|  |
| --- |
| **Szegedi Tudományegyetem**  **Informatikai Intézet** |

|  |  |
| --- | --- |
| **Ipari folyamat szimulációja és irányítása programozható logikai vezérlővel** | |
| Diplomamunka | |
| Készítette: | Témavezető: |
| **Miklós Árpád** | **Dr. Kincses Zoltán** |
| mérnök informatikus szakos hallgató | egyetemi adjunktus |

|  |
| --- |
| Szeged  2021 |

**Ipari folyamat szimulációja és irányítása programozható logikai vezérlővel**

**Diplomamunka mérnök-informatikus MSc szakos hallgató számára**

**Témavezető: Dr. Kincses Zoltán**

**Témakör: ipari informatika, rendszer szimuláció, irányítás**

**Műszaki Informatika Tanszék**

**A feladat leírása, a munka célja:**

A hallgató feladata egy ipari folyamat emulációjának és irányításának elkészítése. A munka célja, egyrészt az Országos Ajtonyi István Irányítástechnikai Programozó Versenyen szereplő technológia emulációjának elkészítése egy HIL szimulátor eszköz segítségével. A munka másik célja az így elkészített rendszer irányításának megvalósítása egy programozható logikai vezérlővel. Az elkészült munka később jól alkalmazható a későbbi PLC versenyekre történő felkészítésben.

**A munkavégzés fontosabb lépései:**

* A HIL szimulátor eszköz és a hozzá tartozó szoftverek megismerése, ismertetése
* Az emulálni kívánt technológia megismerése
* A programozható logikai vezérlő és a hozzá tartozó szoftver megismerése, ismertetése
* A HIL emuláció elkészítése
* Az elkészült szimuláció irányításának megvalósítása programozható logikai vezérlővel
* Hibakezelés
* Tesztelés
* A dolgozat megírása

**A fejlesztéshez rendelkezésre álló erőforrások:**

* OMRON CJ2M PLC, I/O egységek és a programozásához szükséges szoftver
* Lucas Nülle I/O interfész PRO/TRAIN-hez
* Lucas Nülle BORIS szoftver csomag
* Lucas Nülle PRO/TRAIN

**A jelentkezés feltételei:**

* Érdeklődés a PLC alapú irányítások és vizualizációjuk iránt
* Angol nyelvtudás

# Tartalmi összefoglaló

* **A téma megnevezése:**

Egy ipari folyamat emulálása hardware-in-the-loop (HIL) szimulátor segítségével és az emulált ipari folyamat irányítása programozható logikai vezérlővel.

* **A megadott feladat megfogalmazása:**

Meg kell valósítanom egy megfelelően összetett ipari folyamat emulálását és annak irányítását programozható logikai vezérlővel azért, hogy szemléltessem a HIL szimulátorral támogatott fejlesztést és tesztelést. Az alap feladatkiíráson túl meg kell terveznem és meg kell valósítanom egy alternatív megoldást a feladat megoldásához biztosított HIL szimulátor helyett, amelyik egyben kompatibilis is azzal.

* **A megoldási mód:**
  + A feladat megoldásához biztosított szoftverek és hardverek megismerése.
  + A kiválasztott ipari folyamat emulációjának megvalósítása és tesztelése.
  + Az emulált ipari folyamat irányításának megvalósítása PLC segítségével.
  + A HIL szimulátor hardveres és szoftveres elemeinek a tanulmányozása.
  + A HIL szimulátorral kompatibilis szoftver és hardver fejlesztése és tesztelése.
* **Alkalmazott eszközök, módszerek:**
  + Lucas-Nülle I/O interfész, Omron CJ2M-CPU32 PLC, MikroElektronika EasyPIC v7 fejlesztőlap, Tektronix MSO2024B oszcilloszkóp, WinFACT 7 BORIS szimulációs szoftver, CX-Programmer 9.1, mikroC PRO for PIC 6.6.2, IAR Embedded Workbench for Arm 7.10.1, Visual Studio Code 1.54.3 és Qt Creator 4.14.1 fejlesztői környezetek, Altium Designer 17.1 elektronikai tervező szoftver, Eltima Serial Port Monitor 6.0.235 segédprogram
  + dokumentumelemzés, megfigyelés, kísérlet, mérés, fejlesztés, tesztelés
* **Elért eredmények:**

Megvalósítottam a kiválasztott ipari folyamat emulálását a szimulációs szoftverrel és elkészítettem a hozzá tartozó irányítást. Megvizsgáltam az I/O interfész és a BORIS közötti kommunikációt az I/O interfész felnyitása nélkül, majd ez alapján megterveztem egy helyettesítő áramkör alapjait és írtam egy könyvtárat az I/O interfésszel való kommunikációhoz, amivel újra megvalósítottam az ipari folyamat emulálását QML-ben.

* **Kulcsszavak:**

ipari informatika, rendszer emuláció, HIL, irányítás, PLC, technológia-visszafejtés

Tartalomjegyzék

[Tartalmi összefoglaló 2](#_Toc69951478)

[Bevezetés 6](#_Toc69951479)

[1. Felhasznált hardverek és szoftverek 7](#_Toc69951480)

[1.1. Az ipari folyamat emulálásának eszközei 7](#_Toc69951481)

[1.1.1. A Lucas-Nülle I/O interfész 7](#_Toc69951482)

[1.1.2. A WinFACT 7 BORIS szimulációs szoftver 8](#_Toc69951483)

[1.1.3. A Flexible Animation Builder beépülőmodul 10](#_Toc69951484)

[1.2. Az ipari folyamat irányításának eszközei 12](#_Toc69951485)

[1.2.1. Az Omron CJ2M programozható logikai vezérlő 12](#_Toc69951486)

[1.2.2. A CX-Programmer programozószoftver 13](#_Toc69951487)

[1.3. A technológia-visszafejtés eszközei 14](#_Toc69951488)

[1.3.1. Az Eltima Serial Port Monitor segédprogram 14](#_Toc69951489)

[1.3.2. A MikroElektronika EasyPIC v7 fejlesztőlap 16](#_Toc69951490)

[1.3.3. A mikroC PRO for PIC fejlesztői környezet 17](#_Toc69951491)

[1.4. Az alternatív emuláció megvalósításának eszközei 17](#_Toc69951492)

[1.4.1. A Qt Creator fejlesztői környezet 17](#_Toc69951493)

[1.4.2. Az Altium Designer elektronikai tervezőszoftver 18](#_Toc69951494)

[1.4.3. Az IAR Embedded Workbench for Arm fejlesztői környezet 18](#_Toc69951495)

[2. Az emulálni kívánt ipari folyamat 19](#_Toc69951496)

[3. Interaktív vizualizáció készítése a BORIS eszköztárával 21](#_Toc69951497)

[4. Az ipari folyamat emulációja és irányítása 22](#_Toc69951498)

[4.1. Az emuláció megvalósításának bemutatása 23](#_Toc69951499)

[4.1.1. A robotok és a futószalagok vizualizációja FAB segítségével 23](#_Toc69951500)

[4.1.2. A robotok mozgatása 23](#_Toc69951501)

[4.1.3. A karosszériák mozgatása 23](#_Toc69951502)

[4.1.4. A cellák ellenőrzőlogikái 23](#_Toc69951503)

[4.2. Az irányítás megvalósításának bemutatása 23](#_Toc69951504)

[5. A BORIS projektek hordozhatósági problémájának megoldása 24](#_Toc69951505)

[6. Az ipari folyamat emulálásának alternatív megoldása 26](#_Toc69951506)

[6.1. Az I/O interfész működésének behatásmentes visszafejtése 26](#_Toc69951507)

[6.1.1. BORIS és az I/O interfész közötti kommunikáció lehallgatása 26](#_Toc69951508)

[6.1.2. Az I/O interfész működésének utánzása a BORIS számára 26](#_Toc69951509)

[6.1.3. Az I/O interfész működtetése a BORIS használata nélkül 26](#_Toc69951510)

[6.2. Az emuláció megvalósítása alternatív eszközökkel 26](#_Toc69951511)

[6.2.1. A karosszériák mozgatása 26](#_Toc69951512)

[6.2.2. A futószalagok megvalósítása és működtetése 26](#_Toc69951513)

[6.2.3. A daru megvalósítása és működtetése 26](#_Toc69951514)

[6.2.4. A robotok megvalósítása és működtetése 26](#_Toc69951515)

[7. Az I/O interfészt helyettesítő elektronika 27](#_Toc69951516)

[8. Konklúzió 28](#_Toc69951517)

[Irodalomjegyzék 29](#_Toc69951518)

[Nyilatkozat 30](#_Toc69951519)

# Bevezetés

A minőség korunk egyik legmeghatározóbb hívószava. Olyan követelmény, amelyre a piaci versenyképesség fenntartása érdekében folyamatosan kiemelt figyelmet kell szentelni. Habár a fogalma az elmúlt évtizedek során sokat változott, köznapi értelemben a minőség annak a mércéje, hogy egy termék vagy szolgáltatás milyen mértékben elégíti ki a vele szemben támasztott elvárásokat vagy igényeket. Az ipari automatizálásban a minőség általában olyan tulajdonságokhoz köthető, mint a megbízhatóság, a biztonság, a karbantarthatóság és a teljesítmény, amelyeknek a biztosításában kiemelt szerepe van a rendszeres és átfogó tesztelésnek.

A hardware-in-the-loop (HIL) tesztelés egyike azon módszereknek, amelyek az ipari automatizálásban is jól alkalmazhatók tesztelési és fejlesztési célokra. Ez a fajta eljárás a vizsgált eszközt olyan fizikai környezetben működteti, amelynek a jeleit egy virtuális rendszer biztosítja oly módon, hogy azok a valós rendszerrel megegyezőnek tűnjenek. Így nem csak költséghatékonyabbá tehető a tesztelés, hiszen nincs szükség egy valódi rendszerre, de olyan szélsőséges körülmények tesztelését is lehetővé teszi, amelyeknek a valóságban akár súlyos anyagi károkkal járó vagy emberéleteket is követelő következményei lehetnek, ha egy teszt elbukik.

Ez a dolgozat egy összetettebb ipari folyamat segítségével mutatja be a HIL tesztelés és a HIL szimulátorok működését a gyakorlatban. Ennek érdekében az első fejezetben először azok a hardverek és szoftverek kerülnek bemutatásra, amelyek a feladat megoldásához lettek felhasználva. Ezt követi az emulálni kívánt ipari folyamat rövid bemutatása a második fejezetben. A harmadik fejezet a HIL szimulátor szoftverével történő animációkészítésre korlátozódik, ugyanis a HIL szimulátor megismerése Görbedi Ákos kollégával közösen végzett munka volt, és az ő diplomamunkája részletesen ismerteti ugyanennek a szoftvernek a használatát. Az emulált ipari folyamatnak a megvalósítása a negyedik fejezetben kerül bemutatásra, míg a bemutatott megvalósítást is érintő hordozhatósági probléma és annak megoldása az ötödik fejezetben kerül ismertetésre. Végezetül bemutatásra kerül az emulált ipari folyamat alternatív megvalósítása és a HIL szimulátor hardverének helyettesítő elektronikája a hatodik és a hetedik fejezetekben. A feladat kidolgozása közben megszerzett tapasztalatok és az elért eredmények a nyolcadik fejezetben kerülnek összefoglalásra.

# Felhasznált hardverek és szoftverek

A feladat megoldásához felhasznált hardverek és szoftverek a részfeladatok alapján jól csoportosíthatók. Az ipari folyamat emulálásához a HIL szimulátor szoftvere és hardvere lettek igénybe véve, míg az irányításhoz egy széles körben alkalmazott PLC és a programozását lehetővé tevő programozószoftver. A HIL szimulátor technológiájának visszafejtéséhez elsősorban egy olyan szoftver került felhasználásra, amelyik képes a PC és az I/O interfész közötti kommunikáció lehallgatására, majd egy fejlesztőlap a hozzá tartozó fejlesztői környezettel a kommunikáció megfigyeléséből származó adatok helyességének az igazolására. Az alternatív HIL szimulátor és az ipari folyamat alternatív emulálásának megvalósításához a legnépszerűbbnek számító tervezőszoftverek és fejlesztői környezetek lettek felhasználva.

## Az ipari folyamat emulálásának eszközei

A HIL szimulátor két különálló gyártó egymással kompatibilis termékeiből tevődik össze. Az emulációhoz használt szoftver alapvetően egy széleskörűen felhasználható grafikus fejlesztői környezet, amelyik támogatja a bővítmények használatát is, míg a PLC-vel kommunikáló hardver egy olyan eszköz, amelyik rendelkezik egy bővítményként beépülő meghajtóprogrammal ugyanehhez a szoftverhez.

### A Lucas-Nülle I/O interfész

A CO3715-1H típusjelzésű I/O interfész (1.1.1. ábra) a Lucas-Nülle GmbH egyik terméke volt, amelyet a PRO/TRAIN for Windows szoftveréhez fejlesztett ki azzal a céllal, hogy a segítségével összeköthesse a számítógépen futó emulált folyamatot az azt irányító PLC-vel. Fontos tény, hogy ennek a dolgozatnak az írása közben, mint az I/O interfész, mint pedig a PRO/TRAIN for Windows lekerültek a Lucas-Nülle GmbH kínálatából.

Ez az eszköz úgy lett megalkotva, hogy minden olyan minden olyan PLC típust támogasson, amelyik képes az iparban használatos jelszintekkel üzemelni. A ki- és bemenetei 4 mm-es biztonsági aljzatokra, illetve DC‑37 csatlakozókra lettek kivezetve, emellett állapotjelző LED-ekkel rendelkeznek. A számítógéphez soros porton keresztül csatlakoztatható az erre a célra szolgáló DE‑9 csatlakozó segítéségével. Az eszköz további műszaki jellemzői a következők:

* 16 digitális bemenet: +24 V DC / 10 mA
* 16 digitális kimenet: +24 V DC / 100 mA
* 2 analóg bemenet: 0…+10 V / 11 bit
* 4 analóg kimenet: 0…+10 V / 11 bit
* tápellátás: +24 V DC / 1 A
* adatátvitel: RS‑232
* méretek: 297 mm × 227 mm × 60 mm (magasság × szélesség × mélység)
* súly: 1 kg



1.1.1. ábra: A CO3715-1H típusjelzésű I/O interfész

A működtetéséhez a megfelelő tápellátás biztosításán és a számítógéphez való csatlakoztatásán túl szükség van egy speciális meghajtóprogramra is, amellyel beépül a 1.1.2 alfejezetben bemutatásra kerülő szoftverbe. Ezt a meghajtóprogramot a PRO/TRAIN for Windows szállítja, amelyhez eredetileg készült.

Mivel a PRO/TRAIN for Windows a feladat megoldásához nem került felhasználásra, csak a vele együtt települő meghajtóprogram, ezért ez a szoftver ebben a dolgozatban nem kerül bemutatásra.

### A WinFACT 7 BORIS szimulációs szoftver

A BORIS (1.1.2. ábra) az Ingenieurbüro Dr. Kahlert terméke és a WinFACT moduláris programrendszer alapmodulja [1]. Az elnevezése a Block-Oriented Simulation System kifejezésből ered, és elsősorban dinamikus rendszerek szimulációjára szolgál, de a megfelelő hardver eszközökön keresztül valós folyamatokhoz is csatlakoztatható.



1.1.2. ábra: A BORIS kezelőfelülete

A BORIS kezelőfelületét az egyszerűség és az átláthatóság jellemzi, aminek köszönhetően az alapvető használata könnyen elsajátítható. Egy egyszerű szimuláció megvalósításához gyakorlatilag elegendő a megfelelő blokkokat lerakni a munkalapra, majd a megfelelő módon összekötni azokat, és elindítani a szimulációt. A lerakható blokkoknak szinte mindegyike megtalálható a Rendszerblokk Eszköztáron (System Block Toolbar), amelyről a beállításoktól függően egérkattintással vagy vonszolással lehet azokat lerakni a munkalapra. Ezeknek a blokkoknak az összekötése mindig az egyik kimenetre való kattintással kezdődik és egy bemenetre való kattintással végződik. Az összeköttetések átláthatóvá tételéről egy beépített automatikus elrendező gondoskodik, amelyik a blokkok összekötése és mozgatása során igyekszik a legjobb elrendezést megtalálni. Mindemellett lehetőség van külön-külön minden egyes összeköttetés színét is megváltoztatni, amely szintén az átláthatóság növelését szolgálja.

A szimulált folyamatok vezérlésére a BORIS több eltérő lehetőséget is kínál. Legegyszerűbb közülük a kézi vezérlés, amelyet a Vezérlő Eszköztár (Control Toolbar) tesz lehetővé. Ennek az eszköztárnak a funkciói a következők:

|  |  |
| --- | --- |
|  | Standard szimuláció elindítása |
|  | Szimuláció befejezése |
|  | Léptető üzemmód aktiválása és deaktiválása |
|  | Szimuláció léptetése |
|  | Végtelenített szimuláció elindítása |
|  | Szimulációs paraméterek módosítása |
|  | Töréspont beállítása |
|  | Töréspont törlése |

Végtelenített szimuláció esetén a szimulációs folyamat addig fut, amíg valamilyen hatás (pl. a Szimuláció befejezése gomb megnyomása) meg nem állítja azt. Ezzel szemben a standard szimulációk időtartama előre meghatározott és a szimulációs folyamat futása ennek letelésekor automatikusan megáll. A standard szimulációk időtartama és a lépések nagysága más egyebek mellett a szimulációs paraméterek között módosíthatók.

A BORIS rendszerblokkjainak a könyvtára kétféleképpen is bővíthető. Egyfelől a blokkok egy tetszőleges csoportjából bármikor létre lehet hozni egy úgynevezett szuperblokkot, amelyik ezt követően tetszőleges számú munkalapra is letehető. Másfelől lehetőség van egyéni bővítmények hozzáadására is, amelyeket bármelyik programozási nyelven meg lehet valósítani, amennyiben a fordítója képes a megvalósításokból natív DLL (Dynamic Link Library) állományokat építeni.

### A Flexible Animation Builder beépülőmodul

A Flexible Animation Builder (FAB) beépülőmodul (1.1.3. ábra) egy vizualizáció tervező és megjelenítő eszköz, amelyet az Ingenieurbüro Dr. Kahlert fejlesztett ki a BORIS szoftveréhez azért, hogy leegyszerűsítse az interaktív vizualizációk, animációk és felhasználói felületek létrehozását [2].

A BORIS alapból nem rendelkezik olyan eszközökkel, amelyekkel vizualizációkat lehetne készíteni a szimulációkhoz, ehelyett olyan adatmegjelenítő blokkokat kínál, mint a táblázatok, a diagramok vagy a mérőeszközök. Természetesen a bővítmények a szállított funkcionalitásaikhoz biztosíthatnak saját vizualizációkat is, viszont ezeket a vizualizációkat értelemszerűen a bővítmény fejlesztéséhez használt eszközökkel kell megvalósítani és esetlegesen módosítani. Mindezekkel szemben a FAB egy teljesen üres vásznat biztosít a felhasználók számára, amelyre az eszköztárából igény szerint vihetők fel az elemek, a kínálata pedig többek között a következőkből áll:

* primitív alakzatok: körök, vonalak, téglalapok stb.
* médiatartalmak: képek, videók, hangok stb.
* szimbólumok és animációk: motorok, tartályok, keverőlapátok stb.
* adatmegjelenítők: táblázatok, diagramok, mérőeszközök stb.
* vezérlők: nyomógombok, beviteli mezők, jelölőnégyzetek stb.



1.1.3. ábra: A FAB kezelőfelülete

A FAB blokkokhoz legfeljebb 50 darab bemenetet és ugyanennyi kimenetet lehet definiálni, amelyeken keresztül a vizualizálni kívánt adatok (pl. folyadékszint, szelepek állapotai stb.) összeköthetők a vizualizációval, és amelyeken keresztül a felhasználói interakciók (pl. nyomógombok állapotai) összeköthetők a szimulációval.

Mivel a feladat megoldásában a FAB kiemelt szerepet töltött be és Görbedi Ákos diplomamunkája elsősorban a BORIS használatának a bemutatására koncentrál, az interaktív vizualizáció készítés ebben a dolgozatban részletesen is bemutatásra kerül a harmadik fejezetben.

## Az ipari folyamat irányításának eszközei

Az irányítás megvalósításához, összhangban a feladat megoldásához rendelkezésre álló erőforrásokkal, egy CJ2M-CPU32 típusjelzésű PLC került felhasználásra, amelynek a programozása a hozzá tartozó programozószoftver segítségével valósult meg.

### Az Omron CJ2M programozható logikai vezérlő

A CJ2M-CPU32 programozható logikai vezérlő (1.2.1. ábra) az Omron Corporation terméke és a CJ2 termékcsalád tagja, amelyet elsősorban csomagolási és általános gépipari automatizálási folyamatok elvégzéséhez fejlesztettek ki [3].

Ez az eszköz a termékcsaládjára jellemző módon moduláris felépítésű, ami lehetővé teszi, hogy a megfelelő bővítőmodulok segítségével az igényekhez lehessen alakítani. Emellett egy beépített USB-porttal is rendelkezik, aminek köszönhetően egyszerűen és különleges beállítások nélkül lehet a számítógéphez csatlakoztatni. Az eszköz főbb műszaki jellemzői a következők:

* programmemória mérete: 10 Klépés
* adatmemória mérete: 64 Kszó
* utasítás‑végrehajtási idő: 0,04 µs
* kommunikáció: EtherNet/IP, Ethernet TCP/IP, USB
* tápellátás: +5 V DC / 0,7 A
* méretek: 90 mm × 62 mm × 84,5 mm (magasság × szélesség × mélység)
* súly: 190 g



1.2.1. ábra: A CJ2M-CPU32 típusú PLC

A feladat megoldásához a felhasznált PLC digitális I/O egységekkel is ki lett bővítve a felhasznált I/O interfészhez való csatlakoztatás érdekében. A bemeneti egység típusa CJ1W‑ID211, ami 16 darab egyenáramú bemenettel rendelkezik, a névleges feszültsége 24 V, a névleges árama pedig 7 mA. A kimeneti egység CJ1W-OD212 típusú, ami 16 darab tranzisztoros kimenettel (PNP) rendelkezik, a névleges feszültsége a bemeneti egységhez hasonlóan 24 V, a névleges árama pedig 0,5 A.

### A CX-Programmer programozószoftver

A CX‑Programmer (1.2.2. ábra) az Omron Corporation teljes PLC kínálatához tartozó programozószoftver és a CX‑One szoftvercsomag szerves része [4]. Sok más programozási rendszerhez hasonlóan az IEC 61131‑3 szabvány [5] előírásainak megfelelően lett kialakítva, emellett az integrált PLC szimulációs eszközeivel lehetővé teszi a programok tesztelését még a letöltés előtt.



1.2.2. ábra: A CX-Programmer kezelőfelülete

A CX-Programmer több kényelmi funkcióval is rendelkezik, amelyek a programok írását igyekeznek még egyszerűbbé és gyorsabbá tenni. Így például a memória kiosztása és felügyelete automatizált, aminek köszönhetően elegendő a szimbólumok típusát meghatározni, a tárolásukról már maga a szoftver gondoskodik. Támogatja az azonos típusú adatokból álló adatblokkokat (tömböket) és az eltérő típusú adatokból álló adatblokkokat (struktúrákat) is, amelyek akár a funkcióblokkok be- és kimeneti változóiként is megadhatók. Az intelligens programbevitel a szimbólumok kiválasztását teszi egyszerűbbé azáltal, hogy egy prediktív stílusú böngészőt biztosít a felhasználó számára, amiben a szimbólumok nevének beírása közben megjelennek a lehetséges találatok. Képes továbbá a soros, USB és Ethernet/IP kapcsolatokon keresztül kapcsolódó eszközökkel automatikusan összekapcsolódni, megkönnyítve ezzel a programozást és a hibakeresést.

Ennek a szoftvernek a 30 napos próbaverziója ingyenesen is használható. Ez az idő alatt minden funkciója korlátozásmentesen igénybe vehető, ugyanakkor a 30 nap elteltével megszűnik a mentés és a nyomtatás lehetősége.

## A technológia-visszafejtés eszközei

A technológia visszafejtés valójában két további részfeladatra lenne bontható, ami alapján a felhasznált hardvereket és szoftvereket tovább lehetne csoportosítani. Az adatgyűjtéshez ugyanis egy független lehallgatószoftver került felhasználásra, míg a begyűjtött adatok igazolása az I/O interfészt helyettesítő elektronika működő prototípusának megépítésével történt, amihez egy PIC fejlesztőlap és a hozzá tartozó fejlesztői környezet került felhasználásra.

### Az Eltima Serial Port Monitor segédprogram

A Serial Port Monitor (1.3.1. ábra) az Eltima IBC által kifejlesztett segédprogram, amely a soros portok aktivitásának nyomon követesére és elemzésére szolgál [6]. A képességeinek köszönhetően kiválóan alkalmazható a hibakeresésben, a kommunikációs protokollok fejlesztésében, a technológia-visszafejtésben és az oktatásban.

Ennek a szoftvernek a segítségével lehetőség van a soros portok minden aktivitását rögzíteni, beleértve a feléjük küldött vezérlőkódokat (IOCTL), a rajtuk áthaladó teljes adatforgalmat még akkor is, ha az érintett soros portok más alkalmazások által már meg lettek nyitva, illetve azokat az adatokat is, amelyeket az alkalmazások csak megkíséreltek a portokra írni a ténylegesen a portoka íródott adatok mellett.



1.3.1. ábra: A Serial Port Monitor kezelőfelülete

A Serial Port Monitor a rögzített adatok vizualizálásra öt különböző nézetet is biztosít, amelyek a terminál nézet (Terminal view), a vonali nézet (Line view), az adatforgalom nézet (Dump view), a táblázat nézet (Table view) és a Modbus nézet (Modbus view). Mindezeknek a tartalma bármikor elmenthető HTML, egyszerű szöveg vagy CSV formában, illetve összehasonlíthatók a korábbi munkamentekből származó adatokkal is.

A megfigyelés mellett a Serial Port Monitor képes adatokat is küldeni a megfigyelt soros portra úgy, mintha azokat a megfigyelt alkalmazás küldte volna. Ennek segítségével megvizsgálható a soros port és a csatlakoztatott eszköz reakciója is.

Érdemes megjegyezni, hogy a bemutatott funkciók egy része csak Professional és Company licenszekkel érhető el, viszont a 14 napig tartó próbaidőszak alatt minden funkció korlátozásmentesen kipróbálható.

### A MikroElektronika EasyPIC v7 fejlesztőlap

Az EasyPIC v7 (1.3.2. ábra) a MikroElektronika d.o.o. hetedik generációs fejlesztőlapja, amelyet a Microchip Technology Inc. 8‑bites PIC mikrovezérlőihez alakítottak ki főként kezdő felhasználók és hobbisták számára, illetve oktatási célokra.

Ezen a fejlesztőlapon számos modul kapott helyet, amelyek a legkülönfélébb alkalmazások fejlesztéséhez használhatók, beleértve a grafikus megjelenítést, az USB és soros kommunikációt, a hőmérsékletmérést és egyebeket. Rendelkezik két mikroBUS aljzattal is, amelyeken keresztül ezernyi új funkcionalitás adható a fejlesztőlaphoz Click Board bővítőkártyák segítségével. Mindezek mellet rendelkezik egy beépített mikroProg programozóval és áramkörön belüli hibakeresővel is, amelyik a Microchip Technology Inc. több mint 387 mikrovezérlőjét támogatja [7].



1.3.2. ábra: Az EasyPIC v7 fejlesztőlap

Az EasyPIC v7 kapcsolási rajza szabadon hozzáférhető csak úgy, mint a csatlakoztatható bővítőkártyáké is, így a fejlesztőlapon kipróbált megoldások viszonylag kevés ráfordítással átvezethetők a célhardverek világába. Mindezt a bővítőkártyákhoz szintén szabadon hozzáférhető függvénykönyvtárak és példaprogramok teszik még egyszerűbbé.

### A mikroC PRO for PIC fejlesztői környezet

A mikroC PRO for PIC (1.3.3. ábra) a MikroElektronika d.o.o. integrált fejlesztői környezete, amelyet a Microchip Technology Inc. 8‑bites PIC mikrovezérlőinek C nyelven történő programozásához fejlesztettek ki [8].

Ez a fejlesztői környezet több, mint 808 különböző PIC mikrovezérlő programozását támogatja, illetve mindegyik támogatott mikrovezérlőhöz rendelkezik egy konfigurációs felülettel is, amelyen keresztül a mikrovezérlő könnyedén testre szabható. A perifériák és a gyakran csatlakoztatott hardverkomponensek használatához számos függvénykönyvtárral rendelkezik, amelyeknek a használata a beépített súgó és a szoftverrel együtt szállított példaprogramok segítségével könnyedén megismerhető.



1.3.3. ábra: A mikroC PRO for PIC kezelőfelülete

A modern fejlesztői környezetektől elvárható módon a mikroC PRO for PIC is rendelkezik olyan funkciókkal, amelyek kényelmesebbé és hatékonyabbá teszik a fejlesztői munkát. Így például rendelkezik egy intelligens kódkiegészítővel, ami a begépelt karakterekhez automatikusan felkínálja a lehetséges függvényeket, konstansokat, struktúrákat, változókat és egyéb kódelemeket. A függvényhívások helyes paraméterezését a beépített paramétersegéddel támogatja oly módon, hogy a begépelt karakterek felett megjeleníti a függvény szignatúráját. Mindezek mellett az átláthatóság növelése érdekében képes a kódblokkok összecsukására és szétnyitására is.

A mikroC PRO for PIC a hozzá tartozó mikroProg programozó és áramkörön belüli hibakereső segítségével natív támogatást nyújt a hardveres hibakereséshez. Ezzel az eszközzel képes a mikrovezérlők programjait lépésről futtatni, megjeleníteni a regiszterek, EEPROM-ok stb. aktuális értékeit, illetve lehetőséget biztosít a töréspontok igény szerinti beiktatására és eltávolítására is.

Ennek a szoftvernek a demó verziója korlátlan ideig használható ingyenesen, viszont a demó verzióval legfeljebb csak 2 Kszó nagyságú PIC programok fordíthatók.

## Az alternatív emuláció megvalósításának eszközei

### A Qt Creator fejlesztői környezet

A Qt Creator egy keresztplatformos integrált fejlesztői környezet, amelyet a Qt Company fejlesztett ki a Qt alkalmazás-keretrendszerrel folytatott fejlesztés igényeinek megfelelően. Köszönhetően a beépített űrlaptervezőnek, jelenleg a Qt alapú, grafikus felhasználófelülettel rendelkező alkalmazások elsődleges fejlesztői környezete.

Ez a szoftver valójában egy fejlett forráskódszerkesztő C++, QML és JavaScript támogatással,

### Az Altium Designer elektronikai tervezőszoftver

### Az IAR Embedded Workbench for Arm fejlesztői környezet

Az IAR Embedded Workbench for Arm (1.4.1. ábra) az IAR Systems AB integrált fejlesztői környezete, amely több, mint 7000 különböző ARM mikrovezérlő programozását teszi lehetővé C és C++ nyelveken [9].

Ez az egyik legnépszerűbb fejlesztői környezet az ARM mikrovezérlőkhöz, amit elsősorban a széleskörű kompatibilitásának, az egyszerű kezelőfelületének és a magába foglalt többezernyi példaprojektjének köszönhet. A kódminőség támogatása érdekében két beépített eszközt is tartalmas. A C-STAT statikus kódanalizátor



1.4.1. ábra: Az IAR Embedded Workbench for Arm kezelőfelülete

Ezt a szoftver ingyenesen is lehet használni korlátlan ideig, de az ingyenes változat korlátozásokat tartalmaz. Ezzel szemben a 30 napos próbaverzióban minden funkció korlátozásmentesen elérhető, viszont a próbaidőszak után már csak megvásárolt licensszel használható tovább.

# Az emulálni kívánt ipari folyamat

Az emulálni kívánt ipari folyamat egy karosszéria gyártósor, amely egy korábbi Országos Ajtonyi István Irányítástechnikai Programozó Verseny egyik gyakorlati feladata volt. Az emulálni kívánt technológia az irányításához kiadott feladatlap alapján került megtervezésre és megvalósításra a BORIS eszköztárának segítségével.

A gyártósor a gépjármű karosszériák gyártásának egy szakaszát mutatja be, ahol a következő három fő művelet folyik:

* ponthegesztés,
* festés,
* átemelés (leválogatás).

A rendszer három cellából áll, ahol mindegyik cellában az előbbi felsorolás egy-egy folyamata történik. A cellákban futószalagok mozgatják a karosszériákat, illetve a cellák mindegyikében két-két érzékelő (fénykapu) jelzi a karosszériák hollétét. Az első érzékelő jelzi a cellába történő érkezést, a második pedig a munkadarab esetleges megállításának pozíciójában található.

Az első cellában két ponthegesztésre alkalmas robot helyezkedik el, amelyek a rajtuk található hegesztőpisztolyokat is beleértve külön-külön vezérelhetők. A második cellában két festésre alkalmas robot található, amelyek szinkron üzemmódban működnek. Ez azt jelenti, hogy az egyik robot másolja a másik robot mozgását, így ez a két robot irányítási szempontból egynek tekinthető. Ugyanebben a cellában egy operátor is tevékenykedik, aki szabadon beléphet a robotok mozgásterébe azzal a céllal, hogy ellenőrizze a festés minőségét és esetlegesen kézzel korrigálja azt. A harmadik cellában egy daru helyezkedik el, melynek feladata a karosszériák típus szerinti leválogatása.

Az emuláció szempontjából a feladat nehézsége a vizualizáció megvalósítása. A karosszériáknak ugyanis követniük kell a valóságban őket mozgató elemek mozgását ahhoz, hogy valóságosnak tűnjenek, ehhez pedig számon kell tartani, hogy az animációban éppen egy futószalagon helyezkednek-e el, a futószalag mozog-e, a daru megfogója záródott-e a karosszéria körül, átemelés történik-e vagy az átemelés éppen befejeződött-e. További nehézséget jelent a FAB grafikus elemeinek a szegényes kínálata, azokkal ugyanis a vizualizáció nem valósítható meg felismerhető módon.

Irányítási szempontból az operátor jelenléte jelenti a legnagyobb nehézséget. A gyártósor irányítása mellett ugyanis folyamatosan figyelni kell az operátor tartózkodási helyét, mivel bármikor beléphet a robotok munkaterébe, amire az irányításnak gyorsan és megbízhatóan kell reagálnia.

Az emuláció elsődleges célja az irányítás tesztelése és az esetleges hibák felderítése biztonságos körülmények között. Ennek érdekében az emulációba számos korlátozást és ellenőrzést kell beépíteni, amelyek jelzik az irányítás hibáit.

# Interaktív vizualizáció készítése a BORIS eszköztárával

# Az ipari folyamat emulációja és irányítása

Jegyzetek:

* A BORIS egy átgondolt és intuitív fejlesztői környezet, amely az eszközök bőséges kínálatát nyújtja a felhasználó számára. Ha csak az erősségei alapján kellene megítélni, akkor egy professzionális termékként lehetne kezelni, amely kiválóan alkalmazható az oktatásban, a kutatásban és a fejlesztésben is. Ezzel szemben a feladat megoldásához használt verzió olyan hibák és hiányosságok sorával küzd, amelyek alapjaiban kérdőjelezik meg a benne folytatott munka hatékonyságát.
  + A visszavonás és a megismétlés műveletek teljes hiánya.
  + A törlések nem kérnek megerősítést a felhasználótól, még az olyan blokkok esetén sem, amelyek összeköttetésben állnak más blokkokkal. Ez a visszavonás lehetősége nélkül felettébb felkavaró tud lenni.
  + Előfordul, hogy az üzenetablakok, amelyek megkövetelik az elsőbbséget a felhasználótól, takarásban ugranak fel. Ez meggátolja a felhasználót abban, hogy hozzáférjen, és így az alkalmazás egy blokkolt állapotába kerül.
  + Minden hivatkozás abszolút elérési útvonalak formájában van letárolva.
  + **Nem feltétlenül kell:** Alapértelmezésben az általános műveletek gyorsbillentyűi eltérnek a megszokottól, így például a másolás nem a már jól ismert Ctrl + C gyorsbillentyűvel hívható meg, hanem a Ctrl + Ins párossal. Ezek azonban módosíthatóak a beállításokban.
  + **Nem feltétlenül kell:** A blokkok közötti összeköttetések átlátható elrendezése nagyon nehezen megvalósítható. Létezik ugyan egy automatikus elrendező az alkalmazásban, de hibásan működik.
* A BORIS nem egy általános célú programozási nyelv, mint például a C, hanem egy elsősorban rendszerek szimulációjára kifejlesztett speciális fejlesztői környezet, ezért nem is várható tőle, hogy minden feladat megoldható legyen benne.

## Az emuláció megvalósításának bemutatása

### A robotok és a futószalagok vizualizációja FAB segítségével

### A robotok mozgatása

### A karosszériák mozgatása

### A cellák ellenőrzőlogikái

## Az irányítás megvalósításának bemutatása

# A BORIS projektek hordozhatósági problémájának megoldása

A fájlrendszerekben a fájlok vagy a könyvtárak helyét, vagyis azt, hogy milyen könyvtárakon át navigálva találhatóak meg, az úgynevezett elérési útvonal [6] határozza meg, amelynek két változata van. Az abszolút elérési útvonal a gyökérkönyvtárból indulva sorolja fel a fájlig vezető könyvtárak neveit, így a fájlrendszeren belül mindig ugyanazt a helyet jelöli. A relatív elérési útvonal ezzel szemben egy abszolút elérési útvonalhoz viszonyítva sorolja fel a fájlig vezető könyvtárak neveit, így ebben az esetben a fájlrendszeren belül jelölt hely viszonyításfüggő.

A több fájlból álló, összefüggő adatok esetén elterjedt és általánosan követendő gyakorlat a relatív elérési útvonalak használata a belső hivatkozásokhoz. Enélkül ugyanis nem lehetne az ilyen adatokat szabadon mozgatni a fájlrendszerben, mivel a belső hivatkozások sérülnének. A feladat megoldásához használt BORIS verzió esetén pedig pontosan ez az probléma áll fenn.

~~A BORIS lehetőséget nyújt a munkalapok egymásba ágyazására úgynevezett szuperblokkok segítségével. Egy szuperblokk képes a hivatkozott munkalap nyitott végpontjait a sajátjaiként megjeleníteni, ezáltal azok összeköthetőkké válnak a munkalapon levő más blokkok végpontjaival.~~

~~A szuperblokk egy olyan speciális blokk, amelyik képes a hivatkozott munkalap nyitott végpontjait a sajátjaiként megjeleníteni~~

A megvalósításnál a fő szempont az egyszerűség volt, viszont ennek hatására a jelenlegi verzió nem lett felkészítve minden határesetre. Így például, ha a hivatkozások egyike olyan fájlra hivatkozik, amelyik a BORIS projekt gyökérkönyvtárán kívül található, miközben azonos névvel és megfeleltethető relatív elérési útvonallal létezik egy fájl a BORIS projekt gyökérkönyvtárán belül is, akkor az a hivatkozás a korrigálás hatására elromlik, ahogyan azt a 4.2.1. példa is mutatja. Ahhoz, hogy ez a hiba ne történjen meg egy már egy jóval összetettebb algoritmusra lenne szükség, amely képes következtetni a BORIS projekt gyökérkönyvtárának korábbi elérési útvonalát, és ezzel az információval végezni a hivatkozások korrigálását.

*// Korrigálás előtt:*

C:\Program Files (x86)\Kahlert\WinFACT 7\Examples\Demo8.sbl

D:\Studies\BORIS Project\Examples\Demo8.sbl

*// Korrigálás után:*

E:\Archived\2021\Studies\BORIS Project\Examples\Demo8.sbl

E:\Archived\2021\Studies\BORIS Project\Examples\Demo8.sbl

4.2.1. példa: Korrigálási hiba azonos végződésű elérési útvonalak esetén

Tulajdonképpen a BORIS Corrector fontossága az olyan projekteknél kerülhetne előtérbe, amelyek verziókezelő rendszerek segítségével teszik lehetővé azt, hogy egyidejűleg többen is dolgozzanak rajta. Ilyen esetben az egyik lehetőség a megállapodás a projekt gyökérkönyvtárának abszolút elérési útvonalában, a másik lehetőség pedig a BORIS Corrector kibővítése egy olyan funkcionalitással, amellyel a hivatkozásokat közös alakra hozhatná a verziókezelő rendszerbe való feltöltés előtt.

# Az ipari folyamat emulálásának alternatív megoldása

## Az I/O interfész működésének behatásmentes visszafejtése

### BORIS és az I/O interfész közötti kommunikáció lehallgatása

### Az I/O interfész működésének utánzása a BORIS számára

### Az I/O interfész működtetése a BORIS használata nélkül

## Az emuláció megvalósítása alternatív eszközökkel

### A karosszériák mozgatása

### A futószalagok megvalósítása és működtetése

### A daru megvalósítása és működtetése

### A robotok megvalósítása és működtetése

# Az I/O interfészt helyettesítő elektronika

# Konklúzió

# Irodalomjegyzék

|  |  |
| --- | --- |
| [1] | Ingenieurbüro Dr. Kahlert, „WinFACT overview,” [Online]. Available: https://www.kahlert.com/engl/winfact-uebersicht. |
| [2] | Ingenieurbüro Dr. Kahlert, „Flexible Animation Builder,” [Online]. Available: https://www.kahlert.com/engl/flexible-animation-builder. |
| [3] | Omron Corporation, „CJ2M-CPU32,” [Online]. Available: https://industrial.omron.hu/hu/products/CJ2M-CPU32. |
| [4] | Omron Corporation, „CX-Programmer,” [Online]. Available: https://industrial.omron.hu/hu/products/cx-programmer. |
| [5] | Wikipedia, „IEC 61131-3,” [Online]. Available: https://en.wikipedia.org/wiki/IEC\_61131-3. |
| [6] | Wikipedia, „Path (computing),” [Online]. Available: https://en.wikipedia.org/wiki/Path\_(computing). |

# Nyilatkozat