

Debreceni Egyetem

Műszaki Kar

Villamosmérnöki és
Mechatronikai Tanszék



Ipari léptető és ipari szervo motorok irányítása és szinkronizálása szabványos kommunikációs protokoll segítségével

Készítette: **Tóth Xénia Erzsébet**, HM8R9N, mechatronikai mérnök hallgató

Belső konzulens: **Dr. Tóth János**

Tanszékvezető: **Dr. Husi Géza**

Debrecen

2013.

Tartalomjegyzék

Ábrajegyzék	3
Bevezetés	4
1. A dolgozat során felhasznált eszközök	5
1.1. Léptető motor és vezérlő egysége, mint „hangszer”	6
1.2. Szervo motor és vezérlő egysége, mint „hangszer”	8
1.3. Kijelző, mint beviteli eszköz	11
1.4. PLC, ami mindezt vezérli	12
2. A zene fizikai alapjai	13
3. Programozási környezetek	15
3.1. Motorvezérlők	15
3.2. A PLC programozása	19
3.3. A HMI kezelőfelülete	22
4. A TDK során végrehajtott feladatok	24
4.1. Tanulás funkció	25
4.2. Visszajátszási opció	29
4.3. DEMO üzemmód	32
4.4. Sebesség grafikonok	33
5. Fejlesztési lehetőségek	34
6. Összefoglalás	35
Irodalomjegyzék	36

Ábrajegyzék

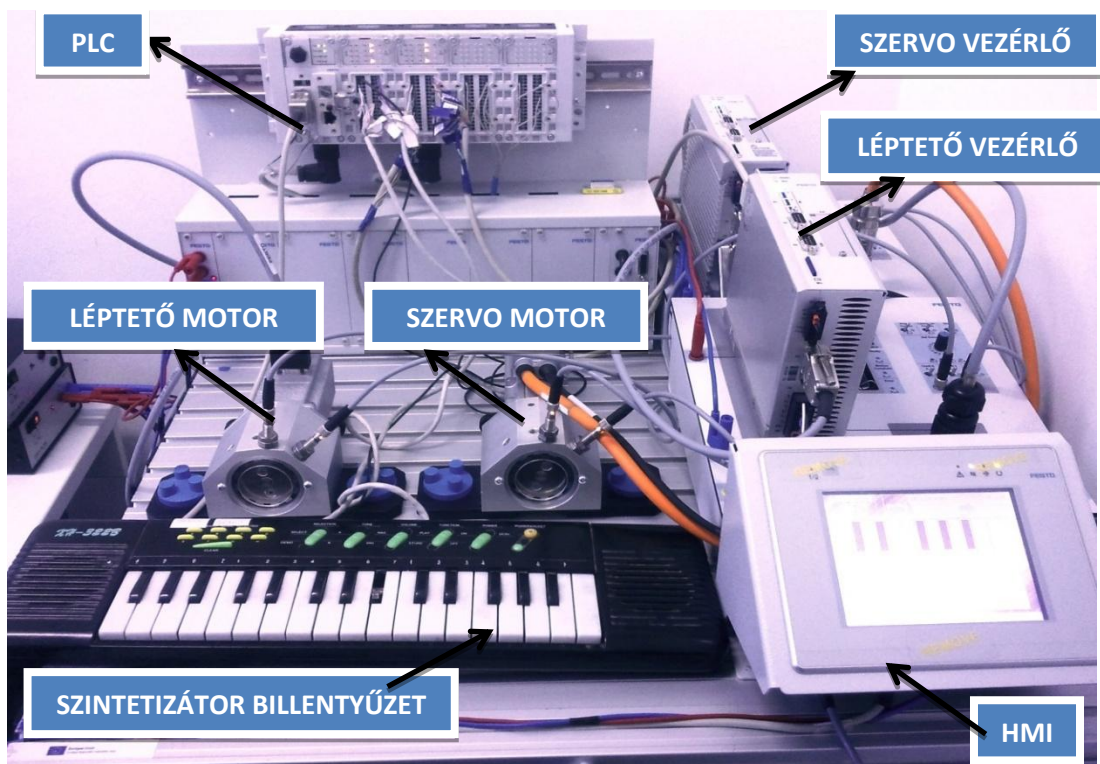
1. ábra: A felhasznált eszközök	5
2. ábra: A léptetőmotor szerkezeti felépítése	6
3. ábra: FESTO EMMS-ST-57-S-SE-G2 léptetőmotor és vezérlője [3]	7
4. ábra: Léptetőmotor gyári adatai [4].....	7
5. ábra: Léptető motor vezérlő egysége	8
6. ábra: A felhasznált motorok nyomaték-fordulatszám viszonyai.....	8
7. ábra: AC szervó motor szerkezeti felépítése	9
8. ábra: FESTO EMMS-AS-55-S-TM szervó motor és vezérlője [6], [7]	9
9. ábra: A megvalósított zárt hatásláncú rendszer	10
10. ábra: A szervomotor gyári adatai [8]	10
11. ábra: Festo FED 500-as kijelző [10]	11
12. ábra: Festo CPX típusú PLC.....	12
13. ábra: Az 'A' hang frekvenciájának duplázódása oktávugráskor	13
14. ábra: A kiegyenlített temperálás egy oktávban [18]	14
15. ábra: Vizsgálat a Visual Analyser programmal	14
16. ábra: Festo Configuration Tool kezelői felülete	15
17. ábra:A vezérlő digitális ki- és bemenetei	16
18. ábra: A profilok beállítása a léptetőmotor esetében	17
19. ábra: A profilok beállítása a szervó motornál	17
20. ábra: Léptetőmotorhoz tartozó Trace Data	18
21. ábra: A szervó motorhoz tartozó Trace Data	18
22. ábra: FST fejlesztő környezete	19
23. ábra: RS-232 DB-25 párhuzamos kábel.....	20
24. ábra: Lefutó él vizsgálatára írt programrészlet	21
25. ábra: FED Designer felhasználói felülete.....	22
26. ábra: Nyomógombra elérhető funkciók.....	22
27. ábra: Grafikonok beállítása	23
28. ábra: A vezérlő HMI főmenüje	24
29. ábra: A tanuló funkcióval ellátott kezelőfelület.....	25
30. ábra: Tanuló program, értékek nullázása	26
31. ábra: A tanulásért felelős programrészlet	27
32. ábra: A visszajátszást bemutató képernyő.....	29
33. ábra: A visszajátszás programhoz tartozó memóriák értékadása.....	29
34. ábra: A visszajátszás program részlete.....	30
35. ábra: Visszajátszás bezárása.....	31
36. ábra: Demo mód	32
37. ábra: Demo programrészlet	32
38. ábra: Motorok aktuális fordulatszámát megjelenítő grafikon működés közben.....	33
39. ábra: A továbbfejlesztett rendszer lineáris egységgel	34

Bevezetés

Dolgozatom témájának kiválasztásánál a célom az volt, hogy a korábbi ismereteimet bővítve újabb a Debreceni Egyetem, Műszaki Karának Villamosmérnöki és Mechatronikai Tanszékén található iparban használatos eszközöket ismerjek meg. Az előző évi TDK-ban egy mérőrendszer kapcsán betekintést nyerhettem egy PLC, a méréshez szükséges szenzorok, illetve egy ipari megjelenítő egység kommunikációjába. A DE-HIGH VOLTAGE csapat tagjaként az elektromos autó fejlesztése során is szereztem tapasztalatokat különféle hajtásokról, motorokról. A megszerzett ismeretek birtokában a választásom a szervó és a léptető motorokra, illetve vezérlő egységeire esett, hogy ezeket azután saját elképzeléseim alapján használjam fel.

Céлом egy olyan összetett rendszer felépítése volt, amelynek betanítható egy felhasználó által meghatározott sorrendi folyamat, mivel a tanulási képességnek az iparban kiemelkedő szerepe van. Lényeges szempont az is, hogy ez később bármikor reprodukálható legyen. Emellett fontos volt, hogy több eszközt használjak fel, amelyeket egymással összehangoltan, szinkronizálva működjenek, valamint ennek beállításain legyen lehetőség módosítani. Mindezek mellett kiemelt jelentőséggel bírt, hogy a külvilággal kommunikáljon egy ipari kijelző (HMI) segítségével, amin egy felhasználóbarát, könnyen kezelhető menüstruktúra legyen. A korábban említett motoroknak egy Programozható Logikai Vezérlő (PLC) végzi a szabályozását. A megfelelő működéshez szükséges az motorvezérlők feladat specifikus konfigurálása. Kitűzött céljaim között szerepelt továbbá az, hogy a PLC programon belül megírt előre definiált műveletsoron, valamint a tanulásért-ismétlésért felelős programrészeken kívül, segítségével lehetőség legyen a két motort egymáshoz képest időben késleltetve működtetni. A megoldandó feladatok közé tartozott az eszközök kommunikációjának megvalósítása is. A fentebb támasztott kritériumoknak megfelelően egy zenét komponálni, rögzíteni, visszajátszani, illetve kánonban is megszólaltatható modellt választottam, így ezzel egyidejűleg a motorok egy alternatív alkalmazási lehetőségét is bemutatva.

1. A dolgozat során felhasznált eszközök



1. ábra: A felhasznált eszközök

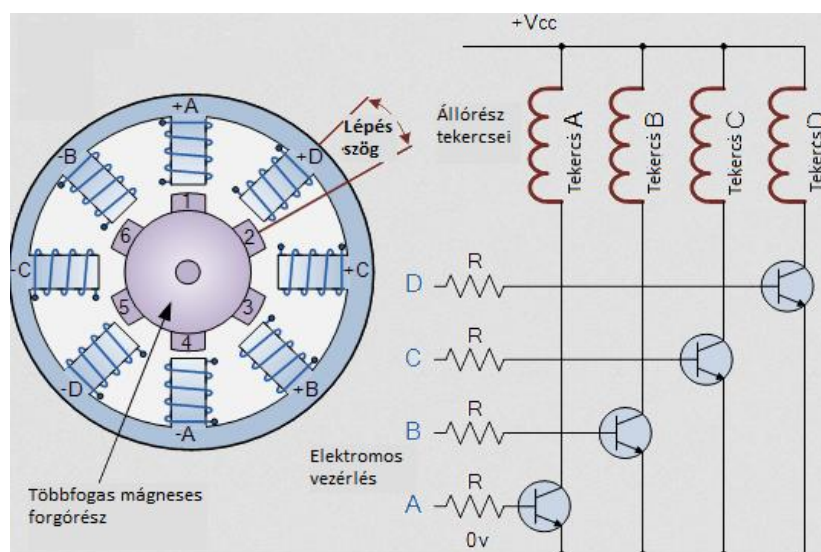
Munkám során a kitűzött feladat megvalósításához az 1. ábrán látható eszközökre volt szükségem. A hangok megfelelő kiválasztására szolgáló, valamint a motorokat közvetve irányító beviteli egység részét képezi a szintetizátor billentyűzet, valamint a HMI, aminek megjelenítő felületén szintén hasonló megjelenésű és elrendezésű nyomógombok láthatóak. A zenét szolgáltató egység a 2 motor, illetve az irányításukat végző vezérlők. Mindezen eszköz vezérlését pedig a PLC végzi. A felhasználó és a rendszer közötti kommunikációt az operátor panel biztosítja, abból a parancsok a PLC-be futnak be, amely felelős a információ továbbításért a motorok vezérlőegységei felé. A motorok pedig közvetlenül a saját vezérlőegységük irányítása alatt állnak.

Az eszközök közötti kommunikáció szabványos ipari protokollokon keresztül zajlik. Az RS-232 (Recommended Standard) szabványos protokollt használok fel a PLC és a motorvezérlők, illetve a HMI és a PLC kommunikációja során. Hátránya lehet a lassúsága, illetve az, hogy csak rövid (15 m) távolságra alkalmazható. Az RS-232 standard kommunikációs protokoll egy soros adatcserét megvalósítva meghatározza a használt feszültség szinteket a készülékek közötti adatátvitel során [1]. TCP/IP

kommunikációs protokollon keresztül valósítom meg a PLC és a PC kommunikációját. Az ipari egységek kapcsolatához szükséges volt a megfelelő kommunikációs beállítások elvégzése.

1.1. Léptető motor és vezérlő egysége, mint „hangszer”

A léptetőmotorok kommutátor (szénkefe) nélküli egyenáramú motorok, működésük olcsó és megbízható, alkalmasak pozicionálási feladatokra. Az állórész tekercselt, míg a forgórész állandó-, vagy lágymágneses anyag. A rotor forgása az álló és forgórész között létrejövő, eltérő irányú mágneses mezők kölcsönhatásából ered. A léptetőmotor elemi elfordulásokat végez az állórész tekercseire kapcsolt impulzusok hatására, tehát a villamos impulzusokat átalakítja mechanikus elfordulássá. Ennek szöge arányos az impulzusok számával, a motor fordulatszáma az impulzusok frekvenciájával és forgásának iránya pedig a sorrendjével. A szemben lévő tekercsek sorba kötését az indokolja, hogy a forgórésznek a tekercsek felé eső részein ellentétes mágneses pólus keletkezzen. A motor bármely pozícióban meg tud állni, vagy forogni valamelyik irányba. Enkóder (szögjeladó) nélkül is működtethetőek, de annak használatával pontosabb, a lépéstévesztést kiküszöbölő rendszer kapunk. A 2. ábrán egy változó mágneses ellenállású léptetőmotor felépítése látható, amely egy négy-pólusú tekercselt állórésszel és egy fogazott lágvas forgórésszel rendelkezik. A pólusok egymás utáni egyenárammal történő gerjesztésére a forgórész elfordul. A pólusszám növelése a forgórész pontosabb pozicionálását teszi lehetővé.



2. ábra: A léptetőmotor szerkezeti felépítése

Az általam használt Festo EMMS-ST-57-S-SE-G2 léptetőmotor (3. ábra) nevének elemei az alábbi tulajdonságokat jelölik: az EMMS mint: motor, ST mint: léptetőmotor, 57: azaz 57 mm-es karima méret, S: teljes hossz (small), SE: Straight connection és rendelkezik enkóderrel, továbbá G2 mint: második generáció. Fogaskerék és fék nélküli kivitel.



3. ábra: FESTO EMMS-ST-57-S-SE-G2 léptetőmotor és vezérlője [3]

A léptetőmotornak egy szegmens lépési maximális tartománya $1,8^{\circ} \pm 5\%$. A forgórész pozíciójának meghatározásáért felelős szenzor egy optikai elven működő szögjeladó.

Névleges feszültség	24-48 VDC
Névleges áramerősség	5 A
Névleges nyomaték	0,8 Nm
Maximális sebesség	2720 rpm
Nyomaték álló helyzetben	0,8 Nm

4. ábra: Léptetőmotor gyári adatai [4]

A motorok vezérléséhez szükséges a saját vezérlő egységük, amely a léptetőmotor estében a Festo CMMS-ST-C8-7-G2 motorvezérlő (3. ábra, 5. ábra), melynek neve a következő tagokból áll: CMM: motorvezérlő, S: standard, ST: léptetőmotor, C8 motor maximális áram felvétele 8A, 7: bemeneti feszültség 48 VDC, G2: második generációs. A vezérlő 24 V DC feszültségről üzemel. Található rajta egy 7 szegmenses kijelző, amely visszaigazolást ad az aktuális programról, illetve hibakódok megjelenítésére is alkalmas. A vezérlőegységben található beépített fékezés szabályozás, valamint mester-szolga mód beállítására is van lehetőség. Képes RS-232/485 és CANopen porton keresztül kommunikálni, a számítógéppel például a RS-232-es protokollon, így a hozzá tartozó szoftverrel belülről kontrollálható a vezérlő. A PLC-vel egy 25 pólusú I/O interfészen teremti meg a kapcsolatot.

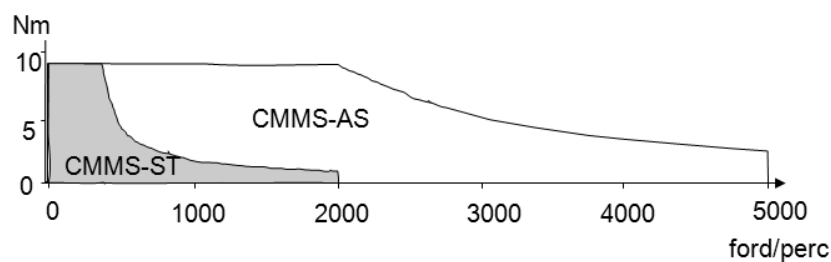


5. ábra: Léptető motor vezérlő egysége

További lehetőségeket nyújt a kézi vezérlő egység (5. ábra), amelyen az analóg vezérlést egy potenciométer teszi lehetővé. A digitális vezérlés egy 16 állású kapcsolóval jön létre, a 0 állással a home, a többivel pedig az előzetesen szoftveresen beállított pozíciók az ahhoz rendelt mozgási viszonyokkal (fordulatszám, gyorsulás, lassulás) érhetők el a start impulzus kapcsolása után. Biztosított még a Start, Stop, és a véghelyzetérzékelők be- és kikapcsolása, megtalálhatóak különböző engedélyező kapcsolók is, mint például a Power Enable, vagy Control Enable, illetve egy a digitális vagy analóg mód közötti váltásért felelős kapcsoló [5].

1.2. Szervo motor és vezérlő egysége, mint „hangszer”

A váltóáramú szervo motorok kefenélküli szinkron motorok, melyeknek egyik legfőbb tulajdonsága az, hogy adott fordulatszámon az áramerősség növelésével nagy nyomaték leadására is képesek. Így nagy tömegek mozgathatók velük, továbbá magas a pozicionálási pontosságuk.

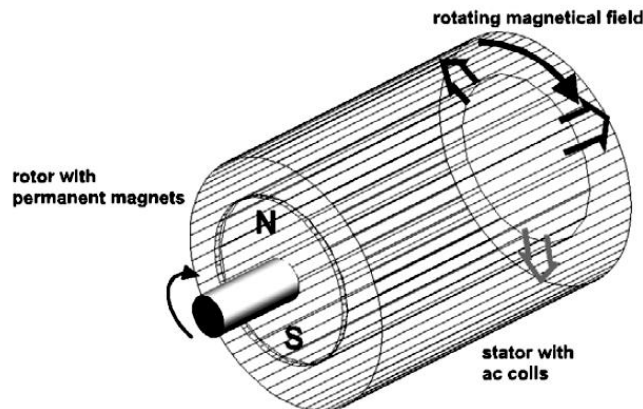


6. ábra: A felhasznált motorok nyomaték-fordulatszám viszonyai

Az iparban igen elterjedtek, fordulatszám tartományuk tágabb határok között mozog, mint a léptetőmotoroké. (6. ábra). Ezekből kifolyólag vezérlésük lehet nyomaték vagy fordulatszám alapú szabályozás. A kefenélküli szervo motor felépítését tekintve egy háromfázisú álló-, és egy állandó mágnesű forgó részből áll (7. ábra). A rotor forgási sebessége függ az állórész tekercselésében indukálódott forgó mágneses mező frekvenciájától. A rotor szinkronban mozog a stator mágneses mezejével, és azt azzal megegyező fordulatszámmal követi.

$$n = \frac{2f}{p} \quad (1)$$

A fenti összefüggésből látható, hogy a motor fordulatszáma (n) egyenesen arányos az állórész frekvenciájával (f), illetve fordítottan a pólusszámmal.

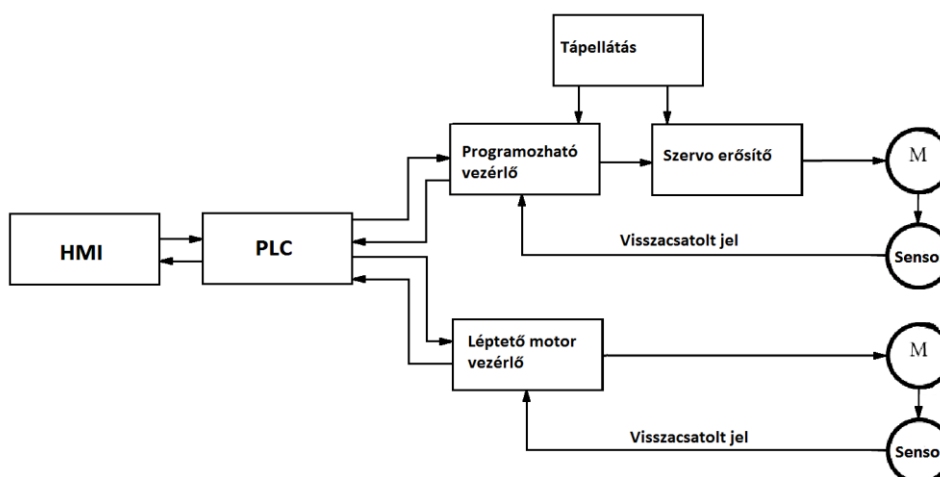


7. ábra: AC szervo motor szerkezeti felépítése

Az általam használt motor egy FESTO EMMS-AS-55-S-TM (8. ábra), amely nevének első fele tisztázza a motor típusát: az EMMS mint: motor, AS mint: szervo motor; vége pedig a további fizikai paramétereit. Megállapítható, hogy ez az 55 mm karima méretű, S (small) hosszúságú, háromfázisú kefenélküli szervo motor tengelyére fel van szerelve egy induktív többfordulatú enkóder is. Ez információkat küld vissza a motortól a vezérlő felé, tehát egy zárt hurkú rendszert használunk.



8. ábra: FESTO EMMS-AS-55-S-TM szervo motor és vezérlője [6], [7]



9. ábra: A megvalósított zárt hatásláncú rendszer

Ez a zárt hatásláncú szabályozási folyamat látható a 9. ábrán. A PLC továbbítja az információkat a motorvezérlők felé a kívánt motorparaméterekről, úgy mint a forgási irányról, a megteendő távolságról, a sebességről és a gyorsulásról. A szenzorok, amik egy-egy enkódert jelölnek, vizsgálják az aktuális valós értékeket, és ezekről egy visszacsatolást küldenek a vezérlőknek. A valós és a szükséges értékek összehasonlítása után eldöntik, hogy szükséges-e korrekció, amennyiben igen, elvégzik azt. Ez folyamatosan ismétlődik, amíg az utasításnak megfelelő értékek be nem állnak. A befejezett mozgásról a visszajelzés befut a PLC-be, ami ezután adja ki a következő parancsot. A felhasználó számára fontos információkat a is kijelzi a HMI.

A szervo motor gyári adatai a 10. ábrán láthatóak.

Névleges nyomaték	0,68 Nm
Névleges sebesség	6600 rpm
Névleges feszültség	360 V
Néveleges áramerősség	1,2 A
Nyomaték álló helyzetben	0,98 Nm
Tartó áram	1,76 A
Maximális sebesség	7330 rpm
Csúcsnyomaték	2,7 Nm
Maximális áram	5 A

10. ábra: A szervomotor gyári adatai [8]

A szervo motorhoz tartozik a Festo CMMS-AS-C4-3A-G2 motorvezérlő (8. ábra), ahol CMMS jelenti a standard motorvezérlőt, AS a váltóárammal való működést (synchronus), C4 a névleges áramerősséget, ami 4 A, a 3A pedig a névleges feszültséget, ami 230 V AC/50...60 Hz, és G2, mint második generáció. A maximális áramerősség értéke 10 A, az RS-232 soros porton létrejövő jelsebesség 9600-115000 bits/s lehet. A vezérlőhöz csatlakozó hálózati feszültség a motorhoz szükséges, a vezérlést pedig +24 VDC megáramlítás biztosítja. A kezelő vezérlőpanelre korábban 1.1. fejezetben a léptető motornál tárgyaltak vonatkoznak [9]. A két motorvezérlő a PLC-hez RS-232-es protokollon, DB-25 párhuzamos kábelen keresztül kapcsolódik.

1.3. Kijelző, mint beviteli eszköz

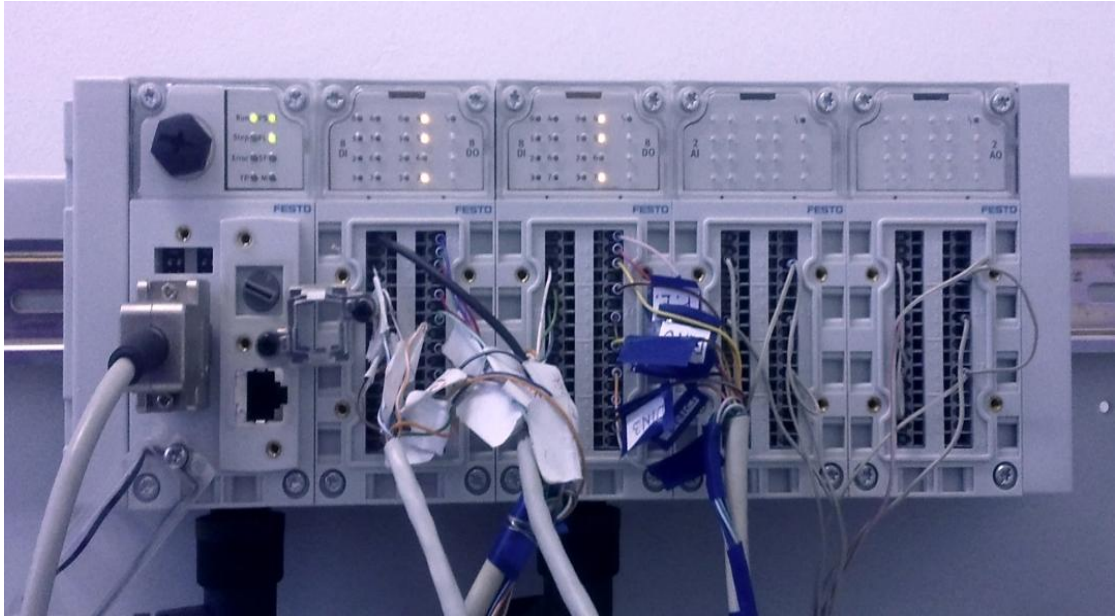


11. ábra: Festo FED 500-as kijelző [10]

A rendszer a felhasználóval való kapcsolatteremtést egy Festo FED-500-as típusú (Front End Display) HMI-vel (Human-Machine Interface) [11] (11. ábra) valósítja meg, ami nyomásra érzékeny (rezisztív), 8 bites színmélységű, 5,6"-os képátlójú, 320x240 felbontású érintő képernyővel rendelkező kijelző. Képes a PLC-ből kiolvasott adatok, grafikus elemek, diagramok kijelzésére, aktiválhat különböző üzemmódokat, és megadhat azokhoz tartozó paramétereket, ezzel a PLC-be is beírva azokat. Ezen operátorpanel választását indokolja az is, hogy kompatibilis az általam használt Festo CPX PLC-vel. A PLC és HMI között valós idejű kommunikáció zajlik (az adatok kevesebb, mint 10 ms-os időközönként frissülnek), valamint konfiguráció során a PLC típusának megadása mellett, a kommunikációs beállítástól függően legfeljebb 38400

Baud/mp-es jelátviteli sebességet választhatunk, A kapcsolat egy RS-232 protokollon, 15 lábbal rendelkező Sub-D csatlakozón keresztül valósul meg [12]. A PC-hez soros porton, 9 tűs Sub-D illesztő felülettel, RS-232 protokollon keresztül kapcsolódik [13].

1.4. PLC, ami mindezt vezérli



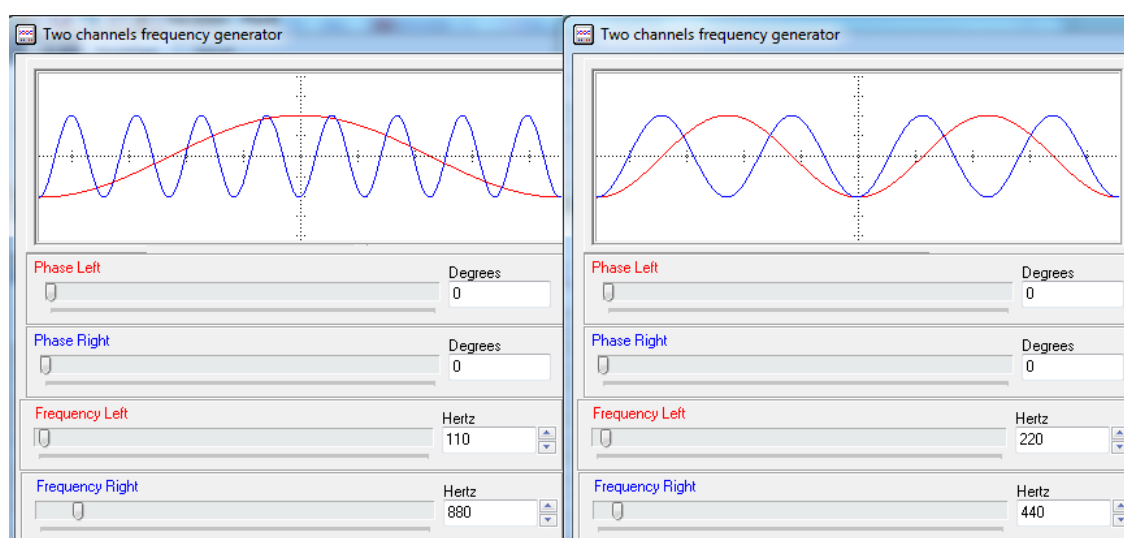
12. ábra: Festo CPX típusú PLC

A vezérlés központi egysége egy moduláris felépítésű FESTO CPX típusú PLC (Programozható Logikai Vezérlő) (12. ábra), amelyből az általam használt kivitel 5 modulból épül fel. Feladata a motorok vezérlése, illetve a kijelzővel való kommunikáció, amit soros porton keresztül valósít meg. A vezérlőegység egy CPX-FEC típusú modul, a bővítő modulokat tekintve, munkám során 16 digitális be-, és 16 digitális kimenetet használtam, ennek megfelelően kettő darab mindezt kezelő 8DI/8DO - Multi I/O modulra volt szükségem. Ezen egységek közötti kommunikáció az adatsínen jön létre, tápellátásukat a láncelemből nyerik, amely az ipari jelszinteknek megfelelően 24 V. Egy sínen egyszerre legfeljebb 11 modul fér el. A PLC a PC-vel TCP/IP protokollon keresztül kommunikál, továbbá ismeri a Modbus/TCP, Easy IP, és HTTP protokollokat is [14].

2. A zene fizikai alapjai

Ahhoz, hogy a kiválasztott modellt helyesen építsük fel, szükséges foglalkozni a zenei háttérrel is, mivel ebben az esetben a diszkrét pozíciók zenei hangokat fognak jelölni.

Az emberi fül hallástartománya 20 Hz és 20000 Hz között van, míg a legérzékenyebb a 300 és a 3000 Hz közötti hangokra. A zenei hangskálában különböző, meghatározott frekvenciájú hangok (zenei hangok) követik egymást, így azok 12 félhangból álló oktávba rendezhetőek. Egy hang alapfrekvenciája oktávugráskor megduplázódik, ez látható a 13. ábrán a zenei A hang 4 különböző oktávban, 110, 220, 440 és 880 Hz-en.



13. ábra: Az 'A' hang frekvenciájának duplázódása oktávugráskor

A hangolási módszerek közül a kiegyenlített temperálást alkalmazva az oktávot 12 egymástól matematikailag pontosan egyenlő távolságra lévő hangra osztjuk fel. Bármelyik oktávban kiszámíthatjuk a zenei hangok frekvenciáját, mivel azok így egy mértani sort alkotnak. A q arányossági tényezővel számolva, ha f_1 jelöli az oktáv 1. hangját, tudva hogy egy oktáv 12 hangból áll, és oktávugráskor a hang alapfrekvenciája kétszereződik, felírhatjuk az alábbi összefüggést:

$$f_{13} = q \cdot f_{12} = q^2 \cdot f_{11} = \dots = q^{12} \cdot f_1$$

$$f_{13} = 2 \cdot f_1 = f_1 \cdot q^{12}$$

$$q^{12} = 2 \rightarrow q = \sqrt[12]{2} = 1,0594631 \quad (2)$$

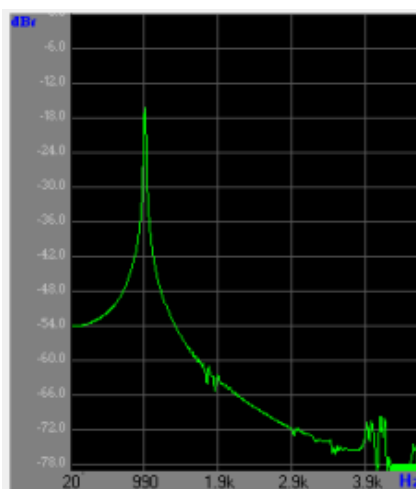
Tehát két szomszédos hang közötti matematikai viszony $\sqrt[12]{2}$. Az ISO 16:1975 szabvány meghatározza a folyamatnál kitüntetett szerepű az A hang frekvenciáját 440

Hz-ben. Ez a hang a C-dúr hangskála 6. eleme, az ún. kamarahang, ez az egységes hangolás referenciahangja, ebből meghatározhatjuk a többi zenei hangot. A 14. ábrán egy oktávot láthatunk, az első hanghoz viszonyított arányok feltüntetésével, a hangok elnevezése balról jobbra haladva: C, C $\sharp$, D, D $\sharp$, E, F, F $\sharp$, G, G $\sharp$, A, A $\sharp$, H.



14. ábra: A kiegyenlített temperálás egy oktávban [18]

A motor forgásakor keletkező hanghullámok átadódnak a levegő részecskéinek, így az egyre magasabb fordulatszámok használatakor a keletkező nagyobb frekvenciájú hangokat is magasabbnak halljuk. A motorok megfelelő fordulatszámának kiválasztásával zenei hangokat tudok definiálni. Ehhez hangoltam be először a korlátozottabb fordulatszám-tartományú léptető motort a Visual Analyser szoftver és a számítógémem hangkártyájának segítségével. Először a referenciahanghoz tartozó fordulatszám értéket volt szükséges megtalálni, aminek frekvenciája a program segítségével ellenőrizhető volt. Az A hang megkapott fordulatszámából a (2) összefüggés használatával könnyedén kiszámítható az egyes hangokhoz tartozó fordulatszám érték, és ellenőrizhető a számítógépes méréssel (15. ábra). Mivel a program 8 profil tárolását teszi lehetővé, így a félhangok használatától eltekintek, tehát a skálám egy oktávot ölel fel a C hangtól a C'-ig.

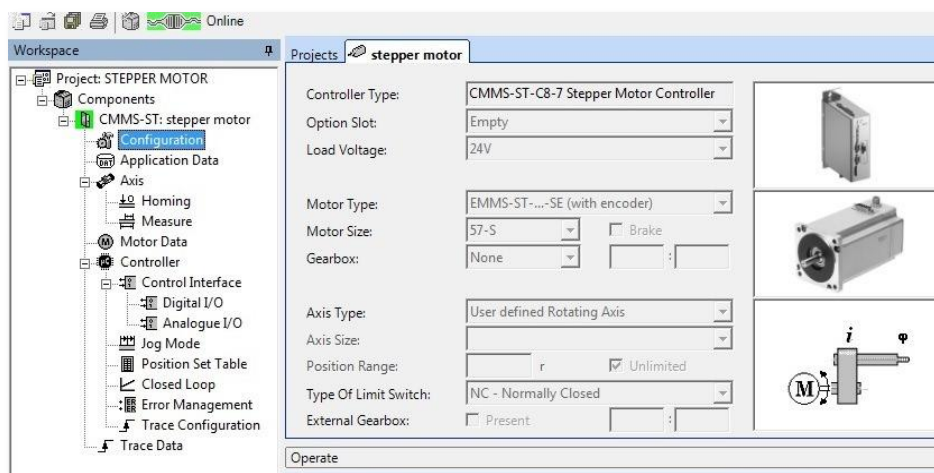


15. ábra: Vizsgálat a Visual Analyser programmal

3. Programozási környezetek

Ahhoz, hogy az általam kitűzött feladatot sikeresen elvégezzem, a fentebbi eszközök egymás közötti kommunikációját meg kellett valósítani. Ehhez elengedhetetlen volt az eszközökhöz tartozó fejlesztői környezetek megismerése, mint az FCT konfiguráló program, az FST programozó szoftver, illetve a FED Designer képernyő készítő alkalmazás. Ezen programok segítségével végeztem el az eszközök megfelelő beállítását az optimális működés érdekében.

3.1. Motorvezérlők



16. ábra: Festo Configuration Tool kezelői felülete

A Festo Configuration Tool szoftver segítségével mind a léptető, mind a szervó motor vezérlője konfigurálható, a programon belül a vezérlő kiválasztása, illetve az adott motor különböző gyári paramétereinek megadásával (16. ábra). Rögzíteni kell a programban továbbá azt is, hogy ne legyenek beállítva végállások, mivel egyik motor esetében sem használok lineáris egységet, ezzel biztosítva a forgó rész zavartalan forgását. A motorok működtetéséhez be kell állítani a home pontjukat, amit az aktuális pozícióként definiáltam.

A program lehetőséget biztosít továbbá arra, hogy meghatározzuk a tengely sebességének határértékét, amit a léptető motor esetében 2000 rpm-ben maximalizáltam, valamint a szervó motor esetében, ami szélesebb fordulatszám tartományon dolgozik, 4370 rpm-ben. A digitális vezérlőjelek alkalmazása esetén 63

állapot tárolható a Position Set Table-ben, az elmozdulás értékével, illetve a mozgás dinamikáját befolyásoló 8 beállított profil valamelyikének használatával együtt. A 0. sorszámú kitüntetett szerepet tölt be, ekkor tér vissza a motor a már korábban beállított 0, azaz home pontjába. A beállítások megkönnyítése érdekében, ezek engedélyezés után közvetlenül a programon belülről tesztelhetők a kiadott vezérlőjelek alkalmazásával is. Ennek során a programban közvetlen visszajelzést kapunk a motor aktuális állapotáról is (pl: pozíció, sebesség, hőmérséklet, stb.).

A digitális I/O módban a vezérlés megvalósításához szükséges ismerni a vezérlő lábkiosztását (17. ábra). A pozíciókat a DIN0-4 bináris kódjaként adhatjuk meg. Szintén definiálhatunk 8 profilt, és ezek kombinációjával elkészíthető a PLC program, ami a megfelelő választja ki a felhasználói igénynek megfelelően. A vezérlő rendelkezik digitális kimenetekkel is, amivel megbizonyosodhatunk arról, hogy a vezérlő egységünk készen áll a parancsok fogadására, arról, hogy befejezte a motor a mozgást, valamint visszajelzést kapunk az esetleges hibákról is. A Mode Selection bitek segítségével választható az egyszerű pozicionálási üzemmód.

Digital Inputs		Digital Outputs
DIN0: Record Select 0	DIN8: Start Positioning	DOUT0: Controller ready for operation
DIN1: Record Select 1	DIN9: Mode Select Bit 1	DOUT1: Motion Complete
DIN2: Record Select 2	DIN10: Record Select 4	DOUT2: Acknowledge Start
DIN3: Record Select 3	DIN11: Record Select 5	DOUT3: Error
DIN4: Enable Power	DIN12: Mode Select Bit 0	
DIN5: Enable Control	DIN13: Stop	
DIN6: Limit Switch 0		
DIN7: Limit Switch 1		

17. ábra: A vezérlő digitális ki- és bemenetei

Célom a vezérlőnként 8 különböző profil jellemzőinek (sebesség, felfutási és lefutási idő-gyorsulás) tárolása a vezérlő egységekben, illetve ezek későbbi felhasználása zenei hangokként. Nem szándékozom lineáris egységet használni munkám során, így a motor konfigurációjában a felhasználó által korlátlanra beállítható forgó tengelyt választottam.

Projects	stepper motor *							
Position List		Position Profiles						
No.	Vel. [rpm]	Accel. [rpm/s]	Decel. [rpm/s]	Smooth [%]	Time [ms]	Start D. [ms]	Fin.Vel. [rpm]	Startcond.
0	635,00	50000,0	50000,0	0	0	0	0,00	Interrupt
1	702,00	50000,0	50000,0	0	0	0	0,00	Interrupt
2	794,00	50000,0	50000,0	0	0	0	0,00	Interrupt
3	841,00	50000,0	50000,0	0	0	0	0,00	Interrupt
4	944,00	20000,0	20000,0	0	0	0	0,00	Interrupt
5	1060,00	20000,0	20000,0	0	0	0	0,00	Interrupt
6	1199,00	20000,0	20000,0	0	0	0	0,00	Interrupt
7	1270,00	20000,0	20000,0	0	0	0	0,00	Interrupt

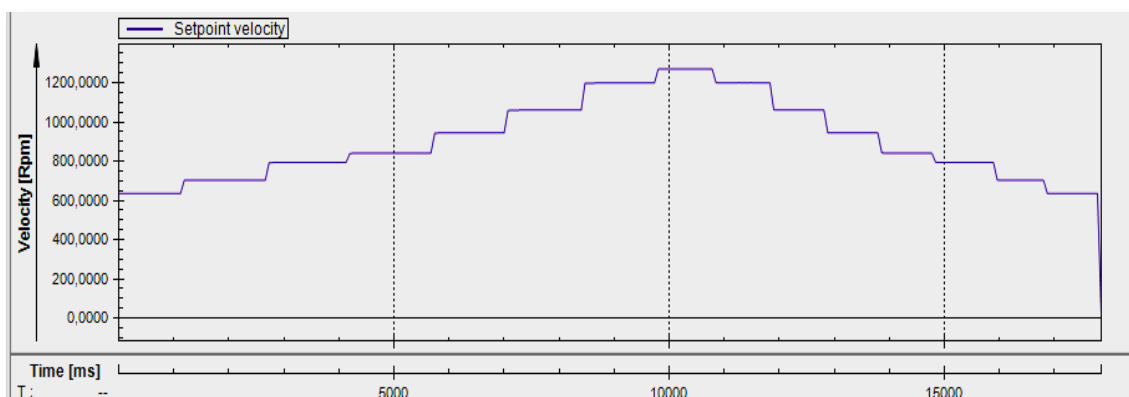
18. ábra: A profilok beállítása a léptetőmotor esetében

Projects	axis							
Position List		Position Profiles						
No.	Vel. [rpm]	Accel. [rpm/s]	Decel. [rpm/s]	Smooth [%]	Time [ms]	Start D. [ms]	Fin.Vel. [rpm]	Startcond.
0	2000,00	419517,0	419517,0	0	0	0	0,00	Interrupt
1	2250,00	419517,0	419517,0	0	0	0	0,00	Interrupt
2	2500,00	419517,0	419517,0	0	0	0	0,00	Interrupt
3	2666,00	319517,0	419517,0	0	0	0	0,00	Interrupt
4	3000,00	319517,0	419517,0	0	0	0	0,00	Interrupt
5	3332,00	219517,0	319517,0	0	0	0	0,00	Interrupt
6	3650,00	219517,0	319517,0	0	0	0	0,00	Interrupt
7	4000,00	219517,0	319517,0	0	0	0	0,00	Interrupt

19. ábra: A profilok beállítása a szervó motornál

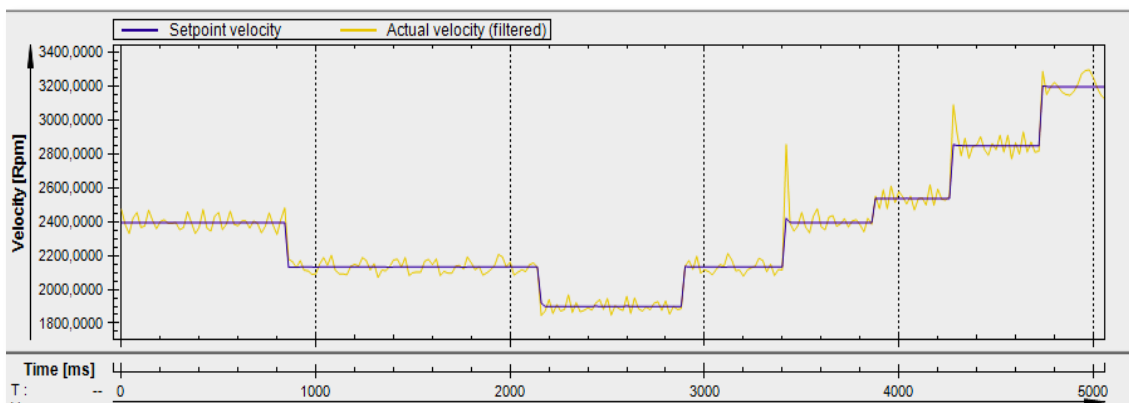
A 18., 19. ábrán látható a 8 profil megfelelő beállítása, mint sebesség [rpm], gyorsítás és lassítás [rpm/s], majd megadva, hogy 2 egymást követő bemenő jel esetén a második megállás nélkül, az előző jel fordulatszámáról tovább gyorsítva, avagy lassítva érje el saját fordulatszámát, pozícióját. Ennek eredményeképp a pozíciók nem egymás lefutása után, szaggatottan játszódnak le, azaz a dallam folyamatos lesz. A gyorsításhoz illetve a lassításhoz jelen esetben ajánlott magas értékeket rendelni, hogy időben hamar érje el az általunk megkívánt fordulatszámot, a hangok csúsztatás nélkül hallatszódnak.

Az első nyolc pozícióhoz rendelt sorban a profilokat, illetve 500 fordulatot állítottam be az elmozdulás értékének, hogy a hosszan kitartott hangok is akadálytalanul megszólaltathatóak legyenek, ne érjenek véget a felhasználó számára idő előtt.



20. ábra: Léptetőmotorhoz tartozó Trace Data

Az FCT lehetőséget biztosít a digitális vezérlés esetében úgynevezett beépített *Trace data* funkcióra. A 20. ábrán látható, hogy különböző paramétereket ábrázolhatunk az idő függvényében grafikonos formában. Jelen esetben a léptető motor sebessége látható egymást követően az első 8 pozícióba állva, majd onnan lépésről lépésre visszatérve a 0 pozícióig, mindezt az idő függvényében.

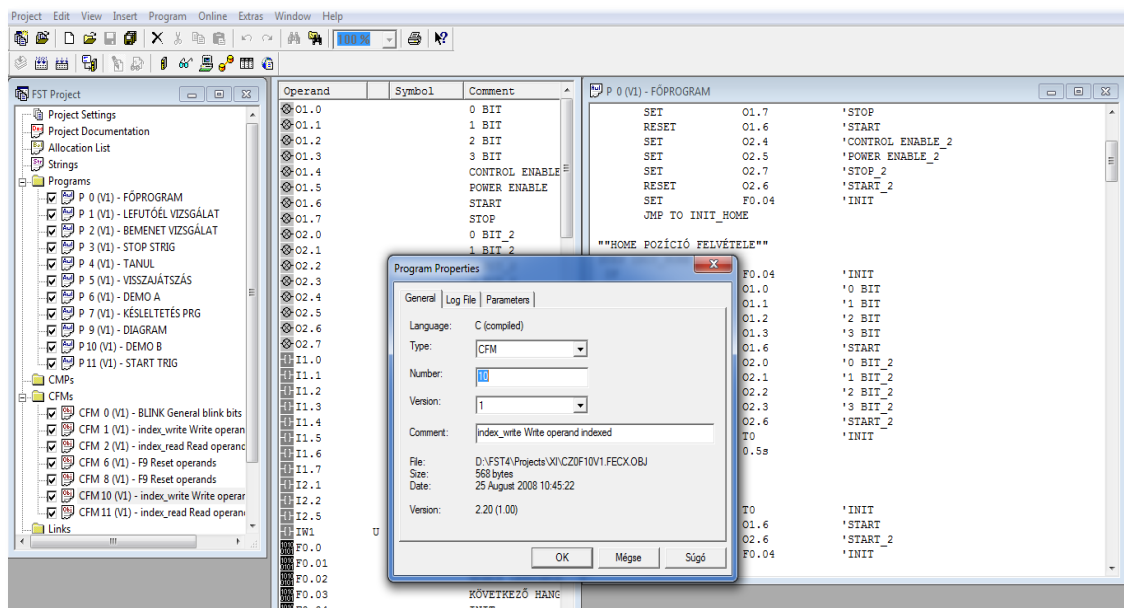


21. ábra: A szervo motorhoz tartozó Trace Data

A 21. ábrán a szervo motor 6 pozíciója látható, a beállított sebességgel (setpoint velocity), valamint a motor aktuális fordulata, a referenciához való pillanatnyi kitérésekkel.

A kézi vezérlő egységen a beállított első 16 pozíciót tudom elérni, ami 4 bemenő bit felhasználásával történik. Ennek megvalósításához először bekapcsolt állapotban kell lennie a tápellátás-, illetve motorvezérlő engedélyezésének, majd minden digitális bemenet futtatásához szükséges a start felfutó éle, illetve a stop jel bekapcsolt állapota.

3.2. A PLC programozása



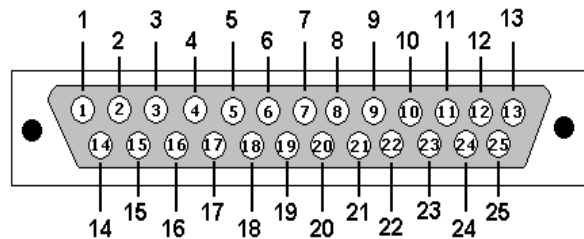
22. ábra: FST fejlesztő környezete

A CPX-FEC PLC programozói környezete a Festo Software Tools 4.21.153 (a továbbiakban FST) nevű szoftver, ami a 22. ábrán látható. Bekonfigurálható benne a PLC, illetve a különböző bővítő modulok típusai. Lehetőség van programozni létradiagramos (LD), valamint az általam is használt lépés szerkezetű programnyelven (STL). Ez utóbbi sorrendi folyamatokat ír le, melynek lényege, hogy azt lépésről-lépésre hajtja végre. Az aktív szekvenciából akkor engedi a programvezérlés tovább, ha annak utolsó feltétele is teljesül, az inaktívakban szereplő feltételek figyelmen kívül hagyásával [16].

A PLC digitális bemeneteket kezelő két moduljába egy átalakított szintetizátor billentyűről, illetve nyomógombjairól, kapcsolóiról érkezik 8-8 impulzus. Az eszköz a célnak megfelelően át van alakítva. A billentyűk lenyomásakor az alatta található PLC tápfeszültségével megtáplált fém érintkező és a pozíciónak megfelelően elhelyezett vezeték rövidre záródnak.

A PLC digitális kimeneteit kezelő moduljaiból továbbítódnak a szükséges bemeneti információk a motorok vezérlői felé, mindez egy átalakított 25 lábból álló párhuzamos kábelon keresztül.

Ekkor a motorvezérlőt, a kézi vezérlőpanelt kikerülve, PLC-vel akarom működtetni. Fontos a működés meghatározása a megírandó program szempontjából. A motorvezérlő lábkiosztása (23. ábra) fellelhető az eszköz gyári adatlapján, illetve ezen adatok méréssel ellenőrizhető.



23. ábra: RS-232 DB-25 párhuzamos kábel

A lábkiosztás: 6: Ground, 7:POS-Bit 1, 8: POS-Bit 3 , 9: Control Enable,
15: Stop, 19: POS-Bit 0, 20: POS-Bit 2, 21: Power Enable, 23: Start

A meghívni kívánt pozíciók sorszáma a 4 position set bitek bináris kódjaként továbbítódik a motorvezérlők felé. A 0001 jelöli az első pozíciót, amihez a C hang fordulatszámának megfelelő profilt rendeltem, majd a skálának megfelelően a 8. pozíciót, a C'-t a 1000 jelöli.

A HMI programozása során használt változókat a programon belül flag-ként definiáltam. A programban 256 számláló, regiszter és időzítő, valamint szó formában is címezhető 10000 flag bit áll rendelkezésre. Egy flagword első bájtját veszem igénybe a HMI-n a PLC bemeneteként szolgáló 8 billentyűhöz, további 8-ra pedig mint a PLC-ből való visszacsatolásra van szükségem. Az operátorpanelen megjelenő gombok állapotát is flag-ek értékeihez rendeltem.

Az előbb tisztázott be- és kimenetekhez rendelt, valamint a különböző definiált változók az allokációs listában kapnak helyet.

A program szerkezetét tekintve egy főprogram kezeli a többi alprogramot, azokat szükség szerint aktívva téve, illetve folyamatosan figyelve a futás közbeni fontosabb állapotokat. Új változó értékének megadása után, a start impulzus ismételt kiadása válik szükségessé.

Első kezelendő problémát jelenti a billentyűzetnél a több gomb egyidejű lenyomásának eshetőségére. Erre megoldás lehet az időben hamarabb lenyomott gomb dominánssá tétele. A program ciklikusságából adódóan figyelni kell, a megfelelő feltételek meghatározására, amely szerint felengedés előtt kiadott újbóli start impulzus ne jöjjön

létre, ellenkező esetben az egy adott folyamatosan nyomva tartott billentyű újbóli indításaiból zajt okoz.

```

IF          I1.7          'C' BILLENTYŰ
OR          F4.07          'C' HMI
THEN RESET  F0.02          'NINCS LENYOMVA BILLENTYŰ
RESET       F0.01          'VAN LEFUTÓ ÉL
RESET       F0.03          'KÖVETKEZŐ HANG
SET         F3.07          'C' VISSZA
JMP TO c1

""LEFUTÓ ÉL VIZSGÁLAT""
STEP C
IF          N            I1.0          'C BILLENTYŰ
AND         N            F4.00          'C HMI
THEN RESET  O1.6          'START
SET         F0.01          'VAN LEFUTÓ ÉL
RESET       F3.00          'C VISSZA
SET         T12           'LEFUTÓ KÉSL
WITH        FW10          'KÉSLELTETÉS IDŐ
SET         F11.02        'LEFUTÓ KÉSL
JMP TO kés1

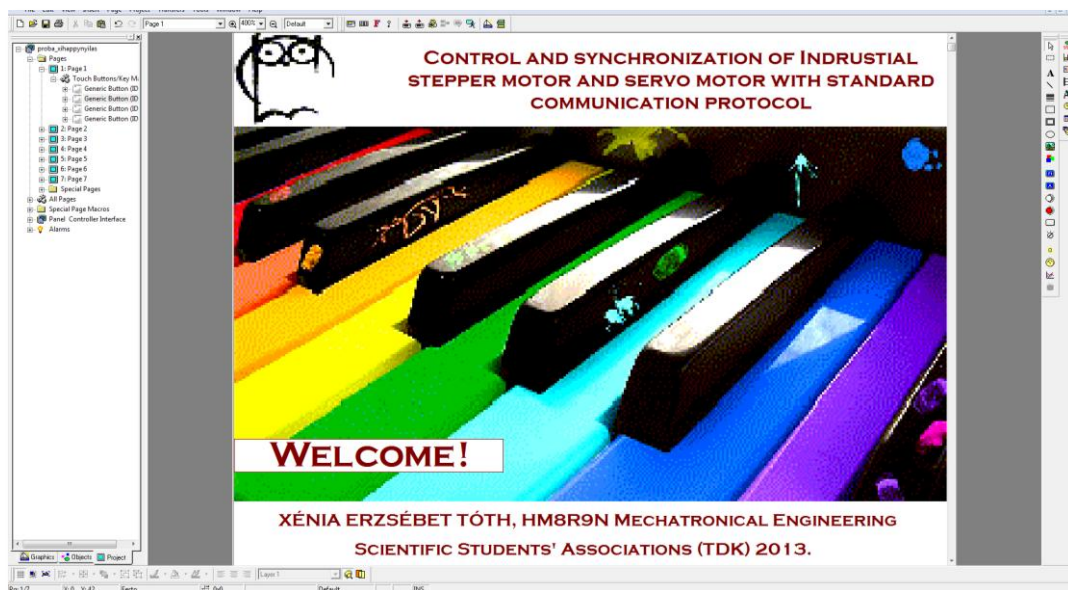
```

24. ábra: Lefutó él vizsgálatára írt programrészlet

A működés során figyelni kell a domináns billentyűk aktívra tétele után a lefutó élüket. Ehhez az FST-ben való programozásban nincs külön a jelek fel-, vagy lefutó élet figyelő függvény, valamint élvezérelt nyomógomb, így egy háttérben folyamatosan futó alprogramba került ennek vizsgálata (24. ábra). Működését tekintve, ha valamely általa állandóan vizsgált változó aktív állapotba került, akkor annak érdekében, hogy azt más ne befolyásolhassa új step-re vált, ahol mindaddig tartózkodik, amíg a bemenet megszűnését, lefutó élet nem érzékeli. A program végén egy választó lépésbe érkezik, amely vagy aszerint léptet tovább, hogy mindegyik billentyű fel volt engedve, vagy 2 lenyomott közül a dominánst engedték el. A különbség az, hogy az előbbi esetben meg kell szakítani a forgást, amit a control enable parancs pillanatnyi elvételével valósítok meg, utóbbi esetben viszont elegendő egy új start impulzus. Külön alprogramot rendeltem a bemenetek vizsgálatához, aminek belépési feltétele a már korábban említett lefutó él, avagy az, ha nincs beérkező impulzus.

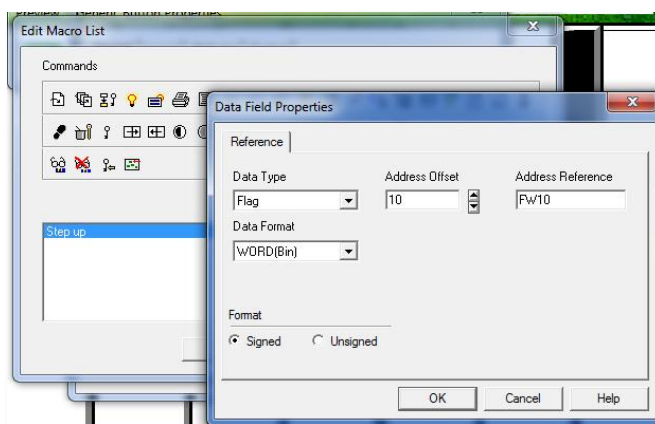
Mivel a kijelző működési elvét tekintve nem kezeli a több ponton való egyidejű érintést, így a fentebbi probléma ebben az esetben nem állt fenn.

3.3. A HMI kezelőfelülete



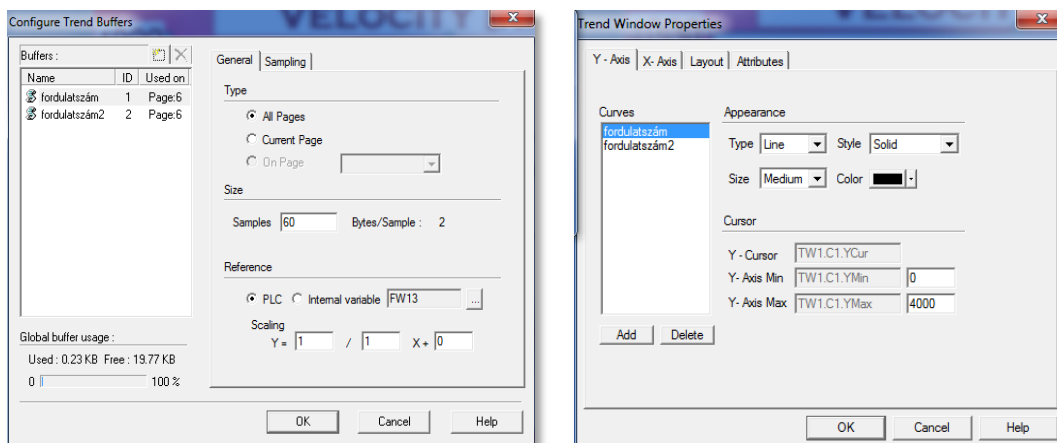
25. ábra: FED Designer felhasználói felülete

HMI csak konfigurációs módban érhető el a számítógépről, melyhez egy funkció nélküli területet kell 3 s-ig nyomva tartani. A megjelenítő felületre a FED Designer nevű szoftverrel (25. ábra) készíthetünk egyedileg tervezett alkalmazásokat, megjeleníthetünk ábrákat, grafikonokat, továbbá saját menüsorral bővíthetjük applikációnkat. Az általam használt operátorpanel bekonfigurálása, a PLC típusának és a kommunikáció során történő jelátviteli sebességnek a megadása, illetve a PLC-ből érkező adatok folyamatos frissítésének beállítása után tölthető fel a létrehozott program a készülékre.



26. ábra: Nyomógombra elérhető funkciók

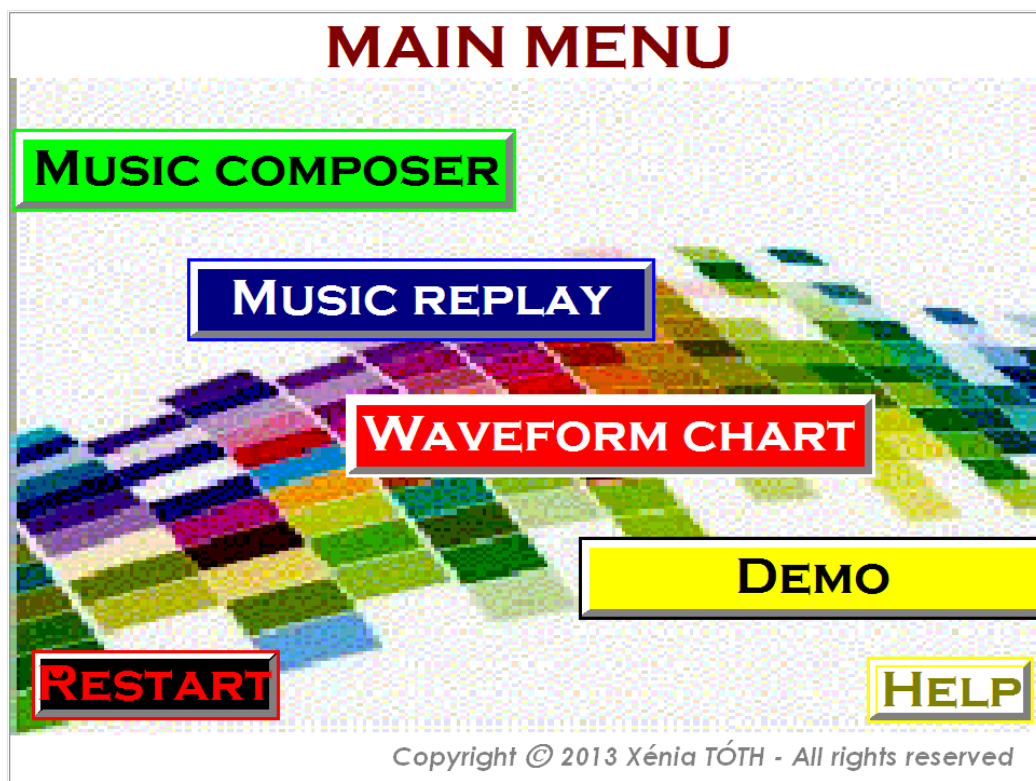
Létrehozhatók különböző képernyők (screen-ek), amelyeken a menüsor nyomógombjai funkciókkal láthatók el, például: jelszóval levédésre, képernyő kontrasztjának beállítására, dátum kijelzésére, a screen-ek közötti váltásra, grafikonok beépítésére, illetve a PLC-ben definiált változók értékének módosítására is van lehetőség. A gombok tulajdonságainál állítható be, hogy a PLC-ben meghatározott változók közül melyiket rendelem hozzá (26. ábra).



27. ábra: Grafikonok beállítása

A grafikonok beépítéséhez meg kell határozni a megjeleníteni kívánt értékhez tartozó PLC megfelelő változóját, a ciklikus frissítésre vonatkozó időtartamot (1s), a diagram Y tengelyéhez tartozó megjeleníteni kívánt legnagyobb értéket, illetve az X tengelyhez tartozó időt (hosszát), valamint a minták számát (27. ábra).

4. A TDK során végrehajtott feladatok

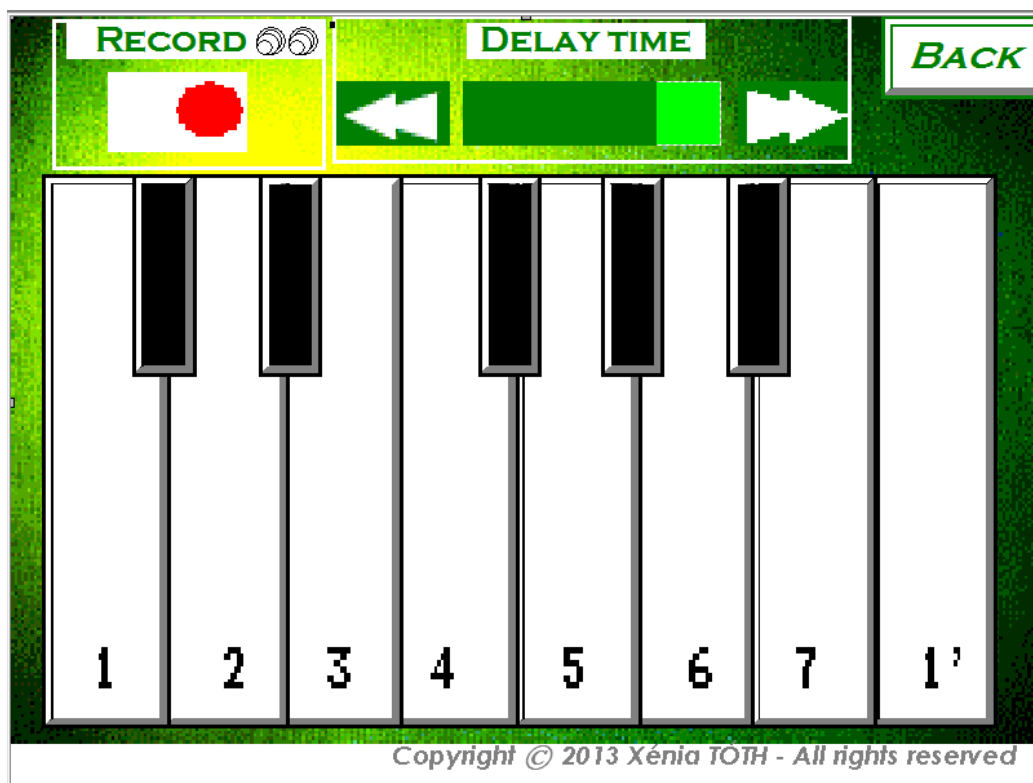


28. ábra: A vezérlő HMI főmenüje

Munkám során a könnyebb kezelhetőség érdekében létrehoztam egy menürendszert (28. ábra), melyben a gombokat megnyomva további almenükbe léphetünk, majd onnan a **Back** segítségével visszatérve újra választhatunk a funkciók közül. A gombok színe megegyezik a hozzájuk tartozó képernyők háttérszínével, ezzel is segítve a felhasználót az eligazodásban.

A **Music composer**–ből elérhető felületen, zenét tudunk lejátszani, illetve azt rögzíteni, amit a **Music replay** megfelelő piktogramjai kiválasztásával visszajátszhatunk. A **Waveform chart** menüpontból a két motor aktuális fordulatszámát, illetve azoknak időbeni változását kísérhetjük figyelemmel, míg a **Demo** módban előre leprogramozott dallamok közül választhatunk. A **Restart** felirattal hiba esetén újraindíthatjuk a rendszert, míg a **Help** a menüpontok nyújtotta lehetőségekről számol be a felhasználónak, a könnyebb tájékozódás érdekében.

4.1. Tanulás funkció



29. ábra: A tanuló funkcióval ellátott kezelőfelület

A 29. ábrán látható felületen lehet a billentyűk segítségével vezérelni a morokat, amelyek közül a feketékhez nincs külön funkció rendelve. Lenyomáskor az aktuális gomb szürkére vált ezzel valóság-hű hatást keltve. Ezek a bemenetek a PLC-ben egy flagword (FW4) első 8 bitjének feleltethetők meg, amik a programban a kezelés szempontjából ugyanúgy viselkednek, mint a zongorabillentyűkhöz tartozó valós fizikai bemenetek, amit a 2.2 fejezetben részleteztem. A **Record** felirat alatt található piros körrel indíthatjuk el magát a tanulási folyamatot, amely ekkor billenőkapcsolóként átvált zöld négyzetre, ennek megnyomásával tetszőlegesen megszakítható a mentési folyamat. A program 122 állapotot képes tárolni a tanulási funkció bekapcsolása során. Az aktív művelet ideje alatt közvetlen a felirat mellett található indikátor lámpa piros színűre vált, visszajelzésként a kezelő számára, valamint megszakításkor az visszatér az eredeti fehér állapotába. A másik LED a felvétel memóriájának megteltkor pirosan világít mindaddig, míg a tanulást újból el nem indítjuk, ekkor a régi adatsor automatikusan törlődik. A **Delay time** funkcióval az egyik motort időben később tudjuk

megszólaltatni a másikhoz képest, a csúszkán beállítottnak megfelelő értékkel, ezzel egyfajta zenei kánont hozva létre. A **Back** feliratot kiválasztva visszatérünk a főmenübe.

Az elvégzett sorrendi műveleteket (vagyis magát a dallamot) a PLC regisztereibe mentem el, majd pedig onnan hívom elő a lejátszás során.

```

STEP init
""ÉRTÉKEK NULLÁZÁSA""
IF      ( I2.1      'TANUL
  AND    N I2.2      'VISSZAJÁTSZÁS
  OR     ( F5.03      'TANUL_IGEN
  AND    N F5.04      'TANUL_NEM

THEN    CFM 8      " Reset operands
  WITH   V1         " 0:R, C, T, F; 1:R; 2:F; 3:T; 4:C

""MEMÓRIA INDEXEK ALAPÁLLAPOTBA ÍRÁSA""
LOAD    V1
TO      R0          'BEMENET_INDEX
LOAD    V124
TO      R123        'IDŐ_MENT_INDEX
RESET   F0.05        'TANUL PRG BEZÁRHATÓ
SET      F0.09        'ENGEDÉLYEZŐ_TANUL
RESET   F5.07        'MEMÓRIA_MEGTELT

""SZÁMLÁLÓ ALAPÁLLAPOTBA ÍRÁSA""
LOAD    V0
TO      CW123        'IDŐ

""KÜLÖN KEZELI A KIJELZOROL ILLETVE A SZINTETIZÁTORRÓL TANULÁST""
IF      I2.1      'TANUL
  AND    N F5.03      'TANUL_IGEN
THEN    JMP TO 0
IF      F5.03      'TANUL_IGEN
  AND    N I2.1      'TANUL
THEN    JMP TO kijelzo

```

30. ábra: Tanuló program, értékek nullázása

A tanulási folyamat megkezdésekor fontos szempont a tároló memória nullázása (30. ábra). Ez abban az esetben szükséges, ha az aktuális műveletet megelőzően, tárhelynek egy nagyobb szegmensét használtuk ki, mint amennyire az újabb felvételkor mentettünk, hogy a visszajátszás során az frissebb művelet elvégzése után ne folytatódjon a tárhely adott pontjától a régebbi, korábban megjegyzett folyamat. Ennek elkerülése érdekében a funkció újbóli bekapcsolásakor, ami a szintetizátor esetében az I2.1 bemenet, a HMI-nél pedig a F5.03 aktívva válását jelenti, törölni kell ezt a memóriát. Első lépésben meghívom az F9 (Reset operands) nevű, a programba már korábban importált függvényt, amit CFM 8-ként deklaráltam. Ekkor a függvény FU32-es bemenetén 0-tól 4-ig számokkal kiválasztható azon változók típusa, amelyek értékét törölni fogja, így az 1-essel a regiszterek nullázását írtam elő. A regisztereknek, amiket a folyamat során a memóriák indexelésére kívánok felhasználni, a soron következő műveletben adok kezdeti értéket. A bemenetek aktuális állapotának rögzítéséért felelős

változók sorszámát határozza meg az *R0*-ás regiszter, amelynek 1-es értékével azt írom elő, hogy a program futásának kezdetekor az *R1*-el kezdődjön a rögzítés. Az *R123*-as pedig az ott töltött időtartamot mentő regiszterek (amely a 124-essel kezdődik) indexéért felelős. *C123*-as számláló rögzíti az aktuális pozíciókhoz tartozó időtartamot. A program futásának zavartalansága érdekében a tanulási folyamatot külön lehet indítani a kijelzőről, valamint külön a szintetizátorról, mindkét gomb lenyomása esetén az időben hamarabb teljesülőnek eszköze fog rögzíteni. A program csak az újbóli indítás esetén tér vissza az init step-be, a működés során a 0-ásban tartózkodik, ha pedig véget ér a folyamat, akkor vált a step 2-be.

```

STEP 0
""A BILLENTYUK (IW1) AKTUÁLIS ÁLLAPOTÁNAK MENTÉSE REGISTEREKBE""
IF      I2.1      'TANUL
  AND    F0.09      'ENGEDÉLYEZŐ_TANUL
THEN    CFM 1      " Write operand indexed
  WITH    V4      " Operand 1=Flagword, 2=Input, 3=Output, 4=Register, 5=Timerpreselect
  WITH    R0      " Index of operand
  WITH    IW1      " New value
  INC     R0      'BEMENET_INDEX
  RESET   F0.09      'ENGEDÉLYEZŐ_TANUL

""BILLENTYU VÁLTOZÁSÁRA TOVÁBBLÉP, IDO MENTÉSE""
IF      I2.1      'TANUL
  AND    ( F0.15      'TRIG_START
  OR      F0.12      ) 'TRIG_NINCS
THEN    CFM 10      " Write operand indexed
  WITH    V4      " Operand 1=Flagword, 2=Input, 3=Output, 4=Register, 5=Timerpreselect
  WITH    R123      " Index of operand
  WITH    CW123      " New value
  SET     F0.09      'ENGEDÉLYEZŐ_TANUL
  INC     R123      'IDŐ_MENT_INDEX
  LOAD    V0
  TO      CW123      'IDŐ

""BLINKELÉS CIKLIKUS INDÍTÁSA""
IF      I2.1      'TANUL
THEN    CFM 0      " General blink bits
  LOAD    FU32      " 4 blink bits
  TO      FW200      'BLINK

IF      N I2.1      'TANUL
THEN    JMP TO 2

""IDO MÉRÉSE""
IF      F200.0      'BLINK 0.25s
THEN    INC     CW123      'IDŐ

""MEMÓRIA MEGHATÁROZÁSA""
IF      ( R0      'BEMENET_INDEX
  >=    V122      )
  OR      ( R123      'IDŐ_MENT_INDEX
  >=    V255      )
THEN    SET     F5.07      'MEMÓRIA_MEGTELT
  JMP TO 2
ELSE    JMP TO 0

STEP 2
IF      N I2.1      'TANUL
THEN    SET     F0.05      'TANUL PRG BEZÁRHATÓ

```

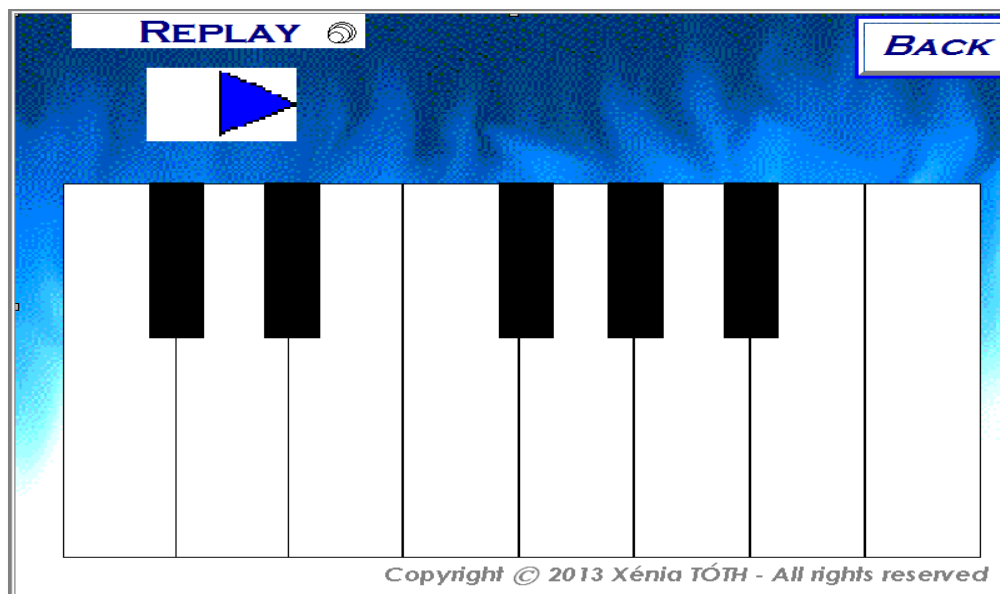
31. ábra: A tanulásért felelős programrészlet

A tanulási folyamatért felelős programrészlet (31. ábra) létrehozásához felhasználtam az *index_write* (Write operand indexed) nevű függvényt (ebben a programban *CFM 1*),

mely azt a célt szolgálja, hogy az általam megadott paraméterű változóba elmenti a kiválasztott értéket. Mivel a program kezdetekor logikai 1 értéket adtam a *F0.09* engedélyező bitnek, így az induláskor az első feltételbe való belépés adott. A függvény alkalmazásakor a *FU32*-es bemenetének 4-es értéket adva a regiszter, mint céltípus kerül megadásra, a *FU33*-as bemenetével kiválasztott *R0*-ás indexű (ami a program indításakor *R0=1*). Mindezek után menti az általam *FU34*-ben meghatározott értéket, vagyis a bemenetek aktuális állapotát (*IW1*), a kívánt helyre (*R1*-be). Ennek elvégzése után növeljük *R0* értékét, mivel ez jelöli azt, hogy hányadik helyre kívánunk beírni, majd ezután már az *R2*-be menthessük. A következő lépésben az engedélyező bit logikai értéke visszaáll 0-ra, hogy a bemenet változásának bekövetkeztéig ne történjen művelet. Ezzel egyidejűleg meghívásra kerül a *BLINK* (General blink bits) függvény (*CFM 0*), amely a *FW200* első 4 bitjét különböző frekvencián „villogtatja”, például az általam használt 0. bit 2 Hz-es frekvencián, vagyis 0.25 másodpercenként vált értéket logikai 1 és 0 között. Az előbbi teljesülésekor beérkező impulzusokkal *C123*-as számláló értéke nő. A bemenet változását kettő általam megírt alprogram figyeli, egyik a start parancsnak (*F0.15 TRIG_START*), másik az összes billentyű elengedésének (*F0.12 TRIG_NINCS*) felfutó élére vált értéket. Amennyiben ezek bármelyike teljesül a *CFM10*-es függvénnyel az előbbi módon mentem a soron következő (jelen esetben *R124*-es) regiszterbe a számláló értékét, illetve növelem az indexet. Az engedélyező bit aktív állapotba lépésével a megváltozott bemenet is mentésre kerül, a folyamat pedig mindaddig folytatódik, míg meg nem szakítom azt, vagy be nem telik a tároló memória, amiről a *F5.07*-es bit ad nekünk információt, ez utóbbi esetben a program addig várakozik feladat nélkül a 2-es lépésben, míg be nem zárom azt a tanulás funkció kikapcsolásával.

Az itt leírt folyamat a függvények használata nélkül csak nehézkesen, sorról-sorra, egyesével történő mentést tett volna lehetővé. Ezzel szemben a függvények alkalmazásával jócskán lerövidült a program, illetve felgyorsult a rögzítés menete!

4.2. Visszajátszási opció



32. ábra: A visszajátszást bemutató képernyő

A visszajátszásért felelős menüpontban láthatjuk a billentyűzet modelljét (32. ábra), ez már nem aktiválható érintésre, kizárólag demonstrációs funkciója van. A kék háromszöggel indítható a korábban megtanított dallam visszajátszása (ekkor az átvált négyzetre), amely során az aktuálisan megszólaltatott hanghoz tartozó billentyű szürkévé válik. Ezek a program szempontjából általam tetszőlegesen formázott LED-ek, amikhez a FW3 első 8 bitjét rendeltem. A **REPLAY** felirat melletti lámpa ad információt a lejátszott dallam befejeztéről, továbbá a folyamatot bármikor megszakíthatjuk a négyzet piktogram lenyomásával. A visszajátszás korlátlan számban áll rendelkezésünkre. A **Back** gombbal visszatérünk a fő képernyőre. A lejátszás mód csak akkor lehet aktív, ha a tanuló mód ki van kapcsolva, és fordítva.

```
"REGISZTER_INDEXÉRT FELELOS MEMÓRIÁK TÖRLÉSE"  
STEP INIT  
IF      ( N I2.1      ) 'TANUL  
      AND      I2.2      ) 'VISSZAJÁTSZÁS  
      OR      ( F5.04      ) 'TANUL_NEM  
      AND      F5.05      ) 'VISSZÁ_IGEN  
THEN LOAD V0  
      TO R0      'BEMENET_INDEX  
      LOAD V124  
      TO R123     'IDŐ_MENT_INDEX  
      RESET F0.06  'VISSZAJÁTSZÁS PRG BEZÁRHATÓ  
      SET F0.07    'ENGEDÉLYEZŐ_VISSZA  
      LOAD V0  
      TO CW122     'IDŐ_VISSZAJÁTSZÁS  
      JMP TO 0
```

33. ábra: A visszajátszás programhoz tartozó memóriák értékadása

A művelet indítása előtt fontos a kezdeti lépésben a tanítás funkcióban szereplő, itt is felhasznált indexelő memóriáknak a megfelelő érték adása (33. ábra).

```

STEP 0
""INDEX_READ FÜGGVÉNY MEGHÍVÁSA, ELSO ÁLLAPOT BETÖLTÉSE"
IF      ( I2.2      ) 'VISSZAJÁTSZÁS
  OR    F5.05      ) 'VISSZA_IGEN
  AND   F0.07      ) 'ENGEDÉLYEZŐ_VISSZA
THEN CFM 2          " Read operand indexed
  WITH V4          " Operand 1=Flagword, 2=Input, 3=Output, 4=Register, 5=Timerpreselect
  WITH R0          " Index of operand
  LOAD FU33        " Value read
  TO   FW4         'HMI ÉS VISSZAJÁTSZÁS ÁLLAPOT
  INC  R0          'BEMENET_INDEX
  RESET F0.07      'ENGEDÉLYEZŐ_VISSZA

""HA LETELT AZ IDO, A KÖVETKEZO ÁLLAPOTBAN TÖLTÖTT IDO BETÖLTÉSE A CW123-BA""
IF      I2.2      'VISSZAJÁTSZÁS
  OR    F5.05      'VISSZA_IGEN
  AND   ( CW122    'IDŐ_VISSZAJÁTSZÁS
    >=   CW123    ) 'IDŐ
THEN CFM 11        " Read operand indexed
  WITH V4          " Operand 1=Flagword, 2=Input, 3=Output, 4=Register, 5=Timerpreselect
  WITH R123        " Index of operand
  LOAD FU33        " Value read
  TO   CW123       'IDŐ
  SET  F0.07       'ENGEDÉLYEZŐ_VISSZA
  INC  R123        'IDŐ_MENT_INDEX
  LOAD V0          'IDŐ_VISSZAJÁTSZÁS
  TO   CW122

""BLINK FÜGGVÉNY MEGHÍVÁSA""
IF      I2.2      'VISSZAJÁTSZÁS
  OR    F5.05      'VISSZA_IGEN
THEN CFM 0        " General blink bits
  LOAD FU32        " 4 blink bits
  TO   FW200       'BLINK

""IDO MÉRÉSE""
IF      F200.0    'BLINK 0.25s
THEN INC CW122    'IDŐ_VISSZAJÁTSZÁS

""MEMÓRIA TÖLLÉPÉS ESETÉN KILÉP""
IF      ( R0      'BEMENET_INDEX
  >=     V122     )
  OR    ( R123    'IDŐ_MENT_INDEX
  >=     V255     )
THEN JMP TO 2
ELSE JMP TO 0

IF      N I2.2    'VISSZAJÁTSZÁS
  AND   N F5.05   'VISSZA_IGEN
THEN JMP TO 2

```

34. ábra: A visszajátszás program részlete

A 0. lépésben látható folyamat (34. ábra) első fontos állomása az index_read (index_read Read operand indexed) függvény (CFM 2) meghívása. Itt a korábbiakhoz hasonlóan meghatározzuk, hogy milyen típusú (a 4-es sorszámmal, ami itt a regisztereket jelöli), hányadik indexű (R0) változó értékét (FU33) akarjuk betölteni a meghatározott helyre (FW4), mindeközben növelve az R0 értékét, és inaktívvá téve az engedélyező bitet. A FW4 bitjei jelölik egyúttal a HMI bemeneteit is, a visszajátszáskor

tehát nem adtam meg új belső változókat a programban, hanem felhasználom az ekkor inaktívakat, hiszen a funkció működése során nincs engedélyezve a szabad játék. A program felépítéséből adódóan, ekkor úgy viselkedik, mintha a HMI-n a megfelelő értékek lennének egymás után lenyomva. Az logikai értékét mindaddig tartja, míg a C122-es, *BLINK* függvénnnyel növelt számláló értéke meg nem egyezik, vagy túl nem lépi a referenciának számító, aktuális pozícióhoz tartozó, FM 11-es függvénnnyel C123-ba betöltött állapot-tartózkodási idővel arányos értéket. Ez azt jelenti, amennyiben megfelelő ideig tartózkodott az aktuális állapotban, ez a feltétel teljesült, akkor betöltésre kerül a következő, illetve az ahhoz tartozó időbeni érték. A program bármikor megszakítható a visszajátszás kikapcsolásával, majd abba visszalépve újból lejátszható mindaddig, míg a tanulás funkció bekapcsolásával felül nem írjuk az addigi memóriát.

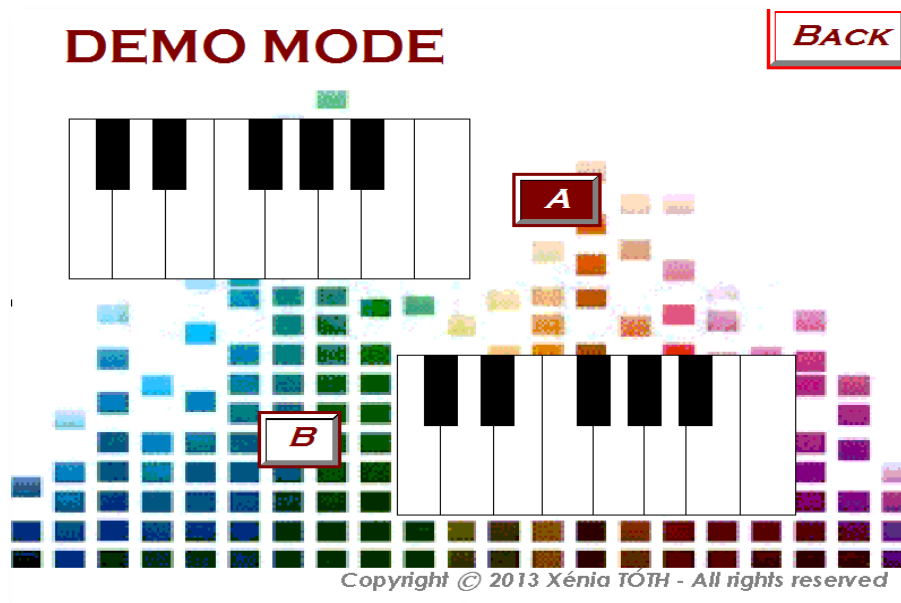
STEP 2				
IF			NOP	
THEN	SET		F5.09	'VISSZAJÁTSZÁS BEFEJEZŐDÖTT
IF		N	I2.2	'VISSZAJÁTSZÁS
	AND	N	F5.05	'VISSZA_IGEN
THEN	LOAD		V0	
	TO		FW4	'HMI ÉS VISSZAJÁTSZÁS ÁLLAPOT
	SET		F0.06	'VISSZAJÁTSZÁS PRG BEZÁRHATÓ
	RESET		F5.09	'VISSZAJÁTSZÁS BEFEJEZŐDÖTT

35. ábra: Visszajátszás bezárása

A lejátszás végeztével a program a STEP 2-ben várakozik a kikapcsolásra, ekkor az F5.09 bit tájékoztat bennünket a művelet befejezéséről (35. ábra).

A késleltető funkció programja az itt bemutatott elven épül fel, azzal a különbséggel, hogy ott a „visszajátszás” indítása nem nyomógombra történik, hanem a beállított idő leteltét követően automatikusan.

4.3. DEMO üzemmód



36. ábra: Demo mód

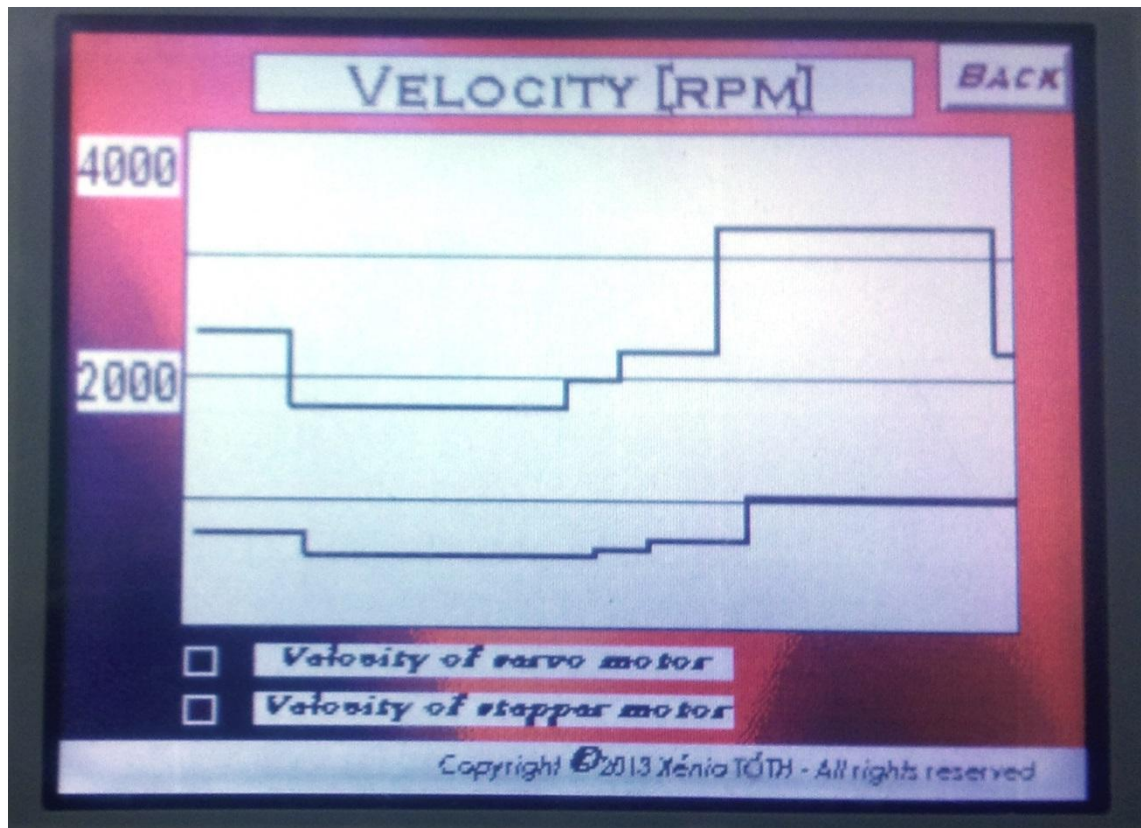
Beépítettem DEMO programként 2 tárolt dallamot, amik a megfelelő jelzésű (A vagy B) nyomógomb működtetett állapotában lejátszhatóak a hozzá tartozó kezelőfelületen (36. ábra), valamint figyelemmel kísérhetők a HMI virtuális billentyűin.

STEP 0	IF		F5.00	'A BEÉPÍTETT
	THEN	SET	F4.04	'G_HMI
		SET	T10	'DEMO
		WITH	0.6s	
STEP 1	IF	N	T10	'DEMO
	THEN	RESET	F4.04	'G_HMI
		SET	F4.07	'C'_HMI
		SET	T10	'DEMO
		WITH	0.6s	
STEP 2	IF	N	T10	'DEMO
	THEN	RESET	F4.07	'C'_HMI
		SET	F4.06	'H_HMI
		SET	T10	'DEMO
		WITH	0.6s	
STEP 3	IF	N	T10	'DEMO
	THEN	RESET	F4.06	'H_HMI
		SET	F4.05	'A_HMI
		SET	T10	'DEMO
		WITH	0.6s	

37. ábra: Demo programrészlet

Itt programon belül a kitartott hangok hosszát, illetve a köztük tartott szünetet az időzítők értékeinek megválasztásával tetszés szerint tudom változtatni (37. ábra).

4.4. Sebesség grafikonok

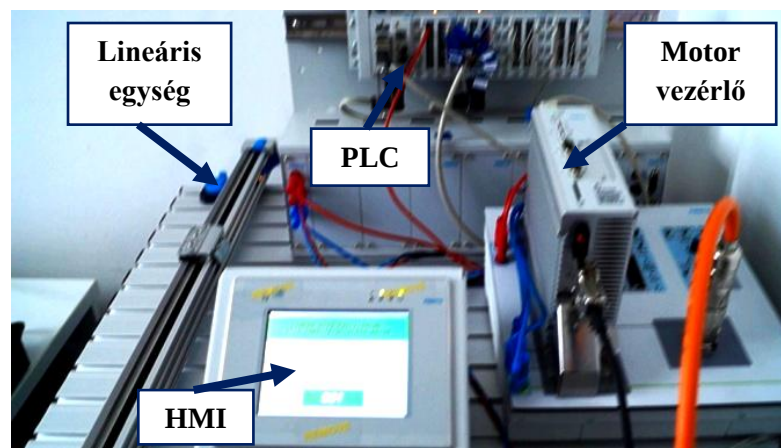


38. ábra: Motorok aktuális fordulatszámát megjelenítő grafikon működés közben

A két motor fordulatszámának értéke, azoknak időbeni változása, továbbá egymáshoz való viszonyuk olvasható le a HMI képernyőjén megjelenő grafikonokról (38. ábra). Látható, hogy a szervó motor (fent) magasabb fordulatszám tartományon dolgozik, míg a léptetőmotor létrejövő legnagyobb fordulatszáma sem közelíti meg azt. A grafikon horizontális tengelye 30 másodperc megfigyelését teszi lehetővé.

5. Fejlesztési lehetőségek

A munkám továbbfejlesztéseként létrehoztam egy hasonló rendszert, ugyanezt a szervó motort felhasználva és beépítve egy lineáris egységet FESTO EGC-50-600-TB-KF-GK (fogas szíjas, 600 mm-es munkazónával) pozicionálás céljából. A folyamatot hasonlóképp egy ipari kijelzővel lehet vezérelni, de itt a fordulatszámok megválasztása mellett kiválasztható a szán pozíciója a lineáris egységen. A mozgássorozat betanításán kívül, új pozíciók felvételére, és azok memóriába való eltárolására is van lehetőség.



39. ábra: A továbbfejlesztett rendszer lineáris egységgel

A szabad kézi mozgatást, és az új pozíciók felvételét a motorvezérlőben található különböző módok közötti váltással tudtam megvalósítani.

Tehát a két rendszer felépítése igen hasonló, annyi különbséggel, hogy ebben az esetben egy-egy hang megfelelője egy-egy pozíció. Szükség volt a vezérlő konfigurációjának specifikus beállítására, illetve a PLC program kismértékű módosítására, továbbá itt nagyobb jelentőséggel bírtak a fel illetve lefutási rámpák pontos beállításai. A lineáris egységből adódó további különbséget jelentette, illetve nagyobb jelentőséggel bírt a véghelyzetérzékelők használata, illetve minden indításkor a home pozíció felvétele [20]. A további fejlesztési lehetőségek közé tartozik a PLC memóriájának további egységekre osztása, hogy több betanított folyamatot tudjon tárolni a rendszer, a jelenlegi egy helyett. Másik fejlesztési lehetőség a motorok fordulatszám-tartományának 8 profilra való lekorlátozása helyett, az FHPP (Festo Handling and Positioning Profile) használata, ami akár az összes motorparaméter szabad hozzáférését, illetve módosítását is lehetővé teszi.

6. Összefoglalás

Célom egy olyan rendszer megvalósítása volt [19], melynek eszközei, részegységei az iparban is hasznosíthatóak. Ide tartozik például az általam létrehozott időben késleltetett indítással egy munkafolyamat megismételtetése. Mindezek mellett az ipari és a hétköznapi gépek felhasználóbaráttá tétele egyre jelentősebbé válik. Fontos, hogy ne csak az eszköz programozója, vagy az ezen tudással rendelkezők tudják a berendezést kezelni, hanem egy átlagos felhasználó is boldogulni tudjon a fontosabb folyamatokkal. Ezt a célt szolgálja a kijelző használata, annak könnyen átlátható kezelőfelülete, illetve az általam létrehozott program tanuló üzemmódja, hogy képes legyen az eszköz egy a felhasználó által meghatározott sorrendi folyamatot eltárolni a saját memóriájába. A tárhelyből kérésre, gombnyomásra előhívva kezelő nélkül önállóan is el tudja végezni a megjegyzett, betanított lépéseket, műveleteket.

Különleges példa a az ipari motorok ilyen célú alkalmazásával a NASA. A Curiosity nevű marsjárója, amely hangszórók híján a minták mozgatásához használt motorjaival játszotta le a Marson a Happy birthday dalt [17].

A folyamat kivitelezése során fontos szempont volt, hogy mindezt az alkalmazási módot saját elképzelés alapján, illetve az eszközöket nem feltétlenül a megszokott rendeltetésük szerint használjam fel. Így más megközelítésből is megismertem ezeket az ipari vezérlőket. A zene megvalósítása során felmerülő speciális problémák kihívás elé állították a programozói tudásomat is, ezek megoldása érdekében elsajátítottam az FST-ben a függvénymeghívás folyamatát is.

Az általam létrehozott program (vezérlés) segítségével bármely más, funkcióját és működését tekintve hasonló ipari környezetben alkalmazható, minimális, az adott feladatra jellemzőváltoztatással megvalósítható vele sorrendi vezérlés, szinkronizálás, mozgássor betanítása, és visszajátszása.

Irodalomjegyzék

- [1] http://ob121.com/bus_rs232.html
- [2] https://www.szabilinux.hu/linux_doksi/node76.htm.
- [3] http://www.festo.com/net/SupportPortal/Files/10272/CMMS-AS_ENUS.pdf
Letöltve: 2013.10.20.
- [4] http://www.festo.com/net/SupportPortal/Files/212956/13_1-18%20-%20Stepper%20motors%20EMMS-ST_ENUS.PDF
Letöltve: 2013.10.20.
- [5] <http://www.festo.com/net/SupportPortal/Files/52033/573125g1.pdf>
Letöltve: 2013.10.20.
- [6] <http://www.electroquip.co.uk/festo-servo-motor/festo-emms-as-55-s-tm-550108>
Letöltve: 2013.10.20.
- [7] http://eplandata.de/portal/en_US/part/FES.572986?cookieset=1
Letöltve: 2013.10.20.
- [8] http://www.festo.com/cat/en-us_us/data/doc_enus/PDF/US/EMMS-AS_ENUS.PDF Letöltve: 2013.10.22.
- [9] http://www.festo.com/net/SupportPortal/Files/10272/CMMS-AS_ENUS.pdf
Letöltve: 2013.10.22.
- [10] http://www.festo.com/cms/en-hk_hk/15648.htm Letöltve: 2013.10.22.
- [11] Ajtonyi István: PLC és SCADA-HMI rendszerek II. Ipari kommunikációs rendszerek II. 2. kötet. Miskolc, AUT-INFO Kiadó, 2008., 408 p
- [12] <http://www.festo.com/net/SupportPortal/Files/52613/705635d2.pdf>
- [13] Gyuricza István, Dr. Ajtonyi István: Programozható irányítóberendezések - Hálózatok és rendszerek - 10484 Műszaki Könyvkiadó, 2002
- [14] http://www.festo.com/cat/hu_hu/data/doc_engb/PDF/EN/CPX_EN.PDF Letöltve: 2013.10.22.
- [15] Tóth Xénia Erzsébet, Dr. Tóth János: Az ipari léptető, szervó motorok újszerű felhasználása, XIX. FMTÜ konferencia, Kolozsvár, 2014.
- [16] Bolla Gyula - PLC alapismeretek: tanfolyami jegyzet, E311 számú tankönyv, Festo Kft. (2010)
- [17] http://index.hu/tudomany/2013/08/06/boldog_szuletesnapot_kivant_maganak_a_curiosity/ Letöltve: 2013.11.12.
- [18] http://www.cs.elte.hu/blobs/diplomamunkak/mattan/2010/lanczendorfer_orsolya.pdf
- [19] Zene szervó és léptető motor segítségével video:
<https://www.youtube.com/watch?v=zJMtiYr7zX4>
- [20] Linear Axis positioning with industrial servo motor video:
<http://www.youtube.com/watch?v=c2RhXUsSxtE>

Köszönetnyilvánítás

*Ezúton szeretnék köszönetet mondani **Dr. Tóth János** Tanár Úrnak szakmai ötleteiért, segítőkészségéért, és építő tanácsaiért, amelyekkel hozzájárult a dolgozatom elkészüléséhez, valamint az eszközök rendelkezésemre bocsátásáért.*