

Debreceni Egyetem

Műszaki Kar

Villamosmérnöki és  
Mechatronikai Tanszék



## **Ember és gép közötti kommunikáció megvalósítása öntanuló robotkarral**

Készítette: **Zilahi Krisztián László**, VLWZND, mechatronikai mérnök hallgató

Belső konzulens: **Dr. Tóth János**

Tanszékvezető: **Dr. Husi Géza**

Debrecen

2013.

# Tartalomjegyzék

|                                                               |           |
|---------------------------------------------------------------|-----------|
| <b>Ábrajegyzék.....</b>                                       | <b>3</b>  |
| <b>Bevezetés .....</b>                                        | <b>4</b>  |
| <b>1. A hardver elemek bemutatása.....</b>                    | <b>5</b>  |
| 1.1. Az RTT robotkarok általános jellemzése .....             | 6         |
| 1.2. A Fishertechnik robotkar modell felépítése .....         | 8         |
| 1.3. A robotkar hajtásához felhasznált elemek.....            | 9         |
| <b>2. A robotkart irányító ipari eszközök .....</b>           | <b>12</b> |
| 2.1. A felhasznált HMI.....                                   | 12        |
| 2.2. A felhasznált PLC .....                                  | 14        |
| 2.3. A PLC és HMI kommunikációja.....                         | 16        |
| <b>3. A dolgozat során megvalósított feladatok .....</b>      | <b>18</b> |
| 3.1. A HMI menü struktúrájának felépítése .....               | 18        |
| 3.2. Az alaphelyzet felvétel .....                            | 19        |
| 3.3. A kézi mozgatás funkció .....                            | 21        |
| 3.4. A betanító üzemmód .....                                 | 24        |
| 3.5. A kézi programozás funkció .....                         | 28        |
| 3.6. A „Mefogó helyzete” opció bemutatása .....               | 29        |
| 3.7. A motorok sebességvezérlése a pozíció függvényében ..... | 30        |
| 3.8. A PLC főprogramjának strukturális szerkezete .....       | 31        |
| <b>4. Fejlesztési lehetőségek .....</b>                       | <b>32</b> |
| <b>5. Összegzés .....</b>                                     | <b>33</b> |
| <b>6. Felhasznált irodalom .....</b>                          | <b>35</b> |

# Ábrajegyzék

|                                                                                                 |    |
|-------------------------------------------------------------------------------------------------|----|
| 1. ábra: Az átalakított Fishertechnik robotkar .....                                            | 5  |
| 2. ábra: Az RTT típusú robotkar és munkatere.....                                               | 6  |
| 3. ábra: A TCP elhelyezkedése a $O_0$ bázis ponthoz viszonyítva .....                           | 7  |
| 4. ábra: Fishertechnik gépelemek .....                                                          | 8  |
| 5. ábra: Az ízületek helyzetének mérését szolgáló elemek.....                                   | 9  |
| 6. ábra: Villanymotorok, hajtómű.....                                                           | 9  |
| 7. ábra: Emitterkövető kapcsolás .....                                                          | 10 |
| 8. ábra: A ki- és bementi feszültségek közötti összefüggés.....                                 | 10 |
| 9. ábra: A motorok bekötésének kapcsolási rajza .....                                           | 11 |
| 10. ábra: Phoenix Contact TP 12T típusú HMI .....                                               | 12 |
| 11. ábra: A VISU+ kezelő felülete.....                                                          | 13 |
| 12. ábra: Phoenix Contact ILC 150 ETH típusú PLC.....                                           | 14 |
| 13. ábra: A PLC és a bővítő modulok.....                                                        | 14 |
| 14. ábra: A PC WORX kezelőfelülete .....                                                        | 15 |
| 15. ábra: A függőleges ízület (z) vezérlésének funkcióblokkjában használt változók listája..... | 16 |
| 16. ábra: OPC Configurator kezelőfelülete.....                                                  | 16 |
| 17. ábra: A PLC és HMI összekapcsolása .....                                                    | 17 |
| 18. ábra: A rendszer kommunikációjának folyamatábrája .....                                     | 18 |
| 19. ábra: A HMI kezelő felületének menüpontjai.....                                             | 18 |
| 20. ábra: Az alaphelyzet felvételnek képernyője .....                                           | 19 |
| 21. ábra: Az alaphelyzet felvétel PLC programja.....                                            | 20 |
| 22. ábra: A kézi mozgatás funkció kezelőfelülete .....                                          | 21 |
| 23. ábra: A függőleges (z) ízület vezérlése.....                                                | 22 |
| 24. ábra: A betenítő üzemmód felülete .....                                                     | 24 |
| 25. ábra: A pozíció rögzítésének lehetséges feltételei .....                                    | 25 |
| 26. ábra: A feltételek figyelése a tanuló programban .....                                      | 26 |
| 27. ábra: A koordináták és a hozzátartozó idők mentése.....                                     | 26 |
| 28. ábra: A megtanult program visszajátszása.....                                               | 27 |
| 29. ábra: A megtanult értékek törlése .....                                                     | 27 |
| 30. ábra: A kézi programozás képernyője .....                                                   | 28 |
| 31. ábra: A megfogó helyzetének megjelenítése .....                                             | 29 |
| 32. ábra: A forgó ízület motorjának sebességvezérlése .....                                     | 30 |
| 33. ábra: A főprogram strukturális szerkezete (részlet).....                                    | 31 |
| 34. ábra: A dolgozat során létrehozott rendszer .....                                           | 33 |

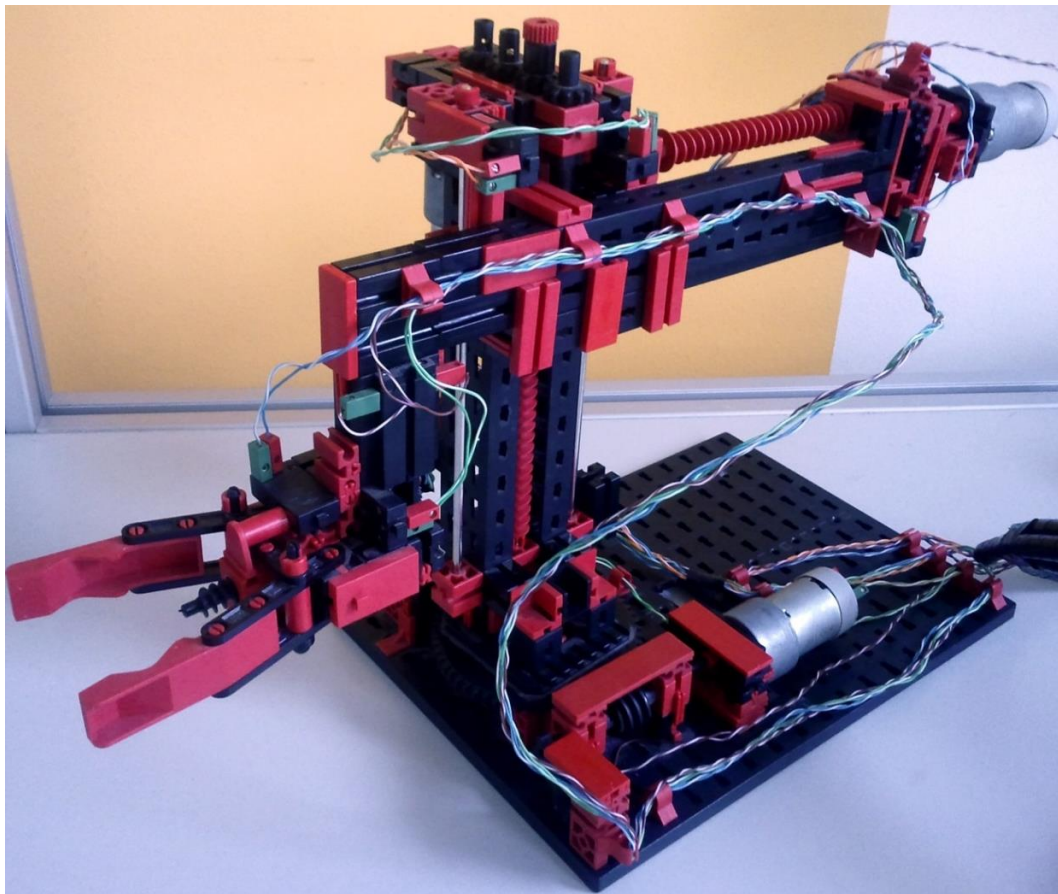
# Bevezetés

Tanulmányaim során a Debreceni Egyetem, Műszaki Karának Villamosmérnöki és Mechatronikai Tanszékén sok lehetőségem adódott különböző oktatási és ipari eszközök megismerésére. A DE-HIGH VOLTAGE, elektromos járműfejlesztő csapatban való részvétel során ismereteket szereztem az iparban használt hajtásokról (szinkron, aszinkron motorok), vezérlőegységeikről (frekvenciaváltók, ipari megjelenítők) és azok kommunikációjáról (TCP/IP MODBUS). Továbbá az előző TDK alatt kifejlesztett PLC-s (Programable Logical Controller) mérőrendszerben felhasznált PLC (Rexroth L20), mérőszenzorok és megjelenítő felületek kapcsolatát is megismertem. Részt vettem a XIX. Országos Irányítástechnikai Programozó Versenyen, ahol tovább bővíthettem és próbára tehettem a PLC programozói tudásomat. A fenti tevékenységek során megszerzett, tapasztalataim inspiráltak, hogy meglévő eszközökből egy újragondolt szerkezetet hozzak létre. Így a tanszék eszközei közül a Fishertechnik robotkar modellre, a Phoenix Contact ILC 150 ETH típusú programozható logikai vezérlőjére és a Phoenix Contact TP 12T típusú ipari operátor paneljére esett a választás.

A kitűzött célok között szerepel a robotkar mechanikájának és elektronikájának megismerése, továbbfejlesztése, a PLC fejlesztőkörnyezetének kezelésének és programozásának elsajátítása. Céлом még a PLC és az operátorpanel közötti kommunikáció megvalósítása és új megjelenítő grafika készítése. A végső szándékom egy betanítási funkcióval rendelkező komplex robotkar létrehozása, mely az ipari kijelzőről irányítható.

# 1. A hardver elemek bemutatása

Ahhoz, hogy sikeresen tudjam a kitűzött célokat elvégezni egy modell robotkart kellett tanulmányoznom, ami a dolgozatom alapjául fog szolgálni. Robotnak tekinthető minden olyan újraprogramozható többfunkciós manipulátor, melyek programozott mozdulatsorok alapján látja el funkcióját. Jelen esetben a robotkar pont-szakasz vezérlésűnek tekinthető, ez azt jelenti, hogy a munkatérén belül a manipulátor megfogójának pozíciója tetszőlegesen választható, de a kívánt pontba történő mozgáskor a pályagörbét a robotkar felépítése szabja meg. Egyszerűbb pályavezérlések megvalósíthatóak az ízületek mozgatásának sorrendi vagy egyszerre történő mozgatásával.



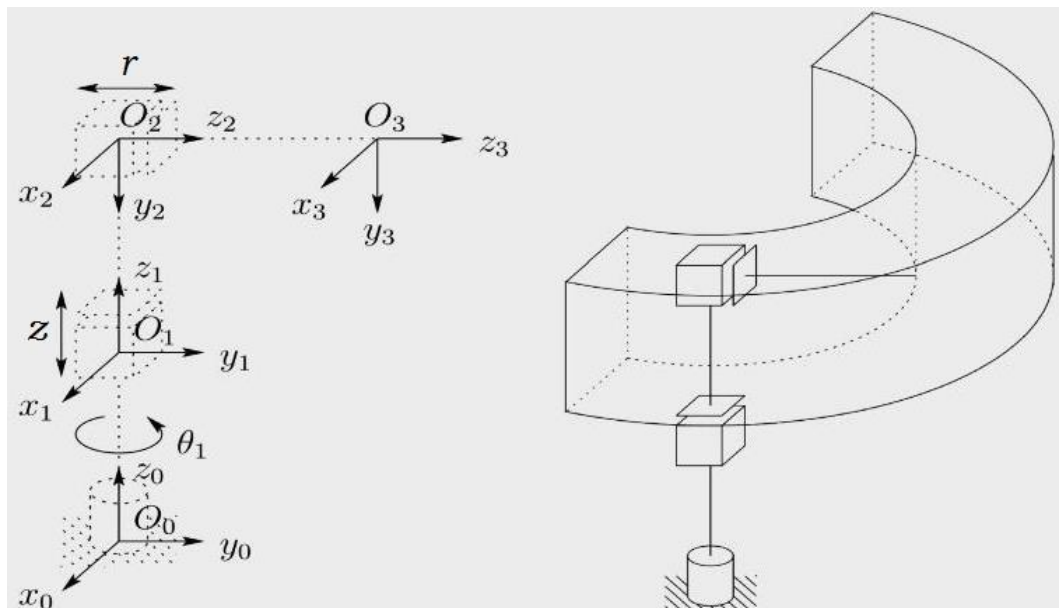
*1. ábra: Az átalakított Fishertechnik robotkar*

Első lépésként megvizsgáltam a hajtáshoz szükséges elemeket, majd az általam kidolgozott elvek szerint építettem át a robot egyes részegységeit (1. ábra). Lecseréltem az ízületeket mozgató motorokat és azok felfogatását. Megerősítettem a robotkar

szerkezeti vázát, így megnőtt a mechanikai stabilitása. Az átalakításoknak megfelelően meghatároztam a hajtások áttételeit.

### 1.1. Az RTT robotkarok általános jellemzése

A TDK dolgozatom során a Fishertechnik nevű cég által gyártott építőelemekből felépített ipari robotkar modellt használtam. A vezérléséhez szükség van a robotkar kinematikájának ismeretéhez.



2. ábra: Az RTT típusú robotkar és munkatere

A robotkar kinematikai láncá szerinti egy RTT típusú (2. ábra), amely egy rotációs és az ahhoz csatlakozó csuklóval és két translációs ízülettel rendelkezik, ennek megfelelően a robot manipulátor 3 szabadságfokkal rendelkezik. A kinematikai lánc nyílt, melynek végén egy megfogó található. Az RTT típusú robotok munkatere egy hengergyűrű (2. ábra), tehát henger koordináta rendszerben dolgozik. A robotkar kinematikájából adódóan érdemes az  $O_0$  ponthoz rögzíteni a bázispontot, mely ponthoz csatoljuk a világi koordináta-rendszert, valamint a robot koordináta-rendszert. A robot koordináták meghatározzák a robot manipulátor effektorának (megfogó) helyzetét. Ennek pozíciója a Descartes-féle koordináta-rendszer koordinátái ( $x$ ,  $y$ ,  $z$ ) segítségével, írható le, orientációja a módosított Euler-szögekkel adható meg ( $\varphi$ ,  $\theta$ ,  $\psi$ ). Mivel a robot manipulátor 3 szabadságfokkal rendelkezik, így az effektor csak az alapvető pozicionálási feladatokat tudja ellátni, külön orientáció beállítására nincs lehetőség. Az ízületek mozgó koordináta rendszereit a Denavit-Hartenberg konvekció alapján vettem

fel (2. ábra). A felvett koordináta rendszerekre felírjuk az egyes ízületekhez tartozó Denavit-Hartenberg transzformációs mátrixokat (1), (2), (3) [1].

A forgó ízülethez tartozó transzformációs mátrix:

$${}^0D_1 = \begin{pmatrix} \cos(\theta_0) & -\sin(\theta_0) & 0 & 0 \\ \sin(\theta_0) & \cos(\theta_0) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1)$$

A függőlegesen mozgó ízülethez tartozó transzformációs mátrix:

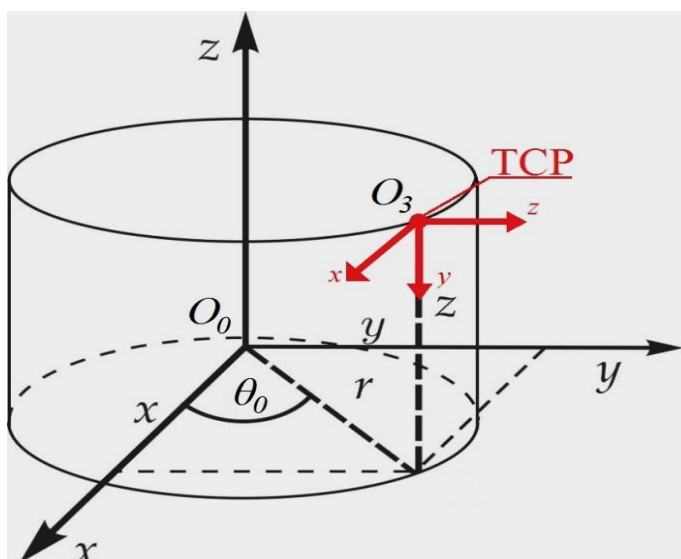
$${}^1D_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2)$$

A vízszintes irányban mozgó ízülethez tartozó transzformációs mátrix:

$${}^2D_3 = \begin{pmatrix} 1 & 0 & 0 & r \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (3)$$

Ezután az  $O_0$  rögzített ponthoz kötött Denavit-Hartenberg féle homogén transzformációs mátrix (4) az egymást követő koordináta rendszerek szorzatával határozható meg.

$${}^0T_3 = {}^0D_1 \cdot {}^1D_2 \cdot {}^2D_3 = \begin{pmatrix} \cos(\theta_0) & -\sin(\theta_0) & 0 & r \cdot \cos(\theta_0) \\ \sin(\theta_0) & \cos(\theta_0) & 0 & r \cdot \sin(\theta_0) \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (4)$$



3. ábra: A TCP elhelyezkedése a  $O_0$  bázis ponthoz viszonyítva

A megkapott homogén transzformációs mátrix egyértelműen leírja a TCP (Tool Center Point), azaz a megfogó középpontjának helyzetét, mely az  $O_3(r \cdot \cos\theta_0, r \cdot \sin\theta_0, z)$  pont, valamint az arra illesztett TCP koordináta-rendszer orientációját (3. ábra). A mátrix három változó paramétert ( $\theta_0, r, z$ ) tartalmaz, mely paraméterek a kar elfordulásának szögét ( $\theta_0$ ), a megfogó magasságát ( $z$ ) és annak kinyúlásának hosszát ( $r$ ) jelentik. A manipulátor vezérlése a fenti paraméterek, azaz a csuklókoordináták változtatásával történik, míg az effektor pozíciója világkoordináták szerint jelenítődik meg.

## 1.2. A Fishertechnik robotkar modell felépítése

A robotkar egyszerű építőelemekből kellett összeállítani. A hajtó motorok forgó mozgása egyszerű gépelemek segítségével alakítódnak át az ízületek mozgásához megfelelő mozgásra.

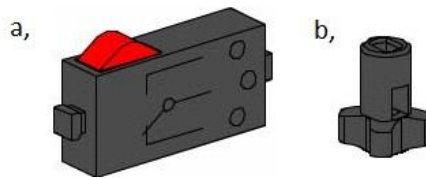


4. ábra: Fishertechnik gépelemek  
(a, csigakerék; b, csiga; c, menetes orsó; d, anya) [2], [3], [4], [5]

A forgó ízület mozgásához szükséges nyomatékot csigahajtáson (4/a,b. ábra) keresztül egy villanymotor szolgáltatja ( $\theta$  szög). A transzlációs (haladó) mozgást végző ízületek mozgását egy villanymotorhoz rögzített menetes orsó (4/c. ábra) és a mozgó ízülethez rögzített anya (4/d. ábra) hozza létre ( $r, z$  ízület).

A megfogó mozgása hasonlóan lett megvalósítva egy csuklós mechanizmuson által, mint az előbb említett ízületeknél. A transzlációs mozgást végző ízületek számára a megfelelő pályát vezetősin segítségével csúszkák biztosítják.

A csigahajtás jellegéből adódóan lassító áttétel valósul meg, ahol a csigakerék fogszáma  $z_1=58$ , tehát 58:1 az áttétel aránya. A motor tengelyéhez rögzített csiga egy körülfordulással az áttétből következően  $360^\circ/58$ , azaz  $6,2^\circ$ -ot mozdit a forgó ízületen ( $\theta$ ). A menetes orsó - anya kapcsolatánál mindkét ízület ( $r, z$ ) esetén a menetes orsó menet emelkedésével ( $P=5 \text{ mm}$ ) egyezik meg az elmozdulás.



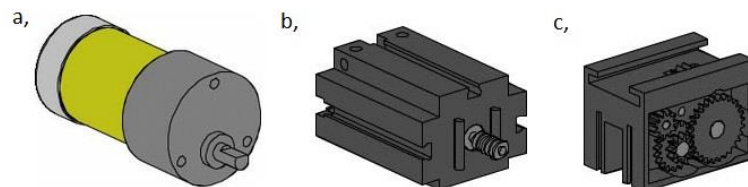
5. ábra: Az ízületek helyzetének mérését szolgáló elemek  
(a, mikro kapcsoló; b, bütykös tengely) [6], [7]

Az ízületek aktuális állapotáról az ízületeken külön-külön elhelyezett bütykös mechanizmus és a hozzákapcsolódó mikrokapcsoló (5/a. ábra) ad információt. A bütykös tengely (5/b. ábra) közvetve a motor tengelyéhez van rögzítve, így fordulat száma megegyezik a motoréval. Kialakításából adódóan egy körülforgás alatt 4-szer hozza működésbe a mikrokapcsolót. A mikrokapcsoló által kiadott impulzusok feldolgozásával az ízület pozíciója meghatározható.

A pontosság érdekében minden ízületen el van helyezve egy-egy végállás kapcsoló (5/a. ábra), tehát a bütykös mechanizmusok által nyújtott információk kalibrálhatók.

### 1.3. A robotkar hajtásához felhasznált elemek

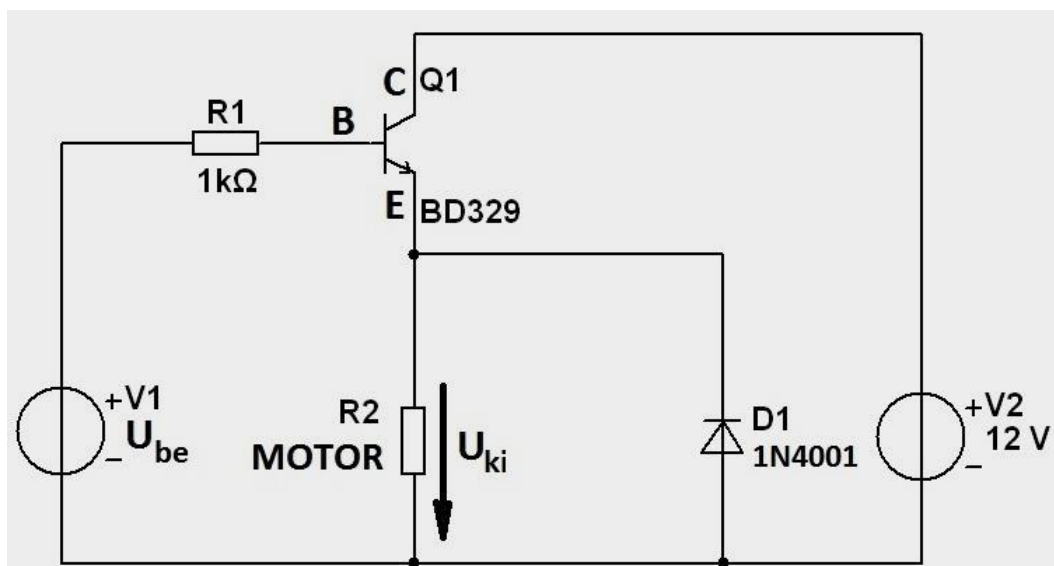
Az ízületek mozgásához szükséges nyomatékot egyenáramú szénkefés villanymotorok (soros DC motorok) szolgáltatják.



6. ábra: Villanymotorok, hajtómű  
(a, 9 V-os motor; b, 24 V-os motor, c, külső hajtómű) [8], [9], [10]

A forgató és a két haladó mozgást végző ízületeknél 9 V-os névleges feszültségű motorok (6/a ábra) kerültek elhelyezésre, ezek belső hajtóműves kialakításúak. A bennük elhelyezett bolygóműves hajtómű 20:1-hez lassító áttétet eredményez. A megfogó szerkezetet egy 24 V-os motor (6/b ábra) működteti. A megfelelő fordulatszám elérése érdekében egy hajtóművön (6/c ábra) keresztül adja át a nyomatékot.

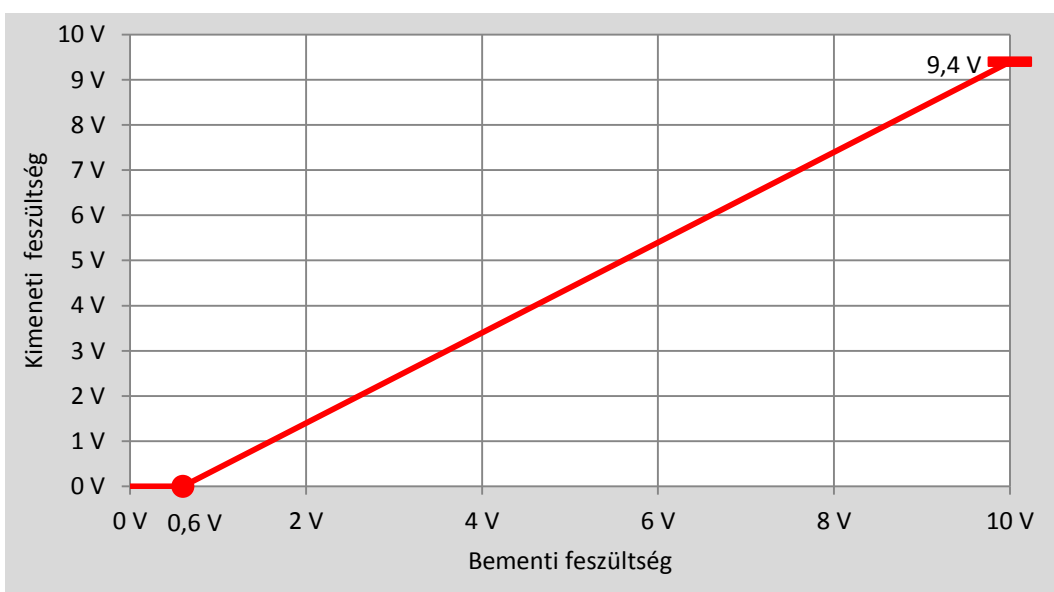
Az ízületek mozgatása során fellépett az a probléma, hogy a motorok áramelvétele után az ízületek tehetetlensége miatt a mozgás még nem fejeződött be, így nem a kívánt pozícióra állt be a robotkar, hanem azon túllendülve lengeni kezdett.



7. ábra: Emittorkövető kapcsolás

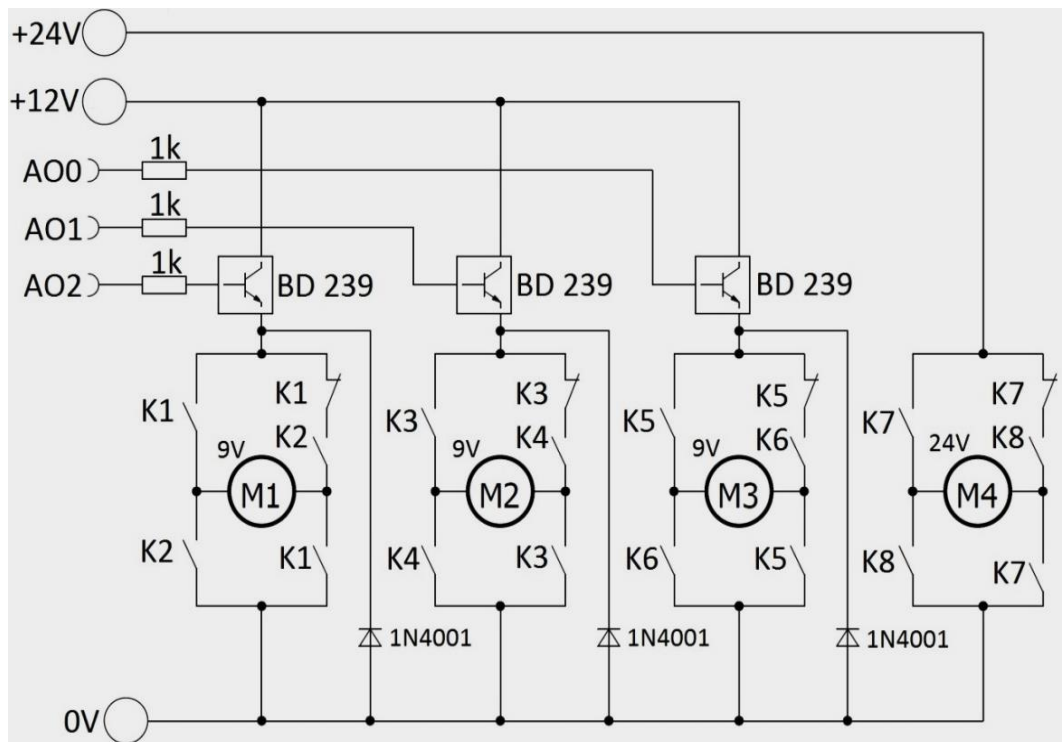
A jelenség kiküszöbölése érdekében a motor sebességét egy emittorkövető kapcsolás (7. ábra) segítségével vezéreltem.

A kapcsolás az áramerősítés célját szolgálja. A bemeneti feszültség ( $U_{be}$ ) a tranzisztor bázisára megy, a kimeneti feszültség ( $U_{ki}$ ) emittoren jelenik meg.



8. ábra: A ki- és bementi feszültségek közötti összefüggés

A be és kimeneti feszültség majdnem megegyezik, csak a bázis-emitter nyitófeszültségének nagyságával (kb. 0,6 V) tér el a kimeneti feszültség (8. ábra). Tehát:  $U_{ki} = U_{be} - 0,6 \text{ V}$ . A kapcsolás tulajdonságaiból adódóan elegendő egy vezérlő jellel beállítani a kívánt motor fordulatszámot [11].



9. ábra: A motorok bekötésének kapcsolási rajza

A motor forgásirányának megváltoztatására reléekkel megvalósított H-hidas kapcsolást alkalmaztam (9. ábra). A kapcsoláshoz 8 darab Finder 24 VDC ipari relét használtam, ezek biztosítják a forgásirányváltást.

## 2. A robotkart irányító ipari eszközök

A robotkar mozgatásához a fizikai építőelemeken kívül olyan programozható eszközöket használtam fel, amelyek segítségével a kezelő vezérelni tudja a megépített robotkart. A felhasználó számára a rendszerrel való kapcsolatot a HMI (Human-Machine Interface) biztosítja. A HMI és a robotkar között a PLC tarja a kapcsolatot, a PLC digitális kimenetei a motort kapcsoló relék tekercseihez vannak bekötve, a roboton lévő mikrokapcsolók jelei a PLC digitális bemeneteibe érkeznek. A HMI és PLC kapcsolata egy INTERBUS - OPC (Object Linking and Embedding for Process Control) szerveren keresztül történik, TCP/IP protokollon keresztül, mely egy ipari kommunikációs szabvány. Az ipari eszközök kapcsolatát és kommunikációját a programozható egységek megfelelő beállításával és fizikai összeköttetések (huzalozások) elkészítésével valósítottam meg [12].

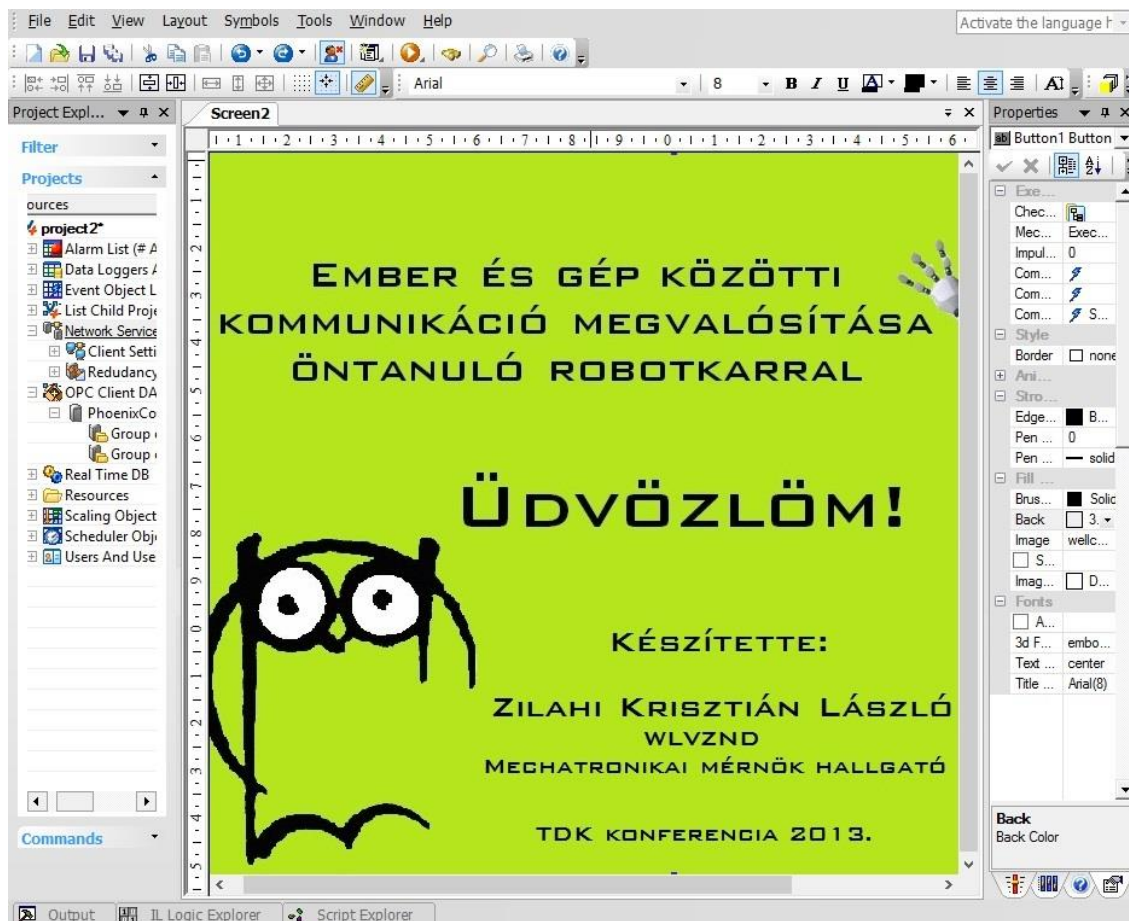
### 2.1. A felhasznált HMI



10. ábra: Phoenix Contact TP 12T típusú HMI [13]

A robotkar vezérléséhez a kezelő felületet egy Phoenix Contact TP 12T típusú ipari operátorpanel (10. ábra), azaz HMI szolgáltatja. A HMI kialakítását tekintve egy 12,1” (30,7 cm) képátlójú rezisztív (nyomás érzékeny) 800 x 600 képfelbontású 16 bites

színmélységű képernyővel rendelkezik. Az operátor panel Windows CE operációs rendszert futtat, melyen keresztül indítható el a megjeleníteni kívánt alkalmazás. A HMI bekapcsolása után a „Config” ikonra nyomva végezhetők el a panel beállításai. Ezen belül a hálózati beállításoknál az általam használni kívánt IP címet (192.168.0.15). Konfiguráltam a későbbiekben részletezett (2.3. alfejezet) OPC-beállításokat, megadtam a PLC típusát és IP címét.



11. ábra: A VISU+ kezelő felülete

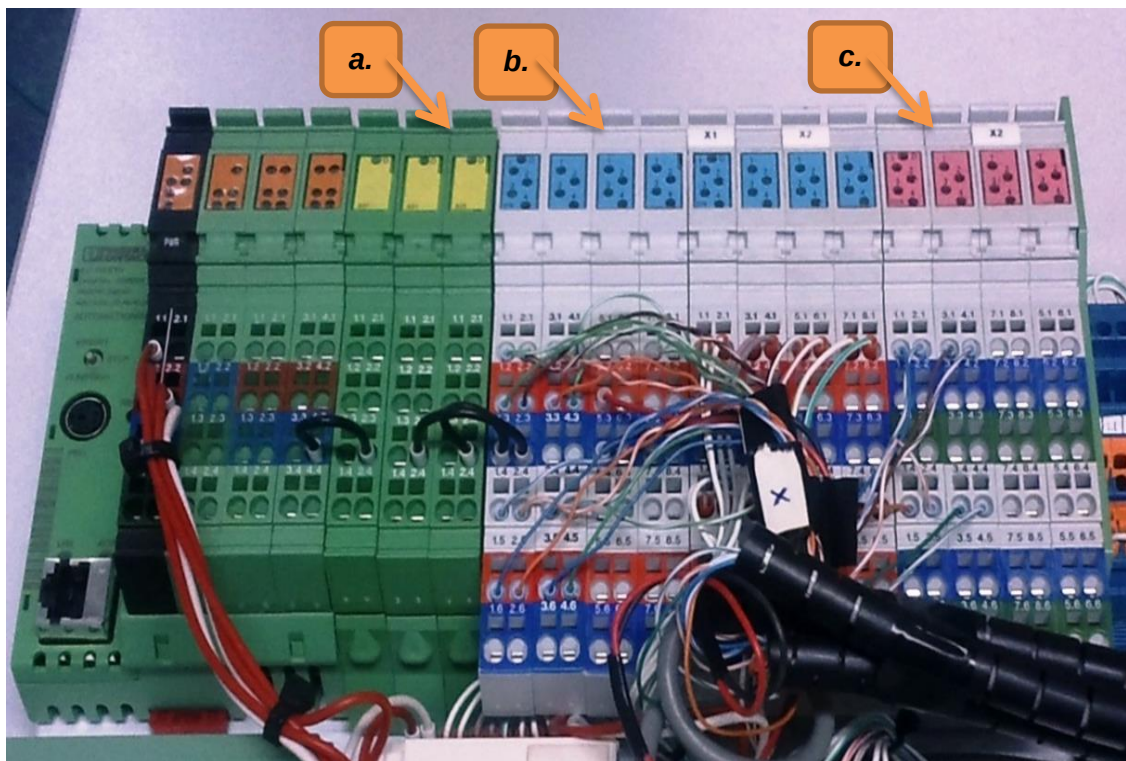
Az alkalmazások a VISU+ nevű szoftver (11. ábra) segítségével hoztam létre. Az applikációk megjelenése tetszés szerint variálható. A megjelenítő felület egyedi menü struktúrákkal tehető tagolttá a kezelő számára. Grafikonok és diagramok formájában a felhasználó számára fontos információk jeleníthetők meg.

## 2.2. A felhasznált PLC



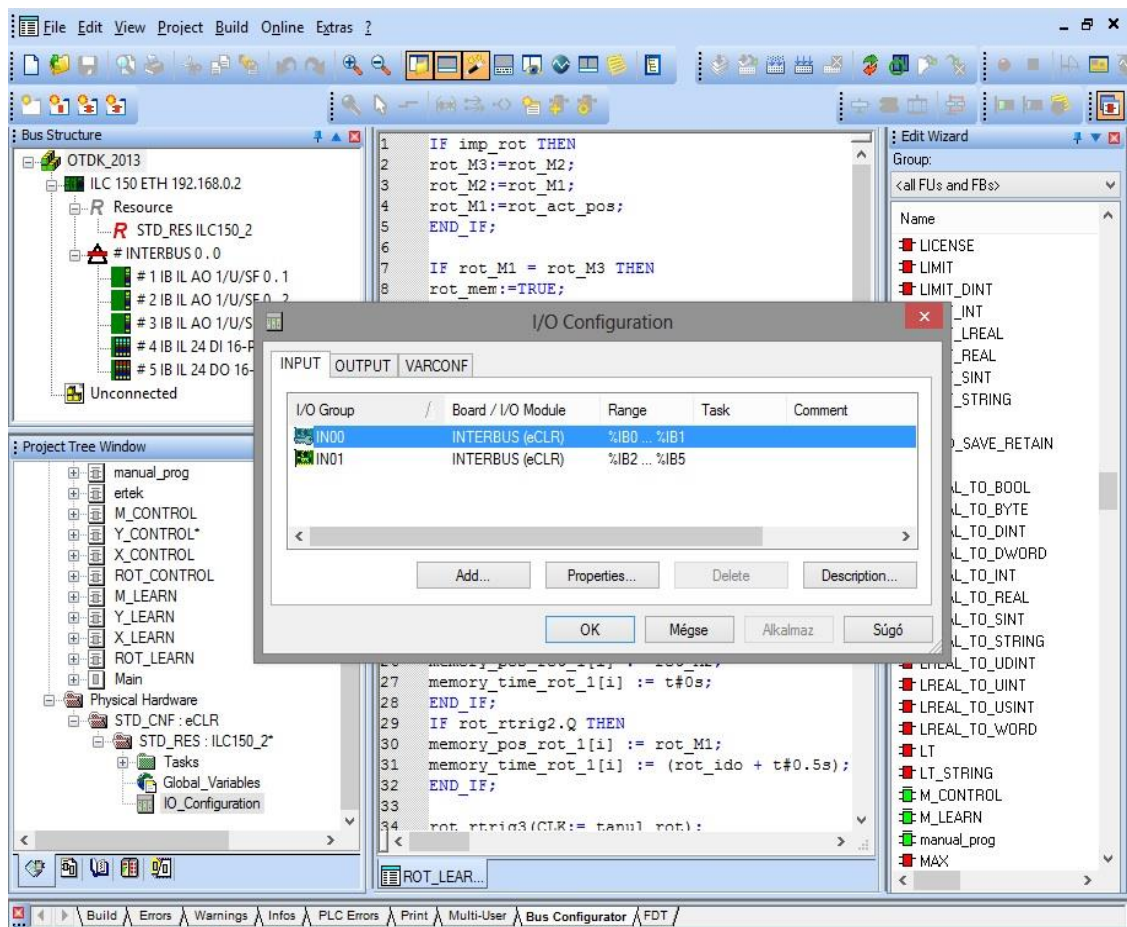
12. ábra: Phoenix Contact ILC 150 ETH típusú PLC [14]

A robot vezérlését egy Phoenix Contact ILC 150 ETH típusú PLC (Programozható Logikai Vezérlő) (12. ábra) látja el. Ez a típus a moduláris felépítésű programozható logikai vezérlők csoportjába tartozik. Igény szerint megfelelő bővítő modulokkal látható el. A robot kar vezérléséhez 8 digitális bemenetre, 8 digitális kimenetre és 3 analóg kimenetre volt szükségem.



13. ábra: A PLC és a bővítő modulok

A digitális bemeneti modul Rexroth R-IB IL 24 DI 16-PAK (13/b. árba), a digitális kimentei modul Rexroth R-IB IL 24 DO 16-PAK (13/c. árba), míg az analóg kimentei modulok Phoenix Contact IB IL AO 1/U/SF típusúak (13/a. árba) lettek.



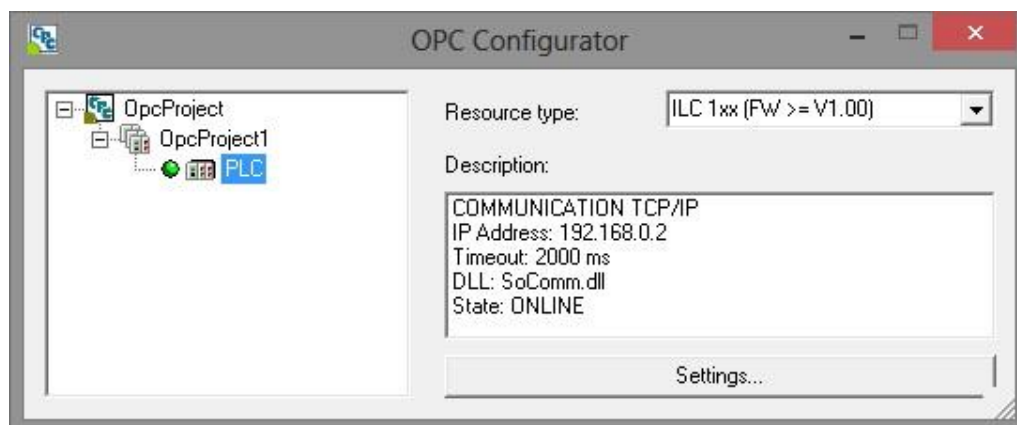
14. ábra: A PC WORX kezelőfelülete

A PLC fejlesztő környezete a PC WORX nevű szoftver (14. ábra), amiben elvégezhető a PLC konfigurációja. Megadható a különböző bővítő modulok típusa, beállíthatók a kommunikációs módok. A keret programon belül készíthető el a PLC programja az IEC 61131-3 nemzetközi szabványnak megfelelően. A programozáshoz 5 féle programnyelvet lehet használni, utasításlistás (IL), létra-diagrammos (LD), funkcióblokkos (FBD), sorrendi folyamatábrás (SFC) és strukturált szöveges (ST) programnyelvek. A fenti programnyelveket egyéni funkcióblokkok létrehozásával tetszőlegesen kombinálhatóak. A robotkar vezérléséhez készült programot az áttekinthetőség érdekében több általam létrehozott funkcióblokkra osztottam, így minden ízület külön részt kapott. A funkcióblokkokon belül FBD programnyelvet alkalmaztam. A ki-, bemeneti-, egyéni változók listáját egy táblázatos kezelőfelületen van lehetőség megadni (15. ábra) [15], [16].

|  | Name             | Type   | Usage      |
|--|------------------|--------|------------|
|  | Default          |        |            |
|  | RS_1_Z           | RS     | VAR        |
|  | RS_2_Z           | RS     | VAR        |
|  | F_TRIG_1_Z       | F_TRIG | VAR        |
|  | R_TRIG_1_Z       | R_TRIG | VAR        |
|  | CTUD_1_Z         | CTUD   | VAR        |
|  | SR_1_Z           | SR     | VAR        |
|  | bool_z_up_par    | BOOL   | VAR        |
|  | bool_z_down_par  | BOOL   | VAR        |
|  | bool_z_endp      | BOOL   | VAR_INPUT  |
|  | bool_man_mod     | BOOL   | VAR_INPUT  |
|  | bool_z_up_man    | BOOL   | VAR_INPUT  |
|  | bool_z_down_man  | BOOL   | VAR_INPUT  |
|  | bool_out_z_up    | BOOL   | VAR_OUTPUT |
|  | bool_out_z_down  | BOOL   | VAR_OUTPUT |
|  | bool_z_home      | BOOL   | VAR_INPUT  |
|  | bool_z_down_endp | BOOL   | VAR_OUTPUT |
|  | bool_z_imp       | BOOL   | VAR_INPUT  |
|  | int_z_wpos       | INT    | VAR_INPUT  |
|  | int_z_act_pos    | INT    | VAR_OUTPUT |
|  | bool_z_up_endp   | BOOL   | VAR        |
|  | int_z_max_pos    | INT    | VAR_INPUT  |
|  | bool_z_imp_trig  | BOOL   | VAR_OUTPUT |

15. ábra: A függőleges ízület (z) vezérlésének funkcióblokkjában használt változók listája

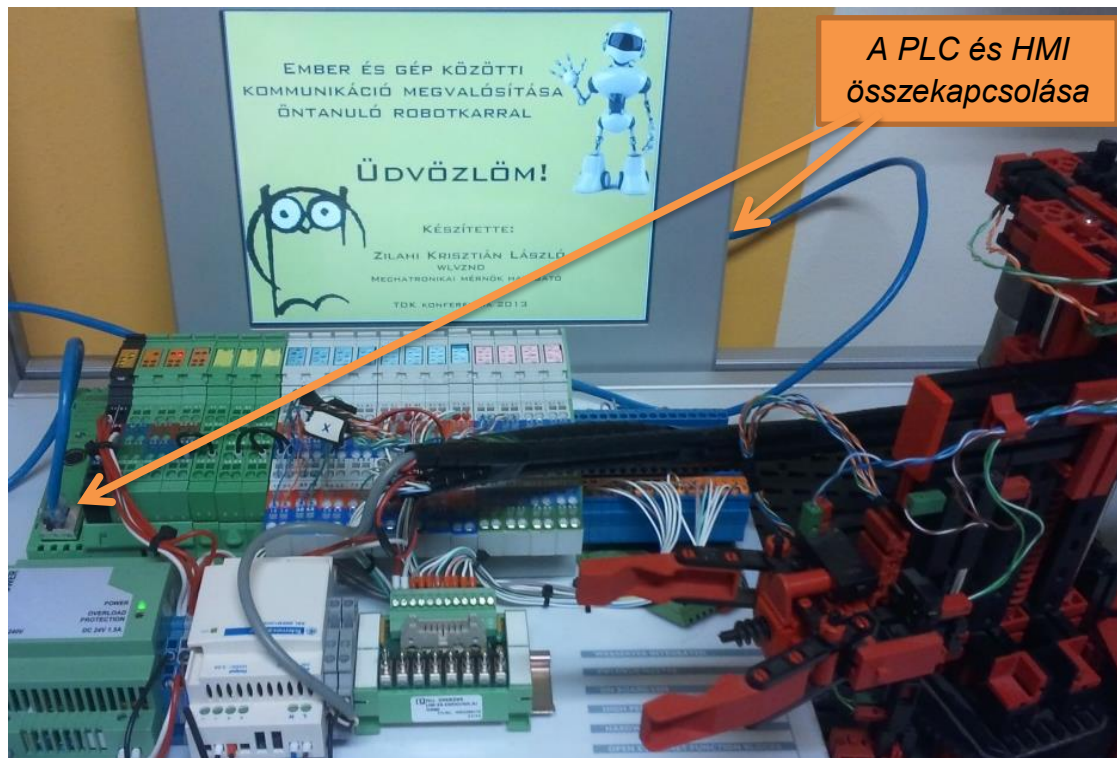
### 2.3. A PLC és HMI kommunikációja



16. ábra: OPC Configurator kezelőfelülete

A PLC és HMI közötti adat kommunikáció TCP/IP protokollon egy INTERBUS - OPC (Object Linking and Embedding for Process Control) szerveren keresztül történik. A Phoenix Contact OPC Configurator nevű alkalmazásával (16. ábra) hozható létre az

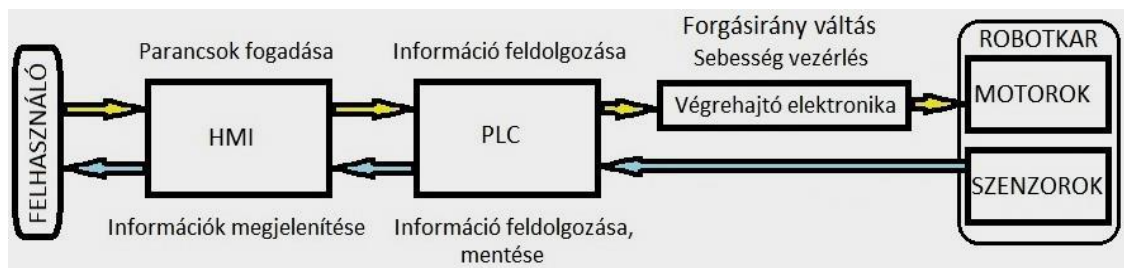
OPC szerver. Itt meg kellett adnom a PLC típusát, IP címét. A PLC tölti be az OPC szerver szerepét, így generál egy valós idejű adatbázist, melyhez a PC WORX-ben megadott változók igény szerint hozzáadhatóak. A valós idejű adatbázis jelen esetben azt jelenti, hogy a változók értékei kevesebb, mint 10 milliszekundumonként frissülnek. Ez a felhasznált eszközök feldolgozási sebességéhez mérten elegendő ahhoz, hogy valós idejűnek tekinthető legyen a rendszer. Az adatbázisban szereplő változók értékeinek beírását, valamint olvasását a PLC is végezheti vagy az OPC kliens funkcióját betöltő HMI [17].



17. ábra: A PLC és HMI összekapcsolása

Fizikailag a kijelző egység és a programozható logikai vezérlő összeköttetését egy RJ45-ös csatlakozókkal ellátott UTP (Unshielded Twisted Pair) kábel látja el (17. ábra).

### 3. A dolgozat során megvalósított feladatok



18. ábra: A rendszer kommunikációjának folyamatábrája

A dolgozatomban a felhasználó és a rendszer közötti kapcsolat (18. ábra) megvalósításához, különböző funkciókat hoztam létre. A HMI felületéről a kezelő információkat kap a rendszerről, valamint utasításokat is adhat. A HMI által továbbított információkat és parancsokat, valamint a robotkaron lévő szenzoroktól származó adatokat is a PLC-re általam megírt program dolgozza fel. A feldolgozott információknak megfelelően a PLC a végrehajtó elektronika segítségével hozza mozgásba a robotkart.

#### 3.1. A HMI menü struktúrájának felépítése



19. ábra: A HMI kezelő felületének menüpontjai

A kezelő felületet menüpontokra (19. ábra) tagoltam. Figyelembe vettem a különböző pontok jelentőségét a felhasználó szempontjából.

- Az első menüpont az „Alaphelyzet felvétel”, melyben a robotkar beáll az alaphelyzetébe, hogy az útmérő rendszerét bekalibrálja.
- A „Kézi mozgatás” pont alatt lehet a robotot manuálisan irányítani, megismerkedni a mozgatásért felelős felülettel.
- A „Kézi programozás” funkcióban a felületen manuálisan megírt program vihető be a PLC-nek.
- A „Betanító üzemmód” esetén a már ismert „Kézi mozgatás” felületére épülő struktúrával mozgatható a robot, kiegészítve egy betanító funkcióval, mellyel tetszőleges mozdulatsorra (ízületenként 1000 pozícióra) tanítható meg.
- A „Súgó” információt nyújt a felsorolt menüpontokban végezhető műveletekről.

A „Kézi programozás”, valamint a „Betanító Üzemmód” pontok jelszavas védelemmel lettek ellátva, hogy azokhoz csak az arra jogosult felhasználónak legyen hozzáférése.

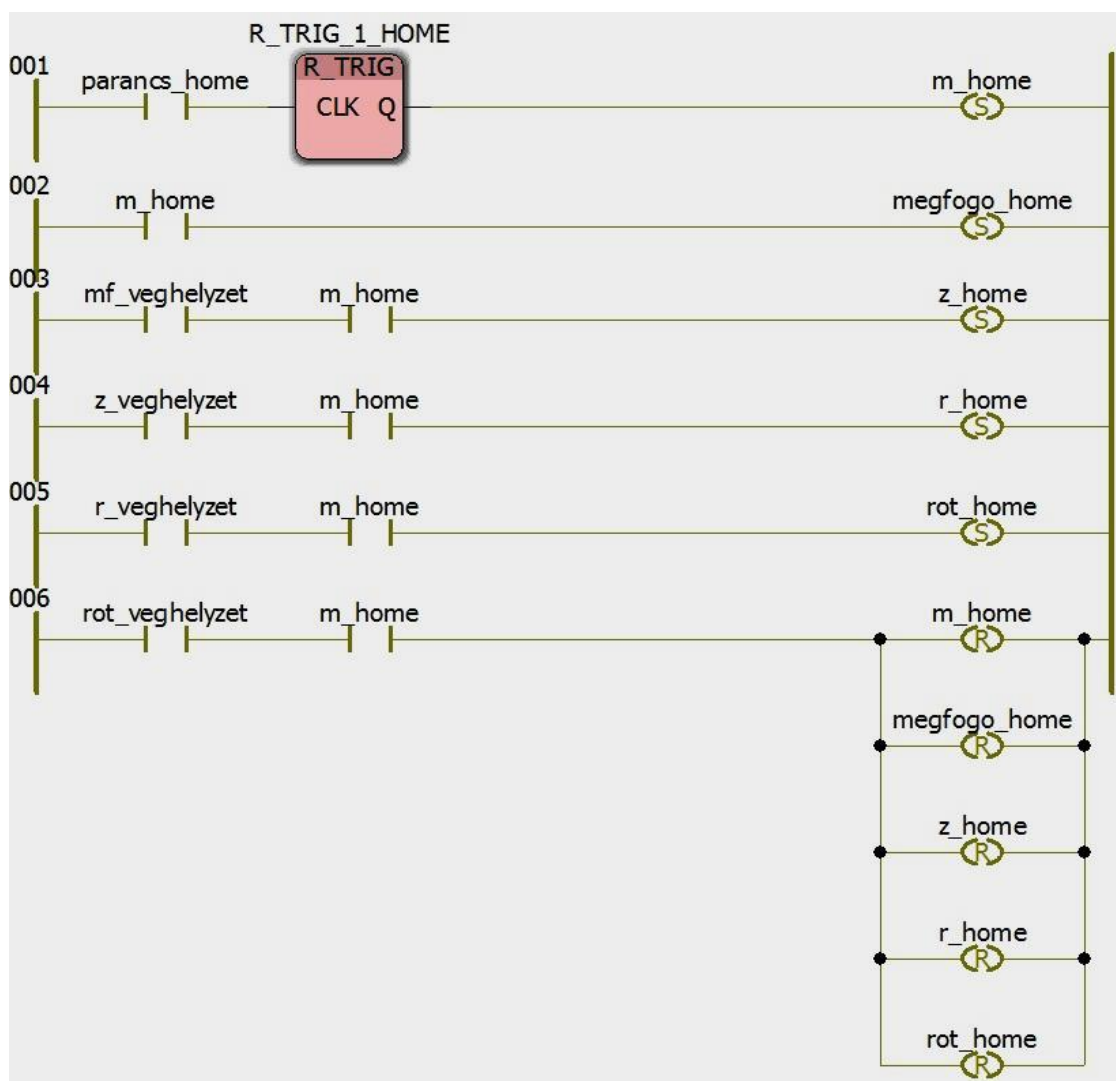
### 3.2. Az alaphelyzet felvétel



20. ábra: Az alaphelyzet felvételnek képernyője

Az alaphelyzet felvétel menü (20. ábra) alatt az ízületek gépi nullpontba történő mozgatása valósítható meg. Az egyes ízületek nullpontjában elhelyezett érzékelőkhöz kalibrálhatók az inkrementális jeladók pozíció meghatározásához szükséges számlálói. A „Gépi nullpont felvétele” gombot megnyomva a PLC-re megírt program segítségével a robotkar meghatározott sorrendbe, azaz a megfogó, a függőleges (z), a vízszintes (r), majd a forgó ízület ( $\theta$ ) áll alaphelyzetbe. Az alaphelyzet elérését ízületenként a kijelzőn egy-egy zöld LED felkapcsolódása jelzi.

A kezelés megkönnyítése céljából egy beépített sűgő gombot helyeztem el a képernyőn, melyben leírtam az éppen aktuális felület feladatát és annak használatát. A „Vissza” gombbal pedig visszatérhetünk az előző képernyőre.

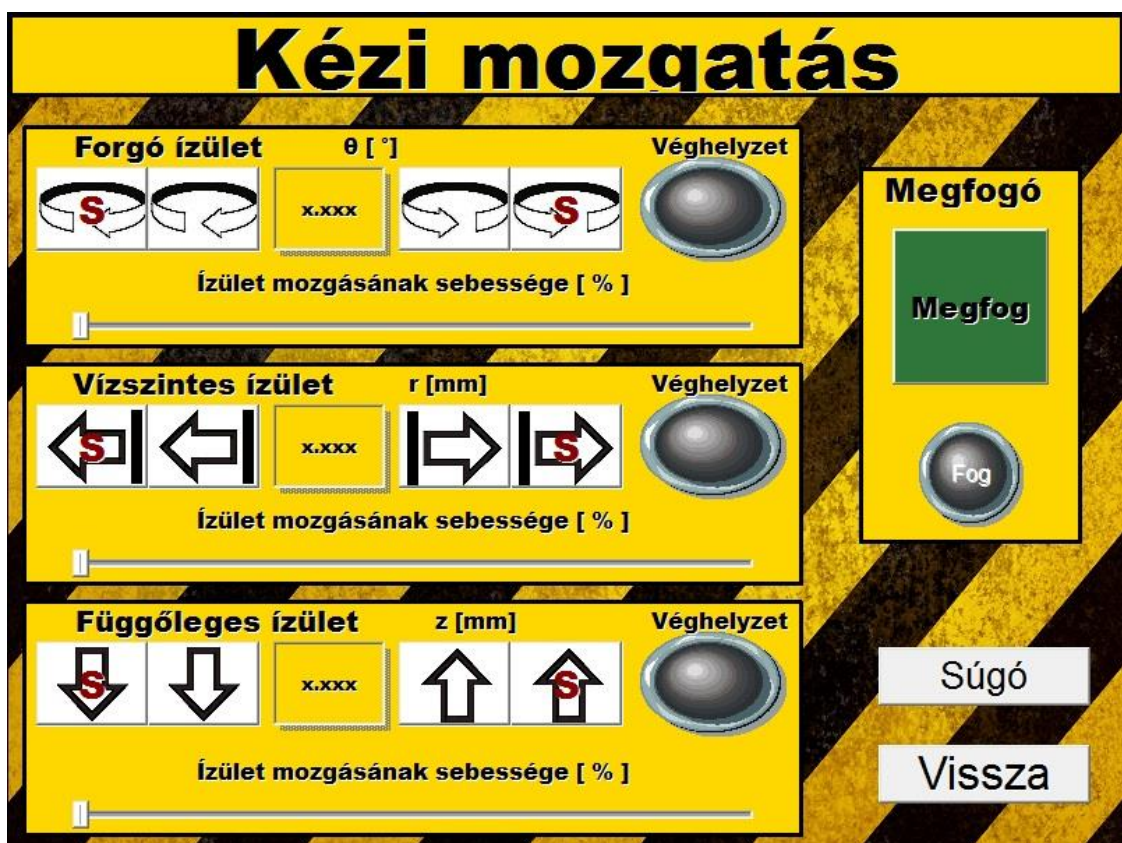


21. ábra: Az alaphelyzet felvétel PLC programja

A PLC program (21. ábra) külön funkcióblokkban került elhelyezésre, melyet Létra-diagrammos (LD) program nyelven készítettem el. A program a sorrendi vezérlés elvén

alapul, az előző folyamat elvégzése a következő folyamat indítási feltétele, így a csuklók a már meghatározott sorrendbe felveszik az alaphelyzetüket. A HMI-n megnyomott gomb (*parancs\_home*) működésbe hozza a megfogó motorját (*megfogo\_home*), mind addig, amíg az effektor véghelyzet érzékelője jelet nem ad (*mf\_veghelyzet*). Ezt követően a függőleges ízület motorja is a véghelyzet irányába mozog (*z\_home*). A z-ízület véghelyzetében (*z\_veghelyzet*) a vízszintes ízület is az alaphelyzete felé indul (*r\_home*). Az r-ízület alaphelyzetbe állása a forgó ízület is megkezdzi az alaphelyzetének felvételét (*rot\_home*). A folyamat befejeztével a használt változók értékét visszaállítottam, hogy akár többször is igénybe vehető legyen a kalibrálási folyamat. A programban elhelyeztem egy biztonsági változót (*m\_home*), mely megakadályozza a folyamat futását, ha az nem aktív.

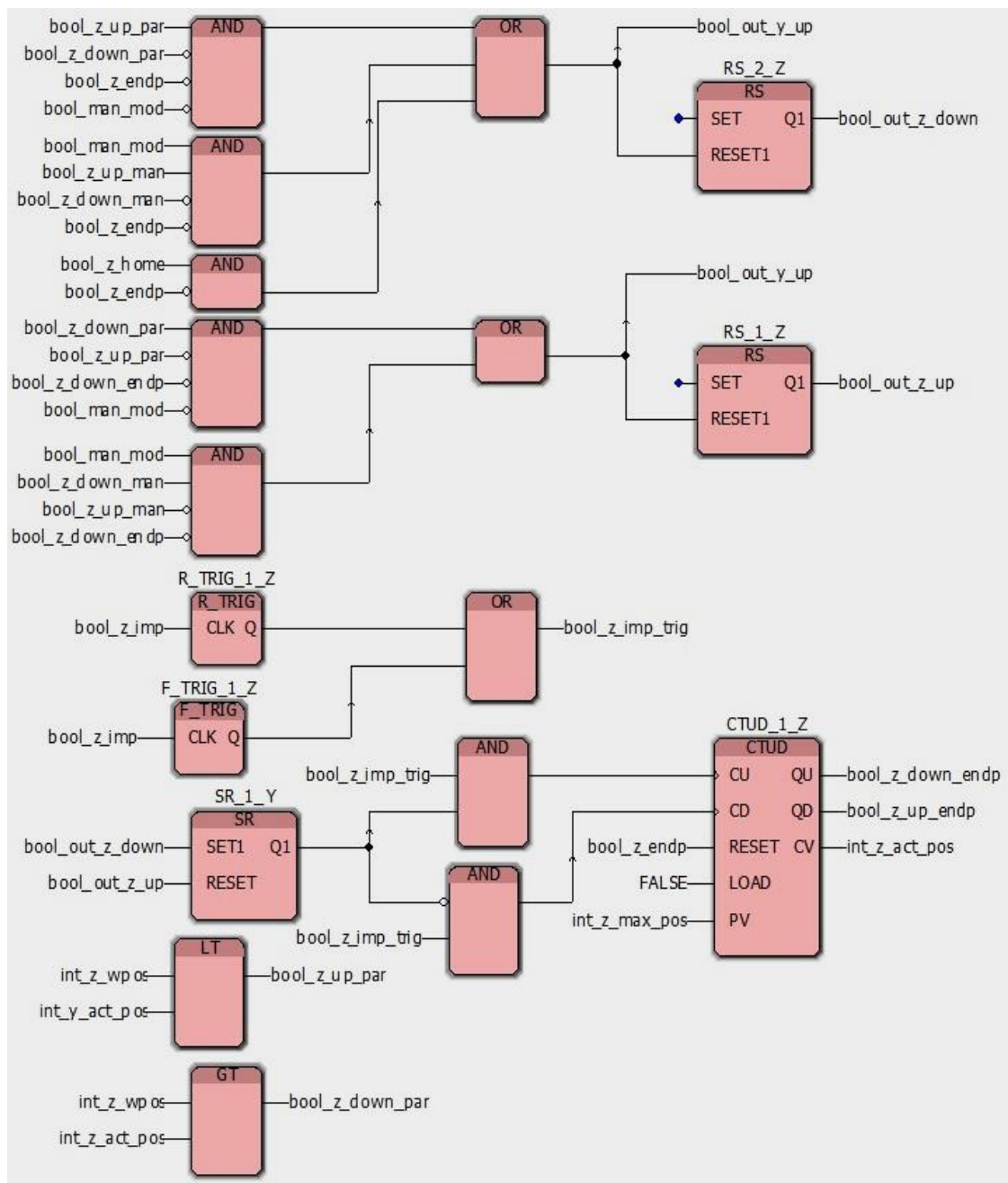
### 3.3. A kézi mozgatás funkció



22. ábra: A kézi mozgatás funkció kezelőfelülete

A kézi mozgatás menüpont (22. ábra) alatt egy olyan kezelő felület található, mellyel az ízületeket különböző parancsokkal mozgató. A gombok az ízületek mozgásának irányaiival megegyező piktogramokkal vannak ellátva, valamint minden irányhoz

elhelyeztem egy nyomógombot és egy kapcsolót is. A kapcsolókra nem működtetett állapotukban az „S” (SET), kapcsolt állapotukban „R” (RESET) jelölés utal. A kezelő tájékoztatására az ízületek véghelyezeteit LED-ek jelenítik meg. A mozgató motorok sebessége külön-külön egy csúszka segítségével állítható be. A megfogó egy kapcsolóval működtethető. Két állapota van: a megfog és az elenged. Állapotáról egy LED biztosít információt. Az ízület aktuális pozíciója a mozgató gombok között, a feltüntetett mértékegységnek megfelelően jelenik meg. Az ízület pozíciójáról a már említett büttyös mechanizmus szolgáltat információt, itt a mikrokapcsoló jele érkezik be egy meghatározott digitális bementen.



23. ábra: A függőleges (z) ízület vezérlése

A pozicionálás pontosságát az impulzusok le- és felfutó éleinek számolásával lehet növelni (23. ábra), tehát az ízületeknél azok motorjainak egy körülforgását 8 részre tudjuk osztani. Ebből következően a forgó ízületnél  $6,2^\circ/8$ , azaz hozzávetőlegesen  $0,775^\circ$ -os, a haladó mozgást végző ízületeknél pedig  $5\text{ mm}/8$ , avagy körülbelül  $0,625\text{ mm}$  mérési és pozicionálási pontosságot lehet elérni.

A programban a különböző funkcióknak megfelelően lettek meghatározva a változók közötti logikai kapcsolatok, így lehetőség van meghatározott irányba vagy egy megadott pozícióba küldeni az ízületet. A folyamat leírását a függőleges ízületet vezérlő programmal (23. ábra) szemléltetem. Egy számláló (*CTUD\_1\_Z*) segítségével a mozgatás irányának függvényében a beérkező impulzusokkal (*bool\_z\_imp\_trig*) növeljem vagy csökkentem annak értékét. Így megadható az ízületek véghelyzet érzékelő nélküli véghelyzete (*bool\_z\_down\_endp*). A motorok működtetésére és forgásirányváltására szolgáló relék különböző parancsokkal, bizonyos biztonsági feltételek mellett működtethető. Lehetőség van a kézi mozgató gombokkal összerendelt változókkal (*bool\_z\_up\_man*, *bool\_z\_down\_man*), valamint a programon belüli logikai feltételt vizsgáló függvényekkel (*LT*, *GT*) jelet adni a reléknek. A programon belüli feltételeket vizsgálva, azaz a számlálók aktuális értékét (*CTUD\_1\_z.CV*) összehasonlítva a többi funkció blokk által megadott kívánt pozíció (*int\_z\_wpos*) értékével, automatikusan eldönti az általam leprogramozott rész, hogy melyik irányba kell az ízületet mozgatni (*bool\_z\_up\_par*, *bool\_z\_down*). A feltételek szerint ki is adja a megfelelő relé számára a vezérlő (*bool\_out\_z\_up*, *bool\_out\_z\_down*) jelet. A kívánt értéket elérve megszűnik a vezérlő jel. A biztonsági feltételek magukban foglalják a véghelyzet elérésénél, a kézi mozgató gombok egyszerre való lenyomásánál a vezérlő jel elvételét, valamint a kézi gombok használatának lehetőségét a kívánt pozícióra való mozgásnál.

### 3.4. A betanító üzemmód

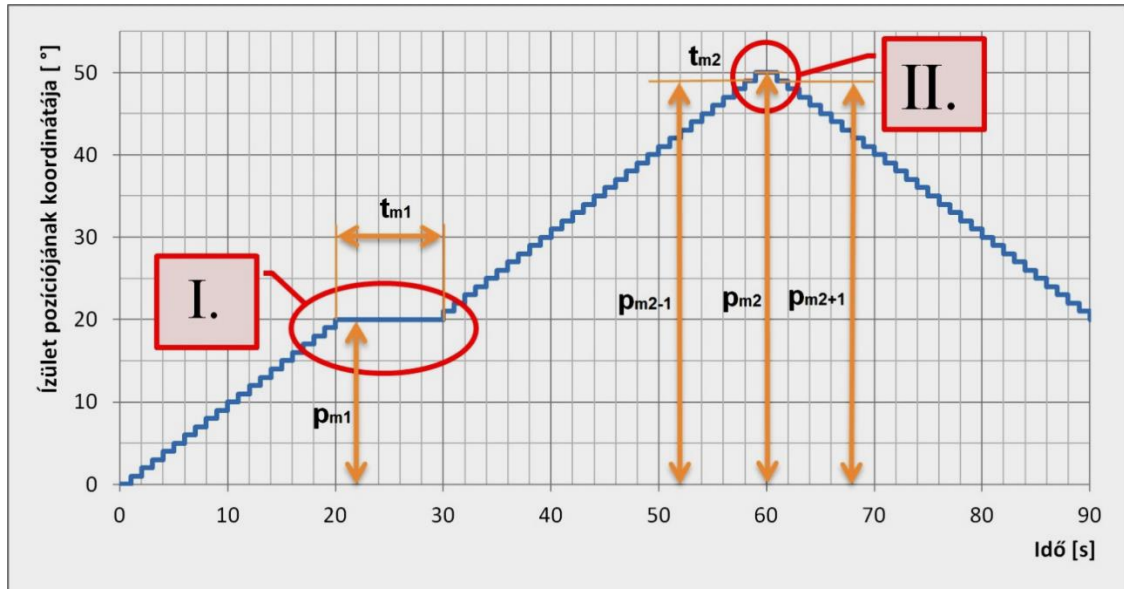


24. ábra: A betenítő üzemmód felülete

Az alábbi funkció egy jelszóval védett rész, mivel itt már kiegészítő funkciók is elérhetőek a kézi mozgatás mellett.

A betanító üzemmód kezelőfelülete felépítését tekintve megegyezik a kézi mozgatás funkcióval, kiegészítve egy betanítási lehetőséggel. a „Betanítás” gombot megnyomva elkezdődik a mozgássor megtanulása (ilyenkor a gombon a „Befejezés” felirat jelenik meg és pirosra színeződik). A tanítási funkció alatt ízületenként 1000 pozíciót tudunk megtaníttatni a rendszerrel. A tanulási folyamat teljesen automata, tehát a kezelőnek csak a mozgatással, kell foglalkoznia. A funkció befejezéséhez a „Befejezés” gombot kell megnyomni. Ezt követően a megtanult mozdulat sort visszajátszhatjuk. Beállítható az, hogy hányszor játszódjon vissza a mozdulatsor. A „Program indítása” gomb indítja a mozdulatsor programját (ez esetben a gomb színe piros és felirata „Program leállítása” lesz). Az aktuális lépés száma és a lefutott programok száma megjelenik a kijelzőn. A lejátszást a „Program leállítást” gomb megnyomásával állítható meg. Újabb betanításnál a régi mozdulatsor törlődik és az új mozdulatsor jegyeztethető meg, mindez automatikusan történik a „Betanítás gomb újabb lenyomásával.

Kielemezve az ízületek mozgásának időbeni változásának lehetőségeit arra a következtetésre jutottam, hogy elegendő két feltételt vizsgálni a mozdulatsor megjegyzéséhez. A számláló mechanizmus miatt, az ízület pozíciójának időbeni változásának értéke nem folytonos (lépcsős függvény).



25. ábra: A pozíció rögzítésének lehetséges feltételei

Az első feltétel, amikor az ízület egy pozíción áll adott ideig (25. ábra I. rész). Ezt úgy vizsgáltam, hogyha az ízület koordináta értéke nem változik  $t = 0,5$  s-on belül, akkor a pozíciót ( $p_{m1}$ ) és a pozíción töltött időt ( $t_{m1}$ ) kell megjegyezni.

A második feltétel szerint lokális szélső értékeket kell keresni (25. ábra II. rész), azaz, ha megváltozik az ízület mozgásának iránya, akkor az ízület koordináta – idő grafikonon lokális minimumok vagy maximumok keletkeznek. Itt elegendő csak a szélsőérték koordinátáját ( $p_{m2}$ ) megjegyezni,  $t_{m2} = 0$  s-ot rendelve hozzá. Ez a pozíció úgy határozható meg, hogy figyelni kell az utolsó két ( $p_{m2-1}$ ,  $p_{m2}$ ) és az aktuális pozíciót ( $p_{m2+1}$ ). Ha teljesül a  $p_{m2-1} = p_{m2+1}$ , akkor a  $p_{m2}$  koordinátát kell megjegyeztetni.

```

1  IF imp_rot THEN
2    rot_M3:=rot_M2;
3    rot_M2:=rot_M1;
4    rot_M1:=rot_act_pos;
5  END_IF;
6  IF rot_M1 = rot_M3 THEN
7    rot_mem:=TRUE;
8  ELSE rot_mem:= FALSE;
9  END_IF;
10 rot_rtrig(CLK:=rot_mem);
11 rot_rtrig2(CLK:=NOT(TON_rot1.Q));
12 TON_rot1(IN:= (tanul_rot AND NOT(imp_rot)), PT:= t#0.5s);
13 TON_rot2(IN:= (tanul_rot AND TON_rot1.Q), PT:= t#1000s);
14 IF TON_rot1.Q THEN
15   rot_ido := TON_rot2.ET;
16 END_IF;

```

26. ábra: A feltételek figyelése a tanuló programban

A korábban leírt feltételeket figyelembe véve elkészítettem a tanuló programot. Először elkészítettem az utolsó koordináták folyamatos figyelését (26. ábra). Itt a számlálандó impulzusok jelére (*imp\_rot*), láncszerűen tovább adódnak az értékek az utolsó pozíciókat tartalmazó memóriák között (*rot\_M1*, *rot\_M2*, *rot\_M3*). A *rot\_M1 = rot\_M3* feltétel teljesülésével egy eseményjelző bit (*rot\_mem*) értéke „IGAZ” értékre vált.

A másik feltétel teljesülését egy időzítővel (*TON\_rot1*) figyeltem. A beérkező impulzusok jele, mindig újraindítja az időzítőt, ha az időzítő értéke eléri  $t = 0,5$  s-ot, akkor az elindít egy másik időzítőt (*TON\_rot2*), mely elkezd mérni az aktuális pozíción (*rot\_M1*) töltött időt (*rot\_ido* + 0,5 s).

```

17 rot_CTU1(CU:=(tanul_rot AND (rot_rtrig.Q OR rot_rtrig2.Q)),
18 RESET:= NOT(tanul_rot), PV:= 1000);
19 i:=rot_CTU1.CV;
20 IF rot_rtrig.Q THEN
21   memory_pos_rot_1[i] := rot_M2;
22   memory_time_rot_1[i] := t#0s;
23 END_IF;
24 IF rot_rtrig2.Q THEN
25   memory_pos_rot_1[i] := rot_M1;
26   memory_time_rot_1[i] := (rot_ido + t#0.5s);
27 END_IF;

```

27. ábra: A koordináták és a hozzátartozó idők mentése

A feltételek teljesülését egy számlálóval (*rot\_CTU1*) számolom. A tanulási folyamat (27. ábra) alatt (*tanul\_rot*), az események bekövetkeztével növelem a számláló értékét (*rot\_CTU1.CV*). A koordináták és idők mentésére ízületenként egy-egy tömböt definiáltam (pl.: *memory\_pos\_rot*, *memory\_time\_rot*). A tömbökbe való mentés során a számláló értékének megfelelően a tömbök sorindexe [*i*] változik, így a megjegyzendő értékek a tömbök aktuális sorindexű sorába menthetők.

```

28 rot_CTU2(CU:=rot_rtrig5.Q, RESET:= jmp_first_lp_rot, PV:= megtanult_lepes_szam);
29 TON_rot3(IN:=(int_out_rot_wpos = rot_act_pos), PT:= TON_rot5_ido);
30 y:=rot_CTU2.CV;
31 rot_rtrig4(CLK:=(int_out_rot_wpos = rot_act_pos));
32 rot_rtrig5(CLK:=TON_rot3.Q);
33 IF lejatsz_rot AND rot_rtrig5.Q THEN
34 int_out_rot_wpos:=memory_pos_rot_1[y];
35 END_IF;
36 IF rot_rtrig4.Q THEN
37 TON_rot5_ido:=memory_time_rot_1[y];
38 END_IF;

```

28. ábra: A megtanult program visszajátszása

A tömbökbe mentett információk visszajátszásához (28. ábra) szükség van egy számlálóra (*rot\_CTU2*) meg a befejezett mozdulatokat számolja. A számláló értéke így mindig az adatokkal feltöltött tömb soron következő sorindexének felel meg. Így kiolvasható a szükséges koordináta és idő érték. A koordináták továbbítódnak (*int\_out\_rot\_wpos*) az ízületet vezérlő funkció blokkba, ahol az értéknek megfelelő pozíciót felveszi az adott ízület, majd a hozzátartozó idő érték betöltődik egy időzítőbe (*TON\_rot3*). A megfelelő idő elteltével a számláló értékét növelem, ezzel előről kezdődik a folyamat, mindaddig míg el nem érem a megtanult pozíciók számát (*megtanult\_lepes\_szam*). Ilyenkor, ha csak egy lett beállítva a program futások számának, akkor megáll a folyamat. Viszont, ha többszöri visszajátszás lett megadva, akkor, ha minden ízület befejezte a mozgás sort, újra indul a folyamat (*jmp\_first\_lp\_rot*), mind ezt annyiszor, ahányszor az be lett állítva.

```

39 rot_rtrig3(CLK:= tanul_rot);
40 IF rot_rtrig3.Q THEN
41 FOR x := 0 TO 1000 BY 1 DO
42 memory_pos_rot_1[x] := 0;
43 memory_time_rot_1[x] := t#0s;
44 END_FOR;
45 END_IF;
46 RETURN;

```

29. ábra: A megtanult értékek törlése

Minden újabb betanításnál törölni kell a memóriák értékét (29. ábra). Ez automatikusan történik a „Betanítás” gomb megnyomásakor (*tanul\_rot*). Egy FOR ciklus segítségével végig nullázódnak a tömbök (*memory\_pos\_rot*, *memory\_time\_rot*) értékei.

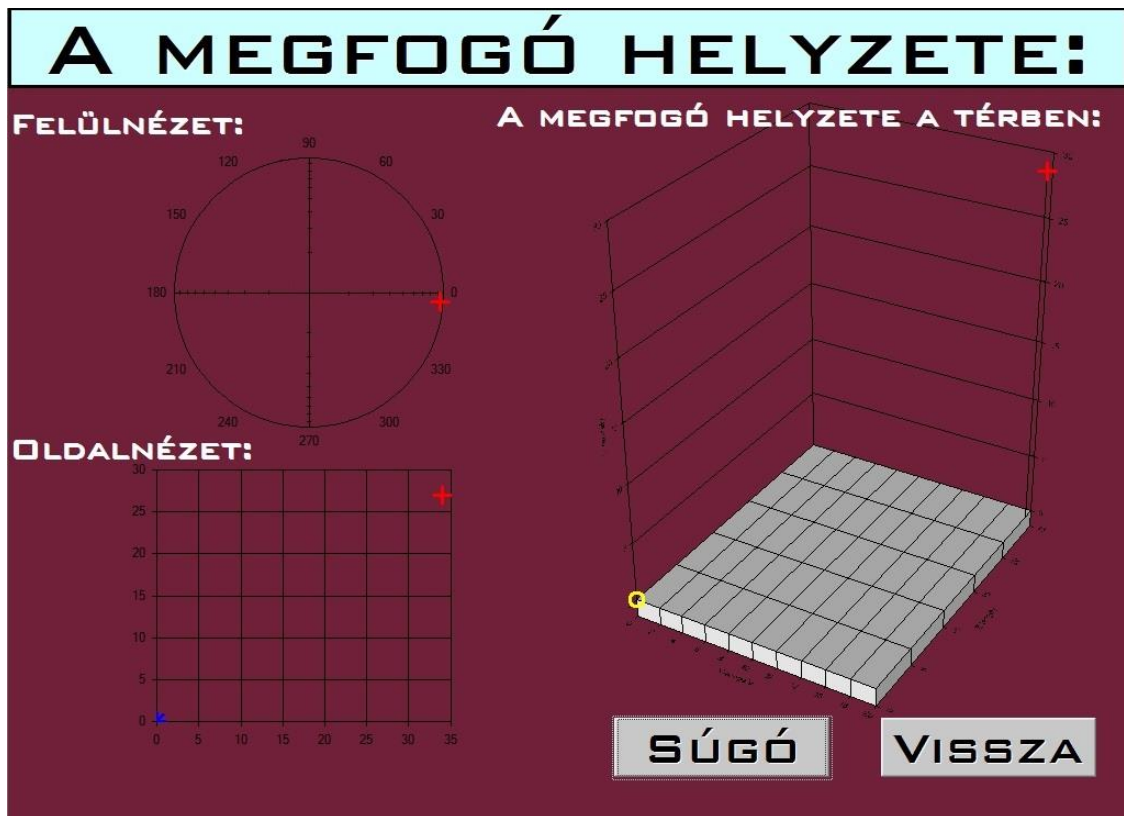
### 3.5. A kézi programozás funkció

30. ábra: A kézi programozás képernyője

A „Kézi programozás” menüpont (30. ábra) alatt manuális program bevitelre van lehetőség. A bevitel a „Beírás” nyomógomb segítségével kezdhető el, ilyenkor a „Lépés” címszó mellett lévő sor a szerkeszthető sor, itt ízületenként megadható a kívánt pozíció értéke és a késleltetési idő. Láthatóak a soron következő („Lépés+1”), valamint az előzőleg bevitt („Lépés-1”) sorok értékeit. A „Lépésszám” oszlopában megjelenik a lépések sorszáma, továbbá a sorszámok feletti gombokkal lehet a sorok között lépkedni. A programozás során meg kell adni, hogy a megírt program hány lépésből áll (maximum 100 lépés), valamint, hogy azt hányszor játssza vissza a rendszer. A „Program indítása” gombbal indítható, a „Program leállítása” gombbal pedig leállítható a folyamat. Megjelenik az aktuális lépés szám és az, hogy hányszor futott már le a megírt a program.

A kézi programozásos rész PLC-re megírt funkció blokkja hasonló struktúrával rendelkezik, mint a betanító üzemmódé, azzal a különbséggel, hogy itt kézzel kell bevinni az adatokat és úgy tárolódnak el egy-egy tömbbe.

### 3.6. A „Megfogó helyzete” opció bemutatása



31. ábra: A megfogó helyzetének megjelenítése

A megfogó helyzete funkció (31. ábra) a „Kézi programozás” és a „Betanító üzemmód” menüpontokból érhető el. Itt a robotkar megfogójának mozgása követhető grafikonokon. A megfogó helyzetét egy piros kereszt jelöli a grafikonokon. Látható a megfogó felülnézete, oldalnézete, valamint 3 dimenziós térbeli elhelyezkedése. A felülnézeti ábrához szükséges a forgó ízület elfordulásának szöge ( $\theta_0$ ) és a vízszintes ízület ( $r$ ) helyzete. Az oldalnézethez a függőleges ízület magasságát, azaz a  $z$  értékét, és a már meghatározott homogén transzformációs mátrixból kifejezett  $x$  koordináta értékét kell tudnom. A térbeli grafikon mindig a megfogó aktuális tér negyedét mutatja világi koordinátarendszere szerint. Szükség van még az  $y$  koordináta értékére, melyet szintén a meghatározott homogén transzformációs mátrixból határoztam meg. Így mind három koordináta ( $x, y, z$ ) rendelkezésre áll a megjelenítéshez.

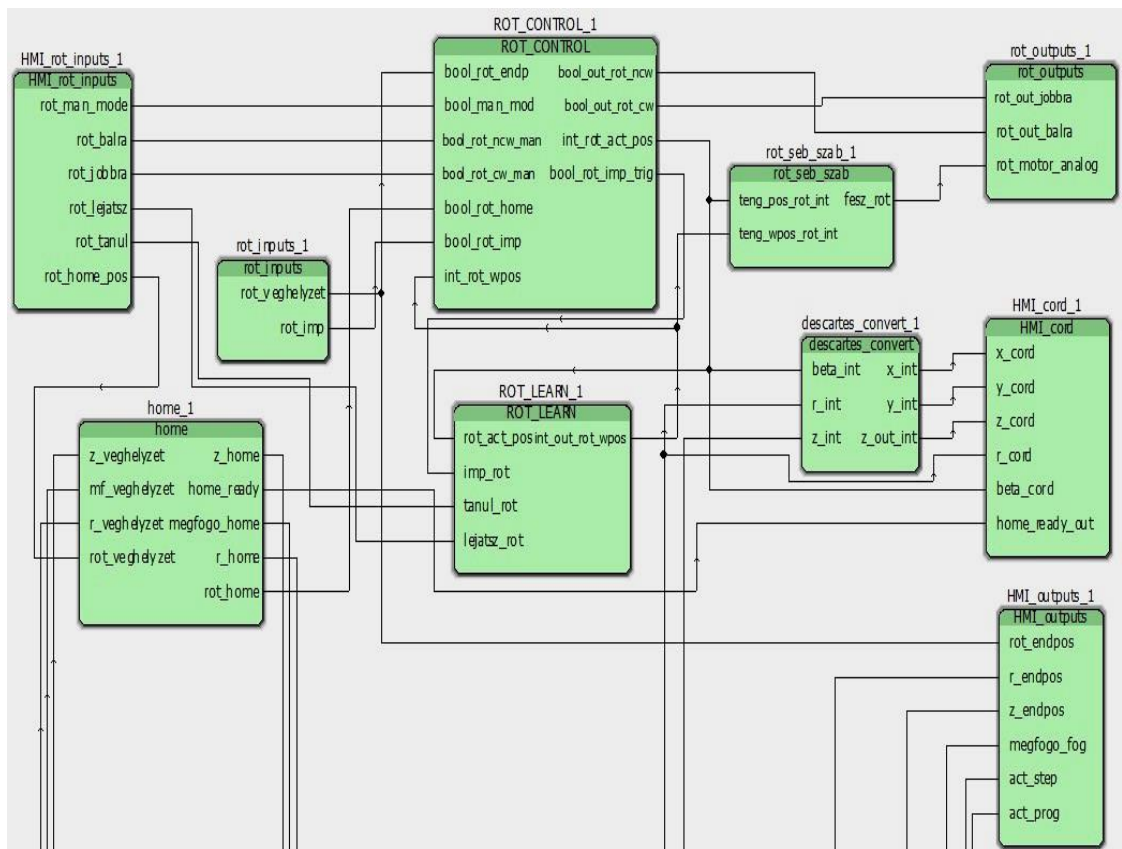
### 3.7. A motorok sebességvezérlése a pozíció függvényében

```
1  teng_pos_rot:= INT_TO_WORD(teng_pos_rot_int);
2  teng_wpos_rot:= INT_TO_WORD(teng_wpos_rot_int);
3
4  IF teng_pos_rot > teng_wpos_rot THEN
5    rot_kulonbseg:= teng_pos_rot - teng_wpos_rot;
6  END_IF;
7
8  IF teng_pos_rot < teng_wpos_rot THEN
9    rot_kulonbseg:= teng_wpos_rot - teng_pos_rot;
10 END_IF;
11
12 IF (rot_kulonbseg <= 15) THEN
13   fesz_rot := 32760 + rot_kulonbseg*(const_1/30);
14 ELSE
15   fesz_rot:= const_1;
16 END_IF;
```

32. ábra: A forgó ízület motorjának sebességvezérlése

A motorok sebességvezérléséhez az 1.3. fejezetben tárgyalt erősítőre és 3 analóg kimenetre volt szükségem. A vezérlés programját a forgó ( $\theta$ ) ízületen mutatom be (32. ábra). A programban figyeltem a felvenni kívánt pozíció (*teng\_wpos\_rot\_int*) és az aktuális pozíció (*teng\_pos\_rot\_int*) különbségének abszolút értékét (*rot\_kulonbseg*). Ha a különbség értéke a meghatározott érték alá csökken, akkor a motorra kapcsolt feszültség értékét csökkentem. A csökkentés mértéke egyenesen arányos a különbséggel. A programrészletben látható összefüggés alapján csökkentem a feszültséget, mely 5 V (az analóg kimenet decimális értéke a programban ilyenkor: 32760) alá nem eshet, mivel annál kisebb feszültség már nem hozza mozgásba a motort. A mozgás során beállított maximális feszültség érték (*const\_1*) (maximum 10 V) hányadának 1/30 és a pozíciók különbségének szorzata adódik az 5 V-nak megfelelő konstanshoz. Az eredmény adja az analóg modul kimeneti feszültségét (*fesz\_rot*).

### 3.8. A PLC főprogramjának strukturális szerkezete



33. ábra: A főprogram strukturális szerkezete (részlet)

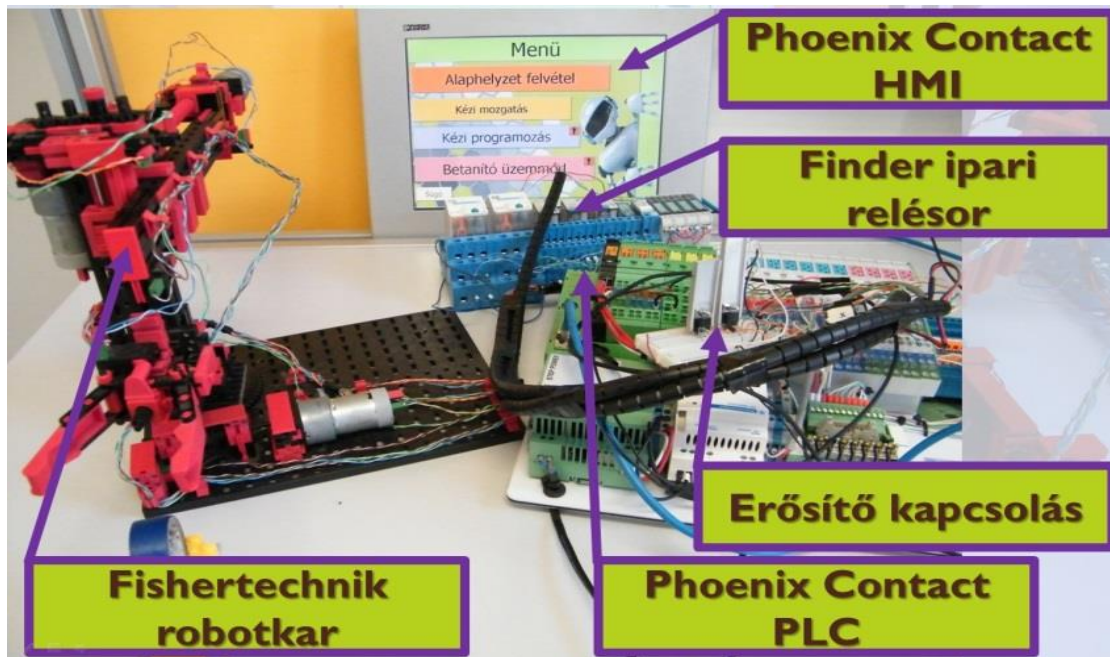
A főprogram (33. ábra) az általam megírt, a dolgozat során bemutatott funkció blokkokat tartalmazza. A blokkok megfelelő be- és kimeneteit huzalozással kötöttem össze, így biztosítva program részek közötti adatátvitelt. A programrészlet baloldalán található a fizikai bemenetek (*rot\_inputs*) és a HMI által adott bemeneti információk (*HMI\_rot\_inputs*), jobb oldalán pedig a fizikai kimenetek (*rot\_outputs*) és a HMI felé továbbított adatok (*HMI\_outputs*). A fizikai be- és kimenetekhez konkrét címeket kellett rendelnem. Például a forgó ízület véghelyzet érzékelője (*rot\_veghelyzet*) a %IX1.0-es, az impulzus jeladója (*rot\_imp*) a %IX1.1-es bemeneti címen, míg az óramutató járásával megegyező forgási irány (*rot\_out\_jobbra*) a %QX6.0-ás kimeneten kapcsolható. A részlet közepén pedig a különböző folyamatokért felelős blokkot találhatók, például a robotkar forgó ízületének vezérlését végző (*ROT\_CONTROL*) és a tanulását végző (*ROT\_LEARN*) program rész.

## 4. Fejlesztési lehetőségek

A dolgozatom során felmerültek lehetőségek, melyeket a rendelkezésre álló időben nem volt lehetőségem megvalósítani, de az alábbi fejezetben szeretném röviden összefoglalni azokat.

- A motor forgásirányváltásáért felelős relé sor helyett tranzisztoros H-hídkapcsolást alkalmazni, így sokkal pontosabban vezérelhető lenne a motor forgásirányváltása.
- Az ízületek pozíciójának meghatározásához szükséges bütykös mechanizmus lecserélése egy finomabb osztású mechanizmusra, mely a motor egy körülforgása alatt több impulzust szolgáltat a vezérlőnek, így növelve a pozicionálás pontosságát.
- Pontosabb útmérő rendszer esetén kialakítható pozíció szabályzás, valamint ez esetben a pályavezérlés megvalósítható. Így előre programozott pályagörbék is felhasználhatóak lennének a programozás során. Továbbá kézi üzemmód esetén világi koordinátarendszer szerint is mozgatható lenne a robotkar.
- A betanítási funkcióban lehetőség kialakítása, több külön mozdulatsor megjegyzésére és azok egymással való kombinálására tetszőleges sorrendben.
- A mozdulatsor megtanulásánál az ízület mozgás sebességének megjegyzése és annak megfelelő visszajátzása.
- A kézi programozás menüpontban a programozható tárak kibővítése, hogy akár több manuálisan megírt program legyen tárolható és visszajátzható.
- A vezérlőegységek kompatibilitássá tétele más konstrukciójú robot karok működtetéséhez (például beállítható ízület véghelyzetek).
- A robotkar munkatartományának beállítási lehetősége, hogy az esetleges állandó akadályokat kikerülje elkerülve az ütközés lehetőségét.
- A HMI mellett Android operációs rendszerű mobilkészülékre és/vagy táblagépre fejlesztett vezérlő felület, mely a készülékben lévő szenzorok segítségével is irányítható lenne.

## 5. Összegzés



34. ábra: A dolgozat során létrehozott rendszer

A dolgozat során megismerkedtem a Fishertechnik robotkar modell felépítésével és alkotó elemeivel. Elemeztem a modell szerkezetét és hajtás láncát, majd azokat a megfelelő módon módosítottam. Építettem egy erősítő kapcsolást a motorok sebességvezérléséhez. Megterveztem a robotkar hajtásának kapcsolási rajzát, majd azt meg is valósítottam.

A vezérléshez ipari eszközöket (HMI, PLC) használtam. Megismerkedtem a HMI és PLC programozó környezetével, elsajátítottam azok programozását. Létrehoztam a PLC és HMI közötti kommunikációt szolgáló valósídejű adatbázist.

Elkészítettem egy megjelenítő alkalmazást, melyen keresztül létre jött a felhasználó és a rendszer közötti kommunikáció. Az alkalmazás segítségével a kezelő igénybe tudja venni az általam megírt különböző funkciókat, mint például a robotkar kézi vezérlése, betanítása, programbevitel.

Az általam elkészített PLC program struktúrát funkció blokkokra tagoltam, így minden blokk egy-egy funkciót lát el, ezzel megkönnyítve a program átláthatóságát. A PLC ennek megfelelően dolgozza fel a beérkező adatokat és adja ki az utasításokat a robotkar felé.

A dolgozat célja az iparban használt eszközökkel megvalósított ember-gép kommunikáció létrehozása volt, melyet egy betanítható rendszerrel egészítettem ki. A betaníthatóság fontos szerepet tölt be az ipari rendszereknél. A sorrendi és idő alapú vezérlések esetén megkönnyíti a folyamatok, mozgások leprogramozását. Elegendő a gépkezelőnek egyszer kézzel végig irányítani, mozgatni a rendszert, majd az automatikusan ismétli azt [18].

Az általam megvalósított rendszer vezérlési része alkalmas a hasonló feladatok ellátására, így kiválóan felhasználható ismétlődő mozgássorok megtanítására és azok visszajátszására.

## 6. Felhasznált irodalom

- [1] Spong Mark W., Hutchinson Seth, and Vidyasagar M. Robot Modeling and Control. New York: John Wiley and Sons, 2005. 496 pp. ISBN: 9780471649908
- [2] [http://www.studica.com/images/product/FischertechnikSpareParts/31390\\_77\\_200x.gif](http://www.studica.com/images/product/FischertechnikSpareParts/31390_77_200x.gif) Letöltve: 2013. november 03.
- [3] [http://www.studica.com/images/product/FischertechnikSpareParts/35072\\_25\\_200x.gif](http://www.studica.com/images/product/FischertechnikSpareParts/35072_25_200x.gif) Letöltve: 2013. november 03.
- [4] [http://www.studica.com/images/product/FischertechnikSpareParts/37926\\_82\\_200x.gif](http://www.studica.com/images/product/FischertechnikSpareParts/37926_82_200x.gif) Letöltve: 2013. november 03.
- [5] [http://www.studica.com/images/product/FischertechnikSpareParts/35115\\_97\\_200x.gif](http://www.studica.com/images/product/FischertechnikSpareParts/35115_97_200x.gif) Letöltve: 2013. november 03.
- [6] [http://www.studica.com/images/product/FischertechnikSpareParts/37783\\_33\\_200x.gif](http://www.studica.com/images/product/FischertechnikSpareParts/37783_33_200x.gif) Letöltve: 2013. november 03.
- [7] [http://www.studica.com/images/product/FischertechnikSpareParts/37157\\_65\\_200x.gif](http://www.studica.com/images/product/FischertechnikSpareParts/37157_65_200x.gif) Letöltve: 2013. november 03.
- [8] [http://www.studica.com/images/product/FischertechnikSpareParts/35481\\_20\\_200x.gif](http://www.studica.com/images/product/FischertechnikSpareParts/35481_20_200x.gif) Letöltve: 2013. november 03.
- [9] [http://www.studica.com/images/product/FischertechnikSpareParts/37488\\_76\\_200x.gif](http://www.studica.com/images/product/FischertechnikSpareParts/37488_76_200x.gif) Letöltve: 2013. november 03.
- [10] [http://www.studica.com/images/product/FischertechnikSpareParts/37272\\_63\\_200x.gif](http://www.studica.com/images/product/FischertechnikSpareParts/37272_63_200x.gif) Letöltve: 2013. november 03.
- [11] U. Tietze, Ch. Schenk: Analóg és digitális áramkörök, kiadó: Műszaki Könyvkiadó, Budapest, 2000. ISBN szám: 9631600106
- [12] Ajtonyi István: PLC és SCADA-HMI rendszerek II. és Ipari kommunikációs rendszerek II. 2. kötet. kiadó: AUT-INFO Kiadó, Miskolc, 2008. 408 p. ISBN szám: 9789636618339
- [13] [http://www.conrad.com/medias/global/partner/1000\\_1999/1900/1930/1933/193398\\_00\\_1000.jpg](http://www.conrad.com/medias/global/partner/1000_1999/1900/1930/1933/193398_00_1000.jpg) Letöltve: 2013. november 04.
- [14] [http://www.tablázat.hu/images/upload/products/org\\_25319\\_1000\\_int.jpg](http://www.tablázat.hu/images/upload/products/org_25319_1000_int.jpg) Letöltve: 2013. november 04.

- [15] Ajtonyi István: PLC és SCADA-HMI rendszerek I. 1. kötet. PLC programozás az IEC 61131-3 szabvány szerint, kiadó: AUT-INFO Kiadó Miskolc, 2007. 290 p. ISBN szám: 9789630631655
- [16] Karl Heinz John, Michael Tiegelkamp: IEC 61131-3: Programming Industrial Automation Systems, kiadó: Springer, London, 2010. 400 p. ISBN szám: 9783642120145
- [17] Ajtonyi István: Ipari kommunikációs rendszerek I. 1. kötet. kiadó: AUT-INFO Kiadó, Miskolc, 2008. 431 p. ISBN szám: 9789630658133
- [18] 3D modell robotkar mozgásának betanítása ipari eszközökkel – videó: <http://www.youtube.com/watch?v=we8ZYFfWjUs>
- [19] Zilahi Krisztián László, Dr. Tóth János: 3D modell robotkar mozgásának betanítása ipari eszközökkel, XIX. FMTÜ konferencia, Kolozsvár, 2014.

## *Köszönetnyilvánítás*

*Ezúton szeretném megköszönni **Dr. Tóth János** Tanár Úrnak, hogy segítségével, szakmai ötleteivel, építő jellegű tanácsaival és észrevételeivel hozzásegített a dolgozat megírásához, valamint biztosította az ahhoz szükséges eszközöket számomra.*