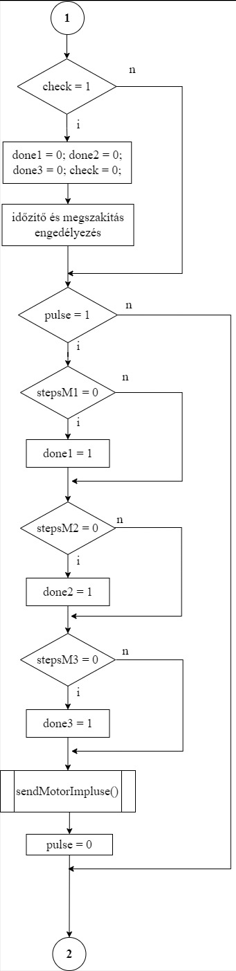
## A főprogram

A főprogram egyszerűbb és érthetőbb leírása érdekében folyamatábrát készítettem. Azonban az olvashatóság miatt három részre kellett bontanom azt. A folyamatábra első része a 8.2.-es ábrán, a második része a 8.3.-as ábrán, a harmadik része pedig a 8.4.-es ábrán látható. A részek kifejtése során igyekeztem az egyes ábrák között átfedést írni, mivel ezek nem külön programok, hanem hárman képzik az egész főprogramot.



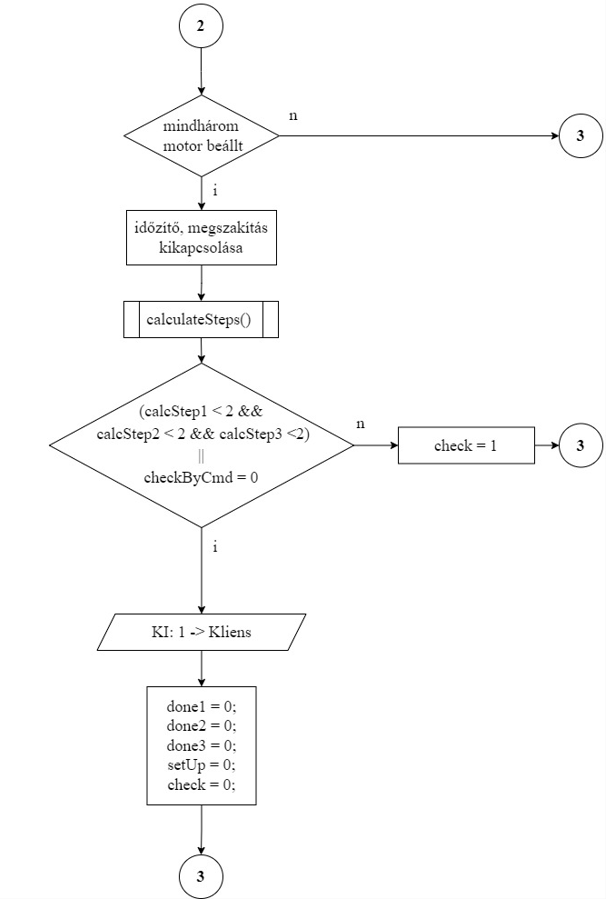
8.. ábra A főprogram folyamatábrájának első része

A Wi-fi kommunikáció és az enkóderek konfigurálása után várjuk, hogy a kliens (Raspberry Pi) felcsatlakozzon a hálózatra. A csatlakozást követően a kliens egy karaktert küld ki, melyet a recChar változóban tárolunk. Maga a kapott adat lehet betű vagy szám. Szám esetében egy String típusú bufferhoz adjuk hozzá az adatot, majd ha egy adott betű érkezik, az azt jelenti, hogy a bufferben rendelkezésre áll az a szám (motorszög), amelyet valamelyik motoron be kell állítani. Az, hogy melyik motoron kell ezt beállítani, azt az ezután kapott betű fogja eldönteni. A bufferből az előző adatot mindig le kell törölni, hogy az újonnan érkező adatban azok a számok ne szerepeljenek. Ha a küldés végén ’e’ karaktert kapunk, akkor a motorvezérlés megkezdéséhez minden adat beérkezett, így további adatfogadásra nincs szükség. A setUp és check flagek 1-es értéket kapnak, illetve megtörténik a szögbeállításhoz szükséges léptetési értékek kiszámítása. Amíg a setUp flag nem 1-es értékű, addig folytatódik az adatfogadás.



8.. ábra A főprogram folyamatábrájának második része

A setUp flag 1-es értéke esetén megkezdődhet a motorok beállítása. A check flag felel azért, hogy elindítsa a motorvezérlést. Egy időzítő és egy megszakítás kerül elindításra, amelyben 25 ms-ént a pulse flag 1-es értékre kerül beállításra. A calculateSteps() függvény meghívásakor a stepsMX változók megkapják az adott motorra kiszámolt léptetési értéket, és ez a sendMotorImpulse() meghívása során mindig 1-el csökken. Ha valamelyik változó értéke 0-ra csökkent, akkor az azt jelenti, hogy a motor beállt a kívánt értékre, arra további impulzust nem kell kiadni, amit a motorhoz tartozó done flag jelez az impulzus kiadó függvénynek. Az impulzusok kiadása után a pulse flaget 0 értékre kell billenteni, hogy a következő impulzus kiadás 25 ms múlva történjen meg. Amíg a pulse flag nem 1, addig impulzus kiadás nem történik.



8.. ábra A főprogram folyamatábrájának harmadik része

Ezután következik egy ellenőrzés, hogy mindhárom motor beállt-e a megfelelő pozícióba. Ha nem, akkor ismét ellenőrizni kell a setUp flag állapotát, és ha lehet, a motorvezérlés folytatódik. Mikor mindhárom motor beállt a kívánt pozícióba, az időzítő és a megszakítás kikapcsolásra kerül. Majd következik egy végső ellenőrzés, hogy szükség van-e korrigálásra. Újra léptetési értékeket számolunk. és ha valamelyik léptetési érték kettőnél nagyobb, vagy a kliens kért korrigálást, akkor a setUpflag ellenőrzése után ismét elkezdődik a motorvezérlés. Ha az összes léptetési érték kisebb, mint 2 a számítás után, vagy a kliens nem kért korrigálást, akkor egy 1-es üzenet kerül kiküldésre a kliensnek, hogy a motorok beállítása sikeresen megtörtént. A setUp és check flagek 0 értéket kapnak, és a klienstől várjuk a következő beállítandó értékeket.

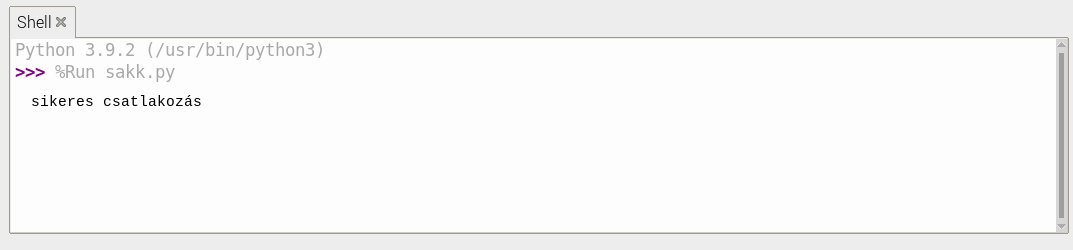
# Tesztelés

A tesztelés első fázisa a rendszer szempontjából egyik legfontosabb pontra, a kommunikációra irányult. Ha a kommunikációs kapcsolat nem jön létre a Raspberry és az ESP32 között, akkor a rendszer nem működőképes.

A kommunikáció tesztelési szempontjai az alábbiak voltak:

1. Az ESP32 a szervert sikeresen létrehozza, arra a Raspberry képes csatlakozni
2. A Raspberry, illetve az ESP32 képes adatot küldeni és fogadni egymástól

Az ESP32-t bekapcsolva rövid időn belül megjelent a Raspberry felületén egy „ESP\_SERVER” nevezetű WiFi hálózat, melyre rá lehetett csatlakozni. A Raspberryn a sakk kód indítását követően az ESP32 tudta fogadni az új klienst, melyről a megfelelő üzenetet kaptuk (9.1. ábra).



9.. ábra Visszajelzés a sikeres csatlakozásról

Ezután következett az adatküldés és fogadás tesztelése, ahol egyszerűen a Raspberryről küldtünk ki adatot az ESP32-nek, majd az üzenet fogadása után a mikrokontrollernek vissza kellett küldenie a kapott üzenetet a Raspberrynek. A teszt akkor volt sikeres, ha az elküldött üzenet megegyezett a kapott üzenettel. A tesztet hat különböző üzenet kiküldésével hajtottuk végre, melyek formátuma a korábban tárgyalt formátummal egyeztek meg. A teszt eredményei azt mutatták, hogy a kiküldött üzenetek teljesen megegyeztek az ESP32-től visszakapott üzenetekkel, így ez a teszt is sikeres volt (9.2. ábra).



9.. ábra Az ESP32-től visszakapott üzenetek

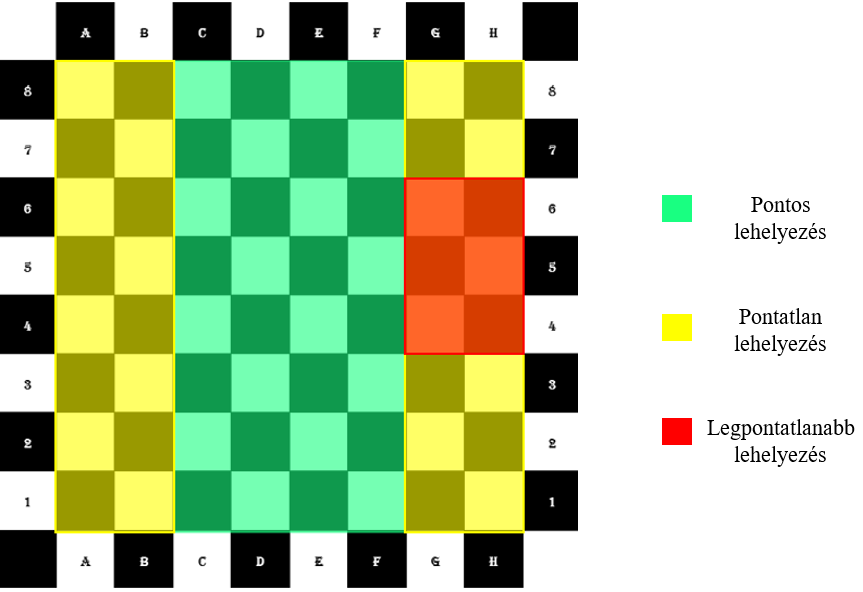
A kommunikáció tesztelése után következett a tesztelés második fázisa, a motorvezérlésnek a tesztelése. Itt a fő szempont az volt, hogy a robot milyen pontosan képes a kart, és ezzel a bábukat pozícionálni.

A motorvezérlés teszt esetei az alábbiak voltak:

1. Egy bábu felvétele, majd ugyan arra a mezőre való visszahelyezése
2. Egy bábu lerakása a tábla minden mezőjére
3. Egy bábuval több lépés megtétele, ami nagyobb, mint két mező

Az 1. sorszámmal jelölt tesztelést egy bábun végeztük el, és a metódust tízszer ismételtük meg. Az enkóderes visszacsatolásból adódóan az egyes esetek eredményei nagyon hasonlóak voltak, szinte ugyan abba a pontra helyezte le a robot a bábut. Az egyes esetek eredményeinek különbségét okozhatta az, hogy a robot egy nehezebb végeffektorral rendelkezik, illetve hosszabb passzív karokkal. Ezért a mozgás során előfordul, hogy maga a végeffektor enyhén rezeg, és így nem mindig pontosan ugyan arra a pontra helyezi le a robot a bábut.

A 2. tesztesetnél a sakktábla bal felső sarkában helyeztünk el egy bábut, és a robot segítségével mindig a következő mezőre helyeztük le. Ezzel képesek voltunk tesztelni, hogy a robot milyen pontosan képes minden mezőre elhelyezni a bábukat. A tesztelésből az derült ki, hogy a pálya középső részén sokkal pontosabban történik a pozícionálás, mint a sakktábla szélein. A 9.3.-as ábrán látható, melyek azok a mezők, ahol pontos, pontatlan lehelyezés történik, illetve hol történik a legpontatlanabb lehelyezés. A pontos esetében a bábu kis hibával kerül a mező közepére. A pontatlan helyeken a bábuk nem középre, hanem a mező valamelyik széléhez közelebb kerülnek. A legpontatlanabb elhelyezések a tábla jobb felső sarkához közelebb történtek. Az oldalsó mezők esetében a pontatlanság adódhat a motorok paramétereiből, terhelhetőségük csökkenéséből. Ez a pontatlanság viszont nem olyan mértékű, hogy a játék menetét, vagy a végeffektor mozgatását befolyásolná.



9.. ábra A lehelyezés pontossága a pályán

A 3. teszt esetén egy olyan lépést teszteltünk, mintha egy futóval, vagy egy bástyával tennénk meg egy nagyobb lépést. Egy bábut a pálya egyik sarokmezőjére helyeztünk, és innen a robotnak a másik három sarokra kellett azt lehelyeznie. A teszt során azt tapasztaltuk, hogy a 2. teszt esetben leírt pontatlanság itt is megjelent, viszont ez a pontatlanság itt se volt olyan mértékű, hogy az a rendszerben hibát okozott volna. A teszt sikeres volt, a robot le tudta helyezni a mezőkre a bábut.

# Összefoglaló

Tesztelések során bebizonyosodott, hogy a rendszert felépítő modulok a feladatukat képesek ellátni. Ezek integrációja után a kész rendszer tesztelési eredményei azt mutatták, hogy az a specifikációban leírtakat hiánytalanul teljesíti. A robot képes a sakktáblát, illetve a játékos által megtett lépést detektálni. A játékos lépésére generál válaszlépést, és ezt a motorok, illetve az elektromágnes megfelelő vezérlésével meg is teszi. Ezen felül az LCD kijelzőn keresztül aktuális információt tud szolgáltatni a játék menetéről a felhasználónak.

A tovább fejlesztés lehetősége minden modulnál lehetséges. A motorvezérlésen olyan fejlesztések hajthatók végre, mellyel gyorsabban és pontosabban tudjuk a bábuk pozícionálását végrehajtani. A kar mozgásán is lehet javítani úgy, hogy az egyes motorokat szinkronizáltan mozgatjuk, melynek megoldására több ötlet is felmerülhet.

A sakkozó robot segítségével a felhasználónak lehetősége van egy iparban használatos eszköz megismerésére, de ezen felül sakktudását is képes fejleszteni. Projektünkkel az 55. Tudományos Diákköri Konferencián részt vettünk, melyen második helyezést értünk el. Ezen felül a 2022.09.30-án megrendezett Kutatók Éjszakáján is bemutattuk a robotot, ahol gyerekek és felnőttek egyaránt kihívták a robotot egy sakkjátékra (10.1. ábra). Emellett 2022.11.03.-án a XXXVIII. Kandó Konferencián is bemutattuk. [29]



10.. ábra Érdeklődők a Kutatók Éjszakáján

# Summary

During the test period it was proven that each module - that forms the system - is able to fulfil its task. After the integration of these modules, the tests done on the complete system showed that it completely satisfies the requirements written in the specification. The robot is able to detect the chessboard and the chess move taken by the player. It generates a response move on the player’s chess move and with the appropiate motor and electromagnet control it is able to take its step too. Also via the LCD display it can provide information about the current status of the game to the user.

The possibility of improvement is present at every module. There are possible improvements that could be done on the motor control, so the positioning of the chess pieces could be done faster and more precisely. The movement of the arm could be improved by synchronized control of each motor. There can be more ideas to solve this.

The user is able to get to know an industrial device, and also to improve their knowledge of chess with the help of the chess-playing robot. We presented this project on the first time at the 55th event of Conference of Students' Research Societies and we got II. place for our work. On 2022.09.30. we also presented our work on Researchers’ Night, where children and adults could challenge the robot for a chess game. We also held a presentation at the XXXVIII. event of Kandó Conference on 2022.11.03. [29]

# Irodalomjegyzék

[1] Irati Zamalloa, Risto Kojcev, Alejandro Hernandez, Inigo Muguruza, Lander Usategui, Asier Bilbao and Víctor Mayoral: Dissecting Robotics - historical overview and future perspectives, 2017. április

<https://arxiv.org/pdf/1704.08617.pdf>

Letöltés ideje: 2022.08.25.

[2] A TRIBUTE TO JOSEPH ENGELBERGER: UNIMATE // The First Industrial Robot

<https://www.automate.org/a3-content/joseph-engelberger-unimate>

Letöltés ideje: 2022.08.26.

[3] lifeboat.com: Karen Hurst: Straight Out of Sci-Fi, Shakey Was the First Mobile Robot Built With AI, 2017.02.16.

<https://lifeboat.com/blog/2017/02/straight-out-of-sci-fi-shakey-was-the-first-mobile-robot-built-with-ai>

Letöltés ideje: 2022.08.26.

[4] KUKA Robotics hivatalos oldala: A KUKA történet, é.n.

<https://www.kuka.com/hu-hu/v%c3%a1llalat/a-kuka-r%c3%b3l/t%c3%b6rt%c3%a9nelem>

Letöltés ideje: 2022.08.27.

[5] EVS hivatalos oldala: Who Invented the First SCARA Robot?, é.n.

<https://www.evsint.com/who-invented-the-first-scara-robot/>

Letöltés ideje: 2022.08.27.

[6] ISO 8373:2021: 2021-11 Robotics – Vocabulary, 3 Terms and definitons: 3.6. Szabvány

<https://www.iso.org/obp/ui/#iso:std:iso:8373:ed-3:v1:en>

[7] Schuster György: Gyártás automatizálás jegyzet, é.n.

<https://uni-obuda.hu/users/schuster.gyorgy/gy%c3%a1rt%c3%a1sautomatiz%c3%a1l%c3%a1s.pdf>

Letöltés ideje: 2022. 08. 28.

[8] GÉP, A gépipari tudományos egyesület műszaki folyóirata: Ipari robot és manipulátorok kutatása és alkalmazása jármű-és autópari mérnöki területeken, áttekintés. 2018/2 LXIX. évfolyam

[9] Zoran PANDILOV, Vladimir DUKOVSKI: Comparison of the characteristics between serial and parallel robots, University “Sv. Kiriil I Metodij”, Faculty of Mechanical Engineering-Skopje, Karpos Ii B.B., P.O.Box 464, Mk-1000, Skopje, Republic of MACEDONIA, 2014

<https://acta.fih.upt.ro/pdf/2014-1/ACTA-2014-1-19.pdf>

Letöltés ideje: 2022. 08. 28.

[10] Xingguo Lu, Ming Liu: Optimal Design and Tuning of PID-type Interval Type-2 Fuzzy Logic Controllers for Delta Parallel Robots, Harbin Institute of Technology, Harbin, Heilongjiang, 2016.04.25.

<https://www.researchgate.net/publication/303469087_Optimal_Design_and_Tuning_of_PID-Type_Interval_Type-2_Fuzzy_Logic_Controllers_for_Delta_Parallel_Robots>

Letöltés ideje: 2022. 08. 29.

[11] Szilágyi Zoltán: Ipari robotika kézikönyv, 2018

<https://www.zoltanszil.hu/wp-content/uploads/2021/07/KO%CC%88NYV_Ipari_robotika_kezikonyv.pdf>

Letöltés ideje: 2022. 08. 30.

[12] Mehdi Azmoun, Ali Rouhollahi, Mehdi Tale Masouleh, Ahmad Kalhor: The 4 DOF Delta manipulator built at Human and Robot Interaction Laboratory University, h.n., 2019. március

<https://www.researchgate.net/publication/331639411_Kinematics_and_Control_of_a_4-DOF_Delta_Parallel_Manipulator>

[13] Jan J. de Jong, Arno H.A. Stienen, Volkert van der Wijk, Martijn Wessels, Herman van der Kooij: A Method for Evaluation and Comparison of Parallel Robots for Safe Human Interaction, Applied to Robotic TMS, Róma, Olaszország, 2012. június

<https://www.researchgate.net/figure/Three-6-DOF-manipulator-concepts-Left-Delta-robot-with-rotation-head-FANUC-M-3iA-6A_fig1_261480208>

[14] cncpart.hu: 57HS76-2804A bipoláris léptetőmotor paraméterei és bekötése

<https://www.cncpart.hu/custom/cncpart/image/data/srattached/82da43425c1617425b4394c85b7f3fc1_57HS76-2804A.pdf>

Letöltés ideje: 2022. 09. 06.

[15] Power Belt: DM556D (93.034.030) típusú léptetőmotor meghajtó FELHASZNÁLÓI KÉZIKÖNYV

<https://www.powerbelt.hu/uploads/source/catalogs/DM556D_felhaszn%C3%A1l%C3%B3i_k%C3%A9zik%C3%B6nyv.pdf>

Letöltés ideje: 2022. 09. 06.

[16] TME webáruház: SP-200 series dokumentáció

<https://www.tme.eu/Document/4fc31215ee0db3ea3b629e14a72e2457/SP-200-SPEC.PDF>

Letöltés ideje: 2022. 09. 06.

[17] AS5048A mágneses forgójeladó adatlap

<https://ams.com/documents/20143/36005/AS5048_DS000298_4-00.pdf>

Letöltés ideje: 2022. 09. 09.

[18] Espressif hivatalos oldala: ESP32 adatlap

<https://espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>

Letöltés ideje: 2022. 09. 09.

[19] manuals.plus weboldal: ESP32-CAM modul adatlap

<https://manuals.plus/m/c64bcd87595b77f0d44f0f6b0bafd03bb5b01313385e3efe3567c267b6390c8c_optim.pdf>

Letöltés ideje: 2022. 09. 09.

[20] Hestore webáruház: LCD-I2C-IF interfész adapter modul leírás

<https://www.hestore.hu/prod_10035516.html>

Letöltés ideje: 2022. 09. 09.

[21] Raspberry Pi weboldala: Raspberry Pi 3 Model B+ termékdokumentáció

<https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf>

Letöltés ideje: 2022. 09. 10.

[22] Raspberry Pi weboldala: Raspberry Pi kamera modul ismertető: Hardver specifikáció

<https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf>

Letöltés ideje: 2022. 09. 11.

[23] EL-PRO-CUS weboldal cikke: What is a Stepper Motor: Types & Its Working, h.n.,é.n.

<https://www.elprocus.com/stepper-motor-types-advantages-applications/?fbclid=IwAR2d_O00pp_GD_JtUfsEl-1xJj7EZXptN52gUUzHgs_88YIKqb27wlxEARw>

Letöltés ideje: 2022. 09. 12.

[24] TME hivatalos oldalának cikke: Léptetőmotorok - típusok és alkalmazásuk, 2020.09.08.

<https://www.tme.eu/hu/news/library-articles/page/41861/Leptetomotorok-tipusok-es-alkalmazasok/>

Letöltés ideje: 2022. 09. 12.

[25] alibaba.com webáruház: 57HS76-2804 stepper motor

<https://www.alibaba.com/product-detail/57HS76-2804-stepper-motor-power-cnc_60582294565.html>

Letöltés ideje: 2022. 09. 12.

[26] Hestore webáruház: OBS-IR infravörös akadályérzékelő

<https://www.hestore.hu/prod_10038536.html?gross_price_view=1&source=gads&gclid=EAIaIQobChMI1aXUw9ea9wIVhOJ3Ch0J9QGBEAYYASABEgLVOvD_BwE>

Letöltés ideje: 2022. 09. 12.

[27] Tihanyi Gergely: Delta-robot tervezése, OE-KVK, 2021

[28] Zardikán Dzsanbolat Najman: Delta robot, OE-KVK, 2020

[29] Kovács Gergő János, Schmidt Péter, Sándor Tamás, Borsos Döníz: Sakk játék megvalósítása delta robot segítségével, XXXVIII. Kandó Konferencia,OE-KVK, 2022.11.03.

# Mellékletek

|  |  |  |  |
| --- | --- | --- | --- |
| **Változó neve** | **Típusa** | **Értéke** | **Magyarázat** |
| ssid | const char\* | „ESP\_SERVER” | A Wi-fi hálózat neve |
| password | const char\* | - | A Wi-fi hálózat jelszava |
| buff | String | Változó | Klienstől érkező számadatokat tárolja |
| angleSensor1 | float | Változó | Az 1-es motoron lévő enkóder által mért motorszöget tárolja |
| angleSensor2 | float | Változó | A 2-es motoron lévő enkóder által mért motorszöget tárolja |
| angleSensor3 | float | Változó | A 3-as motoron lévő enkóder által mért motorszöget tárolja |
| angleToSet1 | float | Változó | Beállítandó szög az 1-es motoron |
| angleToSet2 | float | Változó | Beállítandó szög az 1-es motoron |
| angleToSet3 | float | Változó | Beállítandó szög az 1-es motoron |
| calcStep1 | uint32\_t | Változó | 1-es motor beállításához kiszámolt lépés |
| calcStep2 | uint32\_t | Változó | 2-es motor beállításához kiszámolt lépés |
| calcStep3 | uint32\_t | Változó | 3-as motor beállításához kiszámolt lépés |
| stepsM1 | uint32\_t | Változó | 1-es motor beállításához szükséges lépések száma |
| stepsM2 | uint32\_t | Változó | 2-es motor beállításához szükséges lépések száma |
| stepsM3 | uint32\_t | Változó | 3-as motor beállításához szükséges lépések száma |
| timerTicks | uint32\_t | 25000 | Időzítő értéke az interrupthoz |
| magnetVar | uint8\_t | 0 vagy 1 | Az elektromágnes be ki kapcsolásához szükséges érték |

1. melléklet Adatokat tároló változók

# Ábrajegyzék

[2.1. ábra Unimate, az első szabadalmaztatott ipari robot [2] 9](#_Toc119954033)

[2.2. ábra A Shakey (bal oldalt) [3] és a Famulus (jobb oldalt) [4] 9](file:///C:\Users\gkova\Desktop\szakdolgozat\szakdoga_verziók\szakdolgozat_kgj_v16.docx#_Toc119954034)

[2.3. ábra Első SCARA prototípus [5] 10](#_Toc119954035)

[2.4. ábra Soros manipulátor vázlata 13](#_Toc119954036)

[2.5. ábra Párhuzamos manipulátor vázlata [9] 14](#_Toc119954037)

[2.6. ábra A delta robot vázlata [10] 14](#_Toc119954038)

[2.7. ábra Négy (bal oldalt) [12], illetve hat (jobb oldalt) szabadságfokú [13] delta robot 15](#_Toc119954039)

[4.1. ábra Logikai rendszerterv 20](#_Toc119954040)

[5.1. ábra A fizikai rendszerterv első változata 21](#_Toc119954041)

[5.2. ábra A delta robot 22](#_Toc119954042)

[5.3. ábra ESP32-CAM modullal készített kép 25](#_Toc119954043)

[5.4. ábra A Raspberry Pi-al készített kép 26](#_Toc119954044)

[5.5. ábra A fizikai rendszerterv végső változata 26](#_Toc119954045)

[6.1. ábra Bipoláris léptetőmotor tekercseinek bekötése [24] 27](#_Toc119954046)

[6.2. ábra 57HS76-2804A típusú bipoláris léptetőmotor [25] 28](#_Toc119954047)

[6.3. ábra A meghajtó vezérlési diagramja [15] 29](#_Toc119954048)

[6.4. ábra OBS-IR infravörös akadályérzékelő modul [26] 31](#_Toc119954049)

[7.1. ábra Daisy Chain kialakítás [17] 35](#_Toc119954050)

[7.2. ábra Többszörös Slave-szelekció [17] 35](#_Toc119954051)

[8.1. ábra calculateSteps() folyamatábrája 41](#_Toc119954052)

[8.2. ábra A főprogram folyamatábrájának első része 42](#_Toc119954053)

[8.3. ábra A főprogram folyamatábrájának második része 43](#_Toc119954054)

[8.4. ábra A főprogram folyamatábrájának harmadik része 44](#_Toc119954055)

[9.1. ábra Visszajelzés a sikeres csatlakozásról 46](#_Toc119954056)

[9.2. ábra Az ESP32-től visszakapott üzenetek 47](#_Toc119954057)

[9.3. ábra A lehelyezés pontossága a pályán 48](#_Toc119954058)

[10.1. ábra Érdeklődők a Kutatók Éjszakáján 49](#_Toc119954059)