SZAKDOLGOZAT

Czugh Soma

2023

Szent István Campus, Gödöllő
Cím: 2100 Gödöllő, Páter Károly utca 1.
Tel.: +36-28/522-000
Honlap: https://godollo.uni-mate.hu

Magyar Agrár- és Élettudományi Egyetem
Szent István Campus
Mechatronikai mérnök Szak

Robottal támogatott adagoló állomás PLC programozása és távvezérlésére alkalmas rendszer tervezése

Belső konzulens: Mayerné Dr. Sárközi Eszter
egyetemi adjunktus

Külső konzulens: Juhász András Szilveszter
AV mérnök

Készítette: **Czugh Soma**
EMS3KB
nappali tagozat

Intézet/Tanszék: Gépipari Technológiai Intézet/ Mechatronika tanszék

Gödöllő
2023

MAGYAR AGRÁR- ÉS ÉLETTUDOMÁNYI EGYETEM

Szent István Campus, Gödöllő
Cím: 2100 Gödöllő, Páter Károly utca 1.
Tel.: +36-28/522-000
Honlap: https://godollo.uni-mate.hu

MAGYAR AGRÁR- ÉS ÉLETTUDOMÁNYI EGYETEM

Szent István Campus, Gödöllő
Cím: 2100 Gödöllő, Páter Károly utca 1.
Tel.: +36-28/522-000
Honlap: https://godollo.uni-mate.hu

MŰSZAKI INTÉZET
MECHATRONIKAI MÉRNÖK ALAPSZAK
Gépipari mechatronika specializáció

SZAKDOLGOZAT

feladatlap

Czugh Soma (EMS3KB)

részére

A szakdolgozat címe:

Robottal támogatott adagoló állomás PLC programozása és távvezérlésére alkalmas rendszer tervezése

Feladatkiírás:

Bevezetés, szakirodalom feldolgozás, probléma bemutatás, Robottal támogatott adagoló állomás vezérlése és távvezérlése: az adagoló állomás vezérlésének kidolgozása, robotprogramozás, távvezérlési rendszer kialakítása, Továbblépés lehetőségei, gazdasági számítás, összefoglalás

Közreműködő tanszék: Mechatronika

Külső konzulens: *Juhász András Szilveszter, AV mérnök, Streamnet Zrt.*

Belső konzulens: *Mayerné Dr. Sárközi Eszter, egyetemi adjunktus,* MATE, Műszaki Intézet

Beadási határidő: 2023. november 06.

Gödöllő, 2023. szeptember 04.

Jóváhagyom		**Átvettem**
(tanszékvezető)	(szakfelelős)	(hallgató)

A dolgozat készítőjének külső konzulense nyilatkozom arról, hogy a hallgató az előre egyeztetett konzultációkon megjelent.

Gödöllő, 2023. 11. hó 04. nap

(külső konzulens)

Tartalom

1. Bevezetés 6
1.1. Téma jelentősége 6
1.2. Célkitűzés 6
2. Szakirodalom feldolgozása 7
2.1. Irányitástechnika 7
2.1.1. PLC tulajdonságai 8
2.1.2. PLC funkciója 9
2.1.3. PLC Programnyelvek 10
2.2. Pneumatika 11
2.2.1. Pneumatika főbb jellemzői 11
2.2.2. A sűrített levegővel működtetett eszközök 12
2.2.3. Pneumatika előnyei, hátrányai 14
2.3. Robotika 17
2.3.1. Robotkarok felépítése 17
2.3.2. Ipari robotok 19
2.4. HTML Struktúrájú weblapok 20
2.4.1. HTML weblapok felépítése 21
2.4.2. Statikus webszerverek, FTP típusú honlapok 22
2.4.3. JavaScript Bemutatása 23
3. Robottal támogatott adagoló állomás PLC programozása és távvezérlésére alkalmas rendszer tervezése 24
3.1. MPS Adagoló állomás 25
3.1.1. Az állomás működésének folyamata 26
3.2. CPX PLC 28
3.2.1. Feladatmegoldás 28

3.2.2. PLC Program működése 30

3.2.3. PLC Programok 31

3.3. Robotkar 36

3.3.1. Robotkar felépítése 36

3.3.2. Feladatmegoldás 36

3.3.3. Robotprogram 37

3.4. Webszerver funkciói 39

3.4.1. Webszerver konfigurálása 39

3.4.2. Weblap felépítése 40

3.4.3. Weblap jogosultságok 40

3.4.4. A jogosultságokhoz tartozó oldalak 42

4. Továbblépési lehetőségek 48

5. Gazdasági számítás 49

6. Összefoglalás 50

7. Summary 51

8. Hivatkozások 52

9. Mellékletek 53

1. Bevezetés

Ebben a fejezetben bemutatom a dolgozatom alapvető információit, jelentőségét, illetve, hogy milyen célkitűzéseim vannak a feladattal kapcsolatban.

1.1. Téma jelentősége

A különböző iparok és az őket működtető technológiák is rohamosan fejlődnek. Erre nagyon jó példák az ipari forradalmak. A modern gyártástechnológia elengedhetetlen része azonban kis túlzással az automatizálás lett. Elképzelhetetlen lenne ma egy gyárat vagy akár egy kisebb folyamatot is látni, ami ne lenne automatizálva. Pont ezért forradalmi újítás volt, amikor Richard Morley 1969-ben megépítette az első programozható logikai vezérlőt, más néven a PLC-t. Elég csak akár a vasúti váltokra gondolni, ahol hatalmas huzalozott relés logikai termek voltak kialakítva és ezt egy hozzá képest apró doboz az idő múlásával letudta váltani. De nem csak a helyigény volt az egyetlen szempont, hanem sikerült létrehozni egy egyszerű és gyors programozhatóságú, szabványosított eszközt, ami nagyon gyorsan teret hódított magának.

A mai világban azonban szintén célszerű egy távfelügyeletre, távvezérlésre alkalmas rendszert is létrehozni, hiszen az ember már nincsen helyhez kötve, így akár egy PLC programozónak akár egy folyamatirányító mérnöknek célszerű, ha van egy távoli hozzáférése az adott eszközhöz, így még gyorsabban is tud hibát elhárítani vagy akár az adatokat kezelni. És ezt könnyedén meg is valósítható, hiszen a legtöbb PLC már tud Etherneten kommunikálni az internet segítségével, és így egy egyszerű HTML struktúrájú weblap létrehozásával bárhonnan elérhető lesz a PLC.

1.2. Célkitűzés

Dolgozatom fő célja, hogy a Mechatronika Tanszéken lévő MPS adagoló állomás és Mitsubishi RV-2AJ robotkar munkáját összehangoljam. Ennek érdekében létrehozom a PLC vezérlést és a robotvezérlőnek a programját. A folyamatban lezajló eseményeket egy webszerver segítségével monitorozom és az adagoló állomás távvezérlését egy honlap segítségével valósítom meg.

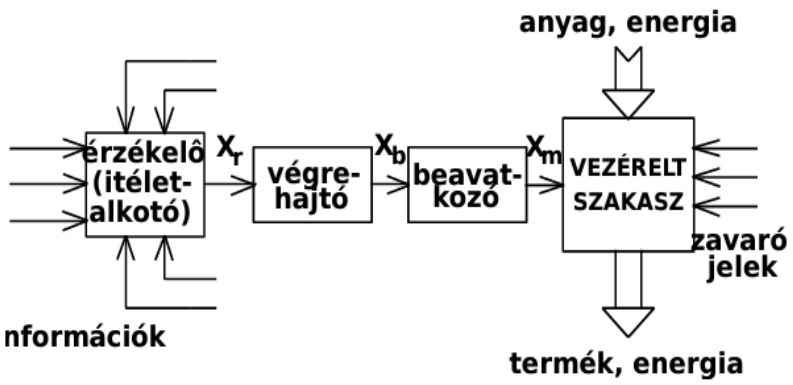
2. Szakirodalom feldolgozása

Ebben a fejezetben szeretném ismertetni azokat témakörökből kiragadott részeket, amik elengedhetetlenek a dolgozatom ismeretéhez, ha meg szeretnénk érteni, hogy az egyes eszközök milyen folyamatokat látnak el, hogyan működnek és miért az adott technológiát használjuk.

2.1. Irányitástechnika

A különböző folyamatok irányítási problémáinak megoldása manapság az egyik legfontosabb kérdés, függetlenül attól, hogy milyen folyamatról beszélünk. Minden folyamatirányítási feladatnál felmerülnek tervezési, elemzési és az üzemeltetési, avagy irányítási feladatok. Ezek során elkerülhetetlenül találkozunk a rendszerek tényleges folyamatirányítási problémáival.

Esetemben egy fontos dolgot kell tisztázni, az pedig a vezérlés. Alábbi ábrán látható a vezérlésnek a nyitott hatáslánca.



1.ábra A vezérlés hatáslánca [1]

A vezérlésre továbbá jellemző, hogy a rendelkező jel kiváltására csak a begyűjtött információk szolgálnak, a berendezés állapotától függ a vezérlési művelet eredménye. A rendelkező jel a környezetből és folyamatokból vett információk feldolgozása hatására jön létre. [1]

2.1.1. PLC tulajdonságai

Az 1970-es években megjelentek a mikroprocesszorok, és ennek hála az irányitási folyamatok felügyelete a huzalozott, hardveres megoldásról áttérhetett a programmal, szoftverrel történő irányításra. Habár eleinte és manapság is egy apró árnyoldala van ennek, hogy míg régebben egy hardveres hibával, vagyis a hibalehetőséggel, százalékkal, tudtak számolni és egész jó valószínűséget lehetett számolni az a szoftveres oldalon nincs így.

A PLC-kel kapcsolatok legfontosabb jellemzők az 1980-as évek közepére kialakultak. A fejlődés azonban azóta sem állt meg, sőt rohamosan fejlődött a technológia, míg az ára elképesztően zuhant. Szabványosították a PLC-ket mind kialakításban, mind pedig a programnyelvükben.

Alapvetően a PLC-ket két nagy csoportba sorolhatjuk:

- Moduláris PLC: ahogy a neve is sugallja ez egy moduláris rendszer, egy alaplabda dugaszolható modulokból áll. Tápegységből, központi egységből (itt található a processzora), kommunikációs modulból, digitális és analóg I/O modulokból és szabályozási feladatokat ellátó modulokból. Természetesen az itt felsorolt modulokat a felhasználó szabadon összeválogathatja, hogy melyikből milyen típusra is van szüksége. És szabadon bővíthető. A programozószoftvert is meg lehet választani. A moduláris PLC-ket leginkább közepes és nagyméretű folyamatokhoz, gyártórendszerekhez fejlesztették ki.
- Kompakt PLC: Egy dobozba szerelt, általában külső tápegységről működő szerkezet. Szabványosan általában 24 Volt feszültséget és egyenáramú tápot igényel. A dobozban megtalálható a processzor, a memória, a kommunikációs csatornák (RS-232 – pont-pont kapcsolat, RS-485/422, ami egy többpontos kapcsolat, lehet USB vagy ami manapság a legelterjedtebb az az Ethernet).
  Itt van lehetőség egy minimális bővítésre, azonban általában az inputok és az outputok számát tudjuk növelni. A kompakt PLC-k közé tartoznak a Mikro PLC-k is, vagy másnéven az intelligens relék, ezek nem bővíthetőek azonban a programozásuk általában egy egyszerű grafikus felületen történik.[2]

2.1.2. PLC funkciója

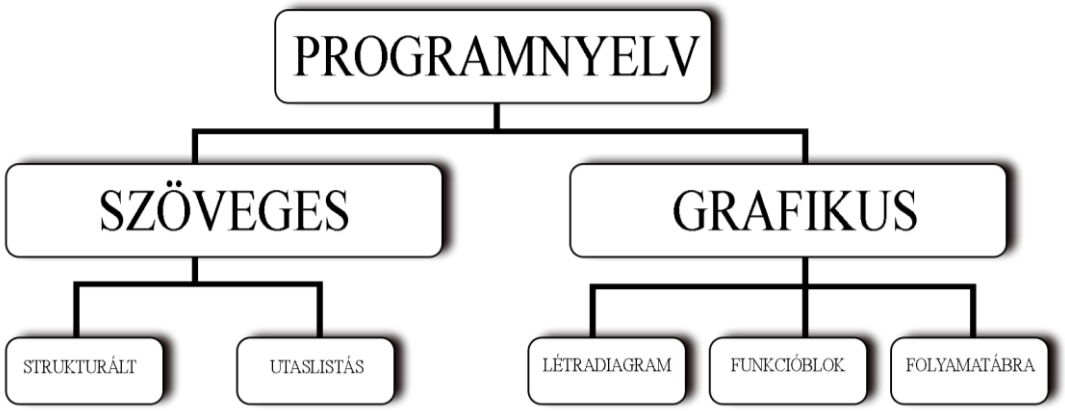
A PLC-k fő célja az irányított folyamatok felügyelete. I/O csatornákon keresztül tudja vezérelni a folyamatot. Emellett kommunikálniuk tud az alrendszerrel, beleértve a távoli perifériákat és a beágyazott vezérlőket, valamint más PLC-kel és a felügyelő számítógépekkel (PLCPC) és a kezelőszemélyzettel (PLC-HMI) is. Régebben a pont-pont összeköttetések voltak jellemzőek, de a modern rendszerek hálózati kommunikációt alkalmaznak. [2]

A PLC-k feladata nemcsak a folyamat üzemeltetése, hanem az irányítás különböző aspektusaira is kiterjed. Ez magában foglalja a rendszer (berendezés, gép) indítását, feltételeinek ellenőrzését (pl. reteszelés), a leállítási folyamatok kezelését (pl. hűtés biztosítása), rendkívüli események reakcióit (pl. vész-stop, védelmi funkciók), hibakódok generálását, üzemóra-számlálást és dokumentálást. Ezeket a feladatokat folyamatosan kell ellátniuk, de vannak időszakos feladatok is, mint például a gépek beüzemelésekor, karbantartásakor vagy meghibásodásakor szükséges kézi-automata-lépés üzemmódok, diagnosztikai és tesztelési funkciók. [2]

A PLC-vel kapcsolatban az egyik legfontosabb dolog, hogy valós időben működjön. Azokat a rendszereket, amelyeknek az időzítése szorosan kötődik a környezeti és valós idejű skálához, valós idejű rendszereknek nevezzük. A valós idejű rendszerek két fő típusa van: a hard-real-time rendszerek, amelyek szigorú időkövetelményeket teljesítenek, biztosítják, hogy a kritikus feladatokat mindig időben befejezzék. A soft-real-time rendszerek csak azt garantálják, hogy a kritikus feladatok prioritással fussanak, időzítési rugalmasságot biztosítva. A PLC-k esetében meghatározzuk, hogy a rendszer milyen gyorsan reagáljon környezeti eseményekre, vagy milyen időzített műveleteket hajtson végre. A modern PLC-kben a valós idejű működést a SCAN ciklus biztosítja. A PLC folyamatosan ismétlődő ciklusokban működik. [2]

2.1.3. PLC Programnyelvek

Az első PLC-k bevezetésekor az automatizálásban áramköri tervekkel dokumentálták a vezérlőáramköröket. A PLC programozásához létradiagramokat használtak, melyek a kezdetleges programozókészülékeken csak képernyőn voltak elérhetők. Az utasítás-listás nyelv volt az elfogadott módszer a egyszerűbb programozókészülékeken. Később, amikor a programozás átkerült a számítógépekre, a funkcionális blokkok segítségével történő programozás vált népszerűvé. [3]



2. ábra PLC Programnyelvek csoportosítása

A különböző programnyelvek mindegyike alkalmas arra, hogy vezérlési feladatokat leírjon. A programozók általában a saját jártasságuk alapján választanak programnyelvet, és nem minden gyártó támogatja az összes nyelvet. Azért, hogy megkönnyítsék a felhasználók munkáját, azon gondolkodnak, hogy szabványosítsák a programnyelveket. Az IEC 1131-3 szabvány nem új nyelveket vezetett be, hanem a már létezőket foglalta egybe egy egységes rendszerben, egységes struktúrával és jelölésrendszerrel. A legtöbb gyártó ma már lehetővé teszi az e szabvány szerinti programozást, miközben megőrzi saját rendszerét, és alternatív megoldásokat is kínál. Ez segíti az irányítástechnikai szakembereket a munkájukban, mivel a kidolgozott megoldásaikat könnyen alkalmazhatják különböző hardvereken. [2]

Az utasításlistás programozási nyelv szorosan kapcsolódik a PLC-k kialakulásához. Az eredeti PLC-k egyszerű bitszervezésűek voltak, ahol az utasítások és adatok egy memóriacímre voltak tárolva, és ezeket kellett meghívni a programban. A PLC program egy sorozat programvezérlési utasításból áll. Minden utasítás két részből tevődik össze: először a

műveleti részből (mit kell csinálni?), majd az operandus részből (mivel kell csinálni?). A műveleti részben határozzuk meg, hogy a processzornak milyen műveletet kell végrehajtania, például adat betöltése (LD - Load). Az operandus részben azt határozzuk meg, hogy melyik bemeneti érték vagy változó felhasználásával kell elvégezni a műveletet. Emellett lehetőség van kiegészítő részek hozzáadására is, mint például a negáció (N) és zárójelek használata. [3]

2.2. Pneumatika

Ebben a részben összefoglalom a pneumatika főbb tulajdonságait, a sűrített levegővel működő eszközök jellemzőit, illetve, hogy milyen előnyei és hátrányai vannak egy ilyen rendszernek.

2.2.1. Pneumatika főbb jellemzői

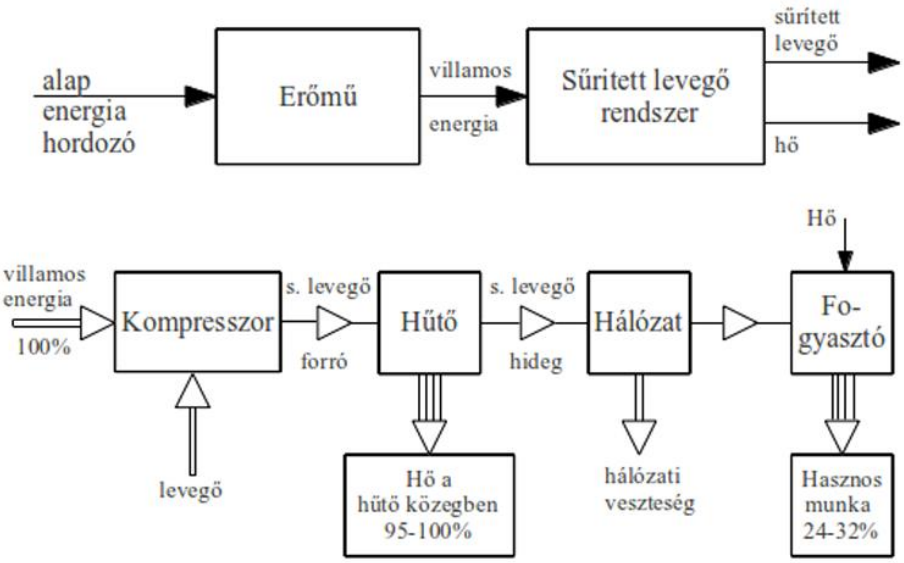
A pneumatika nagyon vonzó a termeléssel foglalkozó szakemberek számára. Ez azért van, mert olyan eszközöket kínál, amelyek képesek pótolni a munkásokat a legegyszerűbb módon. A pneumatika lehetővé teszi a mozgatást, szállítást, emelést, és mindezt gyorsan és biztonságosan. Ráadásul nem kényes a környezeti viszonyokra, így ideális megoldást nyújt az üzemekben és kültéri környezetekben egyaránt. Ezen kívül olyan helyzetekben, ahol az elektronikus rendszerek nem alkalmazhatók (például robbanásveszélyes vegyipari üzemekben vagy bányákban), általában csak pneumatikus és hidraulikus vezérléseket találunk.[4]

A sűrített levegő előállítására általában kompresszorokat, vagy más néven légsűrítőket alkalmaznak. Ezek a berendezések a levegőt a kívánt nyomásértékre préselik össze. A mechanikus energiát, amit ez a folyamat generál, pneumatikus energiává alakítják át. A kompresszorok működtetéséhez általában elektromos motorokat használnak a rögzített rendszerekben, míg a mobil rendszerekben belső égésű motorokat alkalmaznak.

A pneumatikus energia, vagyis a táplevegő, csővezetékhálózaton keresztül jut el a fogyasztóhoz. Fontos megjegyezni, hogy a pneumatikus berendezések rendkívül érzékenyek a szennyeződésekre (pl. kopás) és a nedvességre (pl. korrózió), ezért kiemelten fontos a táplevegő szűrése, tisztítása és páramentesítése.

A kompresszorok légszállításának a mindenkori igényekhez történő illesztése miatt szükség van a szállítás szabályozására. Ezt többféle módon lehet megtenni, az egyik megoldás a

légszállítást (térfogatáramot) a beállítható minimális és maximális nyomáshatárok között változtatni lehet. [4]



3. ábra Sűrített levegő előállításának folyamat ábrája [5]

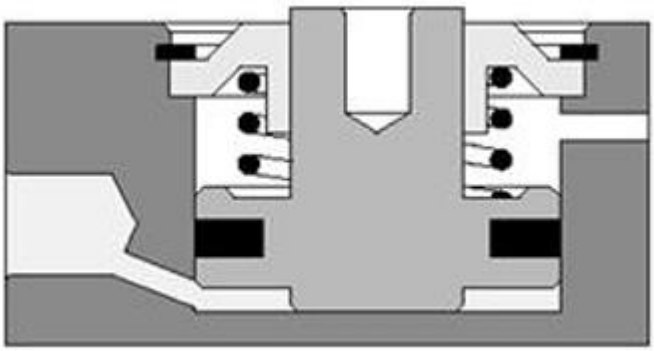
Az előállított sűrített levegő minősége komoly problémát jelent az iparban. A szakemberek rendszeresen szembesülnek azzal, hogy az olajos, vizes vagy szilárd szennyezőanyagokkal szennyezett sűrített levegő milyen komoly problémákat okozhat. A rossz minőségű sűrített levegő előállítása nemcsak energiapazarláshoz vezet, hanem károsíthatja a pneumatikus és végrehajtó elemeket is. Ez termelésleállást és kiesést okozhat, emellett pedig növeli a karbantartási és alkatrészigényt is. Ezért kiemelten fontos, hogy az ipari folyamatok során minőségi sűrített levegőt használjunk annak érdekében, hogy optimalizáljuk a termelést és minimalizáljuk a költségeket. [5]

2.2.2. A sűrített levegővel működtetett eszközök

Azok az eszközök, amelyek átalakítják a pneumatikus energiát mechanikus energiává két fő kategóriába sorolhatók: vannak olyan elemek, amelyek forgó mozgást hajtanak végre, mint például a légmotorok és a forgatóhengerek, illetve olyanok, amelyek lineáris mozgást tesznek lehetővé, mint a pneumatikus munkahengerek. A mozgást, amelyet ezek a végrehajtó

elemek elősegítenek, két fő jellemzővel lehet leírni, egyrészt a mozgás során kifejtett erővel, másrészt a mozgás sebességével. [2]

A pneumatikus munkahengerek egyenes vonalú mozgást valósítanak meg. Szerkezetüket tekintve több főbb típust különböztethetünk meg. Az egyszeres működésű munkahengerek esetében a dugattyúoldalon pneumatikus táplálást alkalmaznak, ami azt jelenti, hogy csak egy irányban (pozitív irányban) tudnak mozogni és munkát végezni. A dugattyúrúd visszatérítését (negatív irányú mozgás) egy beépített rugó biztosítja. Ezek a munkahengerek általában rövid löketűek, legfeljebb 100 mm lökethosszig, mivel a beépített rugó hossza korlátozza a löket hosszát. Ezeket az egyszerű működésű munkahengereket általában kilökésre, sajtolásra, emelésre és adagolásra használják az iparban. [2]

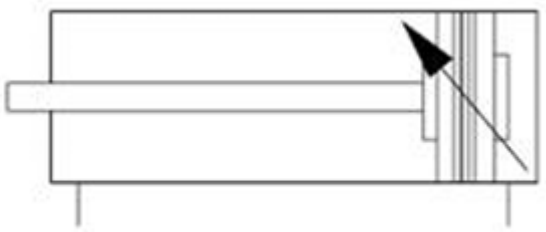


4. ábra Egyszeres működtetésű munkahenger [3]

A kettős működtetésű munkahenger olyan eszköz, amely mindkét dugattyúoldalán rendelkezik tápcsatlakozással. Ennek köszönhetően a levegő energiája lehetővé teszi a munkahenger dugattyújának mozgatását mindkét irányban. Fontos megjegyezni, hogy a kifejtett erő kisebb a dugattyúrúd oldalán, mivel a dugattyúrúd keresztmetszetét le kell vonni a hatásos felületből. A kettős működtetésű hengereket olyan helyeken alkalmazzák, ahol a munkahengernek a negatív mozgása során is szükséges munkát végeznie. Ezáltal hatékonyan kihasználhatják az energiát mindkét irányban történő mozgás során.

A dugattyú merev ütközése a hengerfedélen kárt okozhat. Ezt elkerülve a löketvégbe csillapítást építenek be. Mielőtt a dugattyú elérné a végső pozícióját, egy fékdugattyú lezárja a hengertért, így a levegő nem szabadon áramolhat ki. Ehelyett csak a hengerfedélben található fojtó-visszacsapó szelepen keresztül tud tovább haladni, melynek átmérője

változtatható. Ha nagy tömegeket kell mozgatni, akkor a löketvégi csillapítást külön beépített csillapítóelemmel kell kiegészíteni. [3]



5. ábra Munkahenger löketvégi csillapítással [3]

Az útszelepek fontos feladata, a táplevegő áramlasi irányának állítása, nyitása, zárása. Ezeket a szelepeket ábrázolásuk során szabványos jelekkel és nevekkel látjuk el, ami segít azonosítani a szelep csatlakozóit és pozícióit, valamint azt, hogy hogyan működnek.

Minden szelepnek legalább 2 csatlakozóval kell rendelkeznie - egy bemenettel és egy kimenettel. A szelepek rendszerint 2, 3, 4 vagy 5 csatlakozással ellátottak, a vezérlő és elővezérlő csatlakozásokat nem számítva. A csatlakozások közötti kapcsolatokat nyilak és lezárók ábrázolják a megfelelő jelképes négyzetekben.

A szelep állásainak vagy állapotainak legalább 2 lehetőségük van. A szelepek általában 2 vagy 3 állásúak. [3]

2.2.3. Pneumatika előnyei, hátrányai

A levegő kis viszkozitású, kis súrlódású, ezért nagy sebességgel és hosszú vezetékekben vezethető. Természetes, hogy a technológia folyamatos fejlődésével az ipari folyamatok optimalizálásának és hatékonyságának növelése érdekében új és innovatív megoldásokat kell keresnünk. Ebben a kontextusban a pneumatika kiemelkedő szerepet játszik az ipari automatizálásban és gyártásban. A pneumatika az ipari rendszerek tervezésének és üzemeltetésének számos előnyös aspektusát kínálja.

Az egyik alapvető előnye a pneumatikának az elérhetőség és szállíthatóság képessége. A levegő gyakorlatilag mindenütt korlátlan mennyiségben elérhető, így könnyen hozzáférhetünk a szükséges energiához. Emellett a sűrített levegő könnyen szállítható csővezetékeken nagy távolságokra is, és mivel kicsi a viszkozitása, minimális energiaveszteséggel. A pneumatika rugalmas tárolási lehetőséget is biztosít. A sűrített levegőt tartályokban tárolhatjuk, így nem szükséges, hogy a kompresszor állandóan üzemeljen. Ez nem csak az energiafogyasztást csökkenti, hanem hosszú távon is megkönnyíti a rendszer karbantartását és kezelését. Ezenkívül a pneumatika kiemelkedően jól tűri a hőmérsékleti változásokat. A sűrített levegő nem érzékeny a hőmérséklet-ingadozásokra, ami lehetővé teszi speciális hőmérsékleti körülmények közötti biztonságos alkalmazását.

A pneumatika rendkívül biztonságos módja az energiatovábbításnak. A sűrített levegő robbanás- és tűzbiztos, így nincs szükség drága biztonsági berendezések alkalmazására. Ezenkívül a sűrített levegő tisztasága is kiemelkedő. A levegőtömítésű vezetékeknél és elemeknél nem fordulhat elő szennyeződés, ami különösen fontos az élelmiszeriparban és más tiszta környezetet igénylő iparágakban. Az előnyök között szerepel a pneumatikus rendszerek egyszerű felépítése és olcsó fenntartása. A pneumatikus motoroknak nagyobb energiasűrűsége van, mint a villamos motoroknak, ami például kézi szerszámok esetén nagy előnyt jelent. A pneumatika lehetővé teszi a jellemzők könnyű irányítását is. A sebesség fokozatmentesen vezérelhető, és az erő kifejtése fokozatmentesen állítható. Emellett a pneumatikus készülékek túlterhelhetők anélkül, hogy meghibásodásveszély állna fenn, és a lineáris mozgások egyszerűen megvalósíthatók, ami nagy megbízhatóságot eredményez.

Végül, a pneumatikus elemek korszerű kialakítása lehetővé teszi, hogy "életre szóló" kenéssel legyenek ellátva, így minimális karbantartást igényelnek. [2]

A sűrített levegő felhasználása számos területen elterjedt, azonban hatékony működésének biztosítása számos kritikus tényezőtől függ. Az előkészítés során kiemelten fontos a levegő minősége, hiszen az energiahordozónak semmilyen szennyeződést vagy nedvességet nem szabad tartalmaznia. Ezeknek az elemeknek a leválasztása nélkülözhetetlen az eszközök hosszú élettartamának biztosítása érdekében.

Egy másik fontos szempont az összenyomhatóság, ami azt jelenti, hogy a sűrített levegővel működő rendszerek nem képesek egyenletes dugattyúsebességet biztosítani terhelésfüggetlenül. Az erőkifejtés tekintetében is megvannak a korlátai ennek a

technológiának. A sűrített levegő gazdaságosan csak egy meghatározott erőkifejtésig használható. Az optimális üzemi nyomás mellett (700 kPa, 7 bar) a határterhelés 20.000–30.000 N, 2000–3000 kp körül van, figyelembe véve a dugattyúsebességet és a lökettől függő tényezőket. Az eszközök hatékonysága mellett a költségek is fontos szempontot jelentenek. A sűrített levegő viszonylag drága energiahordozó, ami megfontolásra készteti a felhasználókat.

Ezen túlmenően, a sűrített levegő használata során felmerülő zajproblémákat is kezelni kell. A kipufogó levegő jelentős zajt okoz, azonban a modern hangtompítási megoldások segítségével sikerült ezt a problémát kezelhető szintre csökkenteni. Ezenkívül, a kipufogott levegő energiája is elveszik, azonban résztöltéssel és azt követő expandálással ennek az energiaveszteségnek a mértékét csökkenteni lehet, ami hozzájárulhat a rendszer hatékonyságának növeléséhez. [2]

2.3. Robotika

A robot fogalmát a Nemzetközi Szabványügyi Hivatal (ISO „Manipulating Robots", ISO8373:1996) a következő módon állapítja meg:

„A robot automatikusan vezérelt, újra programozható, három vagy több tengelyű mozgásra és sokoldalú beavatkozásra képes eszköz. A robot lehet rögzített vagy mozgó eszköz".[6]

A modern ipari robotok története az 1930-as évekig nyúlik vissza, amikor az első ipari robotokkal kapcsolatos szabadalmak születtek. Azonban az igazi áttörés csak évtizedekkel később, 1961-ben következett be, amikor a General Motors New Jersey-i gyárában üzembe helyezték az első valóban működő robotot, az Unimate-ot. Azóta a robottechnológia hatalmas fejlődésen ment keresztül, és napjainkban számos ipari területen megtalálhatóak, beleértve a szerelést, a képlékeny alakítást és a minőségellenőrzést is.

Az ipari robotokat az ISO-8373-2021 (2021) nemzetközi szabvány határozza meg, mint olyan önműködő vezérlésű, programozható, többcélú manipulátorokat, amelyek három vagy több vezérelt tengellyel rendelkeznek. A robottechnika interdiszciplináris jellegű, magában foglalva a mechanizmusok klasszikus elméletét, valamint a hidraulika, pneumatika, elektronika, irányítás és számítástechnika területeit.

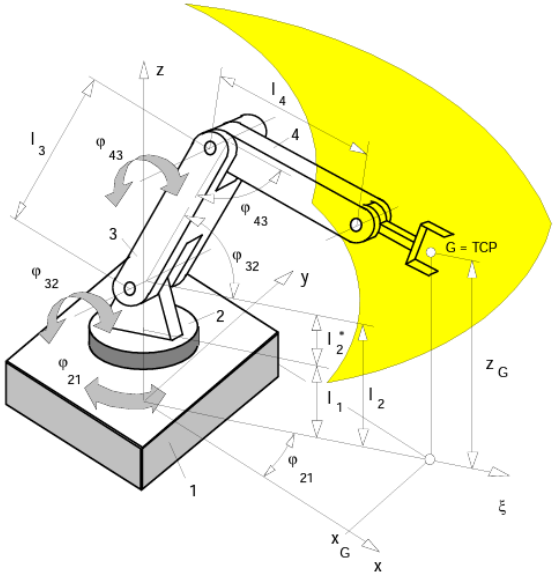
Az intelligens robotok képesek érzékelni környezetük változásait és maguk határozzák meg munkafeladataikat és a feladatok megoldásának módját. Ehhez mesterséges érzékszerveket használnak, például látó modulokat, amelyek optikai információkat feldolgozva csoportban együttműködve is képesek tevékenykedni. A robotok ezen képességei révén az ipari folyamatok hatékonyabbá és rugalmasabbá válnak, és fontos szerepet játszanak a modern műszaki tudományok területén. [7]

2.3.1. Robotkarok felépítése

A csuklókaros robotok tervezésében alkalmazott kinematikai kényszerek elengedhetetlenek a robotok mozgásának és funkcionalitásának biztosításában. Ezek a kényszerek lehetővé teszik a robotkarok rotációs mozgását, ami szükséges azoknak a feladatoknak a végrehajtásához, melyeket a robotoknak képesnek kell lenniük elvégezniük.

Ezek közül a kinematikai kényszerek közül kettő olyan tengelyt foglal magába, melyek egymásra merőlegesek és metszik egymást. Ez az elrendezés lehetővé teszi a robotkarok komplex mozgását a térben, és lehetővé teszi számukra, hogy különböző irányokba mozduljanak. A harmadik kényszer tengelye párhuzamos az előző kettő valamelyikével, ami további mozgási lehetőségeket biztosít a robotkaroknak.

Ezen kényszerek alapján két fő típusú csuklókaros robotot lehet megkülönböztetni. Az egyik típus a függőleges síkú csuklókaros robot, melynek mozgási tartománya főként felfelé és lefelé irányul, lehetővé téve például emelési vagy leengedési feladatok elvégzését. A másik típus a vízszintes síkú csuklókaros robot, mely a földrajzi síkhoz képest vízszintesen mozog. Ennek a típusnak a tervezése és felhasználása olyan feladatokban lehet hasznos, ahol a mozgás fő iránya a vízszintes síkban van, például szállítási vagy anyagmozgatási feladatokban.



6. ábra RRR Típusú robotkar kinematikai vázlata [8]

Ezen kinematikai kényszerek alkalmazása lehetővé teszi a csuklókaros robotok számára, hogy rugalmasan és hatékonyan alkalmazkodjanak különböző feladatokhoz és környezeti feltételekhez. Ennek eredményeként a robottechnológia területén folyamatos fejlődés tapasztalható, amely új és innovatív alkalmazási lehetőségeket kínál az iparban, a gyártásban és sok más területen.[8]

2.3.2. Ipari robotok

Az ipari robotok definíciója alapján az egyik alapvető tulajdonságuk a mechanikus beavatkozás nélküli átprogramozhatóság. Ez azt jelenti, hogy a robotok képesek arra, hogy programjukat módosítsák vagy befolyásolják anélkül, hogy szükség lenne fizikai változtatásokra vagy manuális beavatkozásra. Az átprogramozhatóság különböző módon valósulhat meg, és ezek a lehetőségek a robotok intelligencia szintjét is tükrözik. Az átprogramozhatóság szintje különböző lehet. [8]

- Az első esetben a robotok képesek önállóan működni anélkül, hogy külső programbefolyásolásra lenne szükségük. Ez azt jelenti, hogy a robotok autonóm módon képesek végrehajtani a rájuk bízott feladatokat, és nem igényelnek külső beavatkozást vagy irányítást.
- A második esetben a robotok rendelkeznek a képességgel, hogy különböző programok közül válasszanak ki és alkalmazzanak, attól függően, hogy éppen milyen feladatot kell ellátniuk. Ez a programszelekció lehetővé teszi számukra, hogy rugalmasan reagáljanak a különböző környezeti változásokra és feladatokra, és alkalmazkodjanak azokhoz a körülményekhez, amelyek között dolgoznak.
- A harmadik esetben a robotok rendelkeznek a képességgel, hogy alkalmazkodjanak a környezeti változásokhoz és feladatokhoz azáltal, hogy módosítják vagy adaptálják meglévő programjaikat. Ez a programadaptáció lehetővé teszi számukra, hogy tanuljanak az előző tapasztalatokból és fejlesszék képességeiket az idő múlásával. A robotok ilyen módon képesek optimalizálni teljesítményüket és hatékonyságukat, ami növeli az ipari folyamatok hatékonyságát és rugalmasságát. Az átprogramozhatóság ezáltal nem csupán technikai tulajdonság, hanem az adott robot intelligencia szintjének és alkalmazhatóságának mércéje is egyben.[8]

2.4. HTML Struktúrájú weblapok

A HTML (Hypertext Markup Language) egy szöveges jelölőnyelv, más néven leírónyelv, amelyet weboldalak készítéséhez fejlesztettek ki. Ma már számos változata az internetes szabványok részévé vált a W3C (World Wide Web Consortium) jóvoltából. A HTML eredetileg az SGML (Standard Generalized Markup Language) dokumentumszabványból származik, ami egy általános leírónyelv volt, és 1986-ban vált ISO szabvánnyá. Az SGML általános dokumentumleírást biztosított, és elődje volt a GML (Generalized Markup Language), amelyet az IBM már az 1960-as években kifejlesztett. Az HTML (HyperText Markup Language) az ún. SGML (Standard Generalized Markup Language) jelölőnyelven alapul, míg az XHTML (eXtensible Markup Language) az XML (eXtensible Markup Language) alapú, amely már webes adatbázisok közötti kommunikációt biztosító, bővíthető nyelvet képvisel. Az HTML5 fejlesztése 2004-ben kezdődött és 2014-ben zárult le. Ez a szabvány kompatibilis a korábbi HTML és XHTML verziókkal is, és bár a DOCTYPE átírásával elvileg már HTML5-ös kódot készíthetünk, ha csak ennyit teszünk érte, akkor valószínűleg a dokumentumunk régebbi, elavult elemeket is fog tartalmazni, és nem használja ki az új leírónyelv által kínált új lehetőségeket.

A HTML5-ös technológiát támogatják a mobileszközök is. A HTML korábbi verzióihoz képest a HTML5 jelentős átdolgozásokon ment keresztül. A fejlesztés fő célja az volt, hogy a weboldalak működéséhez ne legyen szükség különféle pluginok telepítésére, mint például az Adobe Flash vagy a Microsoft Silverlight esetében. 2004-ben megalakult a WHATWG (Web Hypertext Application Technology Working Group), amely függetlenül működött a W3C-től. Ők a HTML továbbfejlesztését választották az XHTML helyett. 2007-ben a W3C hivatalosan is kijelentette, hogy a következő hivatalos HTML-szabvány a HTML5 lesz, és nem folytatták tovább az XHTML2 fejlesztését. Ennek eredményeként a HTML5 vált az internetes tartalmak létrehozásának és megjelenítésének alapvető standardjává.[9]

2.4.1. HTML weblapok felépítése

A HTML5 típusú weblapoknak három fő része van. A HTML dokumentumok struktúráját három fő részre lehet osztani. Elsőként, az állomány elején található a !DOCTYPE definíció, ami a dokumentumban használt szabványra utal. Ez az elem meghatározza, hogy melyik HTML verziót kell alkalmazni a dokumentum értelmezése során. A második rész a HTML fejlécét alkotja, amelyet a <head> és </head> elemek közé írunk. Ide kerülnek a technikai és dokumentációs adatok, például a karakterkészlet definíciója, a stíluslapok hivatkozásai és egyéb metaadatok. A fejléc tartalmazza azokat az információkat, amelyeket a böngésző nem jelenít meg közvetlenül az oldalon, de fontosak lehetnek a dokumentum megjelenítése és értelmezése szempontjából. A harmadik egység a HTML törzsrésze, amelyet a <body> és </body> elemek között határozunk meg. Ez a rész tartalmazza azokat az adatokat, amelyeket a böngésző közvetlenül megjelenít a felhasználónak. Ide kerülnek például a szövegek, képek, táblázatok és egyéb multimédiás elemek, amelyek láthatók és interaktívak a weboldal látogatói számára. A HTML törzsrészében definiáljuk a weboldal szerkezetét és tartalmát, amelyeket a felhasználók böngészése során láthatnak és tapasztalhatnak. [9]

```
1    <!DOCTYPE html>
2    <html>
3    <head>
4      <meta charset="UTF-8">
5      <title>Unmatched</title>
6      <link rel="stylesheet" type="text/css" href="unmatched1.css">
7      <link rel="Website icon" type="jpg" href="logo.jpg">
8
9
10   </head>
11 > <body> ···
439  </body>
440  </html>
```

7. ábra HTML felépítése

2.4.2. Statikus webszerverek, FTP típusú honlapok

A statikus weboldalak olyan online tartalmak, amelyek lényegében állandóak és változatlanok maradnak azokban az állapotokban, ahogy készítőik feltöltötték őket a webszerverekre. Ezek az oldalak általában egyszerű HTML kódból állnak, amelyet a weboldal készítője helyez el a szerveren. Amikor a felhasználó lekéri az adott weboldalt, ugyanazt a HTML kódot kapja vissza, amelyet a készítő feltöltött. Ez azt jelenti, hogy minden lekérdezésnél ugyanaz a tartalom és ugyanaz a formázás jelenik meg.

A statikus weboldalaknak jellemzően egyszerűbb kinézetük és felépítésük van. Nem tartalmaznak olyan bonyolult funkciókat, mint például regisztráció, fórum, blog vagy webshop. Nem kérnek be semmilyen adatot a felhasználótól, és nem tárolnak el semmiféle információt adatbázisban.

Ezek a weboldalak statikusak maradnak azáltal, hogy nem igényelnek semmilyen dinamikus programozást vagy adatkezelést. A készítőjük egyszerűen létrehozza az oldalt HTML formátumban, majd feltölti a szerverre, és azután a tartalom változatlan marad, amíg a készítő nem dönt úgy, hogy frissíti vagy módosítja azt. A statikus weboldalak tehát megbízható és egyszerű módját kínálják az információk online megosztásának anélkül, hogy bonyolult dinamikus funkciókat kellene kezelniük. [9]

A webhoszt szolgáltatásokra való regisztráció után egy felhasználói fiókot kapunk, amelyhez hozzáférünk az adminisztrációs felületen és az FTP-n (File Transfer Protocol) keresztül. Ezek segítségével feltölthetjük a weboldalunkhoz szükséges fájlokat és adatokat a szerverre. Emellett kapunk egy webcímet is, amelyen keresztül az oldalunk elérhető lesz az interneten. A minőségi szolgáltatóktól továbbá lehetőségünk van adatbázis-hozzáférés igénylésére is, ha szükségünk van rá. [9]

2.4.3. JavaScript Bemutatása

A webes parancsnyelvek jelentős különbségben vannak a hagyományos programozási nyelvektől. Míg az utóbbiakat először le kell fordítani gépi kódra, hogy a számítógép értelmezze és futtassa azokat, addig a webes parancsnyelvek, például a JavaScript, közvetlenül a böngészőben hajtódnak végre. Ez azt jelenti, hogy amikor egy weboldalt megnyitunk, a böngésző sorban olvassa be és értelmezi a webes parancsokat, majd azonnal végrehajtja azokat.

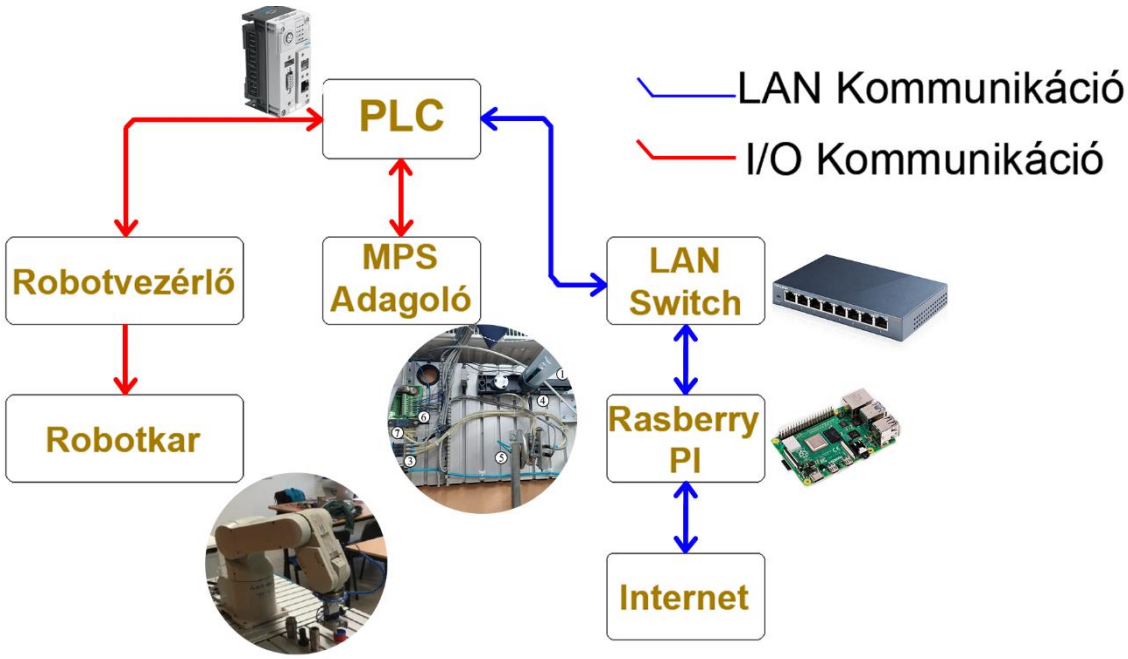
A JavaScript kódot tartalmazó fájl módosítása rendkívül egyszerű folyamat. Csak meg kell változtatnunk a kódot a szükséges módon, majd el kell mentenünk a fájlt. A változtatások azonnal érvénybe lépnek, amint a felhasználó újratölti a weboldalt a böngészőben. Ez a gyors visszajelzési ciklus lehetővé teszi a fejlesztők számára, hogy azonnal lássák és teszteljék a változtatások hatását a weboldal működésére, anélkül, hogy újra kellene fordítaniuk vagy újra kellene indítaniuk az alkalmazást. Ez a rugalmasság és az azonnali visszajelzés lehetővé teszi a fejlesztők számára, hogy gyorsan és hatékonyan dolgozhassanak a webes alkalmazások fejlesztése során. A JavaScript egyik előnye, hogy függvényeket tudunk megjeleníteni a honlapon, ezt teszi lehetővé, hogy például a PLC-vel is kommunikáljon. Ezt vagy a HTML-en belül a <script> </script> tag-en belül tehetjük, vagy pedig létrehozunk egy JavaScript file-t. [9]

```
582  function toggleCharacterSelection100() {
583      var selectAllCheckbox = document.getElementById('selectAllCharacters100');
584      var characterCheckboxes = document.querySelectorAll('.word-image input[type="checkbox"][data-group="battle100"]');
585
586      for (var i = 0; i < characterCheckboxes.length; i++) {
587        characterCheckboxes[i].checked = selectAllCheckbox.checked;
588      }
589      // Frissíti a kiválasztott szavakat
590      selectedWords = Array.from(document.querySelectorAll('.word-image input[type="checkbox"][name="word"]:checked')).map(function (checkbox) {
591        return words.find(function (word) {
592          return word.value === checkbox.value;
593        });
594      });
595
596      updateWordList(); // Frissíti az eredményeket
597
598    }
```

8. ábra Példa egy JavaScript függvényre

3. Robottal támogatott adagoló állomás PLC programozása és távvezérlésére alkalmas rendszer tervezése

A szakdolgozatom célja a Mechatronika tanszéken található MPS adagoló állomás és egy robotkar, illetve egy weblap együttműködésének megvalósítása. Ennek az egésznek az agya a PLC, ami tárolja az előre feltöltött programot és irányítja az adagoló állomást, illetve parancsot küld ki a robotvezérlőnek, ami irányítja a robotkart, mindeközben adatokat küld és kér le a weblapnak. Az adagoló állomáson több szenzor is megtalálható, például végállás érzékelő szenzor és vákuum szenzor, ezen kívül pedig egyszeres és kétszeres munkahengerekkel dolgoztam.

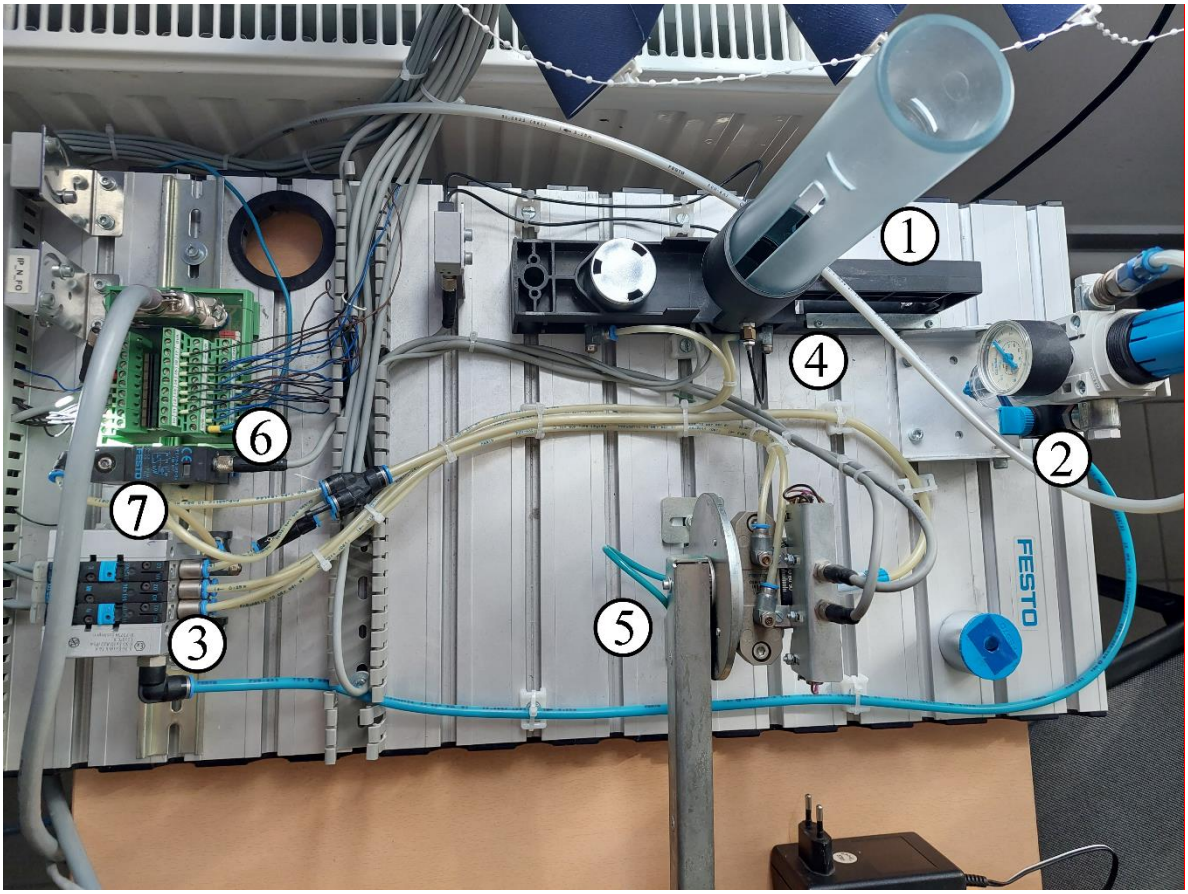


9.ábra A rendszer egyszerűsített kommunikációs rajza

A folyamat röviden úgy írható le, hogy a PLC-re előre feltöltött programok irányítják a folyamatot, első körben az adagoló állomás dolgozik, ami a munkadarabot szállítja egyik helyről a másikra, majd egy adott időpillanatban a robotvezérlő kap egy parancsot, hogy a robotkar is működésbe lépjen. Mindeközben a PLC saját webszerverén nyomon követhető az egész folyamat.

3.1. MPS Adagoló állomás

A rendszer első önálló egysége. Ez a rendszer alapján történnek a mérések, illetve ez a folyamat irányítható és távvezérelhető része, illetve itt található a legtöbb szenzor is. Az adagoló állomás I/O kommunikációt folytat a PLC-vel. A fő feladata, hogy hiba nélkül szállítsa le a munkadarabokat a kezdő pozícióból a vég pozícióba, ha ez sikeresen megtörtént akkor a robotkar fogja tovább szállítani a munkadarabot.



10. ábra Az MPS Állomás

Az egyes számmal jelölt egység a munkadarab adagoló magazin, ebben érkeznek a munkadarabok a kezdő pozíciójukba. A kettes számmal jelölt elem a levegő előkészítő egység, aminek részei: nyomásszabályozó, szűrővíz leválasztó, nyomásmérő óra és egy elzáró szelep. Hármassal jelöltem a szelepszigetet, négyessel az egyszeres működésű munkahengert, ötössel az fordító hengert és végül hatossal a I/O kommunikációért felelős elemet. Hetessel a vákuum kapcsolót. Az állomás elektromos kapcsolása a első, a pneumatikus kapcsolási rajza az első mellékletben található.

3.1.1. Az állomás működésének folyamata

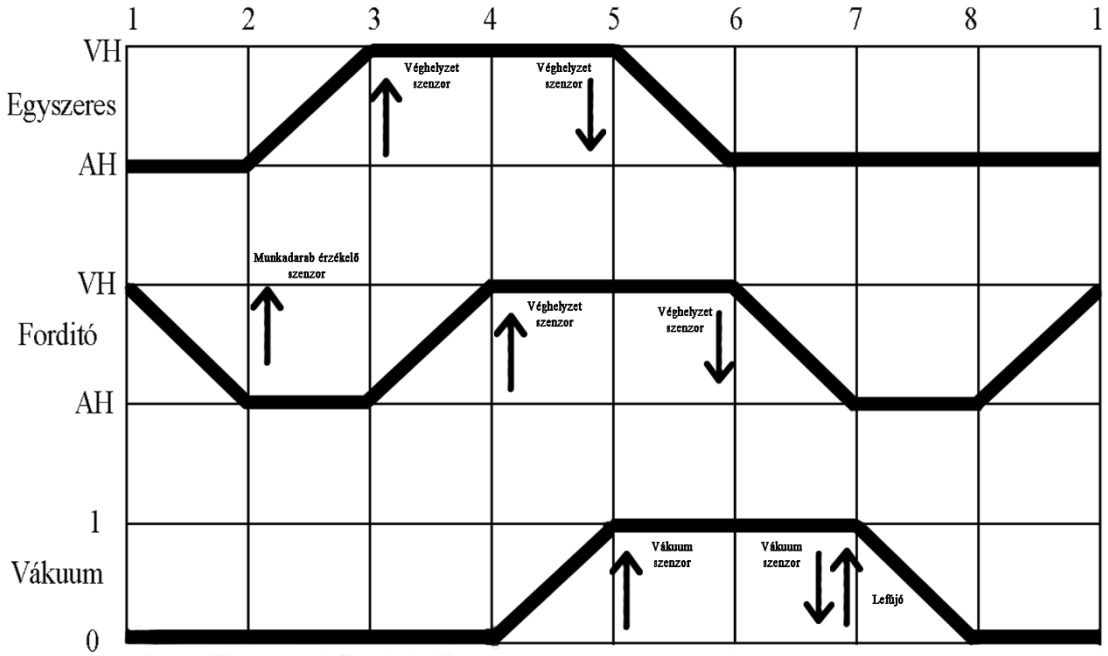
A folyamat a következő kilenc lépésből áll:

1. A rendszernek alaphelyzetben kell lennie, ami azt jelenti, hogy az egyszeres működésű munkahenger, hengere behúzott állapotban van. Az fordító munkahengernek két pozicíója van: Alaphelyzetben van, amikor az adagoló magazinhoz van közelebb és véghelyzetben amikor a másik pozíciójában van. Amiket véghelyzet szenzorok segítségével tudunk érzékelni, illetve se a lefújó se a vákuum szenzor nincs bekapcsolt állapotban és van munkadarab a kezdő pozícióban.
2. Ha az első lépésben minden megfelelő akkor az átfordítónak át kell fordulnia a másik oldalra, amit egy szenzor érzékel.
3. Amikor az átfordító átfordult akkor az egyszeres működésű munkahengernek ki kell tolnia a munkadarabot a felvételi pozícióba és ott tartja.
4. Ha ez megtörtént akkor az átfordítónak megint vissza kell fordulnia a munkadarabért a felvételi pozícióba, ennek során a szívó korongot rászorítja a munkadarab felületére.
5. Az ötödik lépésben a vákuum bekapcsol, a vákuum szenzor ellenőrzi, hogy ez megtörtént-e.
6. Ha a vákuum szenzor jelet adott, akkor az egyszeres működésű munkahenger visszamehet alaphelyzetébe.
7. Amikor a vákuum szenzor aktív és az egyszeres működésű munkahenger is alaphelyzetében van, akkor az átfordító áthelyezi a munkadarabot lehelyezési pozícióba.
8. Ebben a pozícióban a vákuumot ki kell kapcsolni és egy másodperce a lefújót is aktiválni, hogy a munkadarab biztosan a megfelelő helyre kerüljön.
9. A ciklus utolsó lépése, ami igazából a következő ciklus első lépése is, hogy az átfordító megint alaphelyzetébe fordul.

Ezek a lépések szükségesek a folyamat rendeltetésszerű működéséhez. Az egész folyamat két féle képpen indítható, ha az első lépésben lévő taglaltak megfelelőek. Egyrészt a PLC vezérlő felületéről a START gomb megnyomásával vagy pedig a weblapon lévő START gomb megnyomásával.

Ha a rendszer nincsen alaphelyzetben, akkor szintén bármelyik felületen lévő gombok segítségével alaphelyzetbe állítható.

Ezen kívül, a rendszer automatikusan leáll és alaphelyzetbe állítja magát, ha a felhasználó által deffiniált idő elteltével nem érkezik munkadarab a munka állomáshoz. De tartalmaz a kezelőfelület és a weblap is egy ciklus vége, illetve egy vészstop gombot is.



11. ábra Az adagoló út-lépés diagramja

3.2. CPX PLC

A PLC az egész rendszere központja, irányítja az adagoló állomást az állomás minden aktuátorát, ide fut be az összes szenzora. Kommunikál a robotvezérlővel, ellenőrzi a robotkar állapotát és itt történik az utasítások kiadása a robotprogram futtatására. A távdiagnosztikai honlap is a PLC-n fut, ugyanis, a megírt program fájlokat és dokumentumokat, a PLC-re megfelelő könyvtárába feltöltve máris elérhető lesz a kialakított honlap. Ezen kívül egy a Festo PLC-hez kialakított szoftvert is használtam, egy HMI panelt, ami a PLC grafikus kezelőfelületét hivatott pótolni.

3.2.1. Feladatmegoldás

Nulladik lépésként el kell érni, hogy a PLC tudjon kommunikálni egy számítógéppel. A FEC-CPX típusú PLC képes LAN-on kommunikálni és parancsokat fogadni, ezért vagy akár közvetlenül összelehet kapcsolni a PLC-t a számítógéppel, vagy pedig egy bármilyen, nem menedzselhető Switch-el is akár összekötni a két eszközt. Az én esetemben az utóbbit használtam, ugyanis a Rasberry Pi-nak is látnia kellett a PLC-t és a teszteléseket is megkönnyítette. Egy adott hálózatba kell hozni az eszközöket, hogy lássák egymást, ez esetemben a 192.168.0.X tartomány volt, ahol a PLC volt a .1-es a Rasberry Pi a .102-es illetve egy IP vezérlésű kamerát is bele terveztem a rendszerbe, hogy távvezérlésnél élő képet kapjunk a rendszerről, a kamera pedig a .103-as tartományba került.

A FESTO-nak az FST nevű szoftverét használtam a program elkészítéséhez, ami a saját moduljaihoz készült. Illetve egy FEC Soft Control nevű alkalmazását, ami szintén FESTO PLC-hez készült és lényegében nyomógombokat tartalmaz, és a feladatomban a kezelőfelülete lesz a PLC-nek a honlapon kívül.

Miután sikeresen ugyanarra a hálózatra állítottam az eszközöket, először is meg kell határozni, hogy a PLC melyik kimeneteire, illetve bemeneti csatornákra vannak kötve a szükséges aktuátorok és szenzorok, ezt az FST szoftverbe tudtam megtenni, miután Online munkamódra állítottam. Ezután elkezdhettem felvenni az úgynevezett allokációs listát. Ez a lista tartalmazza az összes bemenetet, kimenetet, illetve egyéb olyan operátorokat, amiket használtam a program megírása közben.

Szent István Campus, Gödöllő
Cím: 2100 Gödöllő, Páter Károly utca 1.
Tel.: +36-28/522-000
Honlap: https://godollo.uni-mate.hu

Operátor	Szimbolúm	Komment
	OUTPUT	
O0.0	M_T	MUNKAHENGER TOLOGATÓ
O0.1	V_O	VÁKUUM ON
O0.2	L_L	LEFÚJÓ
O0.3	Á_V	ÁTFORDÍTÓ VH
O0.4	Á_A	ÁTFORDÍTÓ AH
O5.0	S_L	START LÁMPA
O5.1	S_L	STOP LÁMPA
O5.2	I_L	INIT LÁMPA
O5.3	A_L	AUTO/MANUAL LÁMPA
O5.4	W_L	WATCHDOG LÁMPA
O5.5	V_L	VÉSZSTOP LÁMPA
	INPUT	
I0.1	K_AH	KITOLÓ AH
I0.2	M_VH	MUNKADARAB VH
I0.3	V_SZENZOR	VÁKUUM SZENZOR
I0.4	Á_VH	ÁTFORDÍTÓ VH
I0.5	Á_AH	ÁTFORDÍTÓ AH
I0.6	M_SZENZOR	MUNKADARAB SZENZOR
I5.0	S_GOMB	START GOMB
I5.1	S_GOMB	STOP GOMB
I5.2	I_GOMB	INIT GOMB
I5.3	A_GOMB	AUTO/MANUAL GOMB
I5.4	W_GOMB	WATCHDOG GOMB
I5.5	V_GOMB	VESZSTOP GOMB
	FLAGS	
F0.0	I_F	INDITO_FLAG
F1.1	J_FB	JELZO BIT
F7.0	W_FB	WATCHDOG TIMER
FW0	F_FB	FLAG WORD
FW13	1_FW	16-S FLAG WORD
	PROGRAMOK	
P1		MUNKAPROGRAM
P2		STOP LAMPA VILLOGAS
P3		WATCHDOG PROGRAM
P4		WD VILLOGÁS
P5		HIBAKERESŐ PROGRAM
	REGISZTEREK	
R1	R_1	REGISZTER
R7	R_7	HIBAKÓD TÁROLÓ
R20	R_20	REGISZTER
R21	R_21	REGISZTER
R23	R_23	REGISZTER
	IDŐZÍTŐK	
T1	T_1	IDOZITO STOP
T2	T_2	WD IDOZITO
T5	T_5	VÁKUUM IDOZITO
T6	T_6	TIMER
T7	T_7	WD TIMER
T10	T_10	MUNKADARAB ERZEKELO
TP7	TP_7	WD TIMER ERTEK
TW7	TW_7	WD TIMER ERTEK
	SZÁMLÁLÓ	
C0	C_0	SZÁMLÁLÓ BIT
CP0	CP_0	SZÁMLÁLÓ ÉRTÉKE
CW0	CW_0	SZÁMLÁLÓ AKTUÁLIS ÉRTÉKE

12. ábra PLC Programban használt operandusok

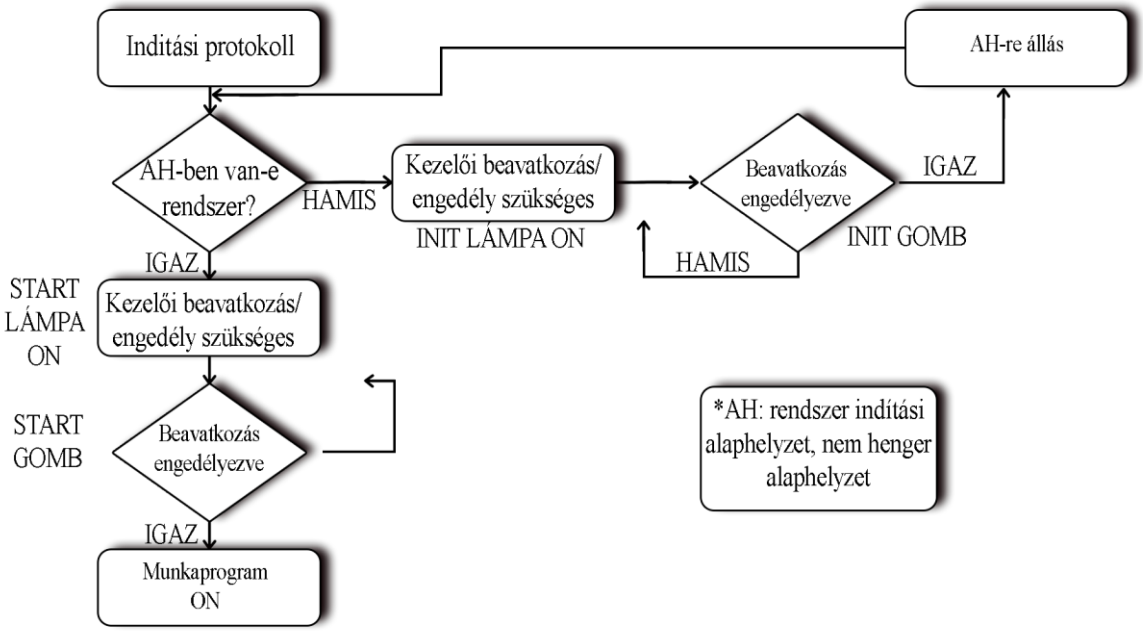
3.2.2. PLC Program működése

Ahhoz, hogy egy PLC program megfelelően működjön nem elég csak, feltölteni a programot a processzora, hanem bizonyos feltételkének megkell felelnie a rendszernek. A rendszer indíthatóságát egy indítási eljárásban foglaltam össze. Első lépésként a mérőállomásnak alaphelyzetben kell lennie. Ez azt jelenti, hogy minden szelepe zárt állapotban van, az aktuátorokban nincs levegő, a lineáris munkahenger az alsó pozícióban van, az átfordító dugattyúja be van húzva, és se a vákuum se a lefújó állapota nem aktív. Ezenkívül a munkadarabnak a tálca pozíciójában kell lennie, amit az Opto-elektronikus szenzor jelzése jelez a rendszernek. A robotkar is alaphelyzetben kell legyen, és erről jelzést kell küldenie a PLC-nek. Csak ezután, miután mindezen feltételek teljesültek, lehet megnyomni a kontroll panelen vagy a honlapon lévő start gombot. Amikor ezek a feltételek mind teljesülnek, a mérőállomás biztonságosan elindul és megkezdődik a munkafolyamat. Ekkor az általános munkaprogram veszi át az irányítást a mérőállomáson, és vezérli azt a végéig. Ez a protokoll biztosítja, hogy a folyamat csak akkor kezdődjön el, amikor minden körülmény optimális és biztonságos a munkavégzéshez. Többek között ez azt jelenti, hogy csak akkor történjen mozgás, ha operátori jóváhagyás történt, ha a rendszer alaphelyzetben van, akkor engedélyezni kel az indítást. Figyelembe véve a jelenlegi konfigurációt és a korlátokat ezeket a biztonsági folyamatokat határoztam meg.

A rendszer biztonsági protokolljának legfontosabb feladata a személyi és vagyon védelem biztosítása. Emellett kiemelt jelentőséggel bír a munkafolyamatok végrehajtása közben felmerülő kritikus hibák azonosítása és azok orvoslása, hogy azok ne okozzanak további problémákat. A protokoll elsődleges célpontja a mérőállomás és az ott dolgozók biztonságának garantálása. Ha személyi sérülés nem történik, a protokollnak az a feladata, hogy az állomást a lehető leghamarabb újra üzemképessé tegye.

A biztonsági protokoll két fő részből áll. Az első rész, vagyis a vészleállítási protokoll. Ebben az esetben a rendszer azonnal kikapcsolja a mérőállomás minden munkafolyamatát, vagy ha valamely munkafolyamat nem tud teljesülni, az érintett részeket azonnal leállítja. Emellett a protokoll légteleníti a munkahengereket, ezáltal minimalizálva a további veszélyek kialakulásának lehetőségét. A protokoll második része egy hatékony hibakeresési rendszer, amely lehetővé teszi a rendszerben bekövetkező problémák gyors azonosítását. Ez a rész a problémák pontos feltárására és az azokkal kapcsolatos gyors intézkedésekre

összpontosít, hogy a rendszer minél hamarabb visszatérhessen a normális üzemeléshez. Ennek az a célja, hogy biztosítsa a mérőállomás zavartalan és biztonságos működését, miközben minimalizálja a leállások és a veszélyhelyzetek kockázatát.



13. ábra Indítási protokoll hatásvázlata

3.2.3. PLC Programok

A feladatmegoldáshoz utasításlistás programnyelvet használtam, ugyanis a Festo CPX PLC ezt a programnyelv típust tudja értelmezni. Ahhoz, hogy megfelelően működjön minden, összesen hat programot kellett létrehoznom:

1. Fő program

   Ez a program felel azért, hogy az összes többi program is megfelelően tudjon dolgozni, ugyanis ez a rész indítja el a munkaprogramot is, de ebből tudjuk kinyerni, hogy akár alaphelyzetben van-e az állomás, külön utasítások vannak, alaphelyzetből való indításra, és nem alaphelyzetből való indításra, ha viszont a alaphelyzetbe állt a rendszer, akkor ez a program felel azért is, hogy egy lámpa felvillanjon, jelezve, hogy a rendszer indítható állapotba került, a start gomb megnyomása után a munkaprogram mellett az úgynevezett watchdog program is elindul. Illetve a fő programomban található a vészstop gombnak a funkció is.

2. Munkaprogram

Ez az alprogram felelős azért, hogy a korábban részletezett munkafolyamatok megfelelően megtörténjenek. Pont ezért több lépésből (Step)-ből áll a program, mindegyik ilyen lépésben a program addig keresi a feltételeket, amíg az meg nem történik, és csak is akkor lép tovább a következő lépésre, ha van olyan feltétel, ami teljesült.

```
STEP 1
 IF                    FW0          'Flag word
        =              V4
 THEN  LOAD            V0
         TO            R20          'regiszter
       LOAD            V0
         TO            R21          'regiszter 2
       LOAD            V0
         TO            CP0          'Számláló értéke
       LOAD            V0
         TO            R23          '12s Regiszter érték
       RESET           F1.1         'Jelzo bit
       LOAD            V0
         TO            FW13         '16 s
```

14. ábra Munkaprogram első lépése

A munkaprogram legelső lépésében lényegében bizonyos értékeket állítunk be, maga a program ebben a lépésben egyetlen egyszer fog végig menni, ahol bizonyos paramétereket fog felvenni. Ezeket hivatott a regiszter mutatni, ugyanis PLC-ben a regisztereket decimális értékekkel tudjuk feltölteni, ami a parametrizálhatóságát adja, jelent esetben bizonyos időközöket tudunk megadni, például, hogy a rendszer hány másodpercig várjon egy munkaradbra mielőtt alaphelyzetbe állna. Ezen kívül itt található a weblap felől jövő start gomb hatása is. Ezt úgy oldottam meg, hogy a nullás flag-nek szintén adok egy decimális értéket, jelen esetben négyet. Miután a szükséges paramétereket felvette a program és az weblapon lévő start gomb is megnyomásra került a következő lépésre léphet a program.

Az összes többi lépés nagyon hasonlít, mindegyikben valamilyen feltételeknek kell beteljesülnie ahhoz, hogy a következő lépésre lehessen térni. Azonban minden lépésnél elindítottam egy watchdog időzítőt is. Ennek az időzítőnek az a feladata, hogy mindegyik lépésben újra induljon, és maga az időköz, amit előzetesen definiáltam, biztosan több idő legyen, mint maga a lépésnek az időtartama, hogy a watchdog időzítő semmilyen körülmények között ne tudjon lejárni, azonban, ha mégis akkor egy hiba üzenetet küldjön a felhasználónak.

```
STEP 10
 IF                          I0.1          'Kitolo AH
 THEN   SET                  O0.3          'Vakum VH
        RESET                O0.4          'Vakum AH
        SET                  T7            'WD_Timer
          WITH               4s
        SET                  F7.0          'Watchdog_Timer
```

15. ábra Részlet a munkaprogramból

Jelen esetben a T7-es időzítőt kapcsolom be négy másodperccel.

A munkaprogramban még két fontos dolog történik, a munkadarab várására szolgáló időzítő parametrizálása, illetve a ciklusszámláló elhelyezése.

```
STEP 15
 IF                          I0.6          'Munkad. szenzor  AH
        AND      N           F1.1          'Jelzo bit
 THEN   SET                  F1.1          'Jelzo bit
        LOAD        (        R23           '12s Regiszter érték
          *                 V100        )
          TO                TP7           'WD_Timer_Ertek_TP
 IF              N           I0.6          'Munkad. szenzor  AH
 THEN   JMP TO 20
```

16. Ábra Parametrizálható regiszter érték

A lépés HA logikai feltétele nézi, hogy van-e munkadarab, az utána lévő utasítás pedig a 23-as regiszter értékét tölti be a watchdog időzítőbe, 100-al szorozva, hogy másodpercet kapjunk eredményül. A 23-as regiszter értékét tudja megadni a felhasználó előre a weblapon egy mezőbe beírva, és az első lépésben tölti be ezt az értéket a PLC a regiszterbe.

A ciklus számláló is regiszterek segítségével történik, ugyan van előre deffiniált számláló a PLC programozó felületében, azonban, hogy ezeket az értékeket a weblapon is megtudjam jeleníteni, a regisztereket kell használni.

```
STEP 95
 IF                          I0.4            'Atfordito jobb
       AND                   I0.1            'Kitolo AH
 THEN  SET                   O0.4            'Vakum AH
       RESET                 O0.3            'Vakum VH
       INC                   R20             'regiszter
       LOAD          (       R21             'regiszter 2
         +                   V1          )
         TO                  R21             'regiszter 2
```

17. ábra Számláló regiszterekkel

A ciklusnak az utolsó olyan lépésébe helyeztem el a számlálót, ahol az utolsó munkafolyamat történik. A számlálóhoz a 21-es regiszter értékét használom, és ebben a lépésben mindig eggyel nagyobb értékét tölt bele a rendszer.

18. Stop lámpa villogás

    A program akkor kapcsol be, mikor a rendszer automatikus üzemmódjában megnyomjuk a stop gombot. Feladata jelezni, hogy az automatizált folyamat, utolsó köre megy végbe, utána a mérőállomás leáll.

19. Watchdog program

    Ez a program akkor lesz aktív, amikor a korábban részletezett lépésekben valamikor az időzítője lejárna. Lényegében a fő feladata, hogy elindítsa a hibakereső programot.

```
IF                          P1              'Munkaprogram
      AND        N          T7              'WD_Timer
      AND                   F7.0            'Watchdog_Timer
THEN  SET                   P4              'WD_Villogas
      SET                   P5              'Hibakereso_Program
      RESET                 P1              'Munkaprogram
      RESET                 F7.0            'Watchdog_Timer
```

18. ábra Watchdog program

19. Watchdog lámpa villogás

    A program akkor kapcsol be, mikor a rendszer automatikus üzemmódjában a watchdog időzítő lejárt, és jelzi, hogy valahol elakadt a folyamat.

20. Hiba behatároló program

    A hiba be határoló program a mérőállomás külső hibából adódó leállásának okát vizsgálja. A program működési elve abból áll, hogy mikor elindul egy adott szakasza a folyamatnak vele párhuzamosan bekapcsol egy időzítő is. Ha nem valósul meg a munkafolyamat, azaz a következő szakasz kezdeti szenzorai nem jeleznek és lejár az időzítő, a rendszer leáll az aktuális pozíciójában, kigyullad a rendszerhiba lámpa. Mindemelett mikor az időzítő lejár a hibakeresési program feltölt egy regisztert, egy a probléma helyére jellemző hibaüzenettel. A program 5 valószínűsíthető hibapontot tud meghatározni, illetve nulladik elemként a hibamentes működést. A lehetséges hibapontok a következők:

    a. 001-es hibakód az, amikor a monostabil munkahenger alaphelyzetbe állítás után is véghelyzetbe maradt.
    b. 002-es hibakód amikor az átfordító se nem alaphelyzetben se nem véghelyzetben nincs.
    c. 003-as hibakód, amikor a munkadarab érkezése ellenére sem jelez a szenzor.
    d. 004-es hibakód, ha a vákuum szenzor az átfordító alaphelyzetében is aktív maradna annak ellenére, hogy nincsen munkadarab.
    e. 005-ös hibakód, ha a monostabil munkadarab elindul véghelyzetébe de ennek ellenére nem érkezik meg véghelyzetébe.

```
"" Atfordito HIBAI
 IF                    O0.4          'Vakum AH
      AND      N       I0.5          'Atfordito bal
      AND              I0.4          'Atfordito jobb
 THEN LOAD             V110
        TO             R7            'Hibakód tároló
```

19. ábra Példa egy hibakereső programra

A hibakódok is ahogy látható regiszterekben vannak tárolva. És a regiszterek értéke alapján iratható ki a megfelelő hibakód és egy esetleges megoldás a proglémára.

3.3. Robotkar

Az adagolóállomás utána végrehajtó szerv. A robot a beérkező munkadarabot a megfelelő vég pozícióba szállítja miután a robotvezérlő program utasítást kapott erre a PLC-től.

3.3.1. Robotkar felépítése

A pakolási feladat ellátáshoz egy RRR típusú, 5 tengelyes Mitsubishi RV-2AJ humanoid robotkart használtam. A robotkar 3 rotációs csuklóval rendelkezik, maximális terhelhetősége 2 kg, 410mm a maximum elérési távolsága. A robotkar effektora egy pneumatikus megfogó. A robotkar további lényeges elemei a vezérlőszekrény, egy teaching pendant, az I/O kommunikációs egység, egy I/O kábel és a pneumatikus interfész, a megfogó végett.

3.3.2. Feladatmegoldás

Először azt a feladatot kellet megoldani, hogy a robot és a PLC képes legyen egymással kommunikálni. Ezt a robotkarhoz kapcsolt I/O terminállal valósítottam meg. A robotkar vezérléséhez 2 bemeneti és 2 kimeneti jelre volt szükség.

Robot bemenet	PLC Kimenet	Leírás
M_IN(10)	O9.6	Robotprogram indítása
M_IN(11)	O9.5	Munkadarab

Robot kimenet	PLC Bement	Leírás
M_OUT(4)	I9.7	Robot program fut
M_OUT(5)	I9.6	Robot és PLC kommunikáció

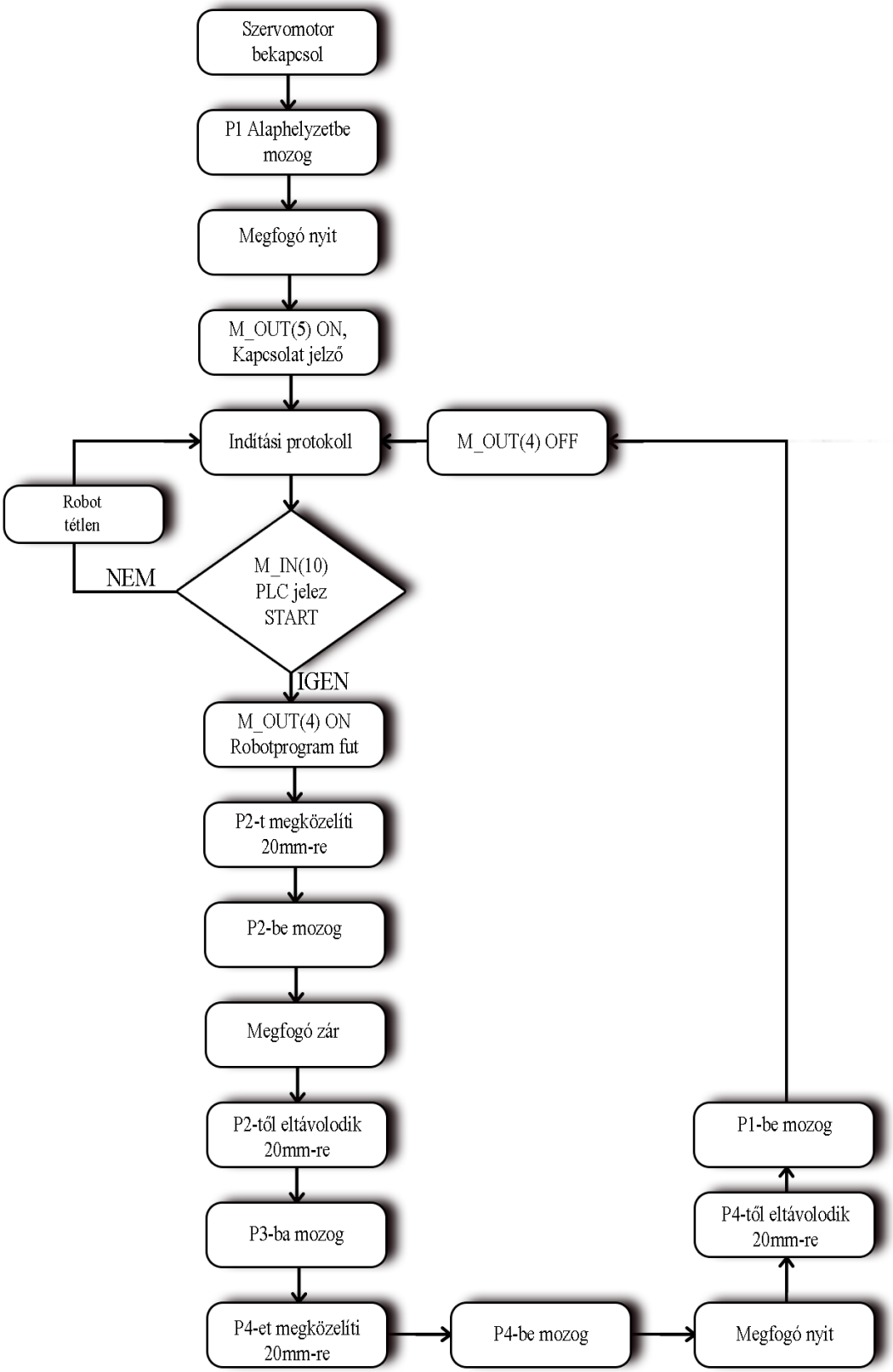
20. ábra Robotprogramhoz szükséges változók

Az eszközök összekapcsolása és a be- és kimenetek felvétele után el tudtam kezdeni a robotkar felprogramozását. Először definiáltam azokat a pontokat, melyekbe a robotkarnak majd mozognia kell a folyamat során. Ezeket a pontokat úgy tudtam meghatározni, hogy a robotkart a teach pendantjával a kívánt pozícióba mozgattam, majd rögzítettem őket a memóriába. A feladat elvégzéshez 4 pontot kellett rögzíteni, melyek a következők:

- P1, a robot alaphelyzetének pontja, mielőtt megkezdődik a munkafolyamata, itt várja az utasítást.
- P2 a pont, ahol a robot felveszi a munkadarabot, azaz a mérőállomás kiadótálcájának vége.
- P3 egy köztes munkapont, erre azért van szükség, hogy a robot ne menjen neki a mérőállomásnak, mikor a munkadarabot az adott pozíciójába akarja juttatni.
- P4 a munkadarab pozíciója.

3.3.3. Robotprogram

A robot feladatelvégzéshez egyetlen programot hoztam létre. A program első egységének a feladata a szervomotor bekapcsolása, a robot alaphelyzetbe állítása és az M_OUT(5) felkapcsolása. Ez azt jelenti, hogy a robotkar a P1 pozícióba mozog, szétnyitja a pneumatikus megfogót és felkapcsolja a kommunikációt jelző kimenetet. Itt a robot addig várakozik, ameddig a PLC nem közli vele, hogy a munkadarab a kiadótálcán van. Ha ez megtörténik elindul a program második egysége. A robot tengelymozgással y koordináta szerinti 20mm távolságra megközelíti a munkadarabot tartalmazó tálca végét, azaz fölé áll a P2 pontnak. Ezután a robot vertikálisan mozog a P2 pontba, ahol egy 0.2 másodperces késleltetés után megfogja a munkadarabot, majd egy 0.5 másodperces késleltetés után vertikálisan elmozog a kiadótálcától 20mm-re, innen pedig tengelymozgással a P3 pontba mozog. Ezután a munkadarabbal elkezdi megközelíteni P4 pontot, amit ugyan úgy teszi, mint a második lépésnél, csuklómozgással megközelíti vertikálisan 20mm távolságra a P4 pontot, majd koordináta mozgással hozzáérinti a munkadarab alját az asztalhoz. Itt ismét 0.2 másodperces késleltetéssel nyit a pneumatikus megfogó, illetve 0,5 másodperces késleltetés előzi meg a P4 ponttól való távolodó koordináta mozgást 20mm-re, majd a robotkar visszaáll alaphelyzetbe. Ha nincs jel és a program lefuttatta a munkadarabra vonatkozó részeket, akkor a robot ahhoz a lépéshez ugrik, ahol kikapcsolja a M_OUT(4) kimenetet, ezzel jelezve, hogy a robotprogram lefutott, leállt.



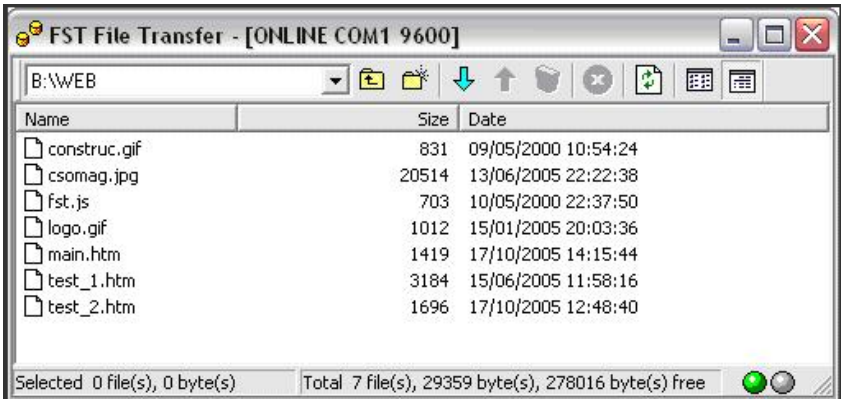
21. ábra Robotprogram hatásvázlata

3.4. Webszerver funkciói

Ebben a témakörben szeretném kifejteni, hogyan is tud kommunikálni és parancsokat fogadni küldeni, egy HTML struktúrájú honlap és maga a PLC.

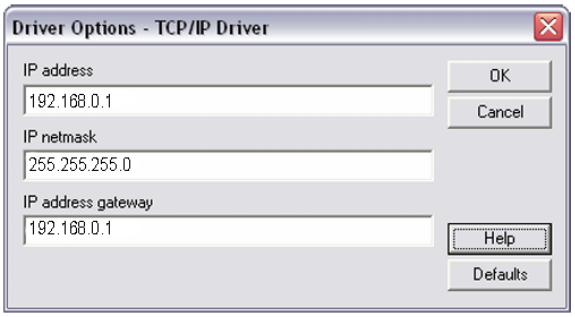
3.4.1. Webszerver konfigurálása

A legfontosabb, hogy a PLC-n levő FTP szerver B könyvtárába (ahol a PLC működéséért felelős megírt program is tárolódik) egy web elnevezésű alkönyvtárt hoztam létre. A beágyazott webszerver ebből a könyvtárból fog dolgozni, itt találja meg a felhasználó által kért oldalakat és az oldalhoz tartozó elemeket (képek, animációk, videók…). Ebbe a könyvtárban kell elhelyezni a JavaScipt számára értelmezhetõ .js fájlt is. Oda kellett figyelnem még arra is, hogy amit főoldalként nyit meg böngészőnk, amikor elérjük a PLC Web-szerverét, annak main.htm nevűnek kell lennie, különben „A lap nem jeleníthető meg” hibaüzenetet kapunk. A további oldalakat, amikre a főoldalból jutunk, már tetszés szerint nevezhetjük el.



22. ábra WEB könyvtár

Ezen kívül nem szabad elfelejteni, az FST szoftverben a TCIPFEC és a WEB_SRVR drivereket hozzáadni a projekthez. Ráadásul a TCPIPFEC-ben meg kell határozni a PLC IP címét, a netmaszkját és az átjáró címét is.



23. ábra PLC IP címének beállítása

Azt, hogy bárhonnan a világról el lehessen érni egy Rasberry Pi-al oldottam meg, amin igazából egy böngésző és egy távoli asztal megosztására alkalmas program kellett csak.

3.4.2. Weblap felépítése

A weblap fő célja, az, hogy a PLC által küldött adatokat valamilyen formában megjelenítse, és parancsokat tudjon visszaküldeni. Erre a célra tökéletesen megfelel a PLC-ben lévő lokális webszerver. A PLC-n közvetlenül tárolhatom a HTML, CSS és JavaScript file-okat.

A kezdőlapot egyszerűen kezelhetőre alakítottam ki.

14. ábra Kezdőlap

Az oldalsó menüpontban a felhasználó választhat a jelenlévő segítségek közül, hogy pontosan mi is, hogyan működik. Illetve a „Belépés” gombra kattintva érthető el akár az admin, akár a vendég felülete is az oldalnak.

3.4.3. Weblap jogosultságok

Mivel magát a honlapot úgy terveztem és alakítottam ki, hogy a világ bármely pontjáról elérhető legyen, ezért bizonyos jogosultsági szerepeket hoztam létre, nevezetesen egy vendég és egy admin felületet. Azért éreztem szükségét ennek, hogy ne tudja bárki irányítani a folyamatokat, hanem a vendég fülön megtekintők csak különböző statisztikákhoz tudjanak jutni. Ahhoz, hogy egyáltalán ezt megvalósítsam el kellett tárolnom azokat az adatok, amik az adminhoz és amik a vendéghez tartoznak. Mivel tudtam, hogy a legegyszerűbb megoldás, az, ha előre definiálom ezeket a bejelentkezési adatokat egy JavaScript függvényben, ezért ezt a megoldást választottam.

```
loginForm.addEventListener('submit', (e) => {
    e.preventDefault();

    const username = document.querySelector('#username').value;
    const password = document.querySelector('#password').value;

    if ((username === 'admin' && password === 'admin')) {
        // Sikeres bejelentkezés az admin felhasználónál

        window.location.href = 'admin.html'; // Átirányítás az admin.html-re
    } else if ((username === 'vendeg' && password === '123')) {
        // Sikeres bejelentkezés a vendég felhasználónál

        window.location.href = 'vendeg.html'; // Átirányítás a vendeg.html-re
    } else {
        // Sikertelen bejelentkezés
        alert('Hibás felhasználónév vagy jelszó!');
    }
});
```

25. ábra A jogosultság megvalósítása

A függvényben először is deklarálni kellett a felhasználó nevét, illetve egy hozzá tartozó jelszót. Egy HA feltételt használtam, hogy az adminhoz tartozó megfelelő jelszót írja be a felhasználó akkor az admin oldalra tovább, ha pedig a vendéghez tartozót akkor a vendég oldalra. Mind a kettő esetében először egy űrlap (loginForm) elküldésére vár, amit egy eseményfigyelő (event listener) figyel. Ha az űrlap elküldésre kerül (submit), akkor az eseményt megakadályozza (e.preventDefault()), megkapja a felhasználónevet és jelszót, majd ezeket ellenőrzi. Ha a felhasználónév és jelszó megegyezik az 'admin' és 'admin', illetve 'vendeg' és '123' értékekkel, akkor az oldalt átirányítja az admin.html-re, illetve vendeg.html-re. Ha a felhasználónév vagy jelszó nem egyezik, akkor egy figyelmeztető üzenetet jelenít meg, miszerint "Hibás felhasználónév vagy jelszó!".

26. ábra Bejelentkezési felület

3.4.4. A jogosultságokhoz tartozó oldalak

Az admin oldalhoz tartozó funkció lista sokkal bővebb, mint a vendéghez tartozó, ugyanis itt a felhasználónak lehetősége van többféle üzemeltetés mód között is váltani, illetve bizonyos időegységet megadni.

27. ábra Admin felület

Az oldal tetején található a rendszer jelenlegi állapota, ami igazából azt mutatja, hogy be van-e kapcsolva vagy nincs. A szöveg ON-ra vált amint változást észlel. A jobb felső sarokban láthatjuk, hogy jelenleg az admin oldalon vagyunk, illetve, ha erre a szövegre kattintunk akkor visszairányít a kezdőoldalra. A bal felső sarokban található a Kamera gomb, amivel élő képet kaphatunk a jelenlegi állapotáról a berendezésről, ha éppen nem vagyunk a közelében.

Két féle bekapcsoló gomb található a honlapon, az egyik egy automatikust üzemeltetést, a másik pedig egy kézi működtetést tesz lehetővé. A kézi működtetés esetén, a rendszer egészen addig nem fog megmozdulni amíg a „munkahengerek irányítása” menüpont alatt egy gomb nem került megnyomásra. Ha az automatikus üzemmódot választjuk, akkor két dolog történik, egyrészt azadagoló állomás elkezdi a munkáját, másrészt a gomb színe és felirata megváltozik kikapcsoló gombbá és ha újra megnyomásra kerül akkor a rendszer az adott ciklus végén leáll. A gombok alapvetően egy JavaScript függvényt hívnak meg, ami továbbít egy adott értéket a PLC-nek.

```
<form>
    <input id="gomb2" type="button" onClick="CISend2('MW0','2')" value="Auto">
</form>
```

28. ábra Gomb HTML struktúrában

A fenti kódrészlet egy egyszerű HTML űrlapot tartalmaz, amelyben egy gomb található. Nézzük meg részletesen, mit jelentenek a különböző részei:

- <form>: Ez egy HTML űrlap eleme, amely lehetővé teszi az adatok beküldését a szerver felé.
- <input>: Ez egy bemeneti mező, amely lehetővé teszi a felhasználó számára az adatok megadását.
- id="gomb1": Ez az input elem azonosítója, amely egyedi azonosítót ad a HTML elemnek, így könnyen hivatkozhatunk rá CSS vagy JavaScript kódban.
- type="button": Ez meghatározza, hogy a bemeneti mező típusa egy gomb lesz. Ez a gomb nem fogja elküldeni az űrlap tartalmát, hanem a felhasználó által meghatározott műveletet fog végrehajtani a gombra kattintáskor.
- onClick="CISend('MW0','2')": Ez a gombra való kattintás eseménykezelője. Amikor a gombra kattintanak, a CISend('MW0','2') JavaScript függvény fut le. Ez a függvény valószínűleg a gomb kattintására definiált egyéb műveleteket hajt végre.
- value="Auto": Ez a gomb szövege, amit a gombon megjelenít. A felhasználó számára látható szöveg, amit a gombon látunk.

Ez a kód tehát egy űrlapot tartalmaz egyetlen gombbal, amelynek egyedi azonosítója van, és amelynek kattintásakor egy JavaScript függvény fut le.

```
function CISend2(MW0,Value) {
  var found;
  var page;


  if ( CheckValue(2) == false )
  {
    return;
  }
    page = MyStrip();

  window.location.href = page + "?ci:" + "M" + MW0 + "=" + 2;
}
```

29. ábra A gombhoz tartozó JavaScript függvény

A CISend2 függvény egy JavaScript függvény, amely két paramétert vár: MW0 és Value. Az első sorban két változó, found és page, deklarálásra kerülnek. A következő sorokban egy feltételes utasítás található (if), amely ellenőrzi, hogy a CheckValue függvény visszatérési értéke hamis-e, vagyis false. Ha ez a feltétel igaz, a függvény visszatér, és nem hajtja végre a további utasításokat. Ha a feltétel hamis (CheckValue(2) igaz), akkor a page változó értékét az MyStrip() függvény visszatérési értéke kapja. Ezután a window.location.href értéke beállításra kerül egy új URL-re, amely tartalmazza a page változót, a MW0 értékét (a "M" string előtaggal) és a Value értékét, amely mindig 2. Ezt az új URL-t a böngésző címsorába irányítják át, így a felhasználó átirányításra kerül a generált URL-re. Ahhoz, hogy a függvény megfelelően működjön szükség van a MyStrip és a CheckValue nevű függvényeknek is a definiálásához.

```
function MyStrip() {
  var found;
  var href;

  href = this.location.href;
  found = href.indexOf("?");
  if ( found > 0 ) {
    href = href.substring(0, found);
  }
  return href;
  }

function CheckValue(Value) {

  if ( isNaN(Value) == true )
  {
    console.log("Value (" + Value + ") is not a number");

    return false;
  }

  if ( (Value < -65535) || (Value > 65535) )
  {
    console.log("Value is out of range");

    return false;
  }
  return true;
}
```

20. ábra Segédfüggvények

1. MyStrip(): Ez a függvény egy URL-címet (location.href) vizsgál, és ha talál a címben egy kérdőjel (?) karaktert, akkor az összes karaktert törli a kérdőjelig tartó részből. Ezután a tisztított URL-címet visszaadja.
2. CheckValue(Value): Ez a függvény egy értéket kap paraméterként, majd ellenőrzi, hogy az adott érték egy szám-e (isNaN(Value) segítségével). Ha az érték nem szám, akkor kiír egy üzenetet a konzolra, amely jelzi, hogy az érték nem szám. Ezt követően ellenőrzi, hogy az érték -65535 és 65535 között van-e. Ha az érték kívül esik ezen a tartományon, akkor szintén kiír egy üzenetet a konzolra, amely jelzi, hogy az érték kívül esik a megengedett tartományon. Ha az érték szám és megfelel a tartománynak, akkor a függvény true értékkel tér vissza, egyébként false értékkel.

Az első függvény tisztítja a URL-címet, a második pedig ellenőrzi egy adott érték szám típusát és tartományát. Az ilyen típusú függvényekkel oldottam meg a PLC és a honlap közötti kommunikációk zömét. Ugyanis a függvényben található MW0 érték egyenértékű a PLC regiszterében található FW0 értékkel, a kezdőbetűből való eltérés a nyelv adja, ugyanis a PLC német, míg a honlap angol nyelven tárolja ezeket a szavakat. A munkahengerek irányítása fül alatt található gombok, ugyanezen szisztéma szerint épülnek fel. Mindegyik gomb mögé egy hasonló JavaScript függvény van rendelve, ami valamilyen decimális értéket küld a PLC-nek és annak megfelelően aktívalja valamelyik bemenetét a PLC-n. Viszont ez a sima gomboknál működik így, ugyanis a rendszert bekapcsoló gombom egy nyomógomb.

Ezt JavaScript függvény segítségével tudtam megvalósítani, esemény figyelőt kellett létrehoznom, ami figyeli, hogy hányszor volt már megnyomva a gomb, és ennek függvényében más-más függvényt hívjon meg.

```
var button1 = document.getElementById("gomb1");
var isButton1Clicked = false;


button1.addEventListener("click", function() {
  if (!isButton1Clicked) {
    // Háttérszín módosítása az első kattintás után
    button1.style.backgroundColor = "green";
    isButton1Clicked = true;
    CISend2 ('MW0'; '2');
  } else {
    // Háttérszín módosítása a második kattintás után
    button1.style.backgroundColor = "red";
    CISend31 ('MW31';'31');
  }
});
```

31. ábra Nyomógomb megvalósítása JavaScripttel

Az első két sor a button1 változót deklarálja és inicializálja. A document.getElementById("gomb1") rész azonosítja azon HTML elemet, amelynek az azonosítója (ID-je) "gomb1". Ezt az elemet hozzárendeli a button1 változóhoz. Ezt követően egy isButton1Clicked nevű változót deklarálnak és értéket adnak neki, ami kezdetben hamis (false). Ezután egy eseményfigyelő (event listener) van hozzáadva a button1 elemhez, amely a "click" eseményt figyeli. Amikor a gombra kattintanak, az eseményfigyelő elindul. Az eseményfigyelőben egy feltétel található, amely ellenőrzi, hogy a isButton1Clicked változó hamis-e vagy sem. Ha hamis (azaz az első kattintás), akkor a gomb háttérszínét zöldre változtatja, beállítja a isButton1Clicked változót igazra, majd meghív egy CISend2 függvényt a paraméterekkel ('MW0'; '2'). Ha a isButton1Clicked változó igaz (azaz a második kattintás), akkor a gomb háttérszínét pirosra változtatja, majd meghív egy CISend31 függvényt a paraméterekkel ('MW31'; '31').

Az idő megadása rész nagyon hasonlóan működik, csak nem flagword-öket töltöttem fel értékekkel, hanem regisztereket. A HTML oldal már csak lekérdezért folytat, aminek a struktúrája a következőképpen épül fel:

```
<p align="center"><font size="5"color="#008000"> Mostani ciklus:
</font><font size="5" color="#FF0000"><FSTCI dzw0></font>
```

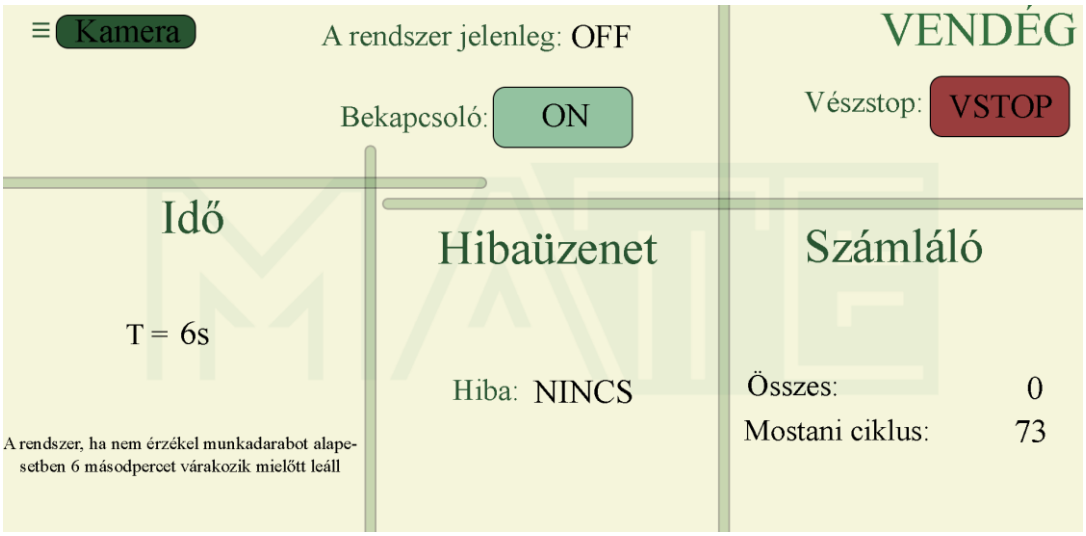
32. ábra Számláló kiíratása a honlapon

Az első sor csak a megjelenítését szolgálja, a második fele végén található <FSTCI dzw0> ami az oldal betöltődésekkor megjeleníti az adott regiszterben található értéket. Mind a kettő számláló ugyanazon az elven épül fel, azzal a különbséggel, hogy a ’Mostani ciklus’ számlálója minden PLC program indításkor nullázódik. Viszont ez egy statikus érték, ami csak akkor töltődik be amikor frissítem az adott oldalt, ezért be kellett állítanom egy HTML utasítást ami három másodpercenként frissíti az oldalt, hogy mindig frissek legyenek az adatok. ’<meta http-equiv="Refresh" content="3;Admin.html">’ Három másodperc alatt egy ciklus biztosan nem következik be.

A hiba üzeneteket is ugyanezzel a módszerrel tudom kiiratni, attól függően, hogy melyik regiszter értékét tölti be a PLC, a megfelelő hibaüzenet fog megjelenni.

Végül az utolsó megtalálható gomb, pedig a kamera gombja, ami egy új HTML oldalra irányit és élő képet ad a rendszerről, mivel a webszerver a Rasberry Pi-ról fut, ezért bármilyen kamera megfelel, amit össze lehet kapcsolni vele, ez lehet USB vagy LAN-on vezérelt kamera is.

Az admin oldalon ezek a funkciók találhatók, ezzel szemben a vendégoldal funkciócsökkentett.



33. ábra Vendég oldal

Ezen az oldalon is elérhetőek az alapvető funkciók, azonban itt csak egy bekapcsoló gomb található, ami az automatikus üzemben fogja a rendszert elindítani. A másik szembe tűnő különbség, hogy nem lehet parametrizálni az időt, az alapvető 6s-os értékkel fog a rendszer működni.

4. Továbblépési lehetőségek

Az egyik legfontosabb továbblépési lehetőség az maga a rendszernek a biztonsági követelménye. Mivel ez egy oktatási eszköz, ezért jelen működésében, ha még egy oktató felügyeletében rendben is van. Biztonsági relék alkalmazása, ami a PLC-nek küldene egy információt arról, hogy egy esetleges leállás után a rendszer újraindítható-e. Védő elemek alkalmazása, hogy ne is történhessen testi sérülés, ha az ember a munkaterületen belülre nyúlna. Ezen kívül, amikor a rendszer leáll teljesen le kellene szellőztetni. Gépészetileg is újra lehetne gondolni magát az MPS adagoló állomást, hogy oktatás technikailag is jobb legyen, illetve esetleges funkció bővítés miatt. A hibás elemek cseréje is fontos.

Szoftveres részen, a honlap kialakítását át lehet alakítani egy dinamikus honlapra, ahol a PLC közvetlenül az internetre tölti fel az adatait, egyszerű relációs adatbázisba, például sql adatbázisba, és a HTML struktúrájú honlapot PhP-ra változtatni, hogy azonnali adatokat tudjunk kinyerni a PLC-ből.

5. Gazdasági számítás

A feladat elvégzéséhez a Szent István Egyetem Mechatronika tanszéke biztosította a adagolóállomást, annak eszközeit, a robotkart, a PLC-t és a hozzájuk tartozó szoftvereket, melyet ezúton is szeretnék megköszönni. Ebből az okból kifolyólag ezeknek bekerülési értéke nem volt. Tehát csak az általam használt számítógépet, a weboldalhoz szükséges szoftvereket és a feladat megoldásához szükséges munkaórákat vettem figyelembe. A feladat elkészítéséhez bármilyen laptop alkalmas volt, mely képes a Windows operációs rendszer futtatására és rendelkezik UTP csatlakozóval. Illetve a távezérléshez használt Rasberry Pi, hozzá tartozó bármilyen HDMI csatlakozóval ellátott monitor és egy egér, billentyűzet, mint periféria, illetve egy switch. A programozási és összeszerelési feladatokat 6 héten keresztül heti 1 nap végeztem napi 8 órában.

Megnevezés	Darabszám	Egységár	Összesen
Munkaóra	48 óra	3 500 Ft	168 000 Ft
Laptop	1 db	70 000 Ft	70 000 Ft
Rasberry Pi	1 db	45 590 Ft	45 590 Ft
Monitor	1 db	30 000 Ft	30 000 Ft
Billentyűzet	1 db	2 000 Ft	2 000 Ft
Egér	1 db	2 000 Ft	2 000 Ft
Switch	1 db	10 690 Ft	10 690 Ft
-	-	-	**328 280 Ft**

6. Összefoglalás

A jelenlegi piacvezető vállalatoknál a termékek előállításában az egyik elengedhetetlen szempont a lehető legalacsonyabb költségek, az elvárt minőség és a gyorsaság. Ezeket a kritériumokat a legkorszerűbb technológiák segítségével próbálják elérni, különösen az automatizált folyamatokkal. Ennek érdekében a szakdolgozatom témája az ipari gyártás és az automatizálás területének mélyebb megértése volt. A kutatás célja az volt, hogy egy teljesen automata rendszert fejlesszek ki, amely képes mérési és adagolási feladatokat végrehajtani emberi beavatkozás nélkül. Ehhez gyártósori körülményeket szimuláltam, és olyan rendszert terveztem, amelyben egy programozható logikai vezérlő (PLC) irányítja a folyamatokat. A mérési feladatokat egy adagolóállomás végzi, míg a munkadarab áthelyezését egy robotkar végzi el. Az egész rendszert egy távdiagnosztikai honlap segítségével lehet figyelni és nyomon követni. Fontos volt, hogy a logikai vezérlő ne csak a folyamatot működtesse, hanem biztonsági intézkedéseket is hajtson végre esetleges problémák esetén. Ezek a biztonsági mechanizmusok biztosítják, hogy a folyamat csak akkor kezdődjön el, ha minden feltétel teljesül, és hogy esetleges balesetek vagy rendszerhibák esetén a rendszert könnyen le lehessen állítani és újra elindítani. Ezen feladatom során a szükséges eszközöket, mint például az adagolóállomást és a robotkart, a Magyar Agrár- és Élettudományi Egyetem Mechatronika Tanszéke biztosította, amit ezúton szeretnék megköszönni. A folyamat során először is a adagolóállomás segítségével a rendszer elvégzi az a folyamatot ahol az adatok mérése történik, majd továbbítja a munkadarabot a robotnak, amely azt a megfelelő pozícióba helyezi. Ennek az automatizált folyamatnak az állapotát és az átpakolt elemek mennyiségét és idejét egy HTML alapú weboldalon keresztül lehet figyelemmel kísérni. Az állomás két üzemmódban működött: egy manuális módban, ahol minden kézi indítással történt, és egy automata módban, ahol az állomás emberi beavatkozás nélkül volt képes a munkadarabok mérésére és parametrizálására. A távvezérlés részét, hogy a világ bármely pontjáról is akár lehessen irányítani, egy miniszámítógép (Rasberry Pi) és egy LAN switch segítségével oldottam meg. Mind a három eszköz képes volt LAN-on kommunikálni egymással, ezért, ha az eszközök rálátnak az internetre, és a miniszámítógépre rácsatlakozunk távoli asztal eléréssel, akkor közvetlenül láthatjuk és irányíthatjuk a folyamatot.

7. Summary

At current market-leading companies, one essential factor in product manufacturing is achieving the lowest possible costs, expected quality, and speed. These criteria are pursued using the most advanced technologies, especially through automated processes. Therefore, the topic of my thesis was to deepen the understanding of industrial manufacturing and automation. The research aimed to develop a completely automated system capable of performing measurement and dosing tasks without human intervention. To achieve this, I simulated manufacturing line conditions and designed a system where a Programmable Logic Controller (PLC) controls the processes. Measurement tasks are performed by a dosing station, while the transfer of workpieces is carried out by a robotic arm. The entire system can be monitored and tracked through a remote diagnostic website. It was important for the logic controller not only to operate the process but also to implement safety measures in case of potential issues. These safety mechanisms ensure that the process only starts when all conditions are met and that, in case of accidents or system failures, the system can be easily stopped and restarted. During this task, necessary equipment such as the dosing station and the robotic arm were provided by the Department of Mechatronics at the Hungarian University of Agriculture and Life Sciences, and I would like to express my gratitude for their support. In the process, the system first performs the measurement where data is collected using the dosing station and then transfers the workpiece to the robot, which places it in the correct position. The status of this automated process and the quantity and time of transferred elements can be monitored through an HTML-based website. The station operated in two modes: a manual mode, where everything was done with manual initiation, and an automatic mode, where the station could measure and parameterize workpieces without human intervention. To enable remote control from anywhere in the world, I used a mini-computer (Raspberry Pi) and a LAN switch. All three devices were capable of communicating with each other over the LAN. Therefore, if the devices have internet access and we connect to the mini-computer using remote desktop access, we can directly see and control the process.

8. Hivatkozások

1. Sumicz Árpád (2021): Magyar Kereskedelmi és Iparkamara gondozásában kidolgozott Villanyszerelő mesterképzésre felkészítő jegyzet., Budapest.

2. Dr. Szabó Tibor (2011): Gépészeti automatizálás., Tatabánya.

3. Dr. Nemes József (2012): Irányítástechnika., Nyugat-magyarországi Egyetem.

4. Balpataki Antal (2019): Járműhidraulika és -pneumatika., Budapest.

5. Veres György (2012): Hidraulika és pneumatika., Pannon Egyetem.

6. Mester Gyula (2016): Robotika., Szeged.

7. Laczik Bálint (2012): Robottechnika., Budapest.

8. Dr. Kulcsár Béla (2011): Robottechnika., Budapest.

9. Kvaszingerné Prantner Csilla (2013): Webfejlesztés., Eger.

10. B. Siciliano, O. Khatib (2008): Springer Handbook of Robotics. Springer, Berlin.

11. Kulcsár B. (2011): Robottechnika I. Typotex Kiadó, Budapest.

12. Dr. Donner Cs. (2000): A Robottechnika alapjai. Gura Nyomda Bt., Zalaegerszeg.

13. Mester Gy. (2003): Intelligens Robotok és Rendszerek. Előadás: Szegedi Tudomány Egyetem, Szeged, 2003.

14. Dr. Ajtonyi I., Dr. Gyuricza I. (2011): Programozható irányítóberendezések, hálózatok és rendszerek. Műszaki Könyvkiadó Kft., Budapest.

15. Bokor J., Gáspár P., Soumelidis A. (2011): Irányítástechnika II. Typotex Kiadó, Budapest.

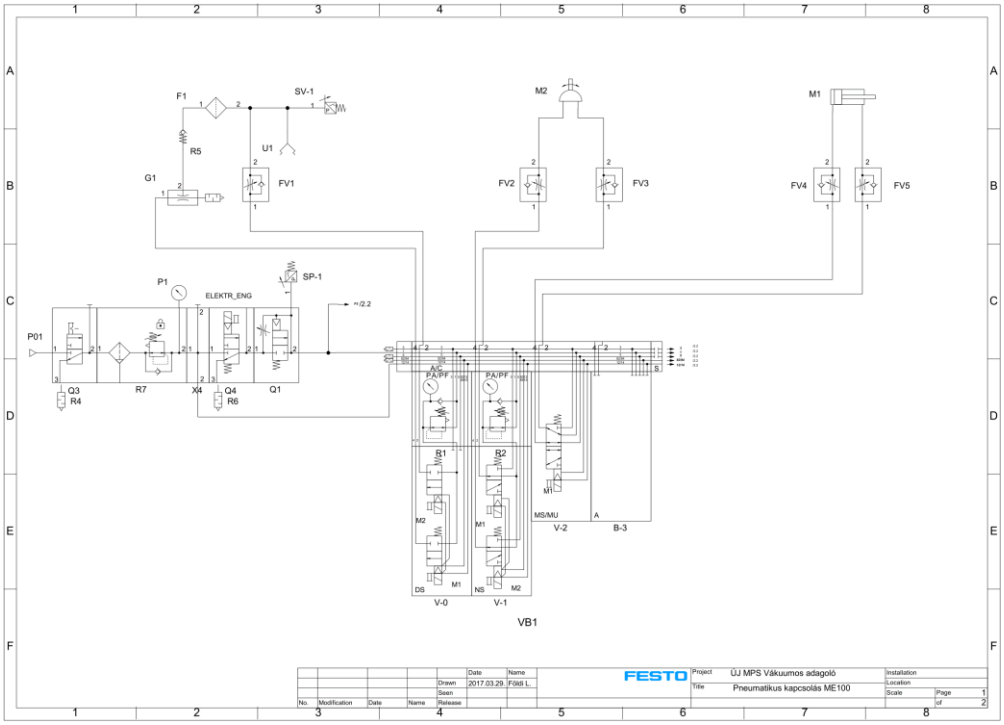
16. W. Bolton (2006): Programmable Logic Controllers. Elsevier Science & Technology, Oxford.

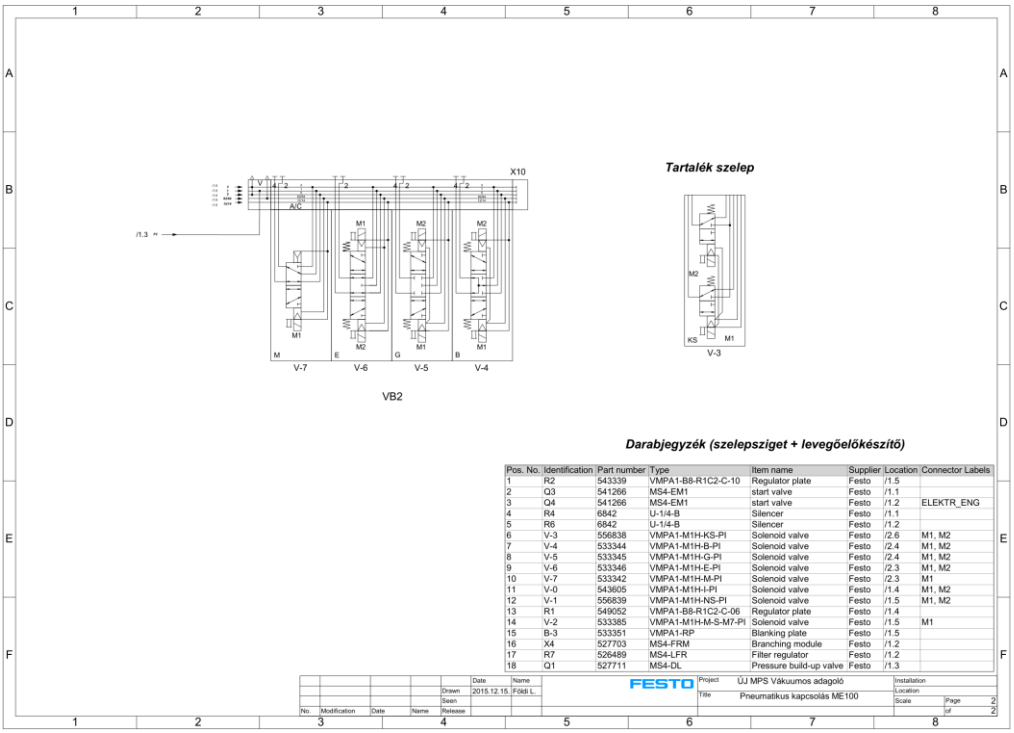
17. F. Ebel, C. Knoblich (2003): Festo Testing Station Manual. Festo Didactic GmbH, Denkendorf.

18. Barber (1997): Pneumatic Handbook, 8th Edition. Elsevier Science & Technology, Amsterdam.

9. Mellékletek

1. MPS Adagoló





Darabjegyzék (szelepsziget + levegőelőkészítő)

Pos. No.	Identification	Part number	Type	Item name	Supplier	Location	Connector Labels
1	R2	543339	VMPA1-B8-R1C2-C-10	Regulator plate	Festo	/1.5	
2	Q3	541266	MS4-EM1	start valve	Festo	/1.1	
3	Q4	541266	MS4-EM1	start valve	Festo	/1.2	ELEKTR_ENG
4	R4	6842	U-1/4-B	Silencer	Festo	/1.1	
5	R6	6842	U-1/4-B	Silencer	Festo	/1.2	
6	V-3	556838	VMPA1-M1H-KS-PI	Solenoid valve	Festo	/2.6	M1, M2
7	V-4	533344	VMPA1-M1H-B-PI	Solenoid valve	Festo	/2.4	M1, M2
8	V-5	533345	VMPA1-M1H-G-PI	Solenoid valve	Festo	/2.4	M1, M2
9	V-6	533346	VMPA1-M1H-E-PI	Solenoid valve	Festo	/2.3	M1, M2
10	V-7	533342	VMPA1-M1H-M-PI	Solenoid valve	Festo	/2.3	M1
11	V-0	543605	VMPA1-M1H-I-PI	Solenoid valve	Festo	/1.4	M1, M2
12	V-1	556839	VMPA1-M1H-NS-PI	Solenoid valve	Festo	/1.5	M1, M2
13	R1	549052	VMPA1-B8-R1C2-C-06	Regulator plate	Festo	/1.4	
14	V-2	533385	VMPA1-M1H-M-S-M7-PI	Solenoid valve	Festo	/1.5	M1
15	B-3	533351	VMPA1-RP	Blanking plate	Festo	/1.5	
16	X4	527703	MS4-FRM	Branching module	Festo	/1.2	
17	R7	526489	MS4-LFR	Filter regulator	Festo	/1.2	
18	Q1	527711	MS4-DL	Pressure build-up valve	Festo	/1.3	

MAGYAR AGRÁR- ÉS ÉLETTUDOMÁNYI EGYETEM

Szent István Campus, Gödöllő
Cím: 2100 Gödöllő, Páter Károly utca 1.
Tel.: +36-28/522-000
Honlap: https://godollo.uni-mate.hu

NYILATKOZAT

szakdolgozat nyilvános hozzáféréséről és eredetiségéről

A hallgató neve:	Czugh Soma
A Hallgató Neptun kódja:	EMS3KB
A dolgozat címe:	Robottal támogatott adagoló állomás PLC programozása és távvezérlésére alkalmas rendszer tervezése
A megjelenés éve:	2023
A konzulens intézetének neve:	Gépipari Technológiai Intézet
A konzulens tanszékének a neve:	Mechatronika tanszék

Kijelentem, hogy az általam benyújtott szakdolgozat egyéni, eredeti jellegű, saját szellemi alkotásom. Azon részeket, melyeket más szerzők munkájából vettem át, egyértelműen megjelöltem, és az irodalomjegyzékben szerepeltettem.

Ha a fenti nyilatkozattal valótlant állítottam, tudomásul veszem, hogy a záróvizsga-bizottság a záróvizsgából kizár és a záróvizsgát csak új dolgozat készítése után tehetek.

A leadott dolgozat, mely PDF dokumentum, szerkesztését nem, megtekintését és nyomtatását engedélyezem.

Tudomásul veszem, hogy az általam készített dolgozatra, mint szellemi alkotás felhasználására, hasznosítására a Magyar Agrár- és Élettudományi Egyetem mindenkori szellemitulajdon-kezelési szabályzatában megfogalmazottak érvényesek.

Tudomásul veszem, hogy dolgozatom elektronikus változata feltöltésre kerül a Magyar Agrár- és Élettudományi Egyetem könyvtári repozitori rendszerébe. Tudomásul veszem, hogy a megvédett és

- nem titkosított dolgozat a védést követően
- titkosításra engedélyezett dolgozat a benyújtásától számított 5 év eltelte után

nyilvánosan elérhető és kereshető lesz az Egyetem könyvári repozitori rendszerében.

Kelt: 2023. év 11. hó 04. nap

Hallgató aláírása

MAGYAR AGRÁR- ÉS ÉLETTUDOMÁNYI EGYETEM

Szent István Campus, Gödöllő
Cím: 2100 Gödöllő, Páter Károly utca 1.
Tel.: +36-28/522-000
Honlap: https://godollo.uni-mate.hu

NYILATKOZAT

Czugh Soma (hallgató, Neptun azonosítója: EMS3KB) konzulenseként nyilatkozom arról, hogy a záródolgozatot/szakdolgozatot/diplomadolgozatot/portfóliót[1] áttekintettem, a hallgatót az irodalmi források korrekt kezelésének követelményeiről, jogi és etikai szabályairól tájékoztattam.

A záródolgozatot/szakdolgozatot/diplomadolgozatot/portfóliót a záróvizsgán történő védésre **javaslom / nem javaslom**[2].

A dolgozat állam- vagy szolgálati titkot tartalmaz: igen nem*[3]

Gödöllő, 2023. év november hó 2. nap

Mayerné Sárközi Eszter
egyetemi adjunktus
belső konzulens
MATE SZIC Műszaki Intézet
Mechatronika Tanszék

[1] A megfelelő dolgozattípus meghagyása mellett a többi típus törlendő.
[2] A megfelelő aláhúzandó.
[3] A megfelelő aláhúzandó.