SZAKDOLGOZAT

Duchai László Pál
Gépipari Automatizálási szakmérnök

Gödöllő
2023

Magyar Agrár- és Élettudományi Egyetem

Szent István Campus

Gépipari Automatizálási Szakmérnök

Ipari kommunikáció és HMI felület megvalósítása Node-RED segítségével

Belső konzulens: Mayerné Dr. Sárközi Eszter

egyetemi adjunktus (MATE)

Külső konzulens: Jankovics Balázs

Művezető

Készítette: **Duchai László Pál**

G9MMOT

levelező tagozat

Intézet/Tanszék: Műszaki Intézet

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

MŰSZAKI INTÉZET
GÉPIPARI AUTOMATIZÁLÁSI SZAKMÉRNÖK

DIPLOMADOLGOZAT
feladatlap

Duchai László Pál *(G9MMOT)*

részére

A diplomadolgozat címe:

Ipari kommunikáció és HMI felület megvalósítása Node-RED segítségével

Feladatkiírás:

Bevezetés, Szakirodalom feldolgozása, Probléma bemutatása, Gravírozó berendezés ipari kommunikációja Node-RED illesztőszoftverrel, HMI felület kialakítása, Továbblépés lehetőségei, Gazdasági számítás, Összefoglalás

Közreműködő tanszék: Mechatronika

Külső konzulens: *Jankovics Balázs, hideghengermű, hengersori vezető, ISD Dunaferr 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 06 nap

(külső konzulens)

Tartalomjegyzék

1. Bevezetés

Az ipari termelési folyamatok optimalizálása és hatékonyságának növelése érdekében számos vállalat automatizálási megoldásokat alkalmaz, beleértve a programozható logikai vezérlő (PLC) rendszereket, bin picking technológiákat és robotikai megoldásokat is. Ezek a megoldások lehetővé teszik a folyamatok felgyorsítását, a hatékonyság és a minőség növelését, valamint az emberi munkaerő terheltségének csökkentését.

A PLC rendszerek nem csak az automatizált vezérlést teszik lehetővé, hanem a gyártósorok egyes részeinek karbantartását és ellenőrzését is segíthetik, valamint a gyártási adatok gyűjtésére és akár azok elemzésére is szolgálnak. A PLC-vel rendelkező gyártósorokban számos érzékelő és aktuátor található, amelyeket a vezérlő egység irányít. A PLC lehetővé teszi a folyamatok automatikus vezérlését, figyelembe véve a különböző szenzorokból érkező adatokat.

A bin picking technológia továbbra is fejlődik, és egyre több gyártó veszi igénybe ezt az automatizálási megoldást. A fejlesztéseknek köszönhetően a megoldások már képesek összetettebb feladatokra, mint például a változó geometriájú vagy színű tárgyak felismerése, ami további lehetőségeket nyit meg az anyagmozgatási folyamatok automatizálásában.

A robotizáció a gyártási folyamatok automatizálásának egyik legfontosabb formája. A robotok lehetővé teszik az egyes munkafolyamatok felgyorsítását és automatizálását, amelyeket emberi erővel nehezebb vagy veszélyesebb elvégezni. A robotok alkalmazása a termelékenység és a minőség növelését is eredményezi.

Szinte minden PLC-ket, valamint kommunikációra képes ipari eszközöket gyártó cég termékei rendelkeznek valamilyen terepi kommunikációs megoldással. Utóbbi esetén sokszor opciós kiegészítőként érhető el a több különböző protokoll valamelyike, de előfordul olyan eset is, amikor nem rendelkezik ilyen kommunikációs protokollal az adott ipari eszköz, viszont mégis szükséges a kommunikációt megvalósítani.

Összességében az automatizáció és robotizáció fontos szerepet játszik az iparban, és folyamatosan fejlődik az újabb technológiák megjelenésével, valamint kiforrottságuk növekedésével. Az egyre összetettebb rendszerekkel, mint például a bin picking, a termelési folyamatok automatizálása hatékonyabbá és gazdaságosabbá válik. Azonban fontos megjegyezni, hogy az automatizáció és robotizáció nemcsak gazdasági előnyöket hoz, hanem

a munkaerőpiacot is befolyásolja. Az egyre fejlettebb technológiák megjelenése új típusú munkahelyeket teremt, de egyben számos hagyományos munkakört is megszüntet.

1.1 Célkitűzés

A gyártási folyamatok elengedhetetlen részegysége a rakodás a különböző göngyölegekről akár szállítószalagokra, akár különböző munkaállomásokra, illetve a termékek követése is. Amennyiben a munkaállomások relatív kis helyigényűek, úgy könnyen előfordulhat, hogy a rendszer kiszolgálását egy dolgozó végzi el, nehéz tárgyak esetén például daruval. Ekkor a dolgozó feladata a göngyöleg helyi kezelése, a munkadarabok folyamatos áramoltatása, illetve a termékkövetési szolgáltatás üzemeltetése is.

A szakdolgozatomban megtervezem egy gyártó-rakodó rendszer esetén a termékkövetéshez alkalmazott, ipari kommunikációs protokollal nem rendelkező részegység és a PLC közötti kapcsolatot, valamint azt is bemutatom, hogy miként dolgoztam ki további kiegészítéseket ezen megoldás számos lehetőségeinek kihasználásával, mint például egy „HMI” felület létrehozása valódi HMI nélkül. A dolgozat tartalmazza a jelenlegi munkafolyamat áttekintését, a kívánt új munkafolyamat leírását, az alkalmazott technológiák és rendszerek ismertetését, valamint az ezeket kiegészítő és protokollillesztő egység elkészítésének és tesztelésének folyamatát.

A munka során alkalmazott rendszerek részét képezi több egyedi berendezés, 6 tengelyes KUKA Robot, PhotoNeo 3D lézer szkenner, HiWin szervóhajtások, Markator pontgravírozó, valamint Siemens S7 1500-as PLC. A dolgozat tárgyát egy mini PC-n futó NodeRED installáció képezi, amelyet a saját böngészős felületén keresztül lehetséges programozni és irányítani.

2. Szakirodalmi áttekintés

2.1 PLC rendszerek

A PLC (Programmable Logic Controller) rendszerek olyan eszközök, amelyeket ipari folyamatok vezérlésére használnak [1]. A PLC-k az iparban elterjedtek, mivel képesek sokféle szenzor, aktuátor és egyéb eszköz irányítására, valamint biztonságos és hatékony működésre [2].

A PLC-k története az 1960-as évekre nyúlik vissza, amikor a gyártósorok automatizálása még relékből álló áramkörökkel történt. Ezek a relés vezérlő rendszerek akár teljes falakat is elfoglalhattak (*1. ábra*), és nem számítottak kifejezetten megbízhatónak. Ezen rendszereket utólag javítani, módosítani gyakorlatilag lehetetlen kihívás volt a bonyolultáguknál fogva [3, 4]. Az 1960-as évek elején egy cég, az Allen-Bradley fejlesztett ki egy eszközt, amely lehetővé tette a relék helyett tranzisztorok használatát a gyártósorok irányítására [5]. Ezt a rendszert a "Programmable Sequence Controller" (PSC) néven ismerték.

1. ábra: Relés vezérlőszekrények [34]

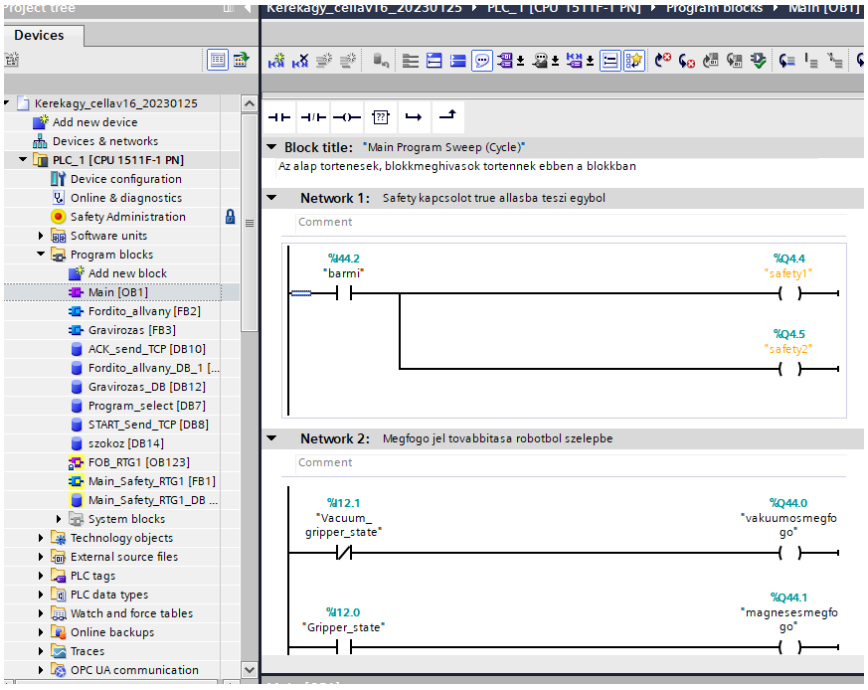
Az első valódi PLC-t 1968-ban fejlesztette ki a Bedford Associates nevű cég, amelyet a GM (General Motors) megbízásából hoztak létre [6]. Az első PLC, amelyet a GM használt, a Modicon néven került piacra, és azóta többek között az Allen-Bradley és a Siemens is sikeres, híres PLC-gyártóvá vált. A PLC-k fejlődése az évtizedek során jelentős volt. Az első PLC-k általában egyszerűbb feladatokra voltak kitalálva, de a technológia fejlődésével egyre összetettebb és nagyobb teljesítményű PLC-k jelentek meg. Ma már a PLC-k széles körben elterjedtek az iparban, és számos gyártó kínál PLC-ket különböző méretű és teljesítményű kivitelben [7].

A PLC-k általában két részből állnak: a hardverből és a szoftverből [8]. A hardver rész magában foglalja a PLC-t, amely általában egy kompakt egység, és tartalmazza a bemeneteket

és kimeneteket, amelyeket a folyamat vezérlésére használnak. A szoftver rész pedig a programozó számítógépén fut, és lehetővé teszi, hogy programozzák a PLC-t, és beállítsák a kívánt működést [9].

A PLC-k számos előnnyel rendelkeznek, beleértve a rugalmasságot, a megbízhatóságot és a skálázhatóságot [5]. Az ipari környezetben a PLC-k a hosszú élettartam és az alacsony karbantartási költségek miatt különösen előnyösek. A PLC-k azonban nem csak az iparban használatosak, hanem számos más területen is, például épületautomatizálásban, közlekedésben, energiatermelésben és még sok más helyen. Egy érdekes példa a mini PLC-k alkalmazása azok számára, akik okosotthont szeretnének létrehozni, amelyhez például a Siemens LOGO mini PLC vagy programozható relé tökéletes választás lehet [10, 11].

Az egyik vezető vállalatnak ma a Siemens számít a PLC-k terén. Az első Siemens PLC-k az 1980-as években jelentek meg. Ma a Siemens PLC-k között számos kivitel van, az egyszerűbb feladatokra alkalmas LOGO PLC-ktől a nagyobb teljesítményű és összetettebb feladatokat is ellátni képes S7-1500 szériás PLC-kig [12].



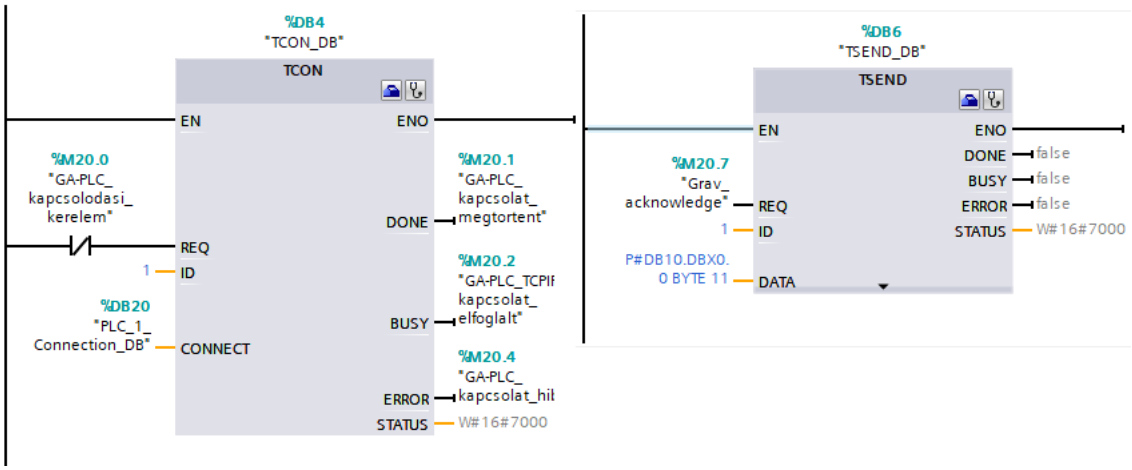
2. ábra: A TIA Portal felülete

A Siemens PLC-ket többnyire a STEP 7, majd később TIA Portal nevű programozói környezettel programozzák, amely egyszerű használatával és sokoldalúságával nagy népszerűségnek örvend a PLC-programozók körében (*2. ábra*). A Siemens PLC-k jellemzően moduláris felépítésűek, ami lehetővé teszi az egyszerűbb PLC-k könnyű kiegészítését új

modulokkal, illetve a nagyobb teljesítményű PLC-k moduláris felépítése lehetővé teszi teljes modulsorok összekötését is.

A PLC programokban gyakran alkalmaznak speciális blokkokat is, amelyeknek belső működését ritkán ismeri a programozó, azonban ez nem is szükséges, mivel jól dokumentált módon ismertetésre kerül ezek használata az adott PLC programozási leírásában. Egy ilyen blokkcsoport az úgynevezett Open User Communication csoport. Az ebben található blokkok az Ethernet porton csatlakoztatott ISO-on-TCP kommunikációt teszik lehetővé. Ez abban az esetben lehet hasznos, ha a rendszerhez csatlakoztatunk egy olyan eszközt, amely ezt is támogatja, azonban (a PLC által preferált) terepi buszos hálózatokat nem.

A TCON blokk segítségével lehetséges létrehozni egy kapcsolatot a cél állomás és a PLC között. Miután a kapcsolat felállt, a PLC azt felügyeli, ellátja egy azonosító számmal, majd ezután a többi kommunikációs blokknak csak ezt az azonosítót kell alkalmaznia a működéshez. A kapcsolat adatait egy adatblokk tárolja, amelyben megtalálhatók például az IP címek is. A TCON blokk bemenete a kapcsolódási kérelem, az adatblokk, valamint az igényelt azonosító, kimenete több státusz bit és hibakód (*3/a ábra*). A kapcsolódás a kérelem felfutó élére indul el. A programozási leírás a hibakódok jelentését és megoldási javaslatait is tartalmazza [13]. Lehetőség van a kapcsolat adatait megadni UI felületen is. A blokk importálása után megjelenik rajta egy „szerszámosláda” ikon (*3. ábra*), amelyre rákattintva az összes fontosabb paraméter beállítható, képes hibás adatmegadás esetén jelezni, valamint segít abban is, mely paraméter mit jelent.



3. ábra: Az Open User Communication blokkok: a) TCON; b) TSEND

A másik fontos blokk ebben a csoportban a TSEND. Fontos bemenetei a küldési kérelem, amely szintén felfutó élre kezdi a küldést, a kapcsolati azonosító, amelyet a TCON létrehozott, valamint az adatcsomag, amelyet el szeretnénk küldeni. Kimenete néhány státuszbit, valamint hibakód (*3/b ábra*). Ez a blokk a DATA bemenetére kötött adathalmazt egyszerűen elküldi a TCON blokkban definiált hálózati partnernek [13]. A küldés során semmilyen további formázást vagy módosítást nem végez el. Épp ezért egyes eszközök nem biztos, hogy fel tudják dolgozni az onnan származó adatokat. Elképzelhető, hogy soremelő vagy „carriage return" karaktereket kell elhelyezni az adatok végén, ha például szöveget szeretnénk továbbítani.

2.2 Terepi buszos kommunikáció

A terepi buszos kommunikáció egy olyan technológia, amely lehetővé teszi a különböző eszközök (például távoli szenzorok jelgyűjtői, mérőműszerek, vezérlők stb.) összekapcsolását egyetlen hálózaton keresztül [14]. A terepi buszok különböző protokollokat használnak, például a PROFINET protokollt, amely az ipari automatizálási rendszerek egyik legelterjedtebb terepi busz protokollja [15].

A PROFINET protokoll lehetővé teszi a gyors és megbízható adatátvitelt az ipari környezetben, valamint lehetővé teszi az adatok és információk általános használatát a különböző rendszerek és eszközök között. A PROFINET protokoll lehetővé teszi a valós idejű adatátvitelt, a redundáns kommunikációt és a hálózati topológiák változatos kialakítását [16].

A PROFINET protokoll alapvetően az Ethernet technológiára épül (IEC 61784-2) [17], és lehetővé teszi az Ethernet hálózatokon keresztüli kommunikációt a különböző ipari eszközök között. A PROFINET protokoll további előnye, hogy támogatja az IEC 61131-3 szabványt, amely a PLC-k programozására, konfigurációjára szolgáló szabvány [18]. Emellett lehetővé teszi a könnyű integrációt a Siemens PLC-kkel, amelyek nagy népszerűségnek örvendenek az ipari automatizálási rendszerek terén.

A PLC-k egy része képes több kommunikációs busz protokoll formára is. Egy Siemens PLC képes lehet a PROFINET-en kívül például PROFIBUS, EthernetIP vagy hagyományos, TCP/IP (esetleg ISO-on-TCP, az Open User Communication csoportból) kommunikációra is [19]. Erre akkor lehet szükség, hogyha egy olyan eszközt szeretnénk csatlakoztatni a PLC-hez, amelyet nem a terepi kommunikációs busz rendszerekbe való integrálásra terveztek vagy akár nem ipari eszközökhöz is.

2.3 A HMI

A HMI jelentése Human Machine Interface, azaz ember és gép közötti felület. Kapcsolatot létesít a felhasználó és a gép között, információáramlást lehetővé téve. Fontos feladata az adott gép kezelésének, irányításának biztosítása, valamint a gép működése során keletkező tájékoztató vagy akár hiba információk megjelenítése. Ilyen felület a személyautónk műszerfala a kijelzőkkel, kapcsolókkal; a számítógép vagy okostelefon kijelzője, valamint ilyen egységeket alkalmaznak az iparban is arra a célra, hogy a dolgozó irányíthassa, parametrizálhassa az adott berendezést, a berendezés pedig működési, diagnosztikai vagy hiba információkat jelenítsen meg a dolgozó számára.

Az ipari gyakorlatban a legelterjedtebb megoldás talán a Siemes Comfort szériás, rezisztív érintős kijelzővel ellátott HMI-je, amely több különböző méretben, akár 7-től 20 hüvelykes képátlóig megtalálható [20]. Ezek programozása a legtöbb esetben a berendezést működtető Siemens PLC-k programozásával együtt történik, szoros szinkronban működve egymással. A Siemens HMI eszközei a projektben létrehozásra kerülő adatblokkokon keresztül kommunikál egymással, de a megfelelően szoros kompatibilitás esetén előfordulhat, hogy a HMI közvetlenül a PLC-n belül található adatblokkokat írja, vagy programblokkokat hívja meg [21].

4. ábra: A Siemens Comfort szériás HMI eszközei [20]

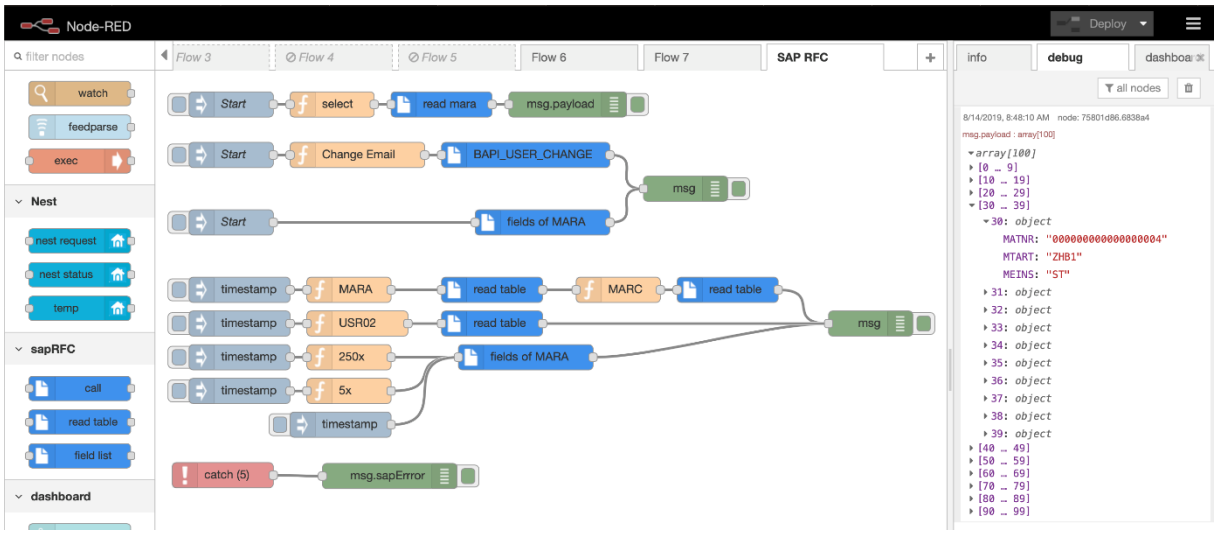
Természetesen sokféleségükben megtalálhatók egysoros LCD kijelzős, gombos verziók is, nagyobb kijelzővel rendelkező, de nem érintésérzékeny verziók, valamint a teljesen érintőképernyős variánsok egyaránt [20]. Utóbbi HMI felületén különböző oldalakat lehet

létrehozni, gombokkal, szövegmezőkkel, grafikonokkal, felhasználói szintekkel és egyéb beavatkozószervekkel. Működése egy egyszerűbb weboldal működésének megfelelő.

2.4 Node-RED

A Node-RED egy programozási eszköz különböző hardvereszközökhöz. API-k és online szolgáltatások új és érdekes módon történő összekapcsolására szolgál. A Node-RED egy folyamatalapú programozási eszköz, amelyet eredetileg az IBM Emerging Technology Services csapata fejlesztett ki, és jelenleg az OpenJS Foundation része [22, 23].

J. Paul Morrison találta ki az 1970-es években, a folyamatalapú programozás egy módja annak, hogy egy alkalmazás viselkedését „fekete dobozok" vagy csomópontok (node-ok) hálózataként írja le [24], ahogy a Node-RED-ben nevezik. Minden csomópontnak van egy jól meghatározott célja; kap néhány adatot, bármilyen módon lehet a beviteli adatokat változtatni, akár már meglévő függvényekkel, vagy személyre szabottan, majd továbbadja ezeket az adatokat. A hálózat felelős a csomópontok közötti adatáramlásért.



5. ábra: A Node-RED felülete [25]

Ez egy olyan modell, amely kiválóan alkalmas a vizuális megjelenítésre (*5. ábra*), és a felhasználók szélesebb köre számára teszi elérhetőbbé. Ha valaki egy problémát diszkrét lépésekre tud bontani, akkor egy folyamatot nézhet, és érzékelheti, hogy mit csinál, anélkül, hogy meg kellene értenie az egyes csomópontokon belüli kódsorokat. Ennek nagy előnye, hogy a legtöbb egyszerűbb automatizálási feladatot akár egy sornyi kód megírása nélkül is le lehet programozni. A különböző típusú csomópontok egy-egy adott feladat ellátására képesek, és amikor ezen csomópontok elhelyezésre kerültek, és a fejlesztő megnyitja az adott

csomópont beállításait, ott rádiógombokkal, paraméterekkel és legördülő menükkel teljesen konfigurálható az adott csomópont viselkedése.

2.4.1 Node-RED alkalmazása

A Node-RED egy Node.js alapú futási környezetből áll, amelyre a webböngészővel hozzá lehet férni a folyamatszerkesztő eléréséhez. A böngészőben úgy hozza létre a fejlesztő az alkalmazást, hogy csomópontokat húz a szélső palettáról egy munkaterületre, és elkezdi összekötni őket. Az alkalmazás egyetlen kattintással visszakerül a futási környezetbe, ahol lehet tesztelni vagy élesben alkalmazni. A csomópontok palettája egyszerűen bővíthető a közösség által létrehozott új csomópontok telepítésével, és a létrehozott folyamok egyszerűen megoszthatók JSON-fájlként. Az egész környezet böngésző alapú szerkesztőt biztosít, amely megkönnyíti a folyamok összekapcsolását a paletta csomópontjainak széles skálájával, amelyek egyetlen kattintással telepíthetők a futás idejére. A JavaScript függvények a szerkesztőben hozhatók létre. A beépített könyvtár lehetővé teszi, hogy hasznos funkciókat, sablonokat vagy folyamatokat mentsen el újra felhasználásra. A könnyű futási környezet a Node.js-re épül, teljes mértékben kihasználva esemény vezérelt, nem blokkoló modelljét. Ez ideálissá teszi a hálózat szélén való futtatást alacsony költségű hardvereken, például a Raspberry Pi-n, valamint a felhőben. A Node csomagtárában több mint 225 000 modullal könnyen bővíthető a palettacsomópontok tartománya új képességek hozzáadásával. A Node-RED-ben létrehozott folyamokat a JSON használatával tárolja, amely könnyen importálható és exportálható másokkal való megosztáshoz. Az online folyamatkönyvtár lehetővé teszi a legjobb folyamatok megosztását a világgal. [25]

Számos automatizálási megoldás került bemutatásra a NodeRED által, valamint lehetővé teszi a nagy adatok gyűjtését az IoT-rendszerekben, és ezért szerverfunkciót is biztosít. Gyakorlatilag egy egyszerűbb és ingyenes, csökkentett funkciókkal rendelkező SCADA (Supervisory Control and Data Acquisition – felügyeleti vezérlés és adatgyűjtési) rendszer létrehozására is alkalmas. A Node-RED adatai belső adatbázisban kerülnek tárolásra. Az alkalmazások képesek az adatokat az adatbázisból lehívni. Továbbá, az összegyűjtött adatok feldolgozhatók, hogy megkapjuk a kívánt információkat, valamint statisztikákat és elemzéseket készítésünk.

A Node-RED alacsony futtatási igényei és ingyenes hozzáférhetősége miatt kifejezetten olcsón telepíthet magának bárki ilyen rendszert, akár különböző fejlesztő NYÁK-ok, Raspberry-k és Arduino-k hálózatával, de ugyanígy képes kommunikálni sok ipari és IoT

szabványon keresztül is. A széles felhasználásának is köszönhető a nagy fejlődési ütem és a széles támogatottság. Több olyan vállalat is aktívan részt vesz benne, amelyek ipari elektronikák fejlesztésével foglalkoznak, mint például a Beckhoff [26], a Bosch Rexroth [27] vagy a Siemens [28]. A Node-RED képes kommunikálni a Siemens S7 PLC-kkel akár közvetlenül is, ismeri az MQTT, a Modbus vagy az OPC-UA szabványokat is.

A Node-RED telepítése parancssoros módon történik, legye akár Windows, akár Linux alapú rendszerre telepítve [29]. A Windows verzió működéséhez a Dokker nevű alkalmazás is szükséges, amellyel illeszthető a Node-RED, bár véleményem szerint ez a megbízhatóság rovására mehet, ezért vélszerű Linux alapú operációs rendszerben működtetni a szoftvert.

Telepítés és indítás után a Node-RED felület elérhető az adott számítógép IP címén (ugyanazon PC esetében a következő IP (localhost) címmel bemutatva): 127.0.0.1:1880. Jelen esetben a 1880-as portot veszi igénybe a Node-RED.

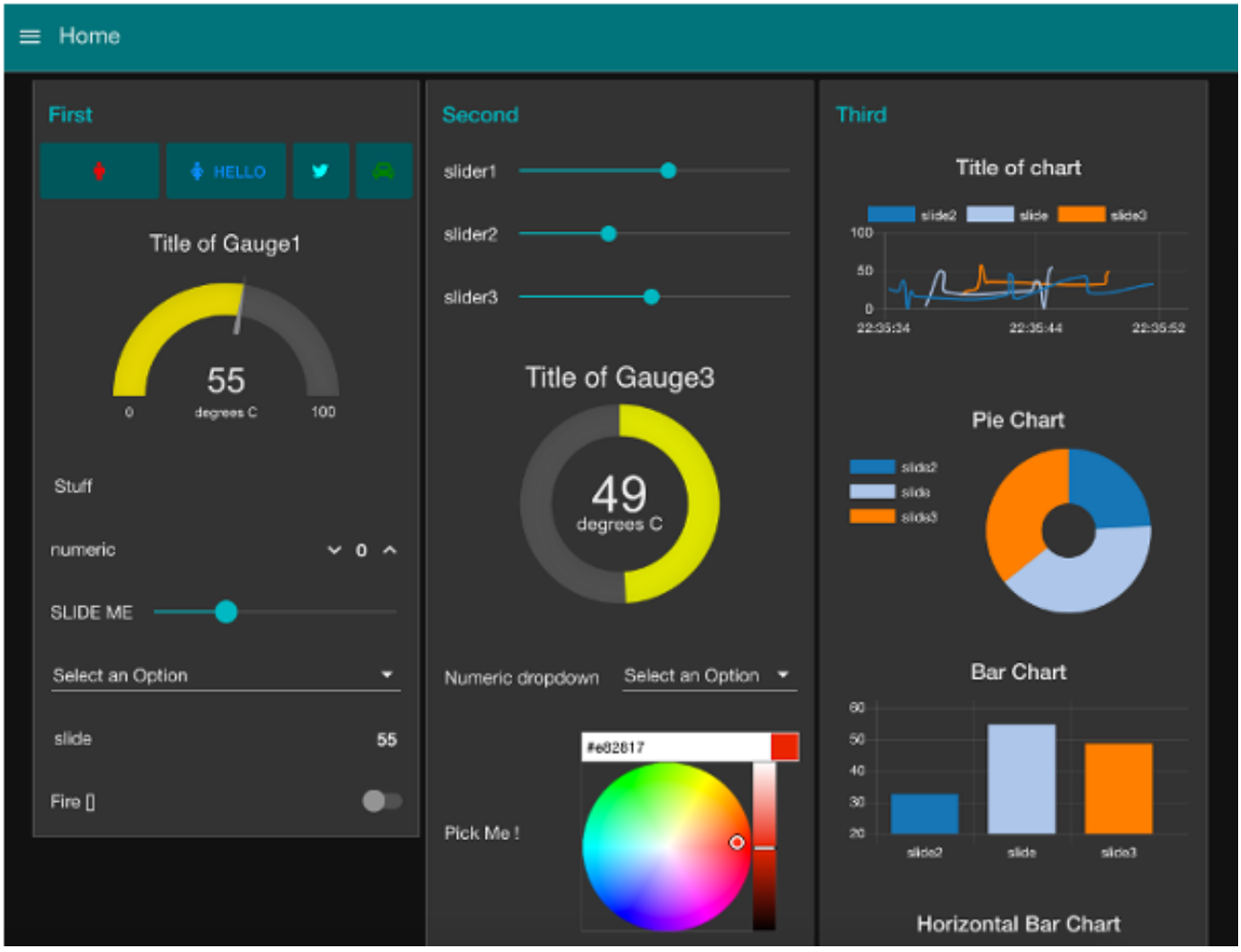
2.4.2 A Node-RED Dashboard

A Node-RED Dashboard egy olyan modulja a Node-RED-nek, amelynek segítségével egy böngészőből kezelhető felületet lehetséges létrehozni. A felületre elhelyezhetők grafikonok, gombok, számlálók, szövegmezők, vagy gyakorlatilag bármi, amit egy weboldalon megszokhattunk már (*6. ábra*). A felületet panelek alkotják, amelyekre a Node-RED csoportokként hivatkozik. Az *6. ábrán* „First”, „Second” és „Third” néven címkézett szegmensek ezek. Ezek a csoportok egységet képeznek, a rajtuk található elemek a csoport paramtereivel együtt változnak. A csoportokat füleken vagy oldalakon lehet elhelyezni, amelyet a böngésző ablak bal oldali hamburger-menüjéből lehet átváltani. A megfelelő beállításokkal egy nagyon könnyen kezelhető felhasználói felületet lehet létrehozni a fejlesztés tárgyát képező automatizációs feladat számára. [30]

A Node-RED Dashboard-ját a 127.0.0.1:1880/ui címen lehet elérni localhost módban.

A Dashboard felülete gyakorlatilag egy rácsos elrendezésen alapul, amelynek alapértelmezett szélessége 6 egység, amelyre egységenként lehet beállítani a komponensek vízszintes és függőleges méretét. Középre igazítás esetén úgynevezett „spacer”, azaz helykitöltő elemekkel érdemes dolgozni, mivel a Node-RED alapértelmezetten mindent a bal felső sarokba szeret tömöríteni.

Több kisebb csoport alkalmazása nem csak azért lehet célszerű, hogy több címkézést és jobb esztétikus élményt kapjon a felhasználó, hanem ezzel elérhető az is, hogy egy széles böngészőablakban a csoportok akár egymás mellé is rendeződhessenek, ahogyan az *5. ábrán* látható. Egy szűk képernyő, például mobiltelefon esetén ezek rendszerint csak egymás alatt kapnak helyet az alapértelmezett 6 egység szélességgel.



6. ábra: A Node-RED Dashboard [30]

Lehetőség van dinamikusan változtatni a Dashboard felépítését is, amennyiben a fejlesztő által definiált peremfeltételek teljesülnek. Például adott esetben el lehet rejteni vagy fel lehet fedni csoportokat is. Ehhez a következő parancsokat lehet felhasználni:

- Oldalak elrejtése
    - {"tabs":{"hide":["oldal_nev_alahuzasokkal_elrejtesre"],
      "show":["masik_oldal_neve_megjelenitesre"],
      "disable":["nem_hasznalt_oldal_neve_kikapcsolashoz"]}}

- Csoportok elrejtése

3. {"group":{"hide":["oldal_nev_csoport_nev_alahuzasokkal_elrejtesre"],
   "show":["oldal_nev_masik_csoport_nev_odaugrasra"],
   "focus":true}}Munkafolyamat bemutatása

A dolgozat során egy festő előkészítő rendszer kerül átalakításra. Ez a munkaállomás a vállalatnál készült termékeket ellátja azok tételszámával és egyedi azonosítójával egy pontgravírozó részegység segítségével, valamint a gravírozás után a munkadarabokat megmossák, amellyel az összes korábbi technológiából származó szennyeződések jelentős részét eltávolítják.

A termékek vasöntvény forgástestek, amelyeket forgácsolási munkálatoknak vetnek alá. Pontosabb leírást a dolgozat titkosítási kötelezettségének elkerülése végett nem tartalmaz a dolgozat. A forgácsolási munkálatok után következik a festő előkészítés, majd a festés. A festő rendszer rendelkezik egy saját tisztító állomással is, így tulajdonképpen az előkészítőben elegendő csak a szennyeződések nagyját, a port és a fennmaradt forgácsot eltávolítani.

3. Munkafolyamat bemutatása

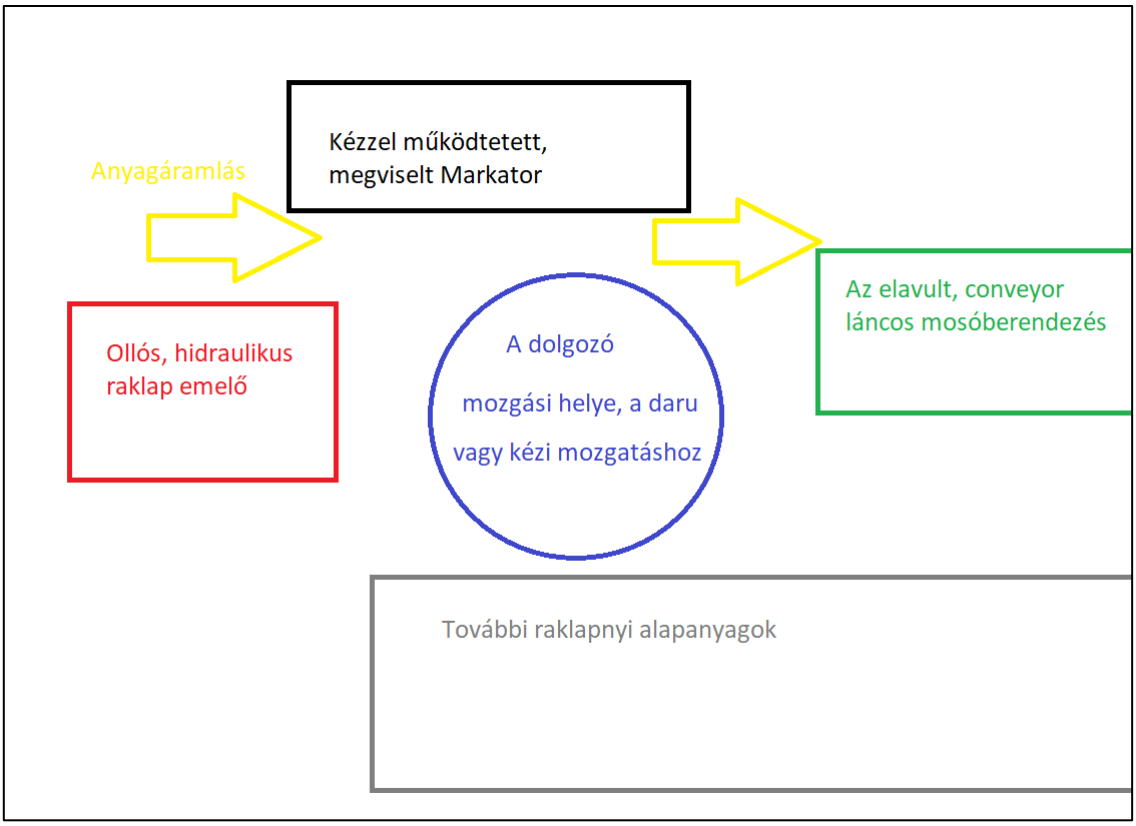
3.1 A jelenlegi munkafolyamat

A jelenlegi munkafolyamat során a termékeket egy targoncás dolgozó hozza a munkaállomáshoz. A termékek, ahogy korábban említésre került, vasöntvény forgástestek, amelyek súlya az öt kilogrammostól az 50 kilogrammosig terjed. Többek között ezért is került a jelenlegi állomás olyan kialakításúra, hogy az EU raklapon érkező termékeket egy hidraulikus raklap emelőre helyezik, ezzel állandó magasságon tartva az aktuális sort a dolgozók számára.

A termékeket a dolgozó a súlyuktól függően vagy kézi erővel, vagy kézi mozgatású daruval helyezi be a gravírozó gépbe. A gép jelenleg csak a tételszámot gravírozza be a termék oldalába, amelyet kézzel adnak meg minden típus váltás során a dolgozók. A gépet manuálisan indítják el, majd az végrehajtja a gravírozást.

A gravírozás végeztével a dolgozó behelyezi a terméket a conveyoros alagútmosóba. Az ebben található láncos pálya lassan átviszi a termékeket sorban egymás után a mosó alagútján, ahol megtörténik a mosás. A mosóban mindössze egy érzékelő található, amely a termék megérkezése esetén bekapcsolja a mosó fúvókát. A pálya folyamatosan halad. A rendszer jelenlegi vezérlése egy egyszerű relés szekrénnyel megoldott.

Az elmosott termékeket a conveyor pálya túlsó végén várakozó dolgozó felhelyezi a festő keretekre, amelyeket utána a festőrendszer már képes fogadni (*7. ábra*). A mozgatás itt is a termék súlyától függően daruval vagy kézzel történik meg. Az állomás működtetéséhez tehát két dolgozó szükséges, egyikük a mosás előtti szakaszon, másikuk a mosás utáni szakaszon tevékenykedik.



7. ábra: A régi előkészítő anyagáramlása és alaprajzi vázlata

3.2 A tervezett munkafolyamat

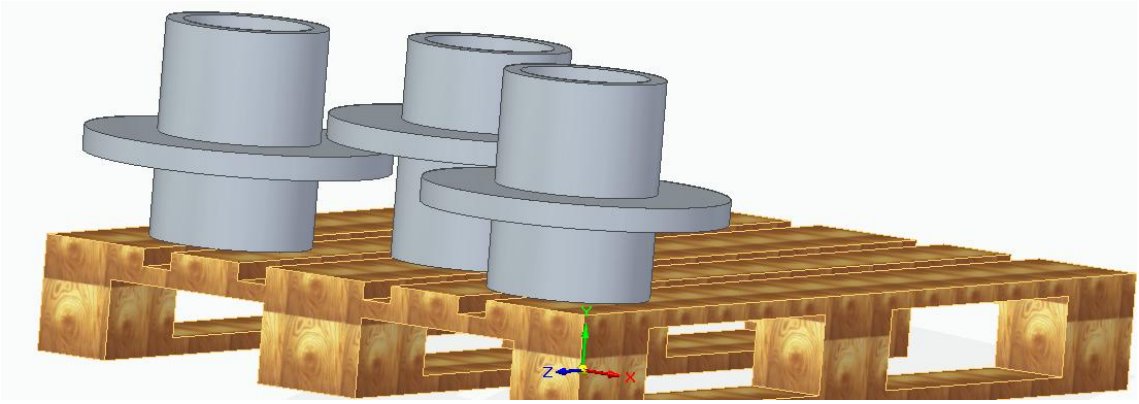
A tervezett, automatizált munkafolyamat hasonlóképpen kezdődik, mint a jelenlegi megoldás. Egy targoncás dolgozó a raklapon érkező termékeket behelyezi a munkaállomásra. A raklapokat ezúttal egy elektromos meghajtású láncos pályára helyezi el. A behordó pálya két elemből áll össze: egy két raklap hosszúságú, és egy egy raklap hosszúságú pályadarabból.

Erre azért van szükség, mert a termékek áramlása alapvetően egy irányba történik, be a cella belseje felé, azonban szükség lesz puffer tárolási lehetőségre is, tehát három raklap egymás utáni betárolására. Ez azért lehet jó, mert a cella akkor is képes dolgozni, ha sokáig nem jön arra targoncás egy új raklappal. Miután már csak a belső oldalon van egy darab raklap, amelyen már a cella dolgozik, úgy egyszerre két raklapot csak abban az esetben lehet behelyezni, ha azok mozgatása külön vezérelhető az aktuálisan feldolgozás alatt lévő raklapéhoz képest. Így az egy raklap hosszú láncos pálya kerül belülre.

A beérkezett raklapot a fölötte elhelyezkedő lézeres 3D szkenner beolvassa, felismeri a kiemelhető termékeket. A szkennerhez tartozó bin picking szoftverben beállíthatók prioritások, ami alapján eldönti a szoftver, hogy melyik felismert terméket vegye ki először a robot.

A kiválasztott termék pozíciójához hozzászámolja a szoftver a robot megfogó geometriáját, annak megfogási pozícióját, majd odavezeti a robotot a termék fölé. A szoftver képes teljes útvonalat generálni a robot számára mind a pozícionálás, kiemelés és a konténerből vagy raklap fölül történő eljárás megvalósításához. A szkenner látóteréből kiérve a robottal egy újabb szkennelés történik, ezzel a számítás és pályagenerálás is megtörténik addig, amíg a robot az aktuálisan kivett terméket kezeli.

A robot ezután a terméket szükség esetén elviheti egy fordító állomásra. Ezen fordító állomásra azért van szükség, mert a termékeket a raklapra sokszor úgy helyezik el, hogy azok egymás mellett minden második darab esetén „fejjel lefelé” vannak, ezzel helyet takarítva meg a raklapon (*8. ábra*). Ennek a hátulütője, hogy a mosás és gravírozás nehezen adaptálható a változó orientációjú termékekre. A megoldást az jelenti, ha egy fordító állomást alkalmazunk, amely a „fejjel lefelé” pakolt termékeket átfordítja a kívánt orientációba.



8. ábra: A raklapon lévő termékek elrendezése egyszerűsített modellekkel

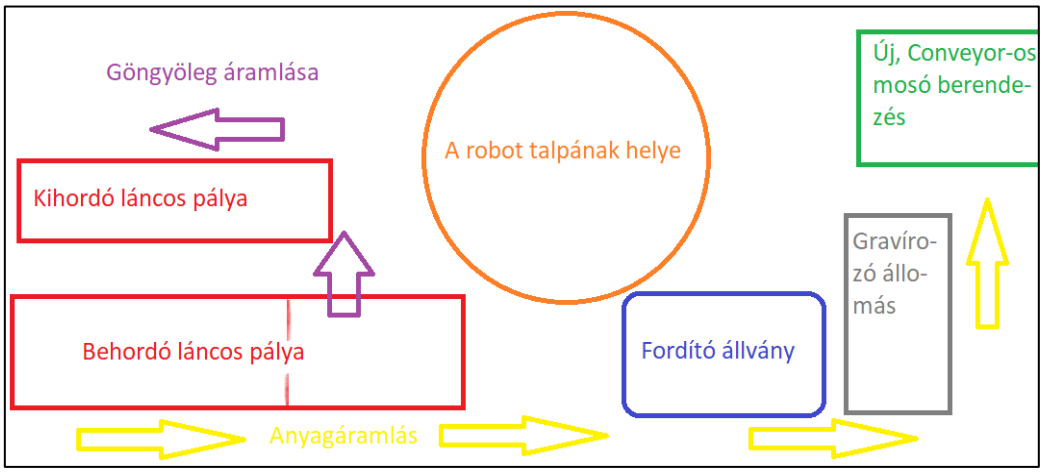
Akár a helyes orientációban kivett, akár a fordítás után helyes orientációban megfogott termékeket ezután a robot felhelyezi egy gravírozó berendezésbe. Miután a robot biztonsági távolságra került a gravírozótól, megtörténik a gravírozási folyamat. A gravírozót egy szervóhajtás közel tolja a termékhez. Ekkor a gravírozó már nem csak a tételszámot üti be a termékbe, hanem generál és beüt egy egyedi azonosítót is hozzá 2D data mátrix formában, amely a későbbi termék-életút nyomkövetéshez lehet hasznos. A termékek hengeres alakja miatt korlátozott az egy termékpozícióban beüthető információ mértéke, ezért egy másik szervóhajtás fordítást végez a terméken annak forgástengelye mentén a tételszám és a 2D kód gravírozása között.

A gravírozott termékeket a robot megfogja, majd átemeli a conveyor pályás mosó berendezésbe. A mosó berendezés saját PLC-vel és vezérléssel rendelkezik majd, így egy

intelligensebb, alaposabb, termékfüggő mosást is képes lehet elvégezni. A dolgozat írásakor a mosó berendezés még gyártás alatt állt, csak az igényelt specifikációk voltak ismertek, a vele történő kommunikáció és vezérlés ezáltal nem képezi a dolgozat részét.

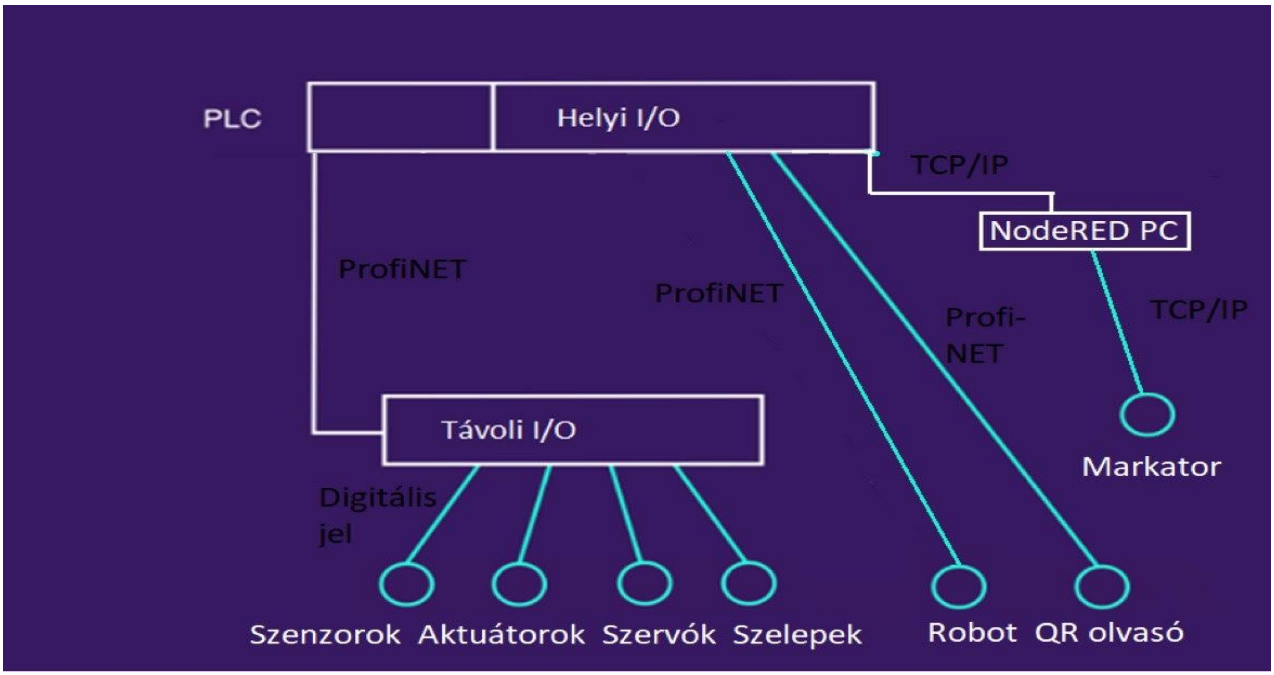
Miután a robot lerakta a terméket a mosóra, következhet a következő termék, amelyet már a szkenner szoftvere ki is választott. Mivel a termékek több rétegben helyezkednek el, elengedhetetlen, hogy elválasztó lemezeket helyezzenek a rétegek közé. Amikor egy réteg leürül, akkor az elválasztó lemezt el kell távolítani, hogy a robot hozzáférjen a következő réteghez. A roboton található megfogó egy olyan egyedi, kombinált rendszer, amelynek mágneses megfogója a termékeket, vákuumos megfogója pedig az elválasztó lemezeket képes megfogni.

A szkenner jelzi, hogy nem talált terméket, ekkor a robot az utolsó darab pozíciójából tudja, hogy hol lesz az elválasztó lemez, amelyet utána egy láncos pályára kihelyez. A láncos pálya a behordó pálya mellett kap helyet (*9. ábra*). Miután az összes termék és elválasztó lekerült a raklapról, a raklapot is a vákuumos megfogó segítségével helyezi ki a robot a göngyöleg kezelő pályára. Egy adott magasság felett a robot jelez, majd a láncos pályát ki kell járatni a cellából.



9. ábra: A tervezett robotcella felépítése

A dolgozat és a projekt készítése során Siemens S7-1500 szériás PLC-t, több ProfiNET terepi modult alkalmazunk, amelyek működését kiegészíti majd a Node-RED mint protokoll illesztő megoldás a PLC és a gravírozó között, valamint mint HMI a kézi típusmegadáshoz, a kézi interakcióhoz és információ megjelenítéshez. Ezen felül a Node-RED rendelkezni fog egy rendszergazda lappal is, amelyet jelszó véd és amellyel kritikusabb kézi vezérlések végezhetők el a cellán belül. A dolgozat során bemutatom ezek működését és fejlesztését, implementációját.



10. ábra: A tervezett kommunikációs architektúra sematikus ábrázolása

3.3 A HMI leendő feladatai

A robotcellához célszerű elhelyezni általában egy HMI képernyőt, amelyen a dolgozó a cella folyamatait, állapotát nyomon tudja követni, egyes beavatkozószerveket is lehetséges elhelyezni, valamint akár emelt jogosultsággal rendelkező funkciók is létrehozhatók itt.

A fő megvalósítandó célok között ezek szerepelnek, létre kell hozni egy informatív HMI felületet ahol a dolgozó láthatja a gravírozó, a hajtások, a robot állapotait, igény szerint kezelni tudja a láncos pályát, illetve egyedi típusmegadásra is lehetősége van.

A megadott jelszó birtokában megjeleníttethető lesz az emelt fokú jogosultságot igénylő adminisztrátor fül, ahol a megjelenő hibaüzenetekre lehet érdemben reagálni, illetve itt elérhető lesz néhány beavatkozó szerv működtetése is, melyet a dolgozó az admin fülbe való belépés nélkül mozgatni nem lenne képes. Ez a felület a komolyabb fennakadások és hibák hatékony elhárításában játszik majd fő szerepet.

A cella működéséhez alapvetően igény van arra, hogy a nagyon kis darabszámban gyártott termékek esetén, vagy ha a roboton karbantartás vagy meghibásodás történik, akkor a kézi rakodás a mosó berendezés irányába, valamint a gravírozó felé adott legyen. A mosó a termékek típusától függetlenül fog mosni, esetleg a későbbiekben a típusok magasságától függően később kapcsolja csak be a mosást a mosófej mozgása során, így a cella jelenlegi készültsági állapota alapján nem jelenthető ki, hogy a mosó alapvetően igényelné kézi adagolás esetén a terméktípus ismeretét. Ezzel szemben azonban a gravírozásra akkor is szükség van, amikor kézi adagolás történik, amelynél a gravírozás programkódját ki kell választani. Ezt a HMI felületen keresztül lehet megtenni majd. A robot üzemen kívülisége esetén a feltanított típusok gravírozását is ki kell tudni választani a HMI-ről, ezzel akár több, mint száz különböző opciót adva a dolgozónak. Ez már önmagában szükségszerűvé teszi a HMI-n keresztül történő választást.

4. NodeRED és PLC összekapcsolása

4.1 A NodeRED telepítése és a mini PC

A NodeRED alapvetően egy Linux alapú PC-n vagy mikroszámítógépen fut a legbiztosabban, ezért a cella ellátásához egy ASUS mini PC-t kaptam a vállalati IT-tól, amelyet már nem használtak irodai célokra, de teljesítménye bőven megfelelő a robotcellában futtatandó NodeRED alá. A PC egy ASUS UN65U VivoMini, amelyben egy 7. generációs Intel i5 processzor található. Más feladata nem lesz, mint a NodeRED és az azon belül elkészült HMI futtatása. Diagnosztikai okokból egy Windows 10 is telepítésre került rá, amelyen a TIA Portal is megtalálható, így egy újraindítást követően akár a PLC programjában is lehet diagnosztikai adatokat végezni. Fontos megjegyezni azonban, hogy ekkor a NodeRED nem fut, ezért, ha olyan diagnosztikára van szükség, ahol a cella minden eleme fut, akkor az elektromos szekrényben található Ethernet switch-re kell a diagnosztikát végző személy laptopjával felcsatlakozni.

Miután az Ubuntu telepítésre került a mini PC-re, gyakorlatilag azonnal kiadható a következő parancs, amellyel a NodeRED telepítése megtörténik (ennek előfeltétele, hogy a *bash* és *curl* parancsokat ismerje a rendszer, amely nem minden esetben alapértelmezett modul az operációs rendszerben, azonban azonnal felajánlja a parancsokhoz tartozó szoftvercsomag telepítését az OS):

```
bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-Node-RED)
```

A telepítés egy pár percet igénybe vesz. A Pi telepítési szkriptjével beállítottam azt is, hogy szolgáltatásként fusson. Ez azt jelenti, hogy futhat a háttérben, és engedélyezve van, hogy rendszerindításkor automatikusan elinduljon. Ahhoz, hogy aktiváljam ezt a funkciót a következő parancsokat adtam meg:

```
sudo systemctl enable Node-RED.service
```

A parancs futtatása után a NodeRED automatikusan indul, így például abban az esetben, ha az alkalmazás nem igényli, nincs is többé szükség a futtató számítógépet perifériákkal ellátni. Elegendő a PC IP címének ismerete a használatához. Ez a megoldás alkalmazható lenne a jelen robotcella esetében is, azonban nem célravezető. A robotcella esetében a HMI felülete működhet a NodeRED-et futtató PC-ról közvetlenül is, amely esetben a böngészőben a http://localhost:1880 címet kell megnyitni.

A fenti IP címen a NodeRED szerkesztő, programozó felülete található meg. Ezt a felületet célszerű ellátni egy adminisztrátori jelszóval, hogy a cella telepítése után véletlenül se férhessenek hozzá a dolgozók, ezzel akár jelentős anyagi károkat okozva. A NodeRED fejlesztői többféle megoldást is kidolgoztak többféle biztonsági fenyegetés ellen [31], azonban ezek egy részére nem lesz szükség, mivel a robotcella nem kerül be a céges belső hálózatba, valamint semmilyen más hálózatba sem, kivétel ez alól a saját lokális hálózat a cellán belül. Az így kihagyható védelmek a HTTP és HTTPS témakört érintik. Mindössze a szerkesztő és az admin API védelme fontos.

A szerkesztő felületet felhasználónév és jelszó kombinációjával lehetséges lezárni, amely esetében a jelszót valamilyen kódolt formában kell eltárolni a NodeRED egyik konfigurációs fájljában. Alább látható két felhasználó beállítása a fájlon belül, különböző jogosultságokkal. A „*” jelöli a korlátlan jogosultságot, míg a „read” csak olvasást tesz lehetővé.

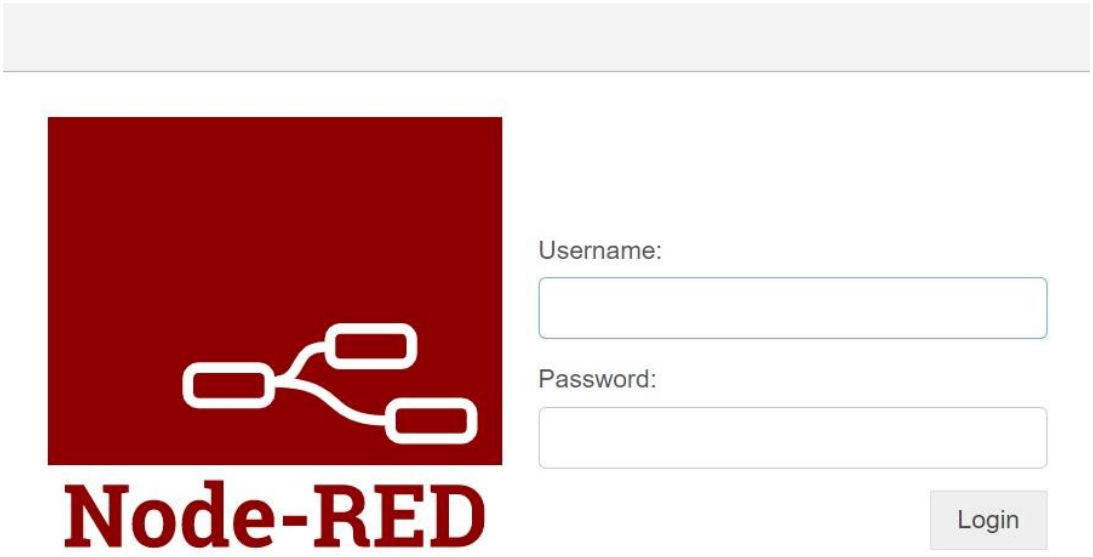
```
adminAuth: {
    type: "credentials",
    users: [
        {
            username: "Rendszer_Gazdi",
            password: "$2a$08$zZWtXTja0fB1pzD4sHCMyOCMYz2Z6dNbM6tl8sJogENOMcxWV9DN.",
            permissions: "*"
        },
        {
            username: "István",
            password: "$2b$08$wuAqPiKJlVN27eF5qJp.RuQYuy6ZYONW7a/UWYxDTtwKFCdB8F19y",
            permissions: "read"
        }
    ]
}
```

Utóbbi jogosultságot megkaphatják a dolgozók, vagy esetleg csak a team vezetők, míg az admin jogosultság a géptelepítők és a folyamatmérnökök jogköre lehet.

A jelszó az alábbi paranccsal generálható le a fent látható hash formátumra:

```
node-red admin hash-pw AzÉnJelszavam
```

A kapott eredményt kell bemásolni a konfigurációs fájlba, majd ezután elmentve azt, valamint újraindítva a NodeRED szolgáltatást, már működni is fog a jelszavas védelem (*10. ábra*). Lehetőség van további biztonsági rétegeket is hozzáadni a jelszavas védelemhez, de a robotcella jelenlegi készültségi szintjén ezek nem indokoltak.

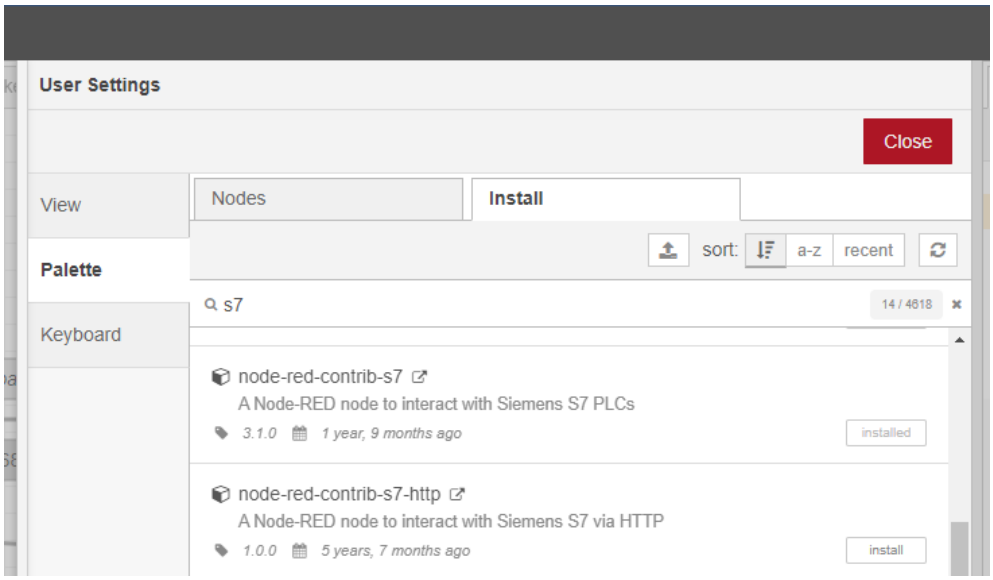


11. ábra: A NodeRED szerkesztőhöz való hozzáférés előtt kéri a jelszót

4.2 A NodeRED konfigurálása a PLC-hez

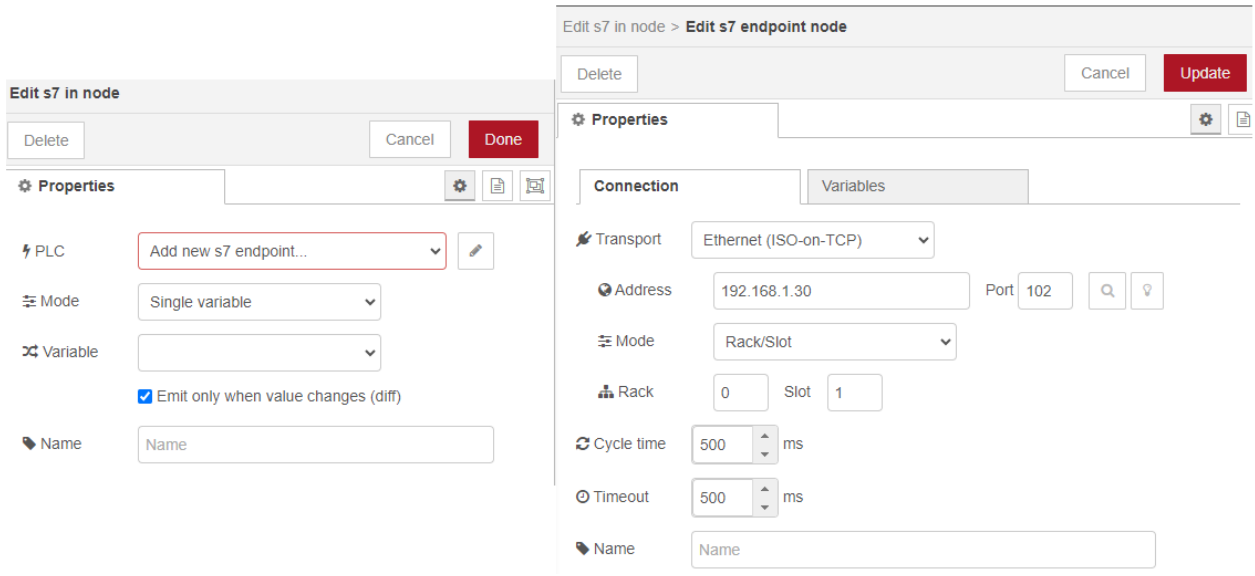
Ahhoz, hogy a cella belső hálózata megfelelően működjön, elengedhetetlen, hogy minden eszköz ismerje a másik eszközt, ha azoknak kommunikálnia kell egymással. Ezt a legegyszerűbb úgy kivitelezni, hogy minden eszköznek fix IP címe van. A cella hálózata a 192.168.1.0/24 hálózatot használja, amelybe minden csatlakozó eszköznek meg van adva a saját statikus IP címe. Legjobb tudomásom szerint a NodeRED nem is lenne képes DHCP szerveren keresztül címzett PLC-vel kapcsolatot teremteni, tehát igényli a fix IP címeket.

A NodeRED szerkesztőbe történő belépés után az üres flow ablak fogad, amelyre elhelyezhetők a folyamat elemei. A képernyő bal sávján (*5. ábra*) találhatók a már telepített és előre telepített node-ok vagy csomópontok, amelyek a flow felületére az egérrel húzhatók fel. A Siemens S7 PLC-kkel való kapcsolódást azonban nem tartalmazza alapértelmezetten a NodeRED, ezért azt importálni szükséges. A képernyő jobb felső sarkában található hamburger-menüben található a Manage Palette menüpont, amelynek az Install fülén belül kell megkeresni a „node-red-contrib-s7” névre hallgató csomagot, majd telepíteni azt (*11. ábra*). Ezután megjelenik a bal oldali menüben az „plc” névre hallgató csoport, amelyben az „s7 in”, „s7 out” és „s7 control” csomópontok találhatók. Az előbbi kettőt alkalmaztam a programozás során.



12. ábra: A NodeRED kiegészítők telepítése

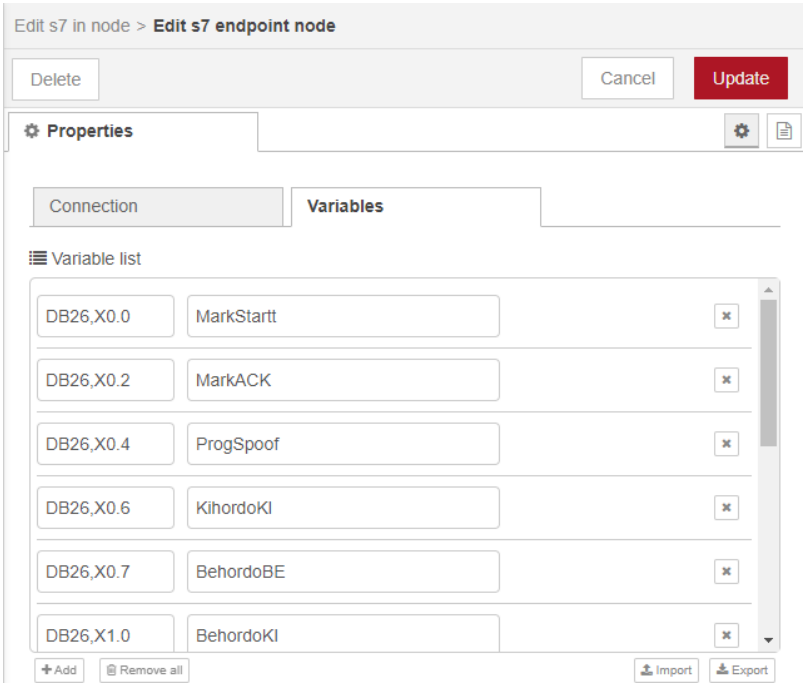
Amikor az első S7-es csomópontot behúztam, még nem ismerte a PLC-t, így azt be kellett konfigurálni. A *12/a ábrán* látható ceruza ikonra kattintva lehet létrehozni és konfigurálni egy s7 végpontot (*12/b ábra*). A PLC IP címét szükséges beállítani első körben, mivel a többi beállítás alapértelmezetten megfelelő lehet. A ciklusidőt és a várakozási időt is 500 ms értékre állítottam, ezzel gyorsabb frissítést és jobb reakciót elérve. Variables fülön a PLC-ben beállított adattábla változóit lehetséges megadni, amelyeket az S7 csomópont ezután figyelni fog.



13. ábra: a, Az "s7 out" csomópont konfigurálása; b, Az S7 végpont létrehozása

A változókat egy táblázatba lehet beírni, és a címzésüket bitenként lehet megadni [32]. Egy változó címzése a PLC-ben lévő adattábla sorszámából és az adattábla adott bitjének sorszámából áll össze. Például a robotcella során a 26-os adattáblát használtam a NodeRED

kommunikációra, így annak első bitje a DB26,X0.0 címen érhető el. Ezután ezt a változót el kell nevezni a NodeRED-en belül (*13. ábra*). Biteknél nagyobb egységeket is meg lehet adni, ekkor azonban csak egy vagy több bájtonként lehet előre haladni, például egy egész szám, amely az adattábla harmadik bájtjától kezdődik, a DB26,I3 címen található meg, és egészen a DB26,X4.7-ig elfoglalja a címeket.



1410. ábra: Az S7 végpont változóinak beállítása

A későbbiekben az itt megadott változóneveket fogják alapértelmezetten felvenni a NodeRED kapcsolódó s7 csomópontjai.

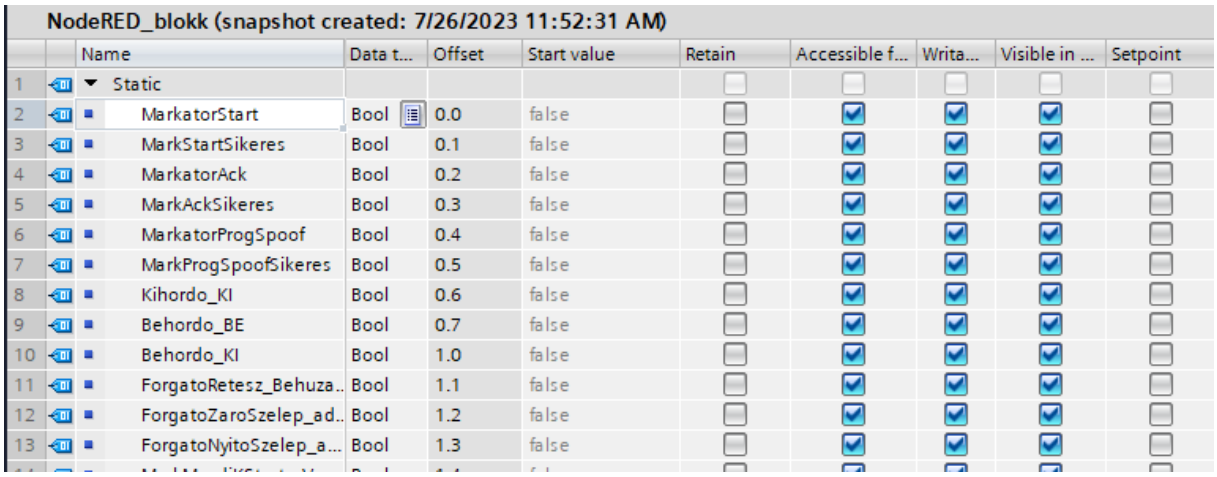
4.3 A PLC konfigurálása a NodeRED-hez

Mivel a dolgozatnak nem képezi részét a PLC programozása, így a dolgozat szemszögéből a PLC már előre a cellához leprogramozásra és konfigurálásra került minden olyan téren, amely nem érinti a NodeRED-et. Ez többek között azt jelenti, hogy a PLC program maga már megírásra került, a PLC alap konfigurációja, a hálózati beállítások, a ProfiNET már készen vannak és működnek. Ezt folytatandó, hogy a PLC a NodeRED-del képes legyen kommunikálni.

Az első lépésként engedélyezni kell azt, hogy külső eszköz belenyúlhasson a PLC-ben történő adatolvasásra és írásra. Ezt a PLC „Protection and Security” menüjében a „Permit access with PUT/GET communication from remote partner” bepipálásával lehet elérni. Ez a funkció

engedélyezi a NodeRED számára, hogy a megfelelő címzéssel ellátott változókat írja és olvassa.

A következő lepésként létre kell hozni egy adattáblát, amelyet a NodeRED-del történő kommunikációra használ a PLC. Ez jelen esetben a NodeRED_blokk [DB26]. Az adatblokk beállításain belül az „Optimized block access” mellől ki kell venni a pipát. Ezzel az adatblokkban létrehozott változóknak már fix logikai címe lesz, amelyet kívülről elérhet a NodeRED. Az „Offset” oszlopban található meg a *14. ábrán*, hogy az adott változó mely bitcímen található meg. Minden változtatásnál újra kell a teljes táblát programozásilag fordítani (Compile).



NodeRED_blokk (snapshot created: 7/26/2023 11:52:31 AM)

	Name	Data t...	Offset	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint
1	Static								
2	MarkatorStart	Bool	0.0	false					
3	MarkStartSikeres	Bool	0.1	false					
4	MarkatorAck	Bool	0.2	false					
5	MarkAckSikeres	Bool	0.3	false					
6	MarkatorProgSpoof	Bool	0.4	false					
7	MarkProgSpoofSikeres	Bool	0.5	false					
8	Kihordo_KI	Bool	0.6	false					
9	Behordo_BE	Bool	0.7	false					
10	Behordo_KI	Bool	1.0	false					
11	ForgatoRetesz_Behuza..	Bool	1.1	false					
12	ForgatoZaroSzelep_ad..	Bool	1.2	false					
13	ForgatoNyitoSzelep_a...	Bool	1.3	false					

15. ábra: A NodeRED-hez társított adattábla felépítése

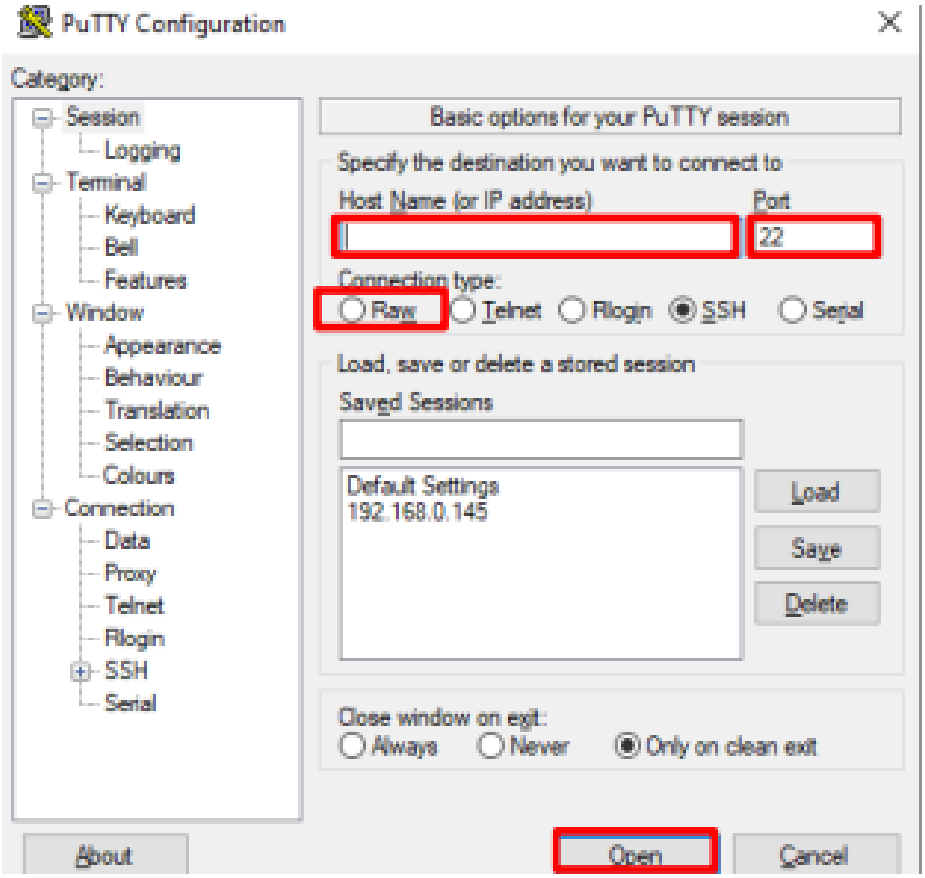
A fentiek létrehozása után a PLC program befejezhető a megfelelő változók ismeretében.

5. A NodeRED folyamat programozása

5.1 A pontgravírozó eszköz feltanítása, állapotfelügyelete

A NodeRED implementálásának fő oka a Siemens PLC és a Markator pontgravírozó (későbbiekben: Markator) közötti kommunikációs nehézségek. Ugyan a PLC-vel sikerült az „Open User Communication" blokkcsomagon belül található TCON, TSEND blokkokat működésre bírni, azaz a kapcsolat és a parancsok küldése a Markator felé szinte hibátlanul működött, azonban semmiféle visszajelzést nem sikerült beolvastatnom a Markator felől a PLC-be. Adatblokkok túlcsordulása és hasonló problémákba futottam bele folyamatosan. Persze a Markator rendelkezett egy részben konfigurálható digitális IO csatlakozással, amellyel adott esetben akár a teljes üzemeltetése is megoldható lenne, azonban a jelenlegi cellában szükséges volt akár száz különböző program közül válogatni úgy, hogy programválasztási hiba esetén azonnal tudja a rendszer, hogy nem mehet tovább a folyamat.

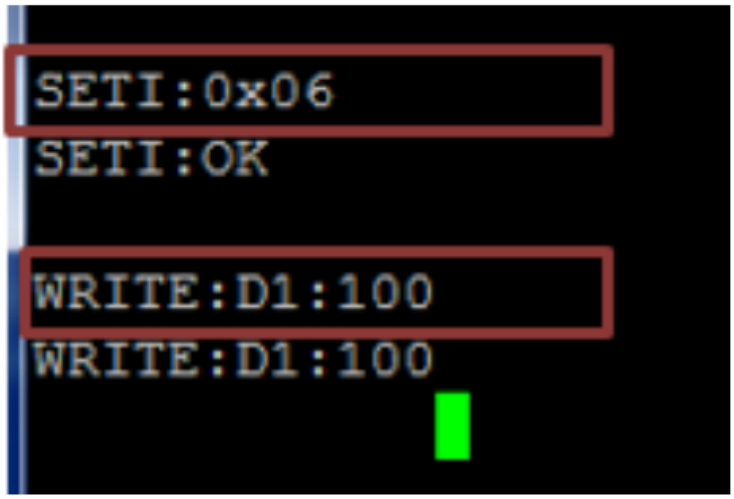
A Markator rendelhető volt digitális IO kártyával és/vagy ProfiNET kártyával, valamint Ethernet kommunikációs opciós csomaggal. A hozzám került Markatorban sajnos a ProfiNET kimaradt. A Markator pontgravírozót olyan körülményes programozni, hogy szinte esélyt sem láttam a folyamat során, hogy azt a digitális IO jelein keresztül irányítsam, a válaszok kezelésével kapcsolatban pedig még komolyabb fenntartásaim voltak. Képes ugyan az Ethernet csatlakozóján keresztül kommunikálni, azonban csak egy nagyon alap szinten. Ezt a PuTTy programon keresztül lehet gyakorlatilag parancssoros módon keresztül használni (*15. ábra*).



16. ábra: A PuTTy konfigurálása a Markatorral való kommunikációhoz [33]

Ezen parancssoros interfészen keresztül rövid parancsokat, kódokat lehet küldeni a Markatornak, amelyeket hasonló válaszokkal viszonoz. A gyártói leírás tartalmaz ehhez egy táblázatot, hogy mely kóddal mely funkciók valósulnak meg [33].

A gravírozás megkezdése előtt szüksége van egy engedélyező jelre a TCP porton keresztül, amelyet a *16. ábrán* is látható „SETI:0x06” paranccsal lehet elérni. Ez tulajdonképpen a TCP porton keresztül működő virtuális IO beállítását végzi el. A Markator válasza erre a „SETI:OK”. Ahhoz, hogy programot ki lehessen választani a típusnak megfelelő kód gravírozásához, változókat is kell átírni. Lehetőség van változóba beírt nevű programokat lefuttatni, így a meglátás az, hogy 001-999 közötti névre hallgató programokat futtat le, amelyeket majd a D1 változóba ment el a Markator, ahogy a *16. ábrán* is látható.



17. ábra: A Markatornak elküldött parancsok keretezve, alatta a válaszok [33]

A „SETI” jellegű parancsok argumentuma mindig 0x01 és 0x0F között van. A Start, a „Stopp” (2 darab „p” betűvel írva, feltételezésem szerint ez a TCP stop jel fordított logikájú kikapcsolása), a Kilépés és az Univerzális Bement állapotainak kombinációját jelenti. A 0x06 az egyetlen, amit a cella alkalmaz, ennél a Start és az Univerzális Bemenet inaktív, a többi pedig aktív.

A start jelet a „START” parancsra is ki lehet adni, valamint egy adott (például D1) változó értékét a „READ:D1” parancsra lehet kikérni. A változót a *16. ábrán* is látható módon a „WRITE:D1:100” paranccsal lehet a 100-as értékkel egyenlővé tenni.

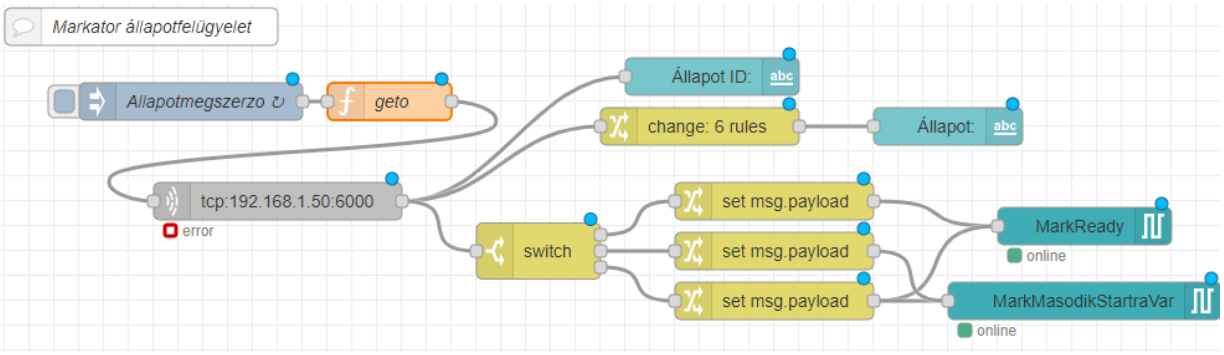
A Markator állapotát a GETO parancs adja meg, amely szintén egy 0x01 és 0x0F közötti intervallumon értelmezhető választ ad, például „GETO:0x03”. A válasz értelmezése a *17. ábrán* áttekinthető.

HEX	Home position	Ready signal	Error	Universal Output
0x00	0	0	0	0
0x01	1	0	0	0
0x02	0	1	0	0
0x03	1	1	0	0
0x04	0	0	1	0
0x05	1	0	1	0
0x06	0	1	1	0
0x07	1	1	1	0

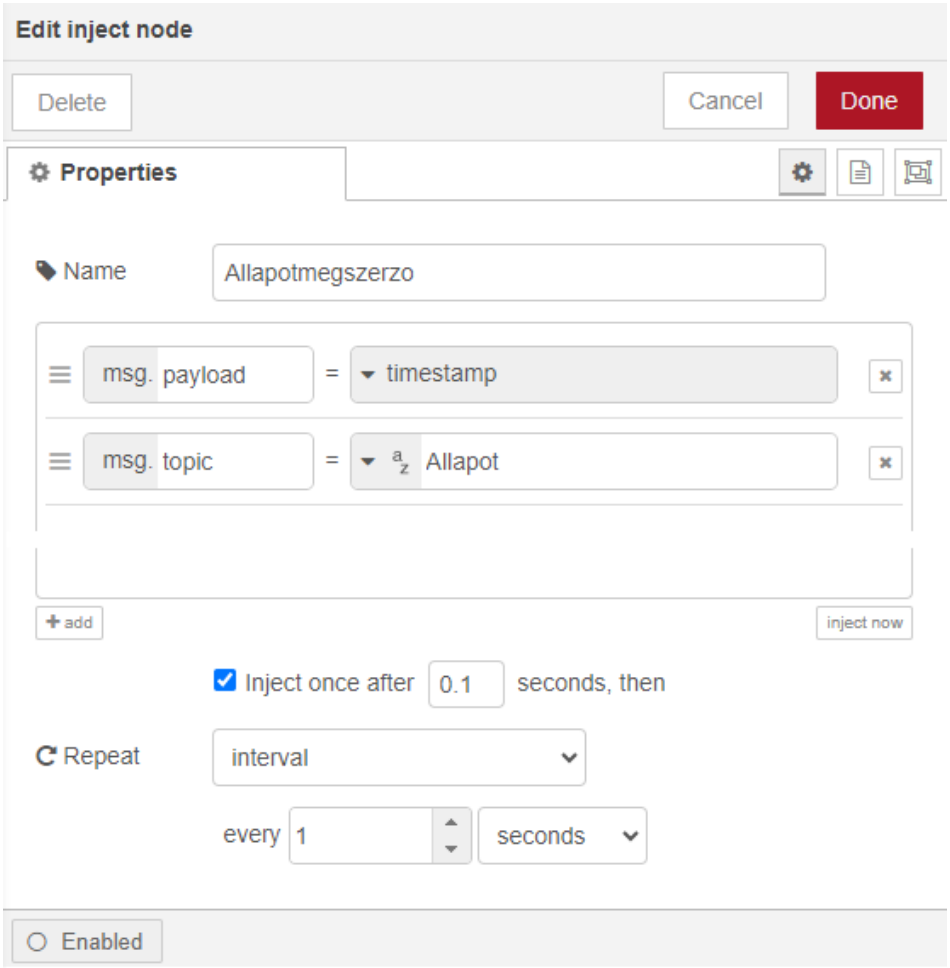
1811. ábra: A Markator GETO parancsra adott válaszában található argumentum értelmezése (részlet) [33]

A kommunikációs problémára a NodeRED nyújtott megoldást, mivel sikerült a Markator és a NodeRED között a problémamentes, kétirányú kommunikáció. A NodeRED bal oldali menüjében található egy „network” csoport, amelyben a hálózati kommunikációs csomópontok találhatók. A feladat ellátására a „tcp request” csomópont felelt meg, mivel ez a csomópont rendelkezi ki és bemeneti oldallal is, így azonnal képes a kiküldött parancsokra érkező válaszokat tovább küldeni. Ebből a csomópontból mindössze kettőt helyeztem el a teljes NodeRED programon belül. Az egyik feladata az, hogy úgymond „pingelje” a Markator-t annak állapotjelentést kérő parancsával, majd azt továbbítsa a HMI felületre. A másik feladata pedig a konkrét működtető parancsok és válaszok feldolgozása. A két csomópont paraméterezése teljesen megegyezik.

Az állapotfelügyeleti folyamat a *18. ábrán* látható módon épül fel. A folyamat egy „timestamp” csomópontból indul ki, amely úgy van konfigurálva, hogy a NodeRED indulása után az alapértelmezett 0,1 másodperces késleltetéssel azonnal küldjön egy üzenetet, majd ezután másodpercenként ismételje azt folyamatosan (*19. ábra*). Az üzenet tartalma az msg.payload változóban kap helyet, amely jelen esetben csak egy időbélyeg.



19. ábra: A Markator állapotfelügyeleti folyamata a NodeRED-en belül



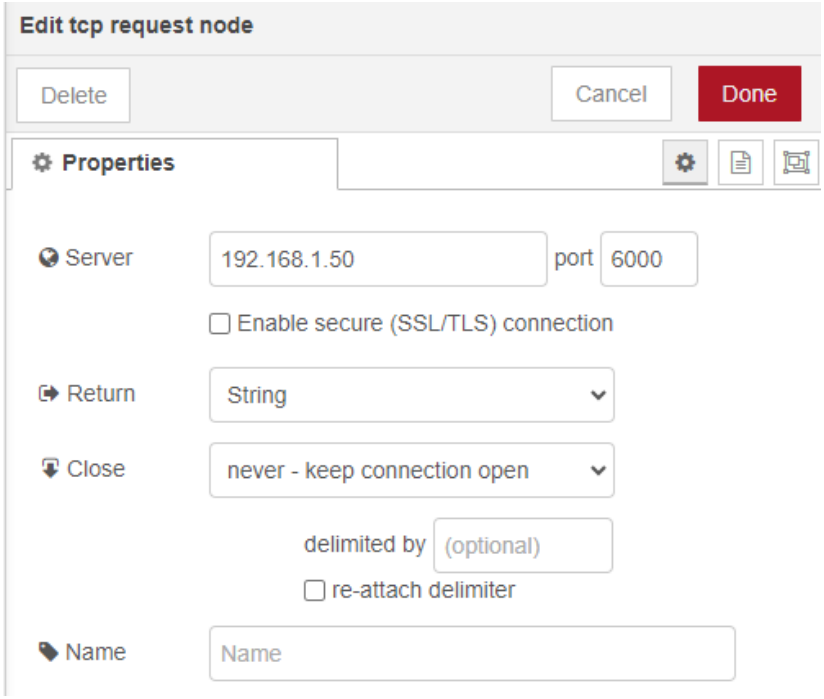
20. ábra: A NodeRED "inject" (beszúrás) csomópontja, másodpercenkénti ismétléssel

A következő lépésben egy Function (függvény) csomópont kicseréli az üzenet tartalmát a Markator állapotellenőrző üzenetére (GETO). A felhasznált parancs alább látható.

```
msg.payload=" GETO¥r" ;
return msg;
```

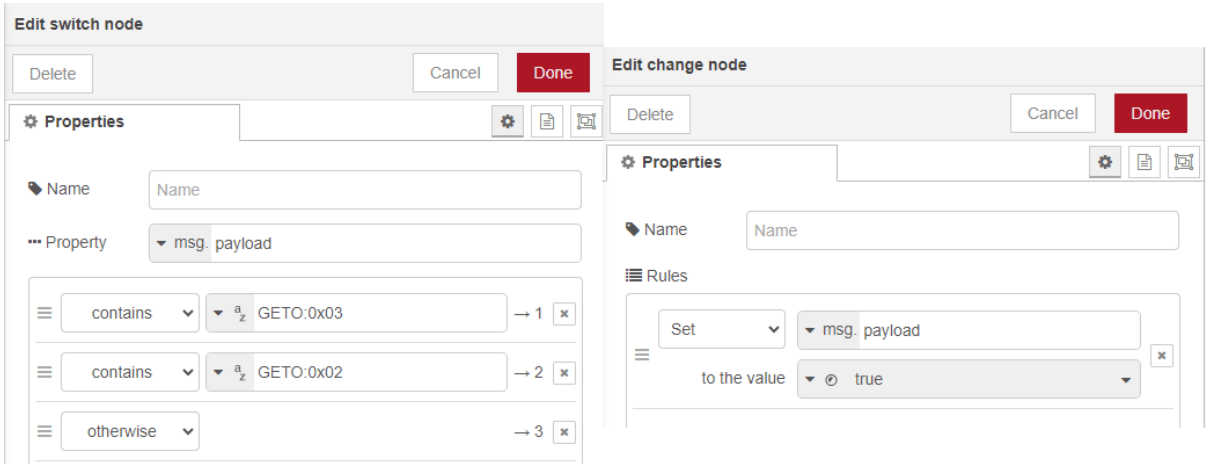
Az üzenet vagy csomag ezután a „tcp request" csomópontba lép be, ahol az továbbításra kerül a Markator-nak. A csomóponton belül beállítható (*20. ábra*) a cél IP cím és port, a

válaszüzenet továbbításának módja, a kapcsolat bezárására lehetőség, valamint a csomópont fantázianeve. Mivel azt szeretném, hogy a kapcsolat sose záruljon le, helyette pedig folyamatosan adatáramlás történjen, ezért az ennek megfelelő opciót választottam ki. A további feldolgozást a válasz üzenetek tekintetében a String típus teszi a legkönnyebbé. A Markator IP címe előre meghatározott és konfigurált, a felhasznált portot pedig a gyártó fixen a 6000-es helyen rögzíti.



21. ábra: A "tcp request" konfigurációja a Markator állapotfelügyeletéhez

A „tcp request” csomópont kitöltése után működik a kommunikáció a PLC és a Markator között. A csomópont kimenetén 3 további folyamatsor indul el (*18. ábra*). Ezek közül a felső közvetlenül, a második egy további feldolgozás során a későbbiekben bemutatott HMI felületre írja ki a Markator állapotát. Az alsó „switch” csomópont felé induló folyamat a válaszok függvényében a PLC-ben található változókat írja felül (*21/a ábra*). Három kimenetet konfiguráltam a csomópontban, ami azt figyeli, hogy a Markator készen áll-e a start jelre és hogy ezt a Home pozícióban teszi-e. A válaszok függvényében a három kimenet közül az egyikre megy az üzenet tovább. Például a hármas kimenetre abban az esetben, ha a Markator nem áll készen a Start jelre, akár hiba miatt, akár mozgásban van. A három kimenet három „change” csomópontba megy, ahol az üzenet tartalmát logikai igaz vagy hamisra cserélem ki (*21/b ábra*), majd az üzenetek továbbmennek egy „s7 out” csomópontba a megfelelő változókat írva a PLC memóriájában.



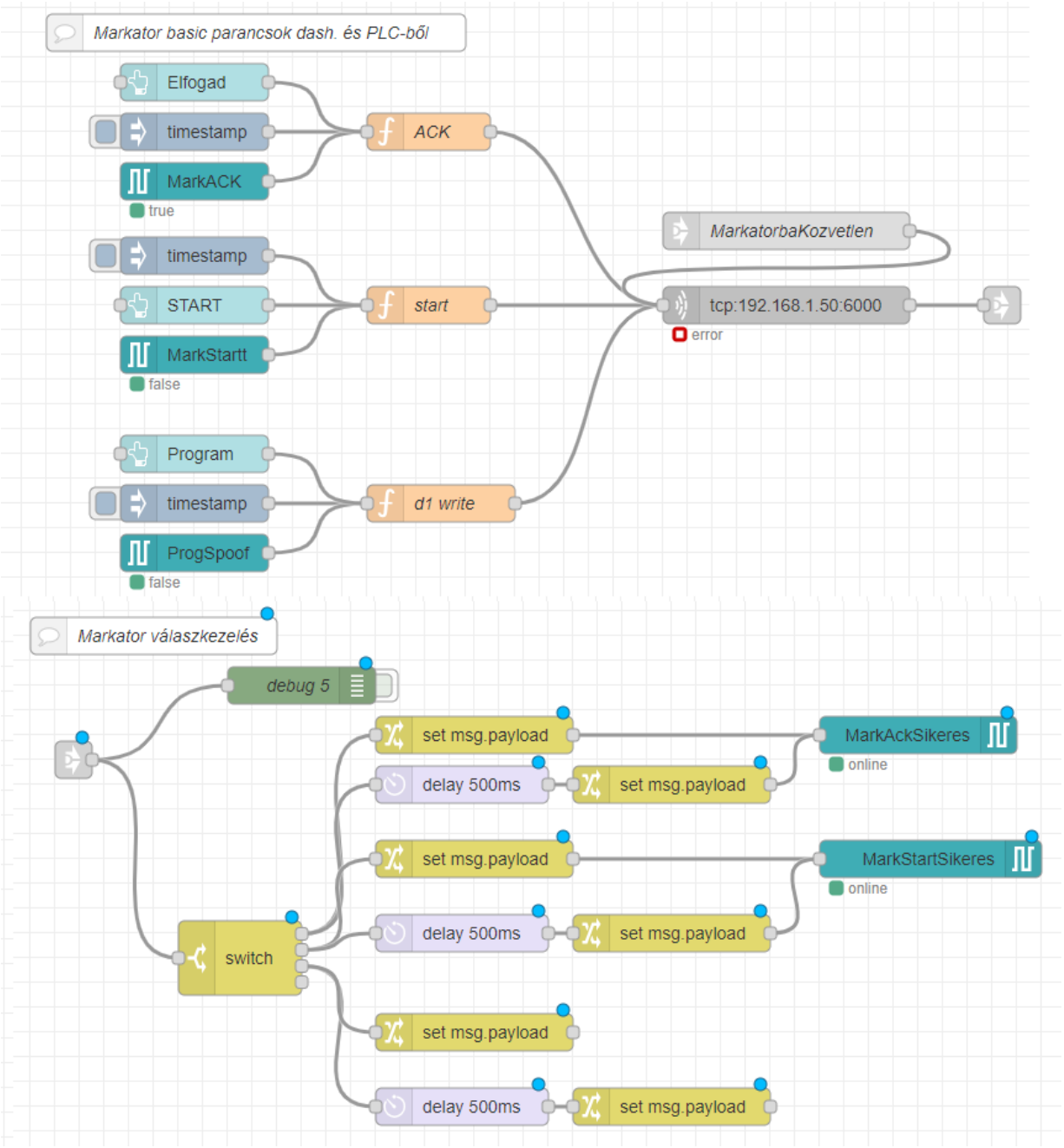
22. ábra: a, A Markator állapotüzenetét vizsgáló "switch" csomópont; b, Az üzenet tartalmát logikai igazra cserélő csomópont

A fentiek programozásával mind a (később bemutatott) HMI felületen keresztül a dolgozó, mind a PLC látja, hogy a Markator épp dolgozik vagy hiba merült-e fel benne, esetleg start jelre vár, utóbbit a Home vagy nem Home pozícióban teszi-e. A Markator programozása alapján két esetben fordul elő az, hogy start jelre vár. Új termék gravírozása esetén Home pozícióban várja a start jelet, majd, amikor a termékre a gravírozandó adatok második fele következik, szintén start jelre vár a Markator, azonban ezt nem Home pozícióban teszi, hanem az utolsó gravírozási pont helyén várakozva. Ebből a PLC és a dolgozó is tudja, milyen művelet következik.

5.2 A Markator irányítása a NodeRED-en keresztül

Az alapvető parancsok kiadása, mint a hibatörlés, a start vagy egy változó írása gyakorlatilag fő és állandó feladata a NodeRED programjának a robotcellán belül. A *22. ábrán* látható ennek teljes folyamata. Mindhárom parancs három különböző forrásból adható ki. Az első módszer kézzel, a NodeRED szerkesztőből a timestamp, azaz időbélyeg nevű csomópontok bal szélén található gombókkal működik. Ez gyakorlatilag diagnosztikai okokból hagytam csak a programban. A második módszer a kezet ábrázoló ikonnal ellátott gomb csomópontok, amelyek a HMI felületre elhelyezett gombok alapján küldenek valamilyen parancsot vagy jelet. A harmadik módszer pedig a digitális jelgöbére hasonlító ikonnal ellátott „s7 in” csomópont, amely a PLC adattáblájában figyeli az adott változók értékét, változás esetén pedig továbbítják azt a kimenetükön.

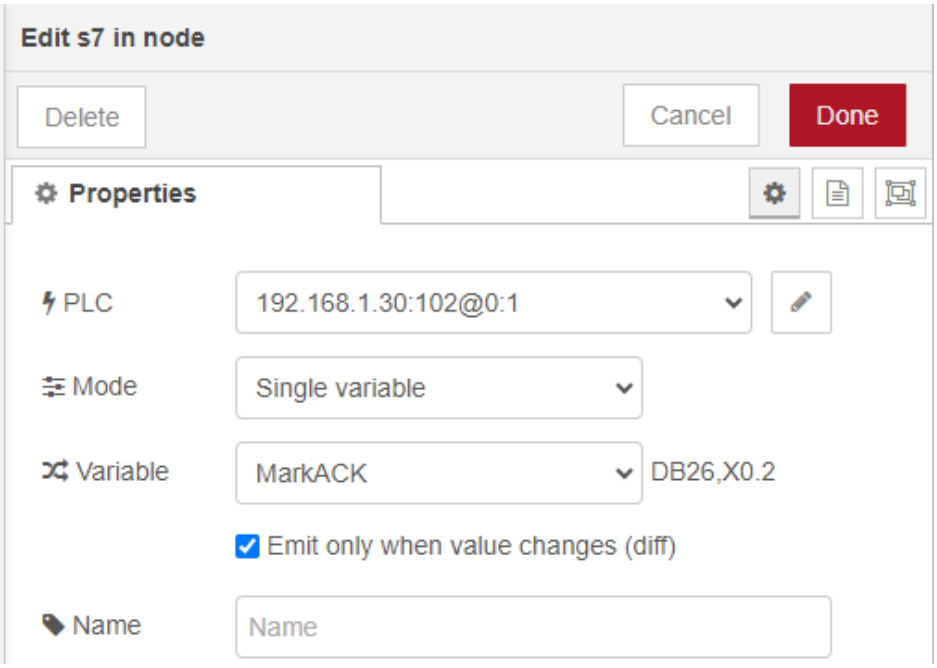
A szerkesztő oldalról működtethető időbélyeg csomópontok a létrehozásuk során nem igényeltek semmilyen módosítást.



23. ábra: A Markator irányításához tartozó fő NodeRED csomópontok és az azokat magába foglaló folyamat

Az „s7 in” csomópontok a *12/b és 13. ábrán* bemutatott s7 végpontot alkalmazzák. A *23. ábrán* látható „Variable” legördülő menüből kiválasztható, hogy mely konfigurált változót figyelje az adott csomópont, valamint az is, hogy csak változás esetén küldje tovább az adatot. A nyugtázás parancsot például a PLC DB26-os adattáblájának 0.2-es bitjén olvashatja ki a

program (*14. ábra*). A további három „s7 in” csomópont is a nevüknek megfelelő változót olvassák ki.



24. ábra: Az "s7 in" csomópont a nyugtázó parancs számára

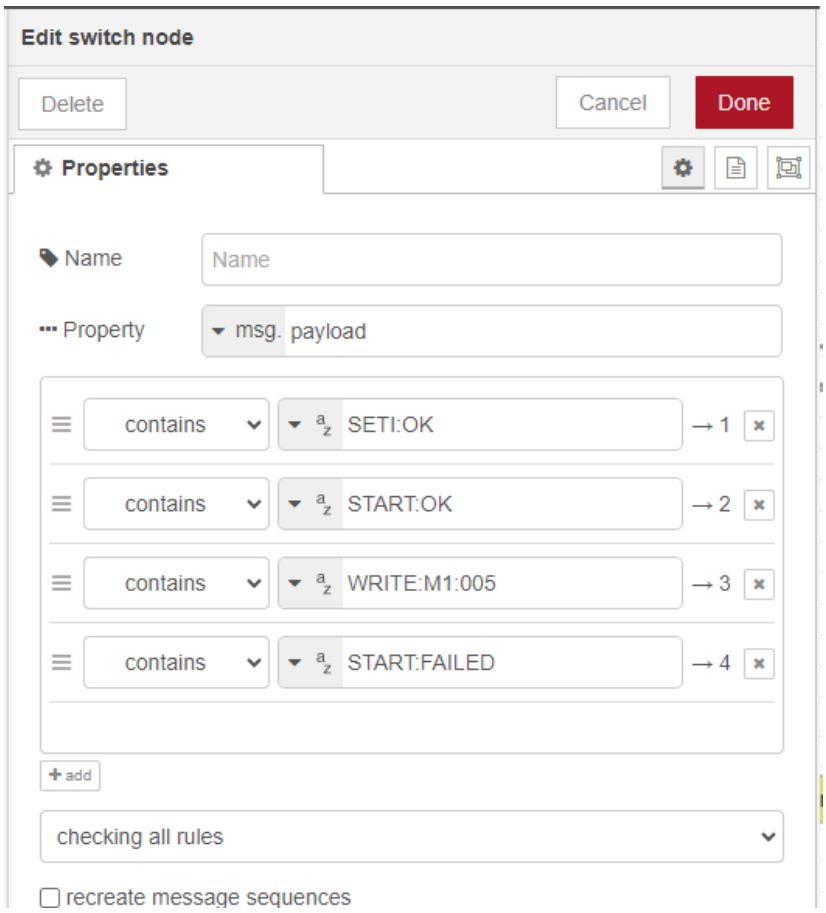
A bemenő parancsok után a folyamatok egy-egy függvény csomópontba kerülnek, amelyek belső tartalma megegyezik. Ha a bemeneteiken a tartalom nem boolean hamis, akkor a kimenetüket az adott folyamatág parancsára cseréljék ki. Valamint ezzel a kóddal abban az esetben, ha a bemenő adat hamis értékű, akkor nincs kimenő adat sem. Az alábbi kódrészlet az M1-es Markator változóba a 005 értéket írja be.

```
if (msg.payload != false) {
    msg.payload="WRITE:M1:005¥r";
    return msg;
}
```

A következő csomópont a már korábban ismertetett „tcp request”, amely a kétirányú kapcsolatot létesíti a Markator és a NodeRED között. Ennek a kimenetére egy link páros egyik fele került. A célja mindössze a szerkesztő felület áttekinthetőbbé tétele, párja a *22. ábra* alsó felén látható, amelyre a Markator válaszai kerülnek továbbításra. Látható ott a felső ágon egy „debug 5” nevű csomópont, amely mindössze diagnosztikai okokból került elhelyezésre, deaktivált állapotban.

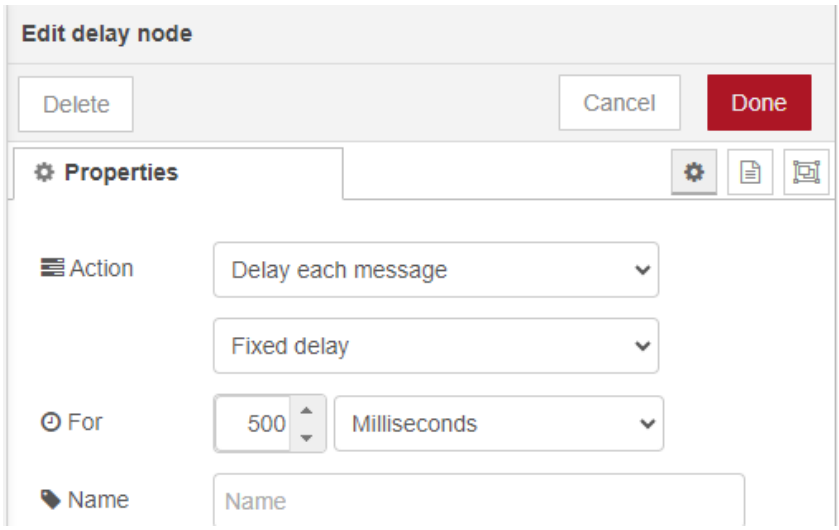
A folyamat fő ágán helyeztem el egy „switch” csomópontot, amelynél a bemenet tartalmát vizsgálom meg. Sajnos ennél a pontnál valamilyen szoftveres hiba okán nem tudtam

egyszerűbben, „change” csomóponttal megoldani a folyamatot. A *24. ábrán* látható, hogy a csomópont azt vizsgálja meg, hogy az üzenet tartalmazza-e az adott válaszhoz tartozó szöveget.



25. ábra: A Markator válaszait kiválogató "switch" csomópont

Az egyes kimeneteken található egy-egy „change” csomópont, amely az üzenet tartalmát boolean igazra cseréli; illetve alatta egy 500 ms késleltetést végző csomópont (*25. ábra*) után egy másik „change”, amely boolean hamisra változtatja a kimenő üzenetét, ezzel az „s7 out” csomópont által írt PLC változó 500 ms ideig igaz volt, majd ezután újra hamissá válik.



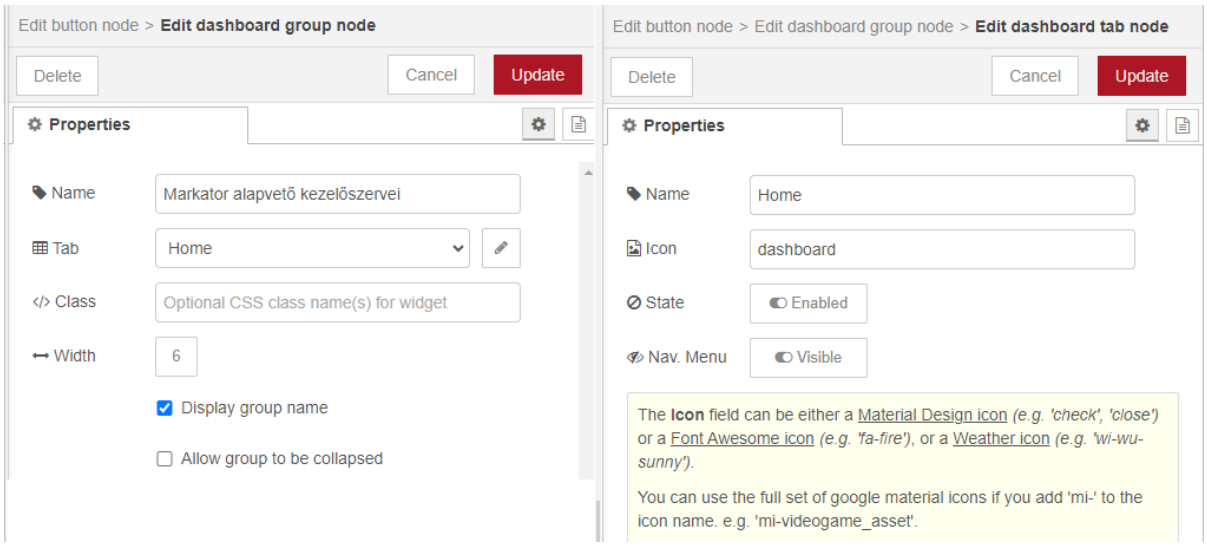
26. ábra: A késleltetés ("delay") csomópont

6. A HMI felület kialakítása

A NodeRED „Dashboard” névre hallgató kezelőfelületét a „Manage Palette” menüpontban lehet telepíteni a korábban ismertetett módon. A listában a „node-red-dashboard” modult kell telepíteni. Ezután a NodeRED a „/ui” URL bővítménnyel elérhető, például 127.0.0.1:1880/ui.

6.1 Alapvető kezelőszervek

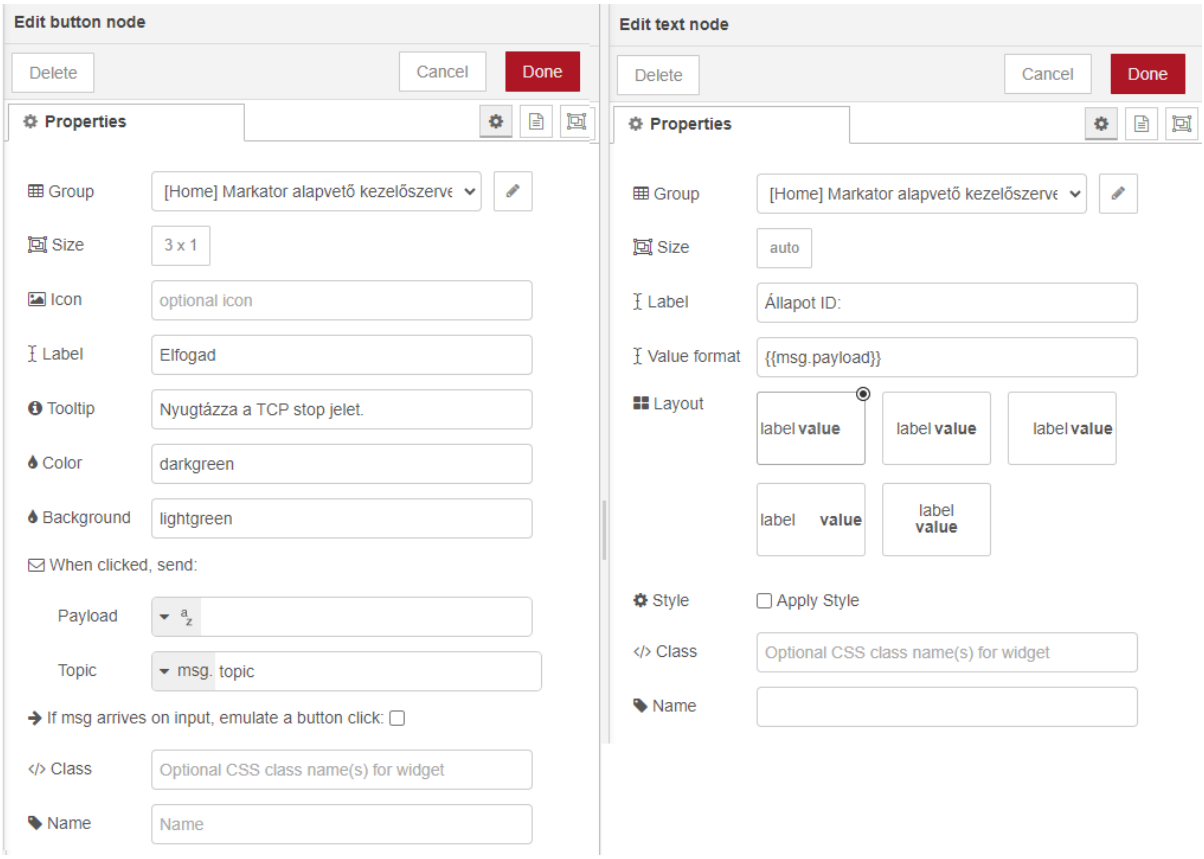
Az első Dashboard elem létrehozásakor létre kell hozni egy csoportot (*26/a ábra*), amelyen az elem elhelyezésre kerül. Az első csoport létrehozásakor egy oldalt is létre kell hozni (*26/b ábra*), amelyre pedig maga a csoport fog kerülni. Az oldal gyakorlatilag a NodeRED Dashboard felületén egy fület jelent, míg a csoport apróbb szegmenseket rajta, ahogy a *6. ábrán* is látható. Az oldalnak és a csoportnak is mindössze egy nevet adtam, a többi paraméter alapértelmezett értéken maradt.



27. ábra: a) Egy Dashboard csoport létrehozása; b) egy oldal létrehozása

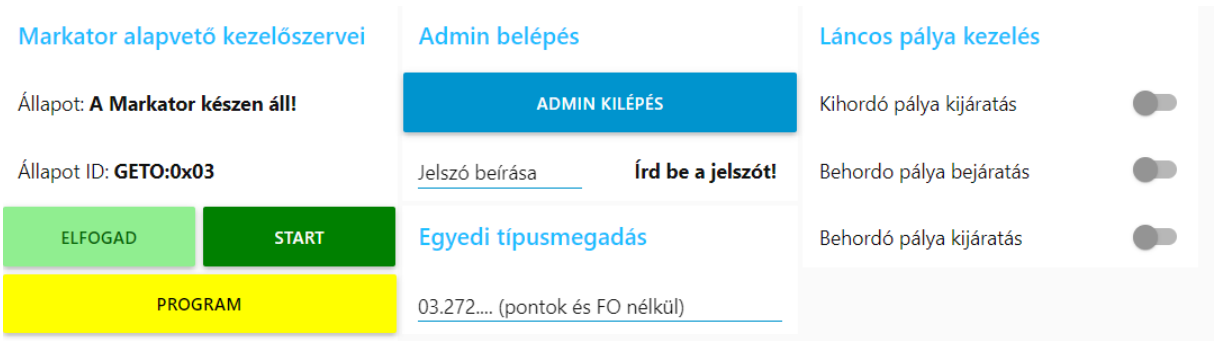
A korábban már említett gombok, amelyek a Dashboard felületen történő kattintás jeleit átviszik a „tcp request” csomópontba a *22. ábrán* üres üzenetet küldenek alapértelmezett formában. A „Markator alapvető kezelőszervei” nevű csoportba, a „Home” oldalon kaptak helyet. A gombokat elneveztem, megszíneztem, valamint a „tooltip” mezőben egy segítő üzenetet írtam be, amely az egérmutató gomb fölé húzására jelenik meg (*27/a ábra*).

A *18. ábrán* látható Dashboard csomópont két darab szövegmező elemet jelenít meg. A korábban már létrehozott csoporton kaptak helyet, valamint elneveztem őket a jelentésüknek megfelelően (*27/b ábra*).



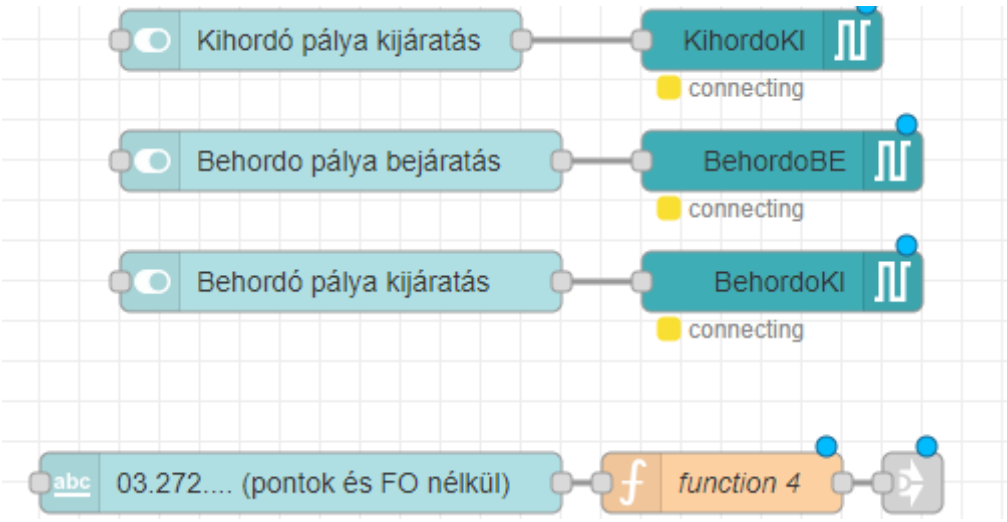
28. ábra: a) Egy Dashboard gomb létrehozása; b) Szövegmegjelenítés létrehozása

Az egyik szöveg kijelző csomópont elé elhelyeztem egy „change” csomópontot is, amely a bemenő kódok szöveges jelentésére cseréli ki az áthaladó üzenet tartalmát. Például a „GETO:0x00” állapotüzenet jelentése az, hogy a Markator épp dolgozik, így az üzenet tartalmát a „A Markator dolgozik!” szövegre cseréli ki. A Markator alapvető kezelőszerveit a *28. ábrán* lehet megtekinteni.



29. ábra: A Markator és a cella alapvető kezelőszervei

A cella további alapvető kezelőszervei jelenleg a behordó láncos pálya előre és hátra járatása, valamint a kihordó láncos pálya kijáratása (*29 ábra*). Ezekhez a műveletekhez egy „switch”, azaz kapcsoló csomópontot alkalmaztam, amely egy kétállású kapcsolónak felel meg. Bekapcsolt állapotában a kiküldött üzenet tartalma boolean igaz, míg kikapcsolt állapotban boolean hamis (*30/a ábra*). A kapcsolók számára létrehoztam egy új Dashboard csoportot a már meglévő Home oldalon „Láncos pálya vezérlés” névvel. Ezek kimenete egy-egy „s7 out” csomóponton keresztül a PLC NodeRED-es adattáblájába kerülnek beírásra.

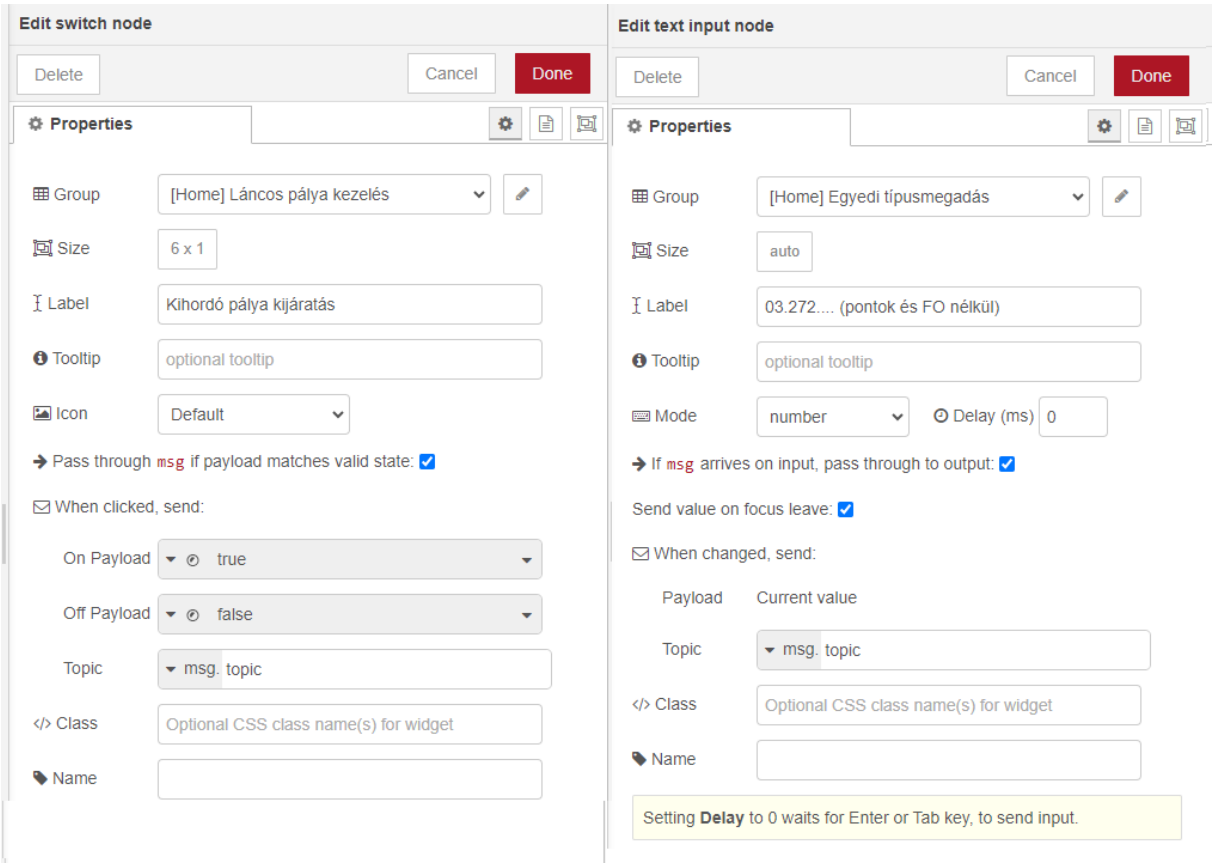


30. ábra: A cella további részét vezérlő elemek folyamata

Talán a legfontosabb eleme a HMI-nek az egyedi típusmegadás. Ezt egy szövegbeviteli csomóponton keresztül a termékek tételszámának végét kell csak megadni, majd enter benyomásával továbbításra kerül a tartalom (*30/b ábra*). A csomópont kimenete egy függvény csomópontba vezet, amelynek programja a bejövő adat ellenőrzése után egy Markator parancsot generál belőle a „WRITE:M1:…” parancs formátumában:

```
var beszam=msg.payload;
if (msg.payload != false) {
    msg.payload = "WRITE:M1:"+beszam+"¥r";
    return msg;
}
```

A függvény után egy link található, amely a bejövő üzeneteket a *22. ábrán* látható „MarkatorbaKozvetlen” linkpáron keresztül közvetlenül beküldi a Markatorba parancs formájában, az erre szolgáló „tcp request” csomóponton keresztül. Ennek kimenete pedig a Markator válaszkezelés folyamatába kerül be. A kezelőszervek a *28. ábrán* tekinthetők meg.



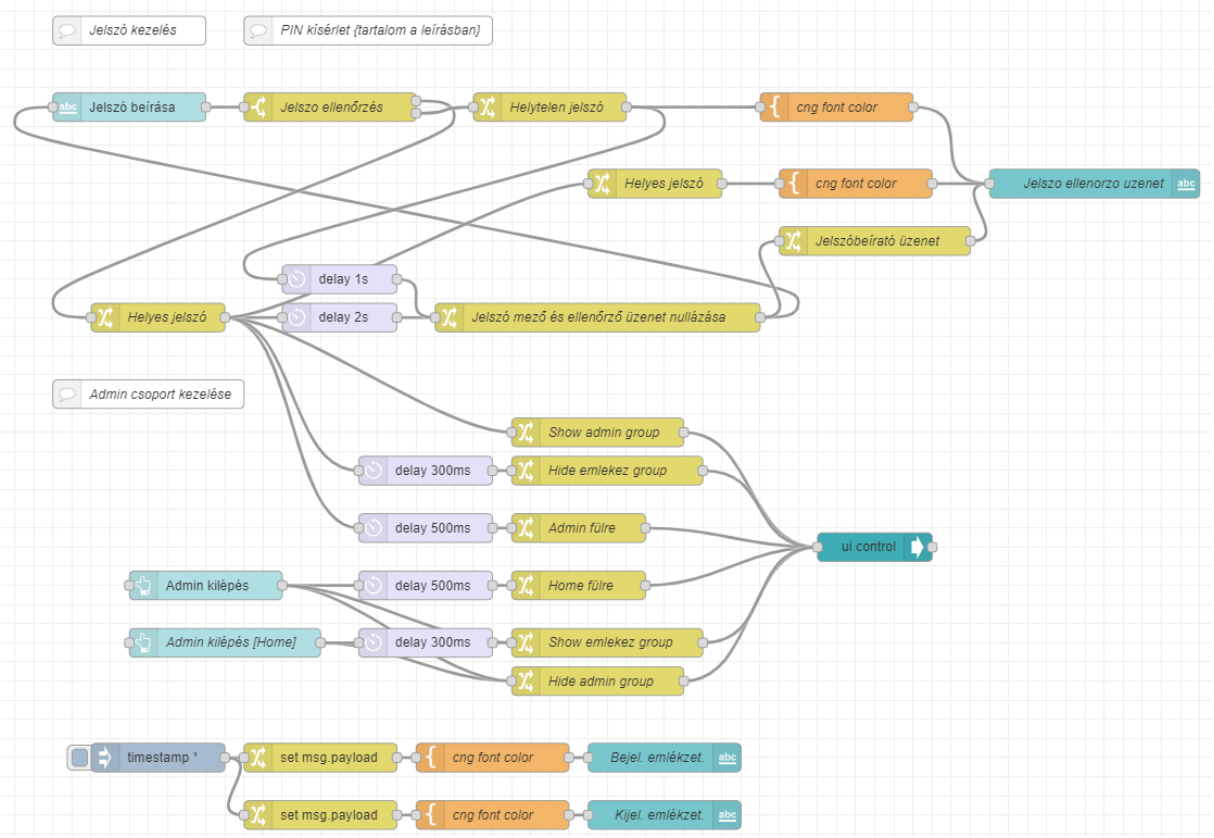
31. ábra: a) A kétállású kapcsoló beállításai; b) A szövegbeviteli mező beállításai

6.2 Rendszergazda felület létrehozása, felépítése

A HMI fejlesztése során felmerült az ötlet, hogy hozzunk létre egy rendszergazdai felületet is, amelyen az aktuátorok, kimenetek változtatásainak különböző előfeltételeit megkerülve közvetlenül lehet beavatkozni például egy elakadt folyamat esetén. Ilyen beavatkozási lehetőség lenne például különböző pneumatikus munkahengerek mozgatása is.

A rendszergazdai felületen található kezelőszerveket úgy kell elhelyezni, hogy ahhoz illetéktelen személyek ne férhessenek hozzá. Alapvetően a NodeRED nem képes ilyen műveletekre, azonban a „ui control” csomóponttal lehetséges befolyásolni a Dashboard kinézetét. A létrehozott NodeRED folyamat meglehetősen bonyolult (*31. ábra*), azonban működése egész egyszerű. Alapesetben a rendszergazdai kezelő csoport el van rejtve, azonban ha a Home oldalon a megfelelő szövegmezőbe a megfelelő jelszót beírja a rendszergazda, akkor annak elfogadása után 0,5 másodperccel átugrik a Dashboard az Admin oldalra, ahol a jelszó beírását követően láthatóvá válnak a kezelőszervek, valamint egy kijelentkezés gomb és egy emlékeztető, hogy ne felejtsen el kijelentkezni a rendszergazda, ha elhagyja a HMI-t. Kijelentkezés után automatikusan a Home oldalra ugrik a HMI, azonban ha véletlen újra az

Admin oldalra kattintana a dolgozó, egy felhívás jelenik meg számára, hogy jelentkezzen be a rendszergazdai jelszóval. Ennek hiányában a kezelőszervekhez nem fér hozzá.

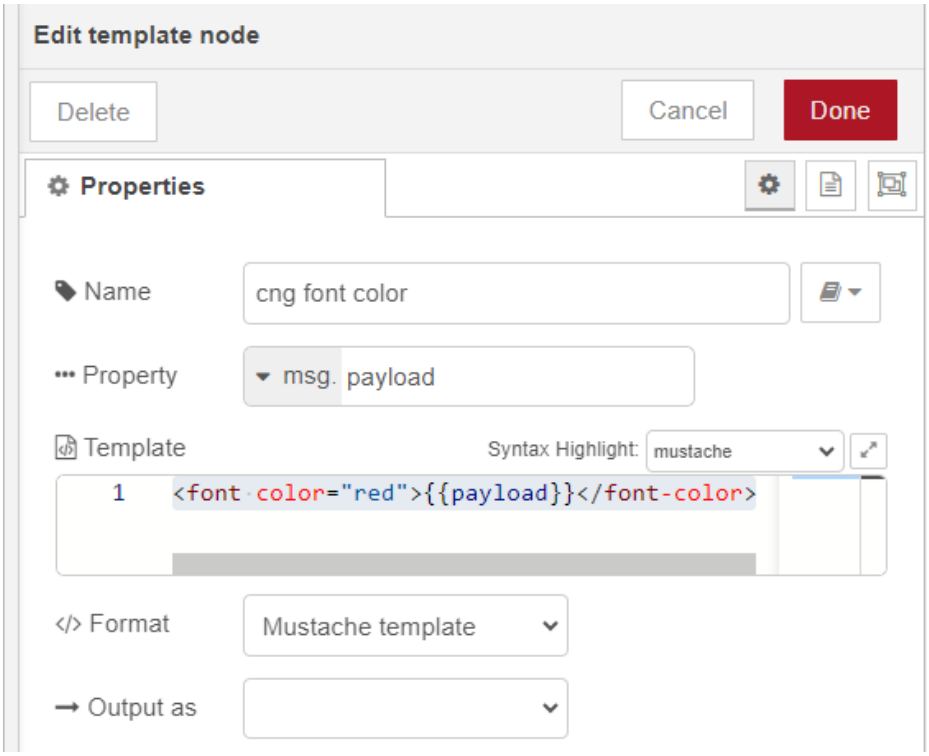


32. ábra: A rendszergazdai kezelőszervek elrejtéséért felelős folyamat

A *31. ábra* alján található egy rövid folyamat, amely feladata mindössze a rendszergazda módba történő ki és bejelentkezésre történő piros színű felhívó üzenet megjelenítése. A folyamat egy időbélyeg csomópontból indul, amely egyetlen dolga, hogy a NodeRED indulásakor egy üzenetet kiküld a kimenetén, amelyre párhuzamosan két darab „change” csomópont került. Ezek az üzenet tartalmát kicserélik az adott felhívás szövegére. Mindkét csomópont után található egy-egy „template” csomópont (*32. ábra*), amely az üzenetet manipulálja. Ebben az esetben a szövegnek ad piros színt:

```
<font color="red">{{payload}}</font-color>
```

A párhuzamos folyamat vége is hasonlít egymáshoz. Egy szöveg csomópontban végződik, amely közül a bejelentkezésre felhívó üzenet az Admin oldal „Emlékeztető” nevű csoportjába írja ki a szöveget, míg a kijelentkezésre emlékeztető szöveget a kezelőszervek mellett jeleníti meg.



33. ábra: A "template" csomópont

A rendszergazdai jelszó kezelése egy „text input” csomópontból indul, amely jelszó formátumban, takart módon fogadja a karaktereket a Home oldal „Admin belépés” nevű csoportjában. Hasonlóan az egyedi típusmegadáshoz, itt is az enter billentyűvel lehet elfogadni a beütött jelszót. Az üzenet ezután egy „switch” csomópontba kerül, ahol, ha a jelszó helyességétől függően egy-egy „change” csomópontba kerül az üzenet, ahol tartalmát a megfelelő üzenetre cseréli a csomópont. Az üzenetet a *32. ábrán* is látható típusú csomópont megszínezi pirossal vagy zölddel, majd szöveg megjelenítésére alkalmas csomópont kiírja ezt az üzenetet a szövegmező mellé.

Helyes és helytelen jelszó esetén is van egy párhuzamos ága az üzenetnek, amelyen egy 1-2 másodperces késleltetés után a jelszó mező tartalmát üresre változtatja egy „change” csomópont, amelynek kimenetét a szövegbeviteli csomópont bemenetére kötöttem. Az üzenet tartalmának törlése után egy másik párhuzamos ág is elindul, amely a színes jelszó ellenőrző üzenet helyére kiírja a felhívást a jelszó beírására.

Helyes jelszó beírás esetén megtörténik a „ui control” csomópont alkalmazása, amelybe közvetlenül, illetve késleltetéssel is érkezik üzenet a helyes jelszó elfogadási helyéről. Minden alkalommal egy „change” csomópont előzi meg az üzenet végállomásba érkezését. Ezen csomópontokban az üzenet tartalmát JSON kódra cseréltem ki, amellyel befolyásolható a

Dashboard megjelenése. A rendszergazdai oldal alap csoportjának megjelenítése, amelyen a kezelőszervek is láthatók:

```
{"group":{"show":["Admin_Alap"]}}
```

Ezzel párhuzamosan, kis késleltetéssel rejtem el a belépésre emlékeztető üzenetet tartalmazó csoportot:

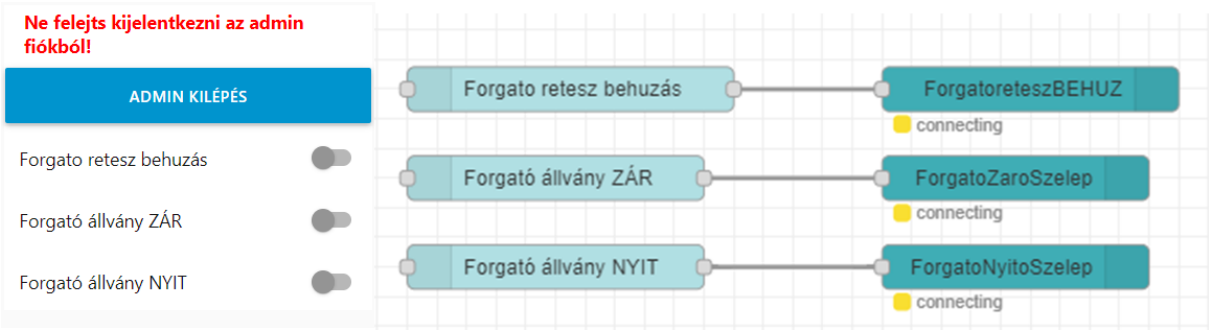
```
{"group":{"hide":["Admin_Emlekez"]}}
```

További késleltetéssel a felületet átugrasztom a rendszergazdai elemeket tartalmazó Dashboard oldalra:

```
{"tab" : "Admin"}
```

A rendszergazdai kezelőszervek közt, valamint a Home oldalon a bejelentkezés csoportban, is található egy kijelentkezés gomb a bejelentkezés csoportban, amelyek megnyomására a kijelentkezési folyamat indul el, azaz a „ui control” elrejti a kényes rendszergazdai elemeket, megjeleníti a bejelentkezésre felhívó üzenetet, valamint visszaugrasztja a Dashboard-ot a Home oldalra, kódja a korábban ismertetettek inverze.

A rendszergazdai kezelőszervek a robotcella jelenlegi készültségi állapotában mindössze néhány kétállású kapcsolónak megfelelő gombot, valamint az azokhoz kapcsolódó „s7 out” csomópontokat tartalmazza (*33. ábra*).



34. ábra: A rendszergazda kezelőszervei: a) A Dashboard-on; b) A szerkesztőben

7. Gazdasági számítás

A teljes rendszert tekintve a beruházás megvalósítására 180.000 Euró lett előzetesen megállapítva, illetve elfogadva. Ebbe beletartozik egyaránt a folyamathoz használt robot, a PLC, a mosó berendezés, a hajtások, a kamerák, a gravírozó berendezés illetve minden egyéb eszköz. Így a teljes rendszert figyelembe véve, dolgozónként évi 25.000 Euró munkabérrel számolva egy évben kb 75.000 Eurót takaríthat meg a cég, mivel 3 műszakban folyik a termelés az üzemben. Ez azt jelenti, hogy 3 év alatt szinte biztosan jövedelmezővé válna ez a befektetés, arról nem beszélve, hogy a munkafolyamatot mennyire felgyorsítanák az elvégzett fejlesztések, illetve a hibakeresés is sokkal könnyebbé válna, ezzel tovább növelve a termelékenységet.

A dolgozatom tárgyát képző ipari kommunikáció megvalósítása, illetve a HMI létrehozása kb. 3 hónapig tartott. „Külsős", a feladat elvégzésére megbízott céget nem kellett felfogadni, a projekt saját munkaerő alkalmazásával lett kivitelezve. Átlagosan heti 30 órát a projektre fordítva ez esetben kb. 360 óra alatt valósult meg a kivitelezés. Ez átlagban 4200 Ft-os bruttó órabér esetén 1.512.000 Ft-jába került a cégnek emberenként. Három emberes, csak ezzel a projekttel foglalkozó team esetén sem éri el az 5.000.000Ft-ot a ráfordítandó munkadíj. Jelen Huf – Eur árfolyammal számítva ez 13.000 Eurót jelent, ami a teljes, tervezett 180.000 Eurós befektetésnek csupán a ~7% -a, így külön a dolgozatom témáját képző feladatok elvégzésére vonatkozva megtérülési időről véleményem szerint nincs értelme beszélni, bőven belefér ezzel együtt is a teljes projektre kalkulált 3 éves megtérülési időbe. Emellett gazdasági szempontból pozitívumként fogható fel az is, hogy a jelenlegi Markator gravírozó helyett szükségtelen volt egy még újabb, illetve drágább megoldást használni, hiszen a kommunikációs probléma áthidalható lett egy ingyenes, könnyen futtatható szoftver használatával.

8. Továbbfejlesztési lehetőségek

Továbbfejlesztési lehetőségekről, még kiforratlan fejlesztési projekt lévén nehéz beszélni, hiszen a projekthez szükséges eszközök közül van ami még beszerzés alatt áll, többek között a mosó berendezés, mely az egész automatizált munkafolyamat egyik szerves részét képezi.

Safety szempontból sok mindent ki kell még alakítani, többek között a munkateret körülvevő kerítést a vonatkozó szabványoknak megfelelően letelepíteni, a rajta lévő elektromos működtetésű ajtókkal, melyen a reteszt nyitott állapotában (ha az ajtó egyáltalán nincs, vagy elégtelenül van bezárva) a munkafolyamat azonnal megszakad. E mellé számos, a munkavégzést biztonságossá tévő tényezőnek kell még megfelelni, többek között a gépkezelőnek rendelkeznie kell a munkafolyamatot azonnal megszakító lehetőséggel, vagyis a berendezést vész-leállító funkcióval kell ellátni. A Safety témakörnél maradva lehetőség lenne egy olyan komplex, biztonságos rendszer létrehozására, ahol is detektálásra kerülne a munkadarab. A gép ekkor tudná, hogy a munkatérben ember nem tartózkodik, így folyamatos, és egyben biztonságos lehet a gyártás. Ezt fényfüggönyökkel, optikai szenzorokkal, illetve a padlóba szerelt alaphelyzetben nyitott (NO) típusú nyomásérzékelő lapokkal kivitelezni lehet. Ezek segítségével megoldható, hogy amennyiben a padló egy bizonyos területére eső terhelés a megengedett értéknél nagyobb lenne, az áramkör azonnal megszakad.

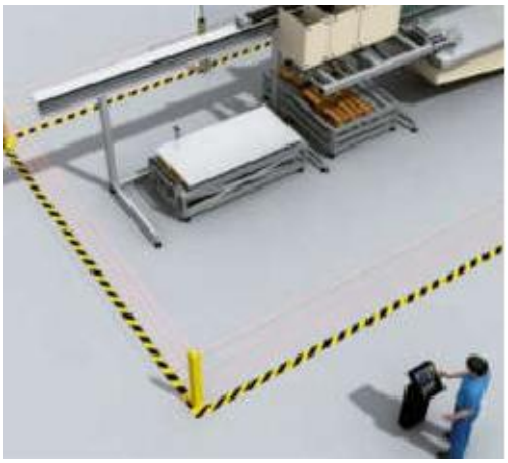
Ezen biztonságtechnikai eszközök használatával lecsökkenthető lenne az a kiesett idő, ami jelenleg a kerítés ajtajának nyitásától tart, az ajtó zárását követő, HMI-n történő nyugtázással folytatódik, majd a kézi folyamat-újraindítással ér véget. Ennek a kivitelezéséhez azonban szükséges lesz azt megoldani, hogy a rendszer önmagától tudja megkülönböztetni a munkadarabot, illetve az emberi jelenlétet, és szükség szerint leállni, illetve amennyiben biztonságos, a munkafolyamatot folytatni.

Fontos azonban azt kiemelni, hogy az ilyen, speciális ismereteket igénylő fejlesztésekkel egy külön dolgozat keretein belül lenne esély érdemben foglalkozni, e komplex kérdéskör kibontása szakdolgozatomnak nem célja.

35.ábra: Fényfüggöny [35]

36.ábra: Vészstop gomb [36]

37.ábra: Veszélyzóna elméleti szemléltetése [37]



38.ábra: Veszélyzóna jelölése a valóságban [37]

9. Összefoglalás

Az ipari folyamatok, legyenek gyártási akár folyamatok, anyagtovábbítással vagy előkészítéssel kapcsolatosak, sok esetben repetitív, vagy akár emberi erővel nehezen kivitelezhető folyamatok. Ilyen folyamat automatizálásának részét végeztem el én is a szakdolgozatom írása közben.

A dolgozatomban egy elavult munkaállomást egy 6 tengelyes ipari robotos kiszolgálási megoldásra cseréltette le a vállalat. A robotot gépi látás vezérli (bin-picking megoldást alkalmazva), melynek következtében a terméket szükség esetén egy forgató állomásba képes helyezi, onnan pedig egy pontgravírozó állomásba. A gravírozás végeztével a robot áthelyezi a terméket a láncos conveyor pályás mosó berendezésbe, majd a folyamat kezdődik elölről.

Mivel a megrendelt komponensek között nem mindegyik rendelkezett a cellában alapvetően alkalmazott ProfiNET kommunikációs protokollal, úgy ezen komponensnek egyedi megoldást kellett létrehoznom a cellába való integrációja érdekében. A szóban forgó komponens maga a pontgravírozó eszköz volt, amely rendelkezett ugyan RJ45-ös porttal, azonban azon csak alapvető, Ethernet jellegű kommunikációra képes.

Az integráció megoldásához a Node-RED programot választottam, amellyel lehetőségem nyílt a Markator pontgravírozó teljes integrációját elvégezni a ProfiNET-es Siemens PLC-vel. A Node-RED-et, mint protokoll illesztő megoldást alkalmaztam, amelyet úgy programoztam fel, hogy a PLC-ben külön erre a célra létrehozott adatblokkot írja és olvassa, annak adatainak megfelelően cselekedjen. A Markator felé leprogramoztam annak indítását, engedélyező jelét és a program típusának választását is a PLC felől, majd a Markator állapotát és válaszait is továbbítottam a Node-RED segítségével a PLC adatblokkjába, ezzel elvégezve a Markator integrációját.

A Node-RED egyik további képessége, hogy lehetséges rajta böngésző alapú felületet, akár HMI-t is létrehozni. Mivel a robotcellában amúgy is szükség volt több különböző kezelőszerv, valamint az egyedi típusmegadás működtetéséhez egy HMI-re, így azt a feladatot is a Node-RED installációnak adtam. A HMI felületre elhelyeztem több, a cella működését befolyásoló gombot, kapcsolót, illetve több különböző állapotmegjelenítő üzenetet és szögmezőt is. Ide került az egyedi típusmegadáshoz tartozó beviteli mező, valamint egy emelt

jogosultságokkal elérhető „adminisztrátor” fül belépési lehetősége is. A speciális fülre jelszó beírásával lehet belépni, amely ezután jelenik csak meg a kezelő számára. Elláttam a Node-RED böngésző alapú szerkesztőjét is egy generált jelszavas védelemmel, így a dolgozók nem tudnak belépni majd ide egyáltalán, míg a helyi vezetőknek olvasási jogosultságot adtam.

A robotcella jelenleg is fejlesztés alatt áll, több komponens hiányzik még a végleges rendszerből. A cellában futó több rendszer programozása is befejezetlen állapotban van még, azonban a teszteléseken egy-egy típusú terméket már képes a robot felemelni, megforgatni, gravíroztatni, majd lehelyezni a földre a mosó berendezés helyére.

10. Summary

Industrial processes, whether they are manufacturing, material handling or preparation, are often repetitive or even difficult to perform by human effort. I did part of such process automation while writing my thesis.

In my thesis, an outdated workstation was replaced by a 6-axis industrial robotic dispatch solution. The robot is controlled by machine vision (using a bin-picking solution), which allows it to place the product into a rotation station when needed, and from there to a point engraving station. Once the engraving is finished, the robot transfers the product to the chain conveyor track washing system and the process starts again.

Since not all of the components ordered had the ProfiNET communication protocol that is standard in the cell, I had to create a custom solution for this component to integrate into the cell. The component in question was the dot-engraving device itself, which had an RJ45 port, but could only provide basic Ethernet-like communication.

To solve the integration, I chose Node-RED, which allowed me to fully integrate the Markator point engraver with the Siemens PLC in ProfiNET. I used Node-RED as a protocol interface solution, which I programmed to read and write a dedicated data block in the PLC and act according to its data. I also programmed its startup, enable signal, and program type selection to the Markator from the PLC, and then I transmitted the Markator's state and responses to the PLC's data block using the Node-RED, thus completing the Markator integration.

An additional capability of Node-RED is that it is possible to create a browser-based interface, even an HMI, on it. Since the robot cell needed an HMI anyway to operate several different controls and the custom type assignment, I gave that task to the Node-RED installation. On the HMI interface, I placed several buttons and switches that affect the operation of the cell, as well as several different status messages and angle fields. I also added an input field for custom type assignment, as well as an "administrator" tab with elevated privileges. The special tab can be accessed by entering a password, which is then only displayed to the operator. I have also provided the browser-based editor of Node-RED with a

generated password protection, so that employees will not be able to access it at all, while I have given local managers read-only access.

The robot cell is currently under development, with several components still missing from the final system. Programming of several systems running in the cell is still incomplete, but in testing, the robot is able to lift, rotate, engrave and then place one type of product on the floor in place of the washing equipment.

NYILATKOZAT

a szakdolgozat nyilvános hozzáféréséről és eredetiségéről

A hallgató neve: Duchai László Pál

A Hallgató Neptun kódja: G9MMOT

A dolgozat címe: Ipari kommunikáció és HMI felület megvalósítása Node-RED segítségével

A megjelenés éve: 2023

A konzulens intézetének neve: Műszaki Intézet

A konzulens tanszékének a neve: Mechantronika 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.

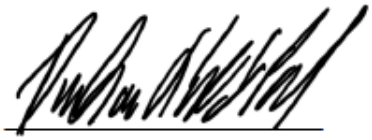
Tudomásul veszem, hogy az általam készített dolgozatra, mint szellemi alkotás felhasználására, hasznosítására a Magyar Agrár- és Élettudományi Egyetem mindenkori szellemitulajdon-kezelési szabályzatában megfogalmazottak érvényesek.

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 11. nap



Hallgató aláírása

NYILATKOZAT

Duchai László Pál (hallgató, Neptun azonosítója: G9MMOT) 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.

9. Irodalomjegyzék

1. Steven O'Hara (2020): *A Brief History of the Programmable Logic Controller (PLC)*. Process Solutions Inc., 2023.04.11. https://processsolutions.com/a-brief-history-of-programmable-logic-controllers-plcs/
2. Stephen Philips Tubbs (2016): *Programmable Logic Controller (PLC) Tutorial, Siemens Simatic S7-1200*. ISBN: 0981975364
3. Juhász Róbert (2013): *Ismerkedés a PLC-vel*. UMSZKI
4. Ben Wayand (2020): *What is a PLC?*. MRO Electric, 2023.04.12. https://www.mroelectric.com/blog/what-is-a-plc/
5. E. A. Parr (1998): *Computers and industrial control*. Industrial Control Handbook. ISBN: 0831130857
6. M. A. Laughton, D. F. Warne (2002): *Electrical Engineer's Reference Book*. ISBN: 9780750646376
7. Gary Dunning (2005): *Introduction to Programmable Logic Controllers.* ISBN: 1401884261
8. John W. Webb, Ronald A. Reis (2002): *Programmable Logic Controllers: Principles nad Applications*. ISBN: 013041672X
9. Gary Anderson (2020): *PLC Programming Using RSLogix 500: Basic Concepts of Ladder Logic Programming*. ASIN: B087ZPVXMG
10. Siemens: *Cost-effective Home Automation*. Siemens GmbH. 2023.04.22. https://new.siemens.com/global/en/products/automation/systems/industrial/plc/logo/home-automation.html
11. Ivan Zeba (2022): *KORIŠTENJE SIEMENS LOGO! KONTROLERA ZA UPRAVLJANJE INOVATIVNIM SMART VENTILACIJSKIM SUSTAVOM S POVRATOM TOPLINE*. Karlovac University of Applied Sciences. URN:NBN: hr:128.057751
12. Siemens: *Simatic S7-1500*. Siemens GmbH. 2023.04.22. https://www.siemens.com/global/en/products/automation/systems/industrial/plc/simatic-s7-1500.html
13. Siemens: *Open User Communication: Other*. Totally Integrated Automation Information System: Programming a PLC – Instructions – Instructions (S7-1200, S7-1500) – Communication (S7-1200, S7-1500).

14. IEC 61158-1:2019: *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*. International Electrotechnical Commission (IEC).
15. PROFIBUS: *More than 20 million PROFINET devices on the market*. PROFIBUS PNO. 2023.04.23. https://www.profibus.com/newsroom/press-news/more-than-20-million-profinet-devices-on-the-market/
16. PROFIBUS (2014): *PROFINET System Description: Technology and Application*. PROFIBUS PNO. Order Number: 4.132
17. IEC 61784-2:2010*: Industrial communication networks – Profiles – Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3.* International Electrotechnical Commission (IEC).
18. IEC 61131-3:2013: *Programmable controllers - Part 3: Programming languages*. International Electrotechnical Commission (IEC).
19. Sigma NSW (2021): *Siemens S7 PLC EtherNet/IPTM Function Blocks.* 2023.04.27. https://www.sigmansw.com.au/s7ethernetip
20. https://www.siemens.com/global/en/products/automation/simatic-hmi/panels.html
21. https://www.scantime.co.uk/tutorials/an-introduction-to-hmi-programming-with-siemens-wincc-tia-portal/
22. Heath, Nick (March 13, 2014). "How IBM's Node-RED is hacking together the Internet of things". *techrepublic.com*. CBS Interactive. Retrieved January 16, 2017.
23. Lewis, Karen (October 17, 2016). "Node-RED visual programming for the Internet of Things (IoT) is now a JS Foundation Project". *IBM Internet of Things blog*. IBM. Retrieved February 7, 2017.
24. Gabe Stein (August 2013). "How an Arcane Coding Method From 1970s Banking Software Could Save the Sanity of Web Developers Everywhere". Retrieved 24 January 2016.
25. https://Node-RED.org/ elérés: 2023.05.01.
26. "Beckhoff Information System - English". *infosys.beckhoff.com*. Retrieved 2023-07-14.
27. "ctrlX AUTOMATION - Node-RED". *developer.community.boschrexroth.com*. 2021-04-19. Retrieved 2023-07-14.
28. "Overview - developer.siemens.com". *developer.siemens.com*. Retrieved 2023-07-14.
29. https://nodered.org/docs/getting-started/
30. https://flows.nodered.org/node/node-red-dashboard
31. https://nodered.org/docs/user-guide/runtime/securing-node-red
32. https://flows.nodered.org/node/node-red-contrib-s7

33. Markator: *Option D: Ethernet*. Markator Operatin Manual. Order Number: 5034 00 063.
34. Juhász: Ismerkedés PLC-vel (http://moodle.jrobi.hu/Feladatok/PLC_EA-5.pdf)
35. https://hu.gzcyndar.net/safety-light-curtain/
36. https://plazmacnc-langvago.hu/termek/veszstop-gomb-dobozzal-felirattal-2/
37. Tóth J. – Biztonsági érzékelők (MATE)