ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

KANDÓ KÁLMÁN VILLAMOSMÉRNÖKI KAR
ELEKTRONIKAI ÉS KOMMUNIKÁCIÓS RENDSZEREK INTÉZET
MŰSZERTECHNIKAI ÉS AUTOMATIZÁLÁSI TANSZÉK

SZAKDOLGOZAT

OE-KVK
2022.

Himics Dávid
T008113/FI12904/K

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Óbudai Egyetem
Kandó Kálmán Villamosmérnöki Kar
Műszertechnikai és Automatizálási Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve: Himics Dávid
Szakdolgozat száma: SZD22030109183916
Törzskönyvi száma: T008113/FI12904/K **Neptun kódja:** [redacted]

Szak: villamosmérnöki (magyar nyelven)
Specializáció: Műszer-automatika specializáció

A dolgozat címe: Összetett rendszerek kialakítása és működtetése Siemens eszközökkel
A dolgozat címe angolul: Setting up and managing a system with Siemens devices
A feladat részletezése: Tervezzen összetett rendszert Siemens eszközökkel, majd dokumentálja a rendszer működtetésének lépéseit!

Intézményi konzulens neve: Sándor Tamás

A kiadott téma elévülési határideje: 2025. március 1.
Beadási határidő: 2022. 05. 15.

A szakdolgozat: Nem titkos.

Kiadva: Budapest, 2022. 03. 21.

.....
Intézetigazgató

A dolgozatot beadásra alkalmasnak találom.

.....
belső konzulens

.....
külső konzulens

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

KANDÓ KÁLMÁN VILLAMOSMÉRNÖKI KAR
ELEKTRONIKAI ÉS KOMMUNIKÁCIÓS RENDSZEREK INTÉZETI
MŰSZERTECHNIKAI ÉS AUTOMATIZÁLÁSI TANSZÉK

HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a szakdolgozat saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja, a titkosításra vonatkozó esetleges megkötések mellett.

Budapest, 2022. május 14.

.....

hallgató aláírása

Tartalomjegyzék

1. Bevezetés

Napjainkban temérdek sok elektronikus eszközt használunk, hogy könnyebbé tegye életünket. Ezek között van számos méretű, fogyasztású, fajtájú, színű, formájú és funkcionalitású készülék. Gondoljunk bele, hogy a zsebünkben lévő telefontól kezdve, az épületben lévő berendezéseken át, a közlekedéssel bezárólag hány ilyen eszközzel találkozunk pusztán a mai nap során. Belegondolni is sok hány ilyen szerkezet segíti mindennapjainkat, és mindezt úgy teszik, hogy teljesen természetessé vált a számunkra.

És hogy ez minek köszönhető? Az elmúlt 40-50 évben elképesztően felgyorsult és globálissá vált a világ. A ma élő embernek nem okoz gondot eljutni a világ egyik feléből a másikba akár pár óra alatt. Könnyedén meg tud vásárolni szinte bármit az internet segítségével. A világpiac és a fogyasztói társadalom fogalma megjelent és virágkorát éli, mivel a megnövekedett igény és a kereslet, ily módon fejlődésre készteti a termelőpiacot. Ilyen és ehhez hasonló fejlődéseknek köszönhető, hogy a gyárakban manapság több a gyártást végző robotok száma, mint a kétkezi munkások száma. Ez annak köszönhető, hogy a robot relatív nagy precizitással végzi el feladatát. Szinte a nap huszonnégy órájában képes dolgozni és csupán időszakos karbantartásra van szüksége, melyet arra képzett szakemberek végeznek.

Ha végig gondoljuk, könnyen beláthatjuk, hogy a technológia fejlődés egyértelműen jelen van. Ennek már most is érezzük hatásait, azonban sokkal fontosabb, hogy olyan szemléletet kövessünk, ahol tudatosan használjuk és alkalmazzuk ezeket az eszközöket. Úgy gondolom a huszonegyedik századi mérnökök egyik fő feladata, hogy az általunk használt eszközöket megfelelően optimalizálja. Természetesen nagyon sokféle dologra lehet optimalizálni, éppen ezért nagyon fontos, hogy mindezt az észszerűség határain belül kell megtenni. A mérnököknek gondolnia kell a jövőre és ezzel párhuzamosan a fenntarthatóságra, hisz „*A Földet nem apáinktól örököltük, hanem unokáinktól kaptuk kölcsön.*” [1]

2. A cég bemutatása

A szakdolgozatomat az evosoft Hungary Kft.-nél írtam meg. Én, személy szerint 2020 novemberében kezdtem gyakornokként, ahol kicsivel több mint egy év tapasztalat szerzés után, most már főállású munkavállaló vagyok. Munkám során főleg tesztelői tevékenységeket folytatok, ahol a TIA Portal fejlesztőikörnyezetet és az új fizikai eszközök működését ellenőrzöm és tesztelem.

Az evosoft, Magyarország egyik vezető szoftver fejlesztéssel foglalkozó vállalata, mely három telephelyen biztosít munkalehetőséget alkalmazottjai számára. Ez a három telephely Budapesten, Miskolcon és Szegeden található. A szervezetnek jelenleg több mint 1500 alkalmazottja van, és emellett a 2021-es évi nettó árbevétele közel 29 milliárd forintra rúg. [2]

A cég hat nagy részegységből áll, melynek egyik része az O3-as ágazat, ahol én is dolgozom. Ez a System Test & Engineering névre hallgat, melynek profiljába értelemszerűen a rendszertesztelés tartozik. Emellett helyet kapott az ágazatban egy úgynevezett External Business projekt is, mely a világpiaci megkeresésekre és megbízásokra biztosít ipari megoldásokat.

Egyetemi tanáraim számára bizonyára nem kell bemutatnom a céget, hisz együttműködési megállapodás keretein belül dolgozik együtt az Óbudai Egyetemmel. „Stratégiai megállapodást kötöttünk az Óbudai Egyetemmel. Számos egyetemi szakmai eseményen erősítjük a vállalat jelenlétét, szakmai rendezvényeket támogatunk, aktív szerepet vállalunk egyetem és ipari partnerei műhely- és kerekasztal-beszélgetésein, hallgatói helyet nyújtunk a kooperatív képzés keretében, és a 2018/2019-es tanévtől duális képzésben is részt veszünk.” [3]

3. A megoldandó feladat ismertetése

Elsődleges célom nem az volt, hogy tökéletesen működő szerkezetet keltsek életre, hanem az, hogy egy egységként működő rendszert tudjak prezentálni, amely bemutatja az abban felhasznált technológiákat. Ezzel a projekttel szeretnék ötletet adni munkatársaimnak, hogy ezen technológiák felhasználásával meg tudják könnyíteni saját, és csapatuk életét. Általános célom egy iránymutatás, melyet kiegészítve, ennél akár egy sokkal bonyolultabb felügyeleti rendszert be tudnak üzemelni.

A feladatom az volt, hogy egy olyan rendszert alkossak meg, ami felügyeli és kezeli a tesztek során használt berendezéseinket. Ez a rendszer megoldást nyújt az áramkimaradás után fellépő magas áramfelvételre, méri és optimalizálja az eszközök energiafelhasználását, és egy esetleges nem kívánt üzemállapot bekövetkeztekor figyelmeztetést, úgynevezett „alarm” -ot generál, sürgősebb esetekben e-mailes értesítést küld az üzemeltetőnek.

A tesztberendezéseinket direkt erre a célra épített állványokon, rack-eken rendszerezzük. Ezek az állványok különböző méretekben találhatóak meg az irodában, függően attól, hogy mennyi eszközt szeretnénk rajtuk elhelyezni. A legtöbb ilyen tartó keret, kb. 2 m magas 1,5 m széles és 80 cm mély. Ezzel a terjedelemmel hozzávetőlegesen 40-60 db modulnak tudunk helyet adni.

4. Technológia áttekintés

4.1. PLC tudnivalók

4.1.1. Alapismeretek

Pár szóban szeretném bemutatni a PLC-t, azaz a Programmable Logical Controller-t. A PLC-t egy ipari számítógépként tudnám a legjobban jellemezni, melynek legfőbb tulajdonsága a megbízhatóság. Ez az a tulajdonság, melynek köszönhetően sok esetben létjogosultsága van az ipari környezetben. Egy olyan eszköz, mely érzékelőktől kapja az információt és a belekódolt logika alapján irányítja az aktuátorokat.

4.1.2. Profibus

A Profibus (Process Field Bus) egy gyártó-független terepi buszrendszer. Az irányítástechnikában szerepet játszó minden cég kínálatában megtalálható. Három fajtáját különböztetjük meg, melyek az FMS, a DP és a PA. Az FMS cella szintű kommunikációra, a DP a terepi eszközökkel való kommunikációra, míg a PA a folyamatszabályzásban használatos. A kommunikáción belül két fő szerepet határoztak meg, melyek a Master és a Slave. A Master felelős az adatforgalom irányításáért a buszon, vagyis ő küldi az üzeneteket. Aktív elemei a busznak. A Slave-ek általában ki-bemeneti eszközök, melyek fogadják az üzeneteket, amit a Master küldött. Képesek adatot küldeni, de ezt csak a Master kérésére tehetik meg. Passzív elemei a busznak. [4]

4.1.3. Profinet

A Profinet egy nemzetközi szabványokon alapuló nyílt ipari Ethernet megoldás. Ez egy kommunikációs protokoll, amelyet az automatizálási környezetben lévő vezérlők és eszközök közötti adatcserére terveztek. A 2000-es évek elején vezették be, és ez a legelterjedtebb ipari Ethernet megoldás. Gyakran emlegetik a Profibus-DP utódjaként is, hisz annak továbbfejlesztése. Sok esetben sokkal előnyösebb tulajdonságokkal rendelkezik. Címzése IP alapú. [5]

4.2. Hibakritikus rendszerek

„Ahol a biztonságos működés az első számú követelmény a technológia veszélyessége miatt, mint például a kazánautomatikák, vasúti szerelvények automatizálása, gáz- és olajszállítással kapcsolatos automatikák, vegyipar, nukleáris erőművek stb., ott a hibakritikus (Fail-Safe) PLC-konfiguráció szükséges. Ha a folyamat jellege olyan, hogy az irányítórendszerben bekövetkező egyedi hibák az élet- és vagyonbiztonság szempontjából veszélyes állapotot hozhatnak létre, a folyamat veszélybiztos irányítórendszert igényel. A veszélybiztos irányítórendszer a hiba fellépésekor a folyamat leállításával képes a veszélyhelyzet kialakítását megakadályozni.” [4]

4.3. Felügyeleti rendszerek

„A központi vezérlőegység egy kifejezetten az épületautomatizálási rendszerek vezérlésére épített, kimeneti és bemeneti csatlakozási lehetőségekkel is rendelkező számítógép.”

„Minden modern épületautomatizáló rendszer rendelkezik riasztási képességgel, amelynek révén a karbantartó személyzet tudomást szerezhet a potenciálisan veszélyes helyzetekről és kezelheti azokat. A rendszer figyelmeztetheti a megfelelő személyeket számítógép (e-mail vagy SMS üzenet), telefonhívás vagy akusztikus riasztás útján, illetve ezek szabadon meghatározott kombinációjával. A riasztásokról biztonsági és biztosítási megfontolásokból az automatizálási rendszerek felvételeket készítenek, rögzítik azok adatait.” [5]

4.4. Árammérés elmélete

A megvalósított feladat során sort kerítettem feszültség, illetve áram mérésére. Míg előbbit meglehetősen egyszerűen véghez lehet vinni, mivel nem kell megszakítani az áramkört, addig az árammérése egy fokkal bonyolultabb folyamat. Ahhoz, hogy meg lessen mérni a berendezések áramfelvételét, egy mérőátalakítót kellett alkalmaznom. Mivel váltakozóáramú a berendezés megtáplálása, így egy mérőtranszformátor segítségével tudtam véghez vinni a feladatot. A mérőtranszformátor nem sokban tér egy hagyományos transzformátortól, működési elve megegyezik. A mérőtranszformátoroknál a primer oldali tekercs menetszáma eggyel egyenlő.

Ezen a vezetéken folyó áram képezi a mérendő áramunkat. Az imént említett vezetőn folyó áram hatására, a mérőtranszformátor vasmagjában mágneses tér fog kialakulni. A vasmagra egy másik, magasabb menetszámmal rendelkező tekercs van csévélve. Ebben a szekunder tekercsben áram fog indukálódni a mágneses fluxus hatására. A szekunder tekercsre egy árammérőt kötve, arányosított áramértéket tudunk mérni. Ez az arány, a két tekercs menetszámától függ. Ez az arányszám a mérőtranszformátorokon fel van tüntetve.

4.5. E-mail küldéséhez szükséges ismeretek

4.5.1. SMTP szerver

Az SMTP az angol Simple MailTransfer Protocol rövidítése. Ez a szabvány egy olyan kommunikációs protokoll, ami az e-mailok interneten történő továbbítását foglalja össze. Az SMTP egy viszonylag egyszerű, szöveg alapú szabvány, ahol egy üzenetnek egy vagy több címzettje is lehet. A külső-SMTP szerver a kiszolgáló szerver, ami a levelezőtől megkapja a levelet és a levélben megadott címzettek fogadó-SMTP szerverének átadja azt. E-mail küldésekor a levelező kliens *(Microsoft Outlook)* SMTP segítségével továbbítja az üzenetet az előre megadott szerver felé, amely szintén SMTP segítségével juttatja el a megfelelő levelezőszerver számára, ahol a címzett postafiókja található. Az alapokat tekintve az SMTP parancsok sokaságát takarja, amelyek elengedhetetlenek az internetes levelezéshez.

Az SMTP az elektronikus levelezőrendszerekben az üzenetek továbbítását végző szabvány. Amikor az SMTP-kliens levelet szeretne küldeni egy másik kliens számára, két irányú kapcsolatot hoz létre az SMTP-levélszerverrel. Ez lehet a levél végső állomása is, de lehet csak egy köztes állomás is. Ha nem a végső állomása, akkor a küldő levélszerver SMTP-protokoll segítségével továbbítja a levelet a címzett kliens levélszerveréhez. A kliensek és a szerverek közötti adatcsere az SMTP-protokoll alkalmazásával történik. Az SMTP az üzenetek továbbítására a TCP-protokollt használja. Fontos megjegyezni, hogy az SMTP-protokollnak nem feladata a leveleknek a címzett klienshez való eljuttatása, ezt a feladatot más protokollok, a POP, illetve az IMAP látják el.

4.5.2. POP

A POP az angol Post Office Protocol version rövidítése. Ez a szabvány egy alkalmazás szintű protokoll, aminek a segítségével az e-mail kliensek (Outlook Express, Thunderbird stb.) letölthetik az elektronikus levelezéseket a kiszolgálóról. Jelenleg a legújabb POP protokoll a POP3, ennek elődje a POP2 és a POP volt. Ennek a szabványnak a segítségével a felhasználó könnyedén letöltheti a leveleket és azokat a saját eszközein olvashatja. Abban az esetben, ha a POP3-at használjuk akkor gyorsabban kereshetünk a leveleink között mintha IMAP-et használnánk, mivel a POP3 esetében a levelek lokálisan a felhasználó számítógépén vannak jelen.

Ha ezt a megoldást választanánk akkor a levelek olvasása korlátozva lenne arra a gépre, ahova letöltjük azokat. Szakdolgozatom során én az IMAP típusú protokollt használtam. Fontos különbség az, hogy az IMAP-el ellentétben, a levelek csak addig elérhetőek a kiszolgálónál amíg le nem tölti a felhasználó. Legfőbb előnye, hogy offline is olvashatóak a levelek. Gyakori azonban, hogy a levelező rendszerek összeomlanak a sok levél miatt, ez adatvesztéssel járhat.

4.5.3. IMAP

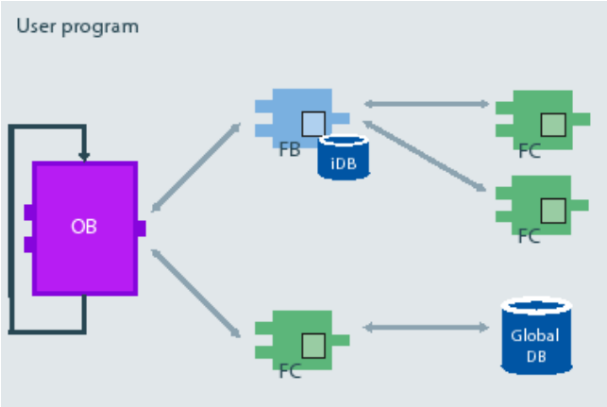
Az IMAP az angol Internet Message Access Protocol rövidítése. Ahogyan a POP3, ez a típusú protokoll is alkalmazásrétegbeli, aminek a segítségével hozzáférhetünk az e-mailjainkhoz. A POP3 mellett ez a legelterjedtebb levéllekérési internetszabvány. Az IMAP esetén a felhasználó nem másolja le a leveleket lokálisan a saját számítógépére, hanem magán e-mail szerveren tárolja azt. Ebben az esetben a levelek bármilyen levelezőprogramból, bármilyen számítógépről vagy bármilyen felhasználói felületről könnyedén elérhető és akár egyszerre is használhatóak. Minden levelet, minden eszközön látni fogunk akkor, ha csatlakozva vagyunk a levelező szerverhez. IMAP hátránya, hogy a szövegtörzsben való keresés lassabb, mint a POP3 esetén.

4.5.4. hMail szerver

A feladat megoldása során a hMail szerver programot használtam a belső hálózaton való levelezőrendszer üzemeltetésére. A hMail szerver egy ingyenes, nyílt forráskódú e-mail szerver a Windows felhasználók számára. Támogatja a közös e-mail protokollokat (IMAP, SMTP és POP3) és könnyen integrálható számos létező webes levelező rendszerrel. Rugalmas spam védelemmel rendelkezik, és a víruskeresőhöz csatlakozva beolvashatja az összes bejövő és kimenő e-mailt. Esetemben a Thunderbird levelező rendszert használtam, mivel a céges Outlook olyan biztonsági funkciókkal van ellátva, hogy lehetetlen volt csatlakoztatni az SMTP szerverrel.

5. PLC programozási ismeretek

Ahhoz, hogy érthetően be tudjam mutatni a PLC programot, szeretnék ismertetni néhány fogalmat, melyek a programozás során használatosak. A programot blokkokba szervezzük a könnyebb átláthatóság és futtatás érdekében. Alapvetően négy féle blokkot tudunk használni programozás során. Az első az „Organisation Block” (OB), vagyis szervező blokktípus, mely egy jellegzetes lila színt kapott a fejlesztői környezetben. Ezen kívül tudunk még használni függvényeket és funkció blokkokat, melyek közül az előbbit zöld, utóbbit, világos kék szín jellemzi. Működésüket az 5.1-es bekezdésben részletezem. Az eddig felsorolt elemekbe magát a programot tudjuk írni, ám szükségünk lesz valamiféle adatterületre is. Erre szolgál az úgynevezett adat blokk, vagyis a „DB” – az angol „Data Block” rövidítése – melyet a sötét kék színezés alapján tudunk könnyedén felismerni, ahogyan az a 2. ábrán is látható.



1. ábra - A PLC által használt blokk struktúra

5.1. Blokk típusok

5.1.1. OB – Szervező Blokkok

Az OB, vagyis szervező blokktípus egy olyan programozási elem, melyben rendszerezni tudjuk a megírt programjainkat. Az OB-k biztosítják a kapcsolatot a PLC operációrendszere és a felhasználó által megírt program között. Három féle OB típust tudunk megkülönböztetni, melyek a következők:

- Startup OB (OB100)
  Egyszer hajtódik végre, amikor a PLC üzemmódja STOP-ról RUN-ra változik. Miután ez megtörtént, a fő programciklus OB végrehajtása kezdődik meg. Általában inicializálásra használják.
- Ciklikus OB-k
  Ide sorolandó a fő programciklus (OB1). Ez a blokk rendelkezik a legmagasabb prioritással, és mindaddig fut, ameddig valami meg nem szakítja azt. Ismétlődő blokk továbbá a ciklikus megszakító blokk, amely egy előre meghatározott időnként megszakítja a fő programciklust, végrehajtva az abba írt programsorokat. Prioritása szabadon állítható a dedikált OB számok kivételével.
- Megszakításos OB-k
  Angolul „event OB” -knak hívjuk őket, ugyanis egy előre definiált esemény hatására szakítják meg a főprogramciklust. Ez az esemény azonban nem periodikus, így aciklikus OB -ként is hívhatjuk őket. Ide tartoznak például a diagnosztika során használt hardveres megszakítás OB.

5.1.2. FB – Funkció Blokkok

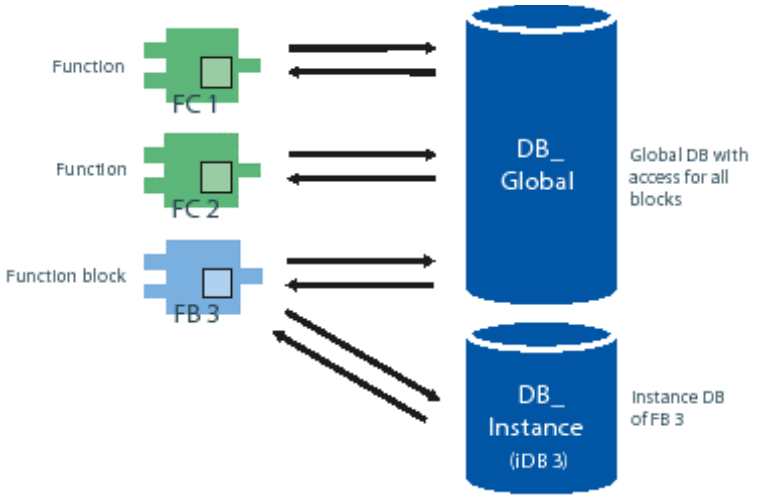
A funkció blokk olyan szubrutin, amelynek több be-, és kimeneti változója is lehet. A fordító program csak egyszer fordítja le a funkció blokkot, és többszöri meghívás esetén csak a bemeneti (feldolgozandó adatok), illetve kimeneti (eredmények) változók részére foglal le programterületet. Ezek a memória területek bármely programból elérhetők. Jelentős különbség a függvényekkel szemben, hogy a funkcióblokkok esetében tudunk használni statikus változókat, vagyis ezek a statikus változók megtartják értéküket a következő programciklusra. Ez amiatt következhet be, mert a funkcióblokk meghívásakor generálódik egy saját adatterület, melyhez csak ez a funkcióblokk fér hozzá.

5.1.3. FC – Függvények

A függvény olyan szubrutin, amelynek több be-, és csak egy kimeneti változója is lehet. A kimeneti változó a CPU munkaregiszterében generálódik, ezért, ha a következő programsor nem dolgozza fel, akkor elvész. A fordító program csak egyszer fordítja le a függvényt, és csak a bemeneti (feldolgozandó adatok) változók részére foglal le programterületet. A függvény esetén nem tudunk statikus változókat használni, mivel a függvény csak temporális adatterülettel rendelkezik. Egy függvény blokkba úgy kell megírni a programunkat, hogy ha az abban lévő adatokat a következő programciklusban is használni szeretnénk, akkor azokat inicializálni kell.

5.1.4. DB – Adat Blokkok

Az adatblokkok értelemszerűen a programadatok tárolására szolgálnak. Az adatblokkok tehát olyan változó adatokat tartalmaznak, amelyeket a felhasználói program használ. A globális adatblokkok az összes többi blokk által használható adatokat tárolják, míg a funkció blokkok által generált adatterülethez, csak a hozzátartozó funkció blokk fér hozzá. Tehát, ahogy a 3. ábrán is látható, egy globális adatterülethez hozzáfér minden függvény és a funkcióblokk, azonban a funkcióblokk adatterületéhez, csak ő maga fér hozzá. Az adatblokkok maximális száma a PLC típusától függően változik.



2. ábra - Globális és funkcióblokk által generált adatblokkok hozzáférhetősége

5.2. Programozási nyelvek

A programozás során sokféle programozási nyelvet használhatunk. Jelenleg hét nyelv közül tudunk választani, ám ahogy fejlődik az iparág, egyre több és több nyelvet kell elsajátítani, ha mindegyikkel tisztában szeretnénk lenni. Alapvetően két nagy csoportra tudjuk osztani ezeket. Az egyik nagy csoport a szöveges (pl. STL, SCL), a másik pedig a grafikus programozási nyelvek. Ide sorolható többek között a LAD vagy az FBD nyelv.

Amikor programozunk, akkor ezeket a nyelveket akár funkcióblokkon belül is változtathatjuk. Ha célszerűbbnek tartjuk, akkor egy aritmetikai bit művelet megírásához szöveges, míg bonyolultabb blokkok esetén inkább a grafikai nyelveket használhatjuk.

5.2.1. LAD – Ladder

A LAD egy grafikus programozási nyelv. Az ábrázolás alapja az áramút terv. A programot hálózatokba tudjuk szervezni. Egy hálózat bal oldalon tartalmaz egy tápsínt, ahonnan a lépcsőfokok erednek. A bináris jelsugarak érintkezők formájában vannak elrendezve a lépcsősíneken. Az elemek soros elrendezése egy-egy lépcsőfokon soros kapcsolatot hoz létre. Az egyidejű ágakon való elrendezés párhuzamos kapcsolatot hoz létre. Az összetett függvényeket blokkok ábrázolják.

5.2.2. FBD – Funkció Blokk

Az FBD egy grafikus programozási nyelv. Az ábrázolás az elektronikus áramköri rendszereken alapul. A programot ebben az esetben is hálózatba tudjuk szervezni. Egy hálózat több logikai műveleti utat is tartalmazhat. A bináris jelsorozatokat blokkok kötik össze. A logika ábrázolása a Boole-algebrában használt grafikus logikai szimbólumokon alapul.

5.2.3. STL – Utasításlistás szöveg

Az STL egy szövegalapú programozási nyelv, amelyet általában logikai blokkok programozására használnak. Nagyon hasonlít az Assembly-hez, mivel egy hardverközeli programozási nyelvről beszélünk. Az STL program esetén is van lehetőségünk hálózatokra osztani a kódunkat. Az egyes STL-utasításokat egy hálózat soraiban programozzuk, ahol soronként csak egy STL-utasítás adható meg. A sorok számozása minden hálózatban eggyel kezdődik, és minden új sorral növekvő sorrendben folytatódik.

Minden utasítás egy parancsot jelent a CPU számára. A CPU az utasításokat fentről lefelé haladva hajtja végre. Az STL-ben két munkaregiszterrel tudunk dolgozni, melyeket akkumulátor1-nek és akkumulátor2-nek neveznek.

5.2.4. SCL – Strukturált szöveg

Az SCL (Structured Control Language) egy PASCAL-on alapuló magas szintű programozási nyelv. A nyelv alapja a DIN EN 61131-3 (nemzetközi IEC 1131-3) szabvány, mely a programozható logikai vezérlők programozási nyelveit szabványosítja.

Az SCL a PLC tipikus elemei, mint például a bemenetek, kimenetek, időzítők vagy memóriabitek mellett komplexebb programozási nyelveket is tartalmaz.

- Kifejezések
- Érték-hozzárendelések
- Operátorok
- Programvezérlés

Az SCL kényelmes utasításokat biztosít a program vezérléséhez, amelyek lehetővé teszik például programelágazások, ciklusok vagy ugrások létrehozását.

Az SCL ezért különösen alkalmas a következő alkalmazási területekre:

- Adatkezelés
- Folyamatoptimalizálás
- Receptek kezelése
- Matematikai/statisztikai feladatok

6. Alkalmazott technológiák

6.1. Szoftver

6.1.1. TIA Portal V17

A Siemens egységes automatizálási fejlesztői platformja a Totally Integrated Automation Portal – későbbiekben TIA Portal. Ez a szoftver egy egységesített fejlesztői környezet, amely magában foglalja a korábban külön-külön használt szoftverkomponenseket. Ilyen szoftverek például a STEP7+, mellyen a PLC programok íródtak, a SIMATIC WinCC, mellyel lehetőség van képi megjelenítők programozására, és ezen komponensek mellett integrálva van még egy nagyon fontos komponens, mely a biztonsági funkciókért felel. Ez az úgynevezett SIMATIC Safety szoftvercsomag, amellyel az eszközök biztonsági funkcionalitását lehet megvalósítani. A TIA Portal alapverziója ezen három szoftverkomponenst tartalmazza, melyek egyszerre települnek és párhuzamosan tudja használni ezeket a mérnök a munkája során, és így egy sokkal átláthatóbb és könnyebben szerkeszthető munkát tud kiadni a kezei közül.

Én a TIA Portal V17-es verzióját használtam a projekt kidolgozása során, mely jelenleg a legfrissebb kiadott főverzió.

6.1.2. Thunderbird

Az egyik legismertebb és legnépszerűbb levelező kliens program. A program legfőbb előnye, hogy teljesen ingyenes, képes nagyon sok bővítmény kezelésére. A bővítmények kezelése nagyon egyszerű, úgy működik ahogyan már a Mozilla Firefox esetében megszokhattuk. Képes egyszerre több e-mail fiók kezelését ellátni, többek között ezért is választottam ezt. Tanítható spam szűrővel rendelkezik, aminek segítségével a kéretlen leveleket ki tudja szűrni automatikusan.

6.2. Hardver

A felhasznált eszközök mindegyikét az evosoft biztosította számomra. Összegük jelentős, éppen ezért a pusztán erre a célra való megvásárlásuk meglehetősen magas költségű lenne. Ám az én helyzetemben ez nem állt fenn, hiszen a felhasznált eszközöket nem kellett újonnan beszerezni, mivel korábbi tesztekből visszamaradt készülékekről beszélünk. Így kézenfekvő volt felhasználni ezen hardvereket.

6.2.1. S7-1518F-4PN/DP

A 1518F PLC-t én választottam ki. Szerettem volna, ha az S7-1500-as termékcsalád egyik tagját használom, mivel mindenki számára könnyen kezelhető kontrollerről beszélünk. Emellett, a képi megjelenítést View of Things segítségével szerettem volna megoldani, amiben ez a PLC segítségemre volt, hiszen támogatja a webszerver funkció ezen szegmensét. Választási szempont volt továbbá, hogy a Profinet mellett Profibus interfésszel is rendelkezzen, ugyanis az ET200SP fejegység, melyet használtam, csak Profibus-os kommunikációra képes. Az utolsó szempont a FailSafe tulajdonság, ugyanis a programomnak vannak elemei, melyek megkövetelik a hibakritikus működést.

3. ábra - 1518F-4PN/DP

6.2.2. ET200SP – IM155-6DP HF

Az ET200 fejet az iparban a távoli pontok eléréséhez használják. Úgynevezett terepi (kihelyezett) eszköz, mely a PLC-től távol lévő ki-bementi perifériákat hivatott kezelni. Esetemben, azért volt rá szükség, mert az energiamérőmodul, melyet kaptam, csak ilyen

ET200SP fejjel kompatibilis. A rack-szekrényen, nem volt Profinet kommunikációra képes eszköz, így a IM155-6DP HF-re esett a választás. (4. ábra)

4. ábra - ET200SP - IM155-6DP HF

6.2.3. AI Energy Meter CT ST

Ahogy az előző pontban is említettem, az AI Energy Meter CT ST az ET200SP termékcsalád modulja, mely ipari energiamérésre szolgál. Alkalmas háromfázisú energiamérésre, de emellett számtalan villamos tulajdonságát képes analizálni a mérendő berendezéseknek. Csak hogy néhányat említsek a teljesség igénye nélkül: frekvencia, fázisonkénti áram, zárlati áram, vonali feszültség, fázisszög, teljesítmény, munkaóra stb. Én ezek közül néhány alap villamos tulajdonságot mértem, melyeket később kijelzésre, és további számításokra használtam.

6.2.4. S7-314C-2 PN/DP

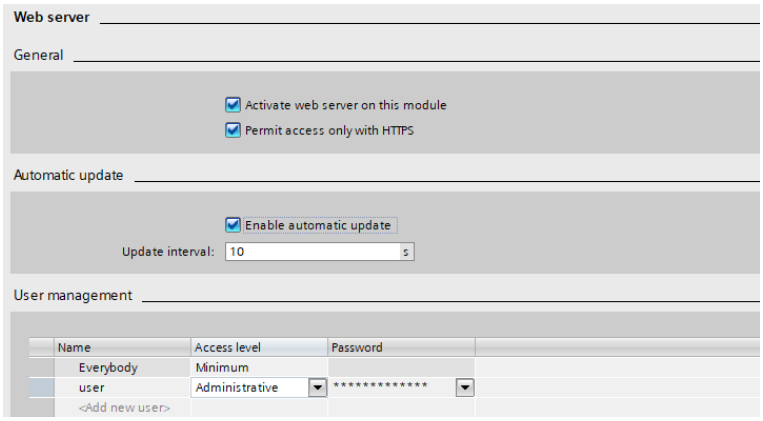
A 314C-2 PN/DP névre hallgató PLC egy S7-300-as termékcsaládba tartozó PLC, mely egy kompakt eszköz. Közvetlenül a készüléken 24db digitális bemenet és 16db kimenet található, míg analógból 5db bemenet és 2db kimenet van rajta. A PLC mögött helyet kapott még két modul, melyek egyenként 16db digitális kimenettel bővítik a csatlakozási lehetőséget. Én egy kihelyezett ki-bementi modulként használtam az eszközt, vagyis feladata az volt, hogy az 1518F PLC által kiadott parancsokat végrehajtsa. A két eszköz Profineten kommunikált egymással, így az ehhez szükséges interfész elengedhetetlen volt számomra.

6.3. Vizualizáció

6.3.1. View of Things

A TIA Portal V17 egyik legérdekesebb újdonsága a felhasználó által meghatározott weboldalak létrehozására szolgáló új rendszer, az úgynevezett View of Things - röviden VoT. A View of Things lehetővé teszi a programozó számára, hogy a WinCC egységes szerkesztője és a TIA Portal előre meghatározott grafikus elemei segítségével a felhasználó weboldalakat hozzon létre. Ha a View of Things eszköztára nem lenne elegendő számunkra, akkor a PLC webszerverére külön is programozhatunk weboldalt, de ehhez HTML és JavaScript ismeret szükségeltetik. Mivel az előre elkészített elemek tárházát elegendőnek ítéltem meg, így én meg tudtam valósítani a képi megjelenítést csupán ezen elemek használatával.

A VoT egy hibrid megoldás, ugyanis egy képi megjelenítőről beszélünk, ám ez a program a PLC memóriájában tárolódik, és a maga PLC futtatja azt. Ebben az esetben nincs szükség egy külön erre a célra dedikált eszközre, mint pl. HMI panel vagy SCADA Runtime. Ahhoz, hogy használni tudjuk a View of Things funkciót, engedélyeznünk kell a PLC webszerverét. Ezt az eszköz konfigurációs beállításainál tudjuk megtenni, ahol a „Web server” fül alatt ki kell pipálni a megfelelő jelölőnégyzetet. Miután ezt megtettük, hozzá kell adnunk legalább egy felhasználót. Ezt egy kicsivel lejjebb a „User management” alatt találjuk. Miután megadtunk egy felhasználót és egy hozzátartozó jelszót, definiálnunk kell ezen felhasználó jogosultságait. Mivel az általam programozott webszerveren használtam olyan elemeket, amelyek módosítani tudják a program futását, így az esetemben mindenképpen engedélyeznem kellett a „Tag access” jogosultságot. Ha ezt megtettük, akkor neki lehet kezdni a weboldal leprogramozásának. Én ezt a 8.2.1. alatt fogom részletezni.



5. ábra - A webszerver funkció engedélyezése

A programunkat a PLC webszerverén keresztül tudjuk elérni. Csupán annyi a dolgunk, hogy egy böngészőn keresztül megnyissuk a PLC webszerverét, amit alapértelmezett esetben a PLC X1 portján beállított címen keresztül tudunk megtenni. Értelemszerűen egy hálózatban kell lennie a PLC-nek és az eszköznek, amelyen el szeretnénk érni a webszervert. A böngészőbe az X1 porton beállított IP címét kell beírnunk úgy, hogy a cím elé a http:// vagy a https:// előtagot gépeljük annak függvényében, hogy milyen titkosítást állítottunk be a PLC webszerverének. Ha ezt megtettük, akkor meg fog jelenni a webszerver webes felülete, melybe a korábban létrehozott felhasználói fiókkal tudunk bejelentkezni. A képernyő bal oldalán találjuk a menüsávot, ahol a „User pages” -re kattintanunk annak érdekében, hogy megnyissuk a View of Things ablakot. Ha ezt megtettük, már elénk tárul a korábban leprogramozott megjelenítőnk.

7. A rendszer kialakítása

A rendszer kialakítását két szempontból közelíteném meg. Az egyik, a villamos megtáplálás, míg a másik az eszközök közötti kommunikáció.

7.1. Villamos kialakítás

7.1.1. Villamos megtáplálás

A villamos megtáplálást azért tartom fontosnak kiemelni, ugyanis a rendszer úgy lett kialakítva, hogy bizonyos esetekben áramtalanítja a rack-szekrényen található eszközöket, néhány kivétellel. A kivétel alatt azt a néhány eszközt értem, melyek a felügyeleti rendszer üzemeltetéséért felelősek.

6. ábra - PS307 típusú tápegységek

A rack-szekrény 230 V váltakozó áramú megtáplálással rendelkezik. Az eszközök működéséhez szükséges 24 V egyenáramot, a szekrény hátuljára szerelt, két darab PS 307, 10 A tápegység állítja elő (6. ábra). A tesztberendezéseket, egy csoport relén keresztül látják el a megfelelő tápfeszültséggel, míg az eszközfelügyelet eszközei közvetlen betáplálást kaptak. A relécsoport egy 314C-2 PN/DP eszközbe vannak bekötve. Ez egy korábbi kialakítás miatt van így, és mivel máshol nincs szükség a 300-as eszközre, így egyszerűbbnek láttam, hogy megtartom a kialakított kábelezést, és az előbb említett 300-as PLC-t egy ki-bemeneti modulként arra használom, hogy aktív állapotba hozzam vele a relécsoport elemeit. A relékimenetek a rack-szekrény elejére vannak kivezetve, ahol banándugó aljzatokban áll rendelkezésre a 24 V egyenáramú feszültség. A rack-szekrényeken nyolc sorban helyezkednek el az eszközök.

A relék két körben biztosítják az áramellátást. Az egyik csoport az első és negyedik, míg a második kör az ötödiktől a nyolcadik sorig látja el feladatát.

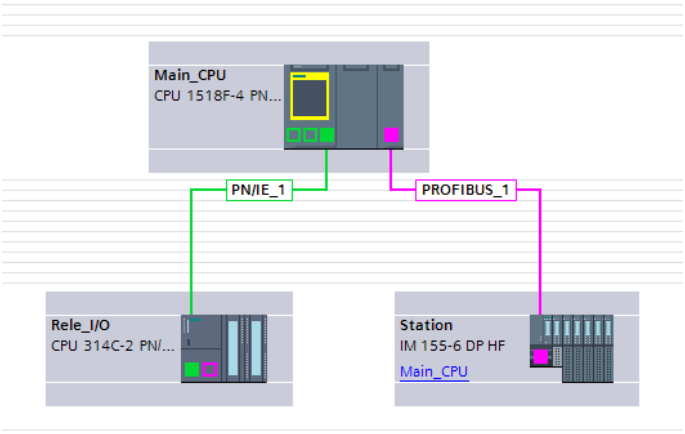
7.1.2. Villamos tulajdonságok mérése

A villamos tulajdonságok mérését, egy ET200 SP „Energy Meter CT HF” analóg mérőmoduljával végeztem. A modul, egy mérőtranszformátor segítségével tudja vizsgálni a villamos jellemzőket. Ez a transzformátor a „7KT 1200” névre hallgat, mely képes 3 fázis mérésére, de én ebben az esetben csak egy fázis analizálására használtam. Ehhez meg kellett bontanom egy hálózati kábel köpenyszigetelését, oly módon, hogy a fázist át tudjam vezetni a mérőtranszformátor tekercsein. Mivel csak egy fázist használtam, így a fázisvezetőt elég volt csak az egyik tekercsen átvezetni. Az energiamérő modulba, 4 kábelt kellett bekötnöm. Ebből kettő a hálózati betáplálásával egy csomópontban lévő nullvezető és fázis kábele, míg a másik kettő a mérőtranszformátor első tekercsének két kábele volt.

7.2. Eszközök közötti kommunikáció

A kommunikáció során két technológiát alkalmaztam. Ez a kettő a Profibus és a Profinet. Az előbbire azért volt szükség, mert a „SIMATIC ET200SP IM155-DP HF” egység nem képes Ethernet alapú kommunikációra, így Profibus DP technológiát használtam.

A másik esetben, a 314C-2 PN/DP eszközzel kellett kommunikálni, mely képes a Profinet alapú információ továbbításra, így nemes egyszerűséggel egy Ethernet kábellel összekötöttem a két eszközt.



7. ábra - Eszközök kommunikációs hálózata

7.2.1. Profibus konfigurálás

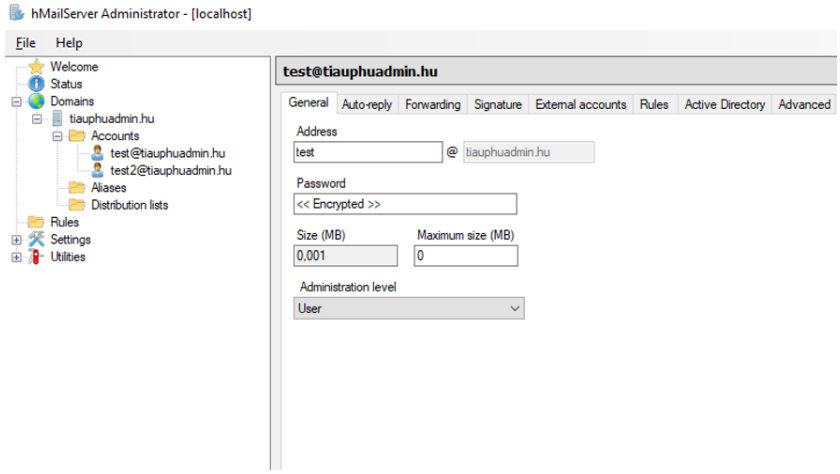
Én a Profibus-DP-t alkalmaztam a feladatmegoldás során. Ahhoz, hogy az energiamérő modulból ki tudjam nyerni az adatokat, konfigurálni kellett a Profibus kommunikációt. Ehhez első lépésben egy Profibus kábelre volt szükségem, mellyel összekötöttem a vezérlő PLC-t és az ET200SP fejegységet. A kábel mindkét végén lévő csatlakozón található egy busz záró ellenálláskapcsoló, melyeket mindkét esetben bekapcsoltam, mivel az adatbusz e között a két pont között található. Ezután Az eszközöket egyenként konfigurálni kellett. A 1518F PLC-t szoftveresen, míg a fejegységet hardveresen kellett beállítani. Az ET200SP előlapján található egy sor mikrokapcsolóból álló Profibus cím beállító egység, mellyel az adott eszköz címét lehet meghatározni. A kapcsolók kettő hatványainak felelnek meg. Ezek kombinációja adja a címet. Az én esetemben a slave eszköz az 58-as címet kapta, melyet úgy kaptam meg, hogy a 32, 16, 8 és 2-es jelölésű kapcsolókat jobbra állítottam, melyeknek összege az 58-at adta. (lásd 8. ábra)

8. ábra - Profibus címkapcsolók

Ezután a TIA Portálban kellett beállítani néhány dolgot. Az ET200 egységnél maradva, be kellett gépelnem, az előzőleg beállított Profibus címét, illetve, hogy slave-ként fog részt venni ebben a kommunikációban. Utolsó lépésként már csak a központi PLC-t kellett DP Master-nek konfigurálni, melyet az eszköz Profibus interfészbeállításainál tudtam elvégezni.

7.3. hMail szerver konfigurálása

Ahhoz, hogy a hMail szervert működésre bírjuk, létre kell hozni egy domain-t. Ez a domain két felhasználót tartalmaz, hogy a „másolat” funkciót is ki tudjam használni. A felhasználók fül alatt lehet kiválasztani, hogy milyen e-mail címe legyen az adott felhasználónak. Itt tudunk jelszót is beállítani, amivel később majd a levelező rendszerbe be tudunk lépni. Különböző szinteket lehet kiválasztani, hogy melyik felhasználónak, milyen jogosultsága legyen.



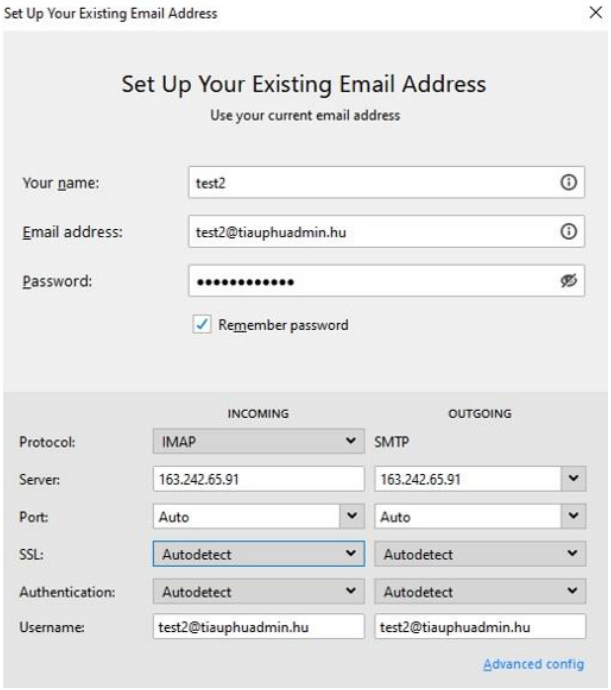
9. ábra - hMail szerver adminisztrátori felülete

Ezután az egyik legfontosabb lépés az a protokoll kiválasztása. Mivel több eszközről szeretném elérni a levelezéseket, ezért az IMAP szabványt választottam.

7.4. Thunderbird fiók konfigurálása

A program telepítése után a hMail szerverben megadott név, e-mail cím és jelszó segítségével tudjuk hozzárendelni az előre kiválasztott felhasználókat a levelezőrendszerhez.

Ahhoz, hogy a megadott szervert megtalálja a levelező program, a protokollt kell először kitölteni. Itt az IMAP protokollt választottam, ezután a szerver IP címét kell megadni. Ide azt az IP címet kell beírni, amelyik számítógépen a szerver fut.



10. ábra - A Thunderbird bejelentkező felülete

A megadott adatok alapján ellenőrzi a program ezeket és ha minden megegyezik a szerverrel akkor hozzákapcsolja a levelező rendszerhez az adott postafiókokat.

8. Működési leírás

8.1. PLC program

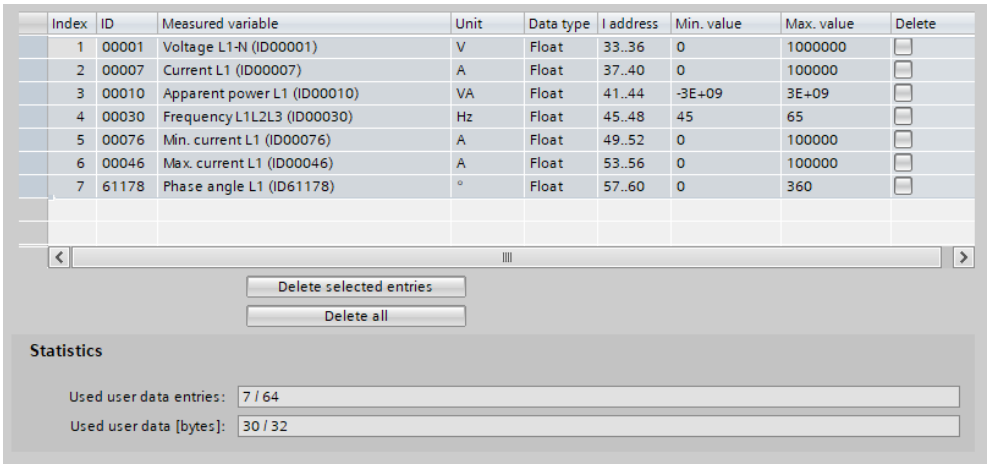
8.1.1. Hőmérsékletmérés

A PLC program talán legegyszerűbb része a hőmérsékletmérés volt. A méréshez egy PT100-as hőmérséklet érzékelő ellenállást használtam, melyet a PLC egyik analóg bemenetére kötöttem. A hőmérsékletmérésnek csupán annyi szerepe van, hogy ellenőrizzem az eszközök megfelelő hőmérsékleti tartományban vannak-e. A programot úgy írtam meg, hogyha a hőmérséklet nem a megfelelő tartományban van, akkor ez egy alarmot generál, így értesítve a kollégáimat, hogy vélhetően egy rendkívüli állapot lépett fel. A megfelelő hőmérsékleti tartományt 15°C és 40°C között definiáltam, vagyis, ha 15°C-nál alacsonyabb, vagy 40°C-nál magasabb a hőmérséklet, akkor felülvizsgálatot igényel a berendezés. Amennyiben a hőmérséklet meghaladja a 60°C-ot, akkor a kontroller PLC azonnal áramtalanítja a rack-szekrényen lévő eszközök nagy százalékát. Csupán azok az eszközök maradnak áram alatt, melyek a felügyeleti rendszer részét képezik.

A PT100-as hőmérséklet érzékelő ellenállás jelét először át kellett konvertálnom integerből real adattípusba. Ezt azért kellett megtennem, mert a hőmérsékletértéket egy tizedes kijelzésben kapjuk meg. Miután megtörtént a konverzió, egy tízes osztást kell még elvégezni, és máris helyes értéket kapunk. Ezt az értéket egy globális változóban eltárolom, majd ennek értékét egy előre definiált blokkal vizsgálom. A blokk neve: „out range”. A blokk bemeneti interfészére három értéket kellett adnom. Az intervallum két szélsőértékét és magát a vizsgálandó változót. Ha a hőmérséklet érték az előre definiált intervallumon kívül esik, akkor egy bool típusú változó igaz értéket kap, amelyet majd az alarm-generálásra fogok használni. (8.1.5.) A 60°C feletti értéket vizsgálatát egy egyszerű komparátorral végeztem. Hasonló az előző esethez, itt is egy bool fog igaz értéket kapni, amennyiben a hőmérséklet 60°C felett van. Ez a bool érték lesz majd felelős az esetleges áramtalanításért.

8.1.2. Villamos tulajdonságok mérése

Az adatok mérését a 6.2.2. pontban említett ET200 fejhez tartozó AI Energy Meter CT HF modullal végeztem. Ahhoz, hogy értelmezhető adatokkal tudjunk dolgozni, egy előzetes konfigurációra lesz szükségünk. Ezen konfiguráció során, be kell állítani a Profibus kommunikációt, meg kell adni a mérőtranszformátor adatait és össze kell állítani egy adat csomagot, hogy mely adatok legyenek kommunikálva a központi PLC felé. Én ebben az esetben áram, feszültség, frekvencia, teljesítmény és fázisszög adatokat mérettem a modullal, melyeket a PLC 33 és 60-as címei között értem el.



11. ábra - A mért adatok összeállítása

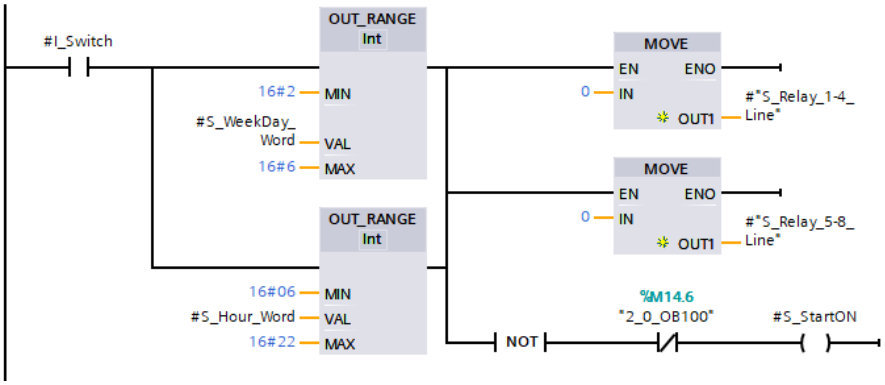
Mivel Profibus kommunikáció állt rendelkezésemre, így egy kissé le voltam korlátozva, hiszen csak 32 bájtnyi adatot tud továbbítani a modul. Az általam összeállított adatcsomag 30 bájt terjedelmű, tehát nem volt lehetőségem több adatot kezelni. Lásd (11. ábra) Ha bővíteni szeretnénk a rendszert, akkor Profinet kommunikáció képes eszközt kell alkalmaznunk, hiszen az 128 bájtnyi adat továbbítására képes. A mért értékeket a PLC bementéről átmásoltam globális változókba, melyeket majd a View of Things programozásánál használtam fel. Lásd (8.2.1.)

8.1.3. Energiafogyasztás optimalizálás

A tesztberendezéseket csak munkaidőben használjuk, így kijelenthetem, hogy meglehetősen pazarló lenne, ha minden eszköz a nap huszonnégy órájában, a hét minden napján rendelkezésre állnának. Éppen ezen megfontolásból merült fel az ötlet, hogy a tesztberendezések tápellátását a munkákaidőhöz igazítom. Egy olyan intervallumot választottam, mely nem okoz kellemetlenséget a kollégáim számára, ha esetlegesen nem reggel nyolc és délután négy óra között szeretnék használni az eszközöket. Ez az intervallum reggel hat óra és este tíz óra közé esett, és ez csak is a munkanapokon értendő. Ehhez ki kellett olvasnom a PLC rendszeridejét, melyet a RD_LOC_T előre definiált funkcióblokkal tudtam megtenni. Ezt a rendszeridőt egy DTL típusú változóban tároltam el. A Date_And_Time adattípussal ellentétben, a DTL-ben már egy különválogatott adatstruktúrába érkeznek az adatok, tehát külön változóba kerül az év, hónap, nap, óra stb. A DTL másik nagy előnye még a „weekday” nevű változója, mely egytől hétig számlál annak függvényében, hogy a hét melyik napját írjuk. A Siemens PLC-k időszámítási rendszere szerint a vasárnap az első nap, a szombat pedig a hetedik, tehát példaképpen, ha csütörtök van, akkor a „weekday” öttel lesz egyenlő. A rack-szekrényre felhelyezésre került egy két állású kapcsoló, mely arra hivatott, hogy energiatakarékos üzemmódba lehessen helyezni a rendszert. (12. ábra)

12. ábra - Energiatakarékos üzemmód kapcsoló

A PLC program megírásakor tehát figyelnem kell az előbb említett kapcsoló állapotát, az időt órában értelmezve, és az imént említett „weekday” változót, ugyanis tudnom kell milyen napot írunk. Erre egy kontaktust és két „out_range” blokkot használtam. (13. ábra)

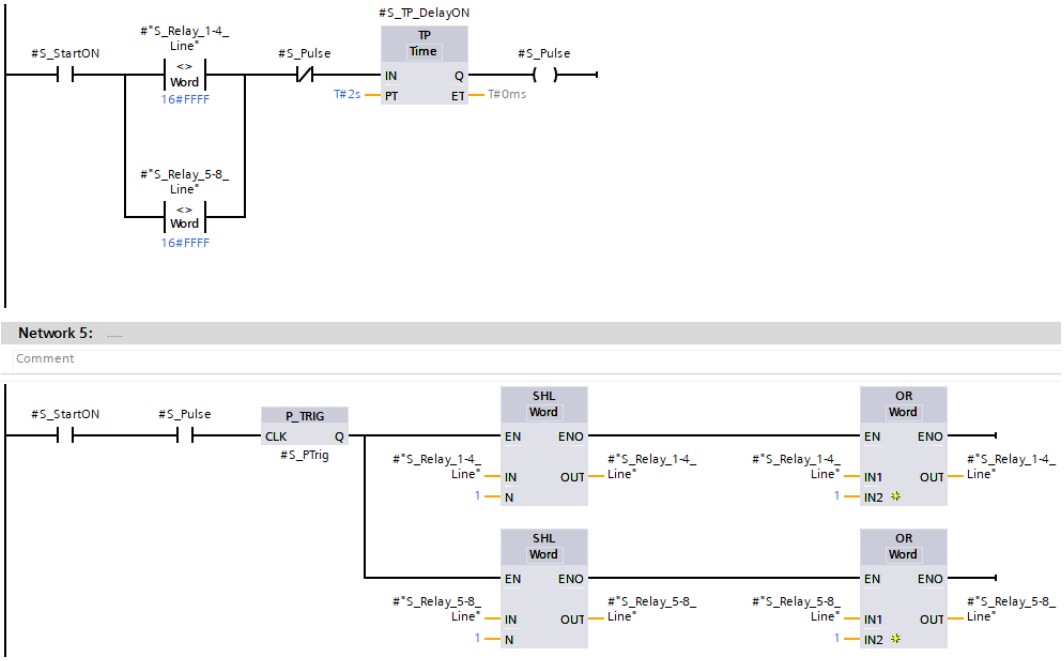


13. ábra - Idősáv vizsgálat programrészlete

A programot úgy írtam meg, még mielőtt bármit vizsgálna a program, szükséges a kapcsoló igaz állapota, vagyis a rendszer energiatakarékos üzemmódban üzemel. Ha ez adott, akkor a következő lépésben megvizsgáltam, hogy a „weekday” kettő és hat közötti értéket kap-e, vagyis hétfő és péntek közötti nap van, mert ha ezen intervallumon kívül esik, akkor a MOVE parancsokkal kikapcsolom a relékimeneteket. Ugyanezen logika alapján az alsó OUT_RANGE blokk, azt vizsgálja, hogy az idő reggel hat és este tíz óra között van-e. A két blokk logika „vagy” kapcsolatban helyezkedik el. Amennyiben a kapcsoló le van kapcsolva, akkor a relékimenetek tiltása megszűnik és a StartON üzemállapotba lép a rendszer.

8.1.4. Bekapcsolási áramfelvétel

Egy lehetséges probléma forrás lehet, ha egy esetleges áramszünet után, a hálózati feszültség visszakapcsoláskor minden eszköz egyszerre kapcsol be. Ilyenkor minden eszköznek egy azon időpillanatban nő meg az áramfelvétele, mely túlságosan megterheli a villamos hálózatot. Ez feszültségesést, extrémebb esetben újbóli áramkimaradást okozhat. Ez amiatt következhet be, mert a túláramvédelemmel ellátott kismegszakítók működésbe lépnek és lekapcsolnak. Ez az eshetőség egy rack-szekrény esetében nem lenne probléma, azonban az irodában megközelítőleg 80-100db ilyen egység található, melyeknek együttes visszakapcsolása már komoly terhelést jelent a villamos hálózatnak. Ezt kiküszöbölendő, egy időben eltolt felkapcsolást programoztam le. (14. ábra)



14. ábra - Időben eltolt felkapcsolás megvalósítása

Ez a megvalósítás két fő programrészletből áll. Az első, egy impulzusgenerátor funkciót lát el, míg a második egy a relékimenetek egyenkénti felkapcsolásáért felelős. Az impulzusgenerátor csak akkor fog működni, ha a StartON változó igaz értéket kapott. Ez például akkor következik be, amikor az energiatakarékos üzemmódban az előre definiált időintervallumokon kívül esik az aktuális idő. Ekkor a következő feltételhez érkezik a programfutás, mellyel azt vizsgálom, hogy fel van-e kapcsolva az összes relékimenet. Ha nincs, akkor egy TP pulzusgenerátor blokkal, két szekundum hosszú négyszög jelet generáltatok, melyet a második programrészletben használok fel.

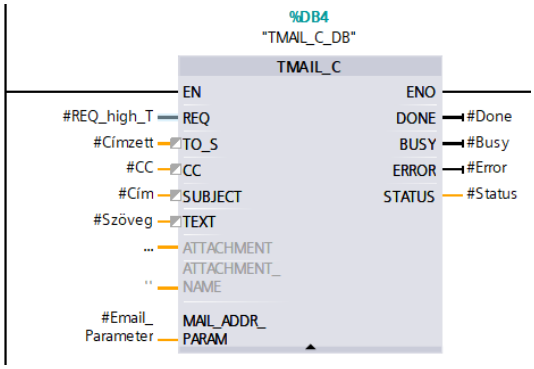
A programrészlet második felében egyenként felkapcsoltatom a reléket. Ezt úgy valósítottam meg, hogy a két másodperces impulzus jelnek a felfutó élekor eltoltam a kimenet értéket egy bittel balra, majd egy logikai vagy kapcsolat segítségével igaz állapotot adtam a nulladik bitnek, majd eltolás után az elsőnek bitnek és így tovább egészen addig, amíg a hetedik helyiértéken lévő bit is igaz értéket nem kapott. Ha ez megtörtént, akkor az impulzusgenerátor már nem fog lefutni, így a nem terhelve a PLC processzorát.

8.1.5. Figyelmeztető jelzés

A működés során felléphetnek nem megfelelő üzemállapotok, melyeknél célszerű, ha az üzemeltető valamiféle értesítést vagy jelzést kap a probléma fennállásáról, meglétéről. Én két figyelmeztetést programoztam le, melyek a hőmérséklettel kapcsolatosak. Ehhez a Program_alarm nevű blokkot használtam, mely egy előre létrehozott szöveg listára hivatkozva, bejegyzéseket generál a PLC diagnosztikai bufferében. Ezeket a szöveglistás üzeneteket, el lehet küldeni e-mail formájában is. A két figyelmeztetés a 40°C, és a 60°C feletti hőmérsékleti tartományba lépéskor generálódik.

8.1.6. E-mail küldés

Az e-mail-es értesítési funkciót a korábban generált bool értékek felhasználásával indítom. Ezt a TMail_C blokk REQ bemeneti paraméterére programoztam. Továbbá, meg kellett adom a címzett e-mail címét, másolat esetén a CC e-mail címet, a küldendő szöveget, és a levél címét. Ezeket előre definiáltam. A két különböző e-mailhez két külön álló blokkot használtam, így az előbb említett értékek statikusak. Utolsó lépésben meg kellett adom az e-mail szerver paramétereit, melyet az Email_Parameter struktúra tárol. Ebbe a struktúrába kellett bevinnem a hMail szerver adatait, hogy az e-mail azon keresztül érkezzen meg a megadott e-mailcímekre.



15. ábra - TMail_C blokk felparametrizálva

8.2. Vizualizáció

A vizualizáció működését már nem a PLC program működéseként magyaráznám, ugyanis hagyományos értelemben véve, ez már nem a PLC program része. A képi megjelenítőket jellemzően külön szokták programozni, és ezután integrálni a rendszerbe.

8.2.1. VoT működése

A képi megjelenítőt csak azután kezdtem el leprogramozni, miután kész volt a PLC program. A vizualizációt azért tartottam célszerűnek, mert információs felületet biztosít a felhasználó számára. Ezt oly módon teszi, hogy nem kell speciális szoftver az adatok kinyeréséhez vagy módosításához (pl. TIA Portal), hanem egyszerűen egy böngésző segítségével el tudjuk érni a kívánt funkciót. A webfelületet nem szerettem volna túlbonyolítani. Csupán egy képernyőből áll, melyen jól láthatóak az adatok. A programozáshoz, a TIA Portal beépített képernyőszerkesztő programját használtam. Először is létre kellett hoznom egy képernyőt, melyet főképernyőnek állítottam be, vagyis ez a képernyő fogja fogadni a felhasználót, amint megnyitja az oldalt. Ennek felbontása FullHD, vagy a meglehetősen elterjedt 1920 x 1080 pixel. A képernyőfrissítés ciklusidejét 250ms-ra állítottam, mely kétszer gyorsabb, mint az alapértelmezett beállítás.

16. ábra - Webalapú megjelenítő a VoT keretrendszerben

A képernyő tetején egy általános információkkal ellátott sáv kapott helyet, melyen a rack-szekrény azonosítására szolgáló felirat és az idő található. Az időmegjelenítést kétféleképpen is megoldottam, melyből az egyik digitális, a másik pedig analóg kijelzésű. Ezt azért tettem meg, mert a két időérték forrás eltér. Míg az analóg annak az eszköznek az idejét jelzi ki, melyről a böngészőt futtatjuk, a digitális a PLC rendszeridejét írja ki.

A fejléc alatt két hasáb kapott helyet, melyből a baloldalit a villamosértékék kijelzésére, a jobboldalit pedig egyéb információk megjelenítésére használtam. Ezeket statikus szöveg jelöli. A mért értékeket közvetlenül a PLC rendszermemóriájából jeleníti meg a rendszer, vagyis egy valós idejű adatkijelzésről beszélünk. Ahhoz, hogy a megfelelő értékeket lehessen látni, mindegyik adatnak egy-egy szövegdobozt hoztam létre. Ezeknek a szövegdobozoknak a tartalmát dinamizáltam és mivel csak kijelzésre szolgálnak, ezért „output” -ként állítottam be őket. Ezzel azt értem el, hogy a felhasználó nem tud belekattintani a szövegdobozba, így ezáltal nem tudja átírni az abban szereplő értéket. Mivel a kinyert adatoknak van tört értéke, így a kijelzési formát „floating-point” -ként konfiguráltam. Ezt követően meg kellett adnom a kijelezni kívánt érték mértékegységét. Ahhoz, hogy a kijelzett adatok egyértelműek legyenek, eléjük egy egyszerű szöveget helyeztem, mely egyértelműen utal az adathoz tartozó információkra.

A jobb oldali blokkban két fontos információ kapott helyet. Az egyik az energiatakarékos üzemmód állapota, míg a másik a PT100-as ellenállással mért hőmérsékleti érték. Az üzemmódváltó kapcsolót, úgy valósítottam meg, hogy ne csak kijelzésre szolgáljon, hanem akár aktiválni is tudjuk a funkciót. Ahhoz, hogy ezt meg lehessen tenni, a megfelelő jogosultsággal kell bejelentkezni a weboldalra. Ha a webfelületen található billenőkapcsolót átbillentjük, akkor a háttérszíne megváltozik, ezzel jelezve a felhasználó számára, hogy az interakció sikeres volt.

9. Számítások

A 8.1.3. pontban megvalósított program alapján számításokat végeztem, melyek számomra nagyon meglepő eredményeket hoztak. A fogyasztásmérő modul által mért teljesítményadat átlagosan 335 W. Amennyiben az energiatakarékos üzemmód ki van kapcsolva, akkor a rendszer a nap 24 órájában, a hét minden napján fogyasztja az áramot.

Ez azt jelenti, hogy a napi energiafogyasztása a következő:

$$W_{Ki\_napi} = P \cdot t = 335\ W \cdot 24\ h = 8040\ Wh = \mathbf{8,04kWh}$$

Ez egy hónapra vetítve (átlagosan 30nap/hónap):

$$W_{Ki\_havi} = W_{Ki\_napi} \cdot n = 8{,}04\ kWh \cdot 30 = 241200\ Wh = \mathbf{241,2\ kWh}$$

Ha az energiatakarékos üzemmódot alkalmazzuk, akkor a napi 24 óra helyett, csak 16 órában fogyaszt áramot a rack nagyrésze, mely a következővel egyenlő:

$$W_{Be\_napi} = P \cdot t = 335\ W \cdot 16\ h = 5360\ Wh = \mathbf{5,36\ kWh}$$

Mivel a hétvégi napokat is figyelembe vettem az energiagazdálkodás szempontjából, így egy 30 napos hónapban átlagosan 21 nap munkanap van, mivel 9 napot a hétvége ölel fel. Tehát ha egy teljes hónapra vetítjük ennek a berendezésnek a villamos fogyasztását, azt így néz ki:

$$W_{Be\_havi} = W_{Be\_napi} \cdot n = 5{,}36\ kWh \cdot 21 = 112560\ Wh = \mathbf{112,56\ kWh}$$

Ha az előző értékeket százalékosítom, akkor:

$$\Delta W_{\%} = \frac{W_{Ki\_havi} - W_{Be\_havi}}{W_{KI\_havi}} \cdot 100\% = \frac{241{,}2\ kWh - 112{,}56\ kWh}{241{,}2\ kWh} \cdot 100\% \cong \mathbf{53,3}\%$$

Ez alapján megállapítható, hogy a rack-szekrényen elhelyezett eszközök energia felhasználása több mint 53%-kal csökken. Úgy gondolom ez arányaiban nagyon nagy eltérés, és számottevően környezetbarátabb a szerkezet működése.

10. Tesztelés

Ahogy haladtam a program megírásával, folyamatosan teszteltem az egyes programrészleteket. Csak akkor haladtam tovább, ha megfelelően működött az adott komponens. A programokat logikailag egymásra épülő sorrendben írtam meg.

10.1. Hőmérséklet mérés

Elsőként a hőmérséklet mérést programoztam le. Már a programozás során szükségem volt folyamatos tesztelési műveletek elvégzésére, hogy tudjam jó irányba haladok-e. A program írása közben, következetesen letöltöttem a programot és ellenőriztem milyen értékeket és eredményeket kapok a hőmérséklet mérés során. Elsőként a hőmérsékletértéket vizsgáltam, mely első ránézésre helyesnek tűnt, de a biztonság kedvéért felmelegítettem a kezemmel, melyet online nézetben a monitoron ellenőrizni tudtam. Ezután, a hőmérsékletértékekből képzett bool értékeket ellenőriztem, hogy meglelő értéket kapnak-e az adott hőmérséklet intervallumokban. Az ehhez szükséges magas hőmérsékletet egy bögre forró vízzel állítottam elő. Amint az hőérzékelő elérte a 40°C-t, a kritikus hőmérséklet jelzésére szolgáló bool igaz értéket kapott. Hasonló volt a helyzet a 60°C-os küszöbhőmérsékletnél is. Miután mindezek helyes működéséről meggyőződtem, késznek nyilvánítottam a programrészletet.

10.2. Energiatakarékos üzemmód

Talán a legnehezebben tesztelhető program, az energiatakarékos üzemmód volt. Miután elkészült a program, úgy ellenőriztem a megfelelő működést, hogy a PLC rendszer idejét állítottam át. Az éjszakai órákra való lekapcsolás tesztelése egészen gördülékenyen alakult, ám amikor a hétvégéket kezelő kódot teszteltem, nem várt működést tapasztaltam. A program látszólag jól működött, ám a hétvége egy nappal el volt csúszva. Pénteken és szombaton teljesen lekapcsolt minden. Ekkor ellenőriztem a Siemens időkezelését, melynél azt tapasztaltam, hogy vasárnappal kezdődik a hét, és szombattal zárul. Miután ezen információk ismeretében, módosítottam a programot, már tökéletesen működött. A szimulációs környezet után valós körülmények között is ellenőriztem a működést, mely meglelő eredményeket hozott.

10.3. View of Things

A View of Things tesztelését is hasonlóan végeztem. Ahogy haladtam a program megírásával, folyamatosan letöltöttem a programot és ellenőriztem azt. Első körben csupán egy kezdőképernyő került letöltésre. Ezzel azt szerettem volna ellenőrizni, hogy elérem-e egyáltalán a webfelületet. Miután konstatáltam, hogy igen, folytattam a megjelenítő létrehozását. Lépésről lépésre, blokkonként haladtam. Az első ilyen nagy blokk a fejléc volt, melyen az idő kijelzése nem volt megfelelő eleinte, ugyanis mást mutatott a digitális és az analóg óra. Egy kis kutakodás után észrevettem, hogy az eltérés, a PLC helytelen konfigurálásából ered, ugyanis az időzóna nem volt megfelelően beállítva. Miután ezt orvosoltam, a két idő megegyezett.

Ezt követően a mért értékek kijelzését ellenőriztem. A kijelzett értékek eleinte értelmezhetetlenek voltak. Ez amiatt volt, mert rosszul állítottam be a kijelzés módját, hiszem floating-point helyett, az alapértelmezett word-ös paraméter maradt aktív. A korrigálás után, már helyesnek tűnő mérési eredmények jelentek meg. Szimuláltam egy üzemállapotot, melyben lekapcsoltam a fogyasztók nagyrészét, ezzel csökkentve az áramfelvételt. A kijelzett áram nagysága lecsökkent, úgyhogy működését megfelelőnek ítéltem meg.

10.4. E-mail küldés

Az email küldés vizsgálatakor, manuálisan változtattam a TMail_C blokk bemeneti paramétereit. Létrehoztam két e-mail fiókot, melyek test@tiauphuadmin.hu és test2@tiauphuadmin.hu címmel szerepeltek. Ezt a két e-mailcímet a címzett és a másolatot kapó címzettnek állítottam be. Miután kitöltöttem a szöveget és a címet, a REQ változónak igaz értéket adtam. Mivel a TMail_C blokk felfutó élre vezérelhető, így csak egy e-mail került kiküldésre. Ezeket a Thunderbird levelezőprogramban láttam.

11. Lehetséges fejlesztések

11.1. Tűzjelzés

Egyik fejlesztési ötletem eredetileg a rendszerterv része volt. Azt szerettem volna megvalósítani, hogy az épületben kiépített tűzvédelmi rendszer jelét hasznosítva, biztonságos üzemállapotba kerüljenek a rack-szekrények. Ehhez egy, a tűzjelzéskor jelet adó kábelre lett volna szükségem, de sajnos az épületüzemeltetés erre nem biztosított lehetőséget, mivel ez veszélyeztette volna a már működő rendszer biztonságát. Biztos vagyok benne, hogy kellő körültekintéssel magoldható lenne ötletem.

11.2. A mért adatok feldolgozása és megjelenítése

A következő fejlesztési javaslatom, az a mért adatok tárolására, feldolgozására és megjelenítésére vonatkozik. Nagyon szemléletes lenne, ha mért adatok egy feldolgozás után grafikonon ábrázolva megjelennének a webes vizualizációs felületen. Ehhez egy trendrajzoló funkcióra lenne szükség, mely nincs lefejlesztve a View of Things eszközkészletébe. Ezen grafikonon nagyon jól nyomon követhető lenne, az áramfelvétel és az energiatakarékos üzemmód hatékonysága és működése. A grafikon mellé elhelyeznék, egy exportáló gombot, mellyel kimutatásokat lehetne kinyerni a rendszerből. Ehhez azonban megfelelő formátumban kellene archiválni a mért adatokat.

11.3. Havi jelentések e-mail formájában

Egy további hasznos funkció lehetne, ha az üzemeltető nem csak sürgős esetekben kapna értesítést, hanem havi jelentések formájában is informálódna a rack-szekrény adatairól. Ezt nagyon könnyen meg lehetne valósítani. Ehhez egy kis program írása szükséges. A PLC-kben lehetőség van úgynevezett „datalog” -ot készíteni, melyet egy adatgyűjtő fájlként tudnék magyarra fordítani. Ez egy .csv kiterjesztésű fájlt jelent, amit könnyen lehet exportálni Excelbe. Az S7-1500-as PLC esetén maximum 1.000.000.000 bájt méretű datalog hozható létre. Természetesen a fájl maximális mérete függ a memóriakártya kapacitásától is, mivel ezeket a fájlokat a memóriakártyára vagy a belső CPU memóriába menti el. A TMail_C lehetőséget ad csatolmányok küldésére is. A .csv kiterjesztésű állományt minden hónap elsején küldeném el e-mail-es csatolmányként a rendszer üzemeltetőjének.

12. Összegzés

Néhány mondatban szeretném összegezni gondolataimat, melyek a szakdolgozat megírása közben fogalmazódtak meg bennem. Úgy gondolom kellő ambícióval álltam neki a megoldandó feladatnak. Ennek köszönhetően elmondható, hogy az elvárásoknak megfelelően és maradéktalanul végeztem el a feladatot. Ahogyan „A megoldandó feladat ismertetése" pontban említettem, a fő célom az volt, hogy egy olyan rendszert tudjak prezentálni munkatársaim számára, mely iránymutató, gondolatébresztő és jó továbbfejlesztési alapot ad. Ugyan a prezentálás még nem történt meg, de remélem, hogy a munkám elnyeri majd kollégáim tetszését.

A megoldandó feladat több kisebb problémakört tartalmazott, melyekre sikerült megoldást találnom. Ezekre külön-külön programrészletet írtam. A programról tehát elmondható, hogy az egyes részek nem bonyolultak, de egészében nézve, egy komplex projektet alkot. Egy kis kihívást okozott, hogy ezeket a programrészeket összehangoljam, ám a végeredménnyel elégedett vagyok.

Megoldottam a rack-szekrény felkapcsolásakor létrejövő magas áramfelvételt, az energiafelhasználás optimalizálást, az értesítés küldést, néhány biztonsági funkciót és a vizualizációt. Ezeket időben eltolt felkapcsolással, időszak vizsgálattal, alarm generálással, safety funkciók alkalmazásával és a View of Things használatával valósítottam meg.

A feladat megoldása során több ismeretlen technológiával találkoztam, melyeknek használata eleinte nehézkes volt, megismerésük után viszont készségszinten tudom ezeket alkalmazni. Véleményem szerint sok hasznos ismerettel gazdagodtam, és sokat fejlődtem a szakdolgozat megírása közben.

Végezetül szeretnék köszönetet mondani az evosoft Hungary Kft. -nek, hogy lehetőséged és témát adott a szakdolgozat megírásához. Továbbá meg szeretném megköszönni a konzulenseim munkáját. Külön köszönet illeti Végh Alex Barnabást, aki szaktudásával, ismeretségeivel és ismereteivel segítette munkámat az elmúlt pár hónapban.

13. Summary

I would like to summarise in a few sentences the thoughts that came to me while writing this thesis. I think I was ambitious enough to tackle the task at hand. As a result, I have completed the task in accordance with expectations. As I mentioned in the "3. A megoldandó feladat ismertetése" section, my main objective was to present a system to my colleagues that would provide guidance and a good basis for further development. Although the presentation has not yet taken place, I hope that my work will be appreciated by my colleagues.

The problem to be solved contained several small problems, which I managed to solve. For each of them I wrote a separate program part. So the program parts are not complicated, but as a whole it is a complex project. It was a bit of a challenge to coordinate all these parts of the program, but I am satisfied with the final result.

I solved the high power consumption when switching on the rack, power consumption optimization, sending notifications, some security features and visualization. I implemented these by time shifted power-up, period analization, alarm generation, safety functions and View of Things.

In the process of solving this task, I encountered several unfamiliar technologies, which were difficult to use at first, but once I got to know them, I was able to use them at a skill level. In my opinion, I have gained a lot of useful knowledge and improved a lot while writing this thesis.

Finally, I would like to thank evosoft Hungary Kft. for giving me the opportunity and the topic to write this thesis. I would also like to thank my advisors for their work. Special thanks to Alex Barnabás Végh, who helped me with his expertise, knowledge and insights during the last few months.

Irodalom jegyzék

[1] - Mondás, mely a híres környezetvédelmi aktivista, David Browertől származik.

[2] - www.ceginformacio.hu/cr9310190891 – evosoft Hungary Kft.

[3] - www.evosoft.hu/kik-vagyunk - Az evosoft weboldala.

[4] - evosoft belső tananyag

[5] - us.profinet.com/profinet-explained/ (megtekintve: 2022.05.02)

[6] - evosoft belső tananyag

[7] - hu.wikipedia.org/wiki/%C3%89p%C3%BCletautomatiz%C3%A1l%C3%A1s (megtekintve: 2022.04.25)

Ábrajegyzés