Dinamička kontrola i monitoring data blokova na Siemens S7-1500 PLC-u uz pomoć s7.NET knjižnice

Vozila, Mateo

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Sveučilišni prijediplomski studij računarstva

Završni rad

Dinamička kontrola i monitoring data blokova na Siemens S7-1500 PLC-u uz pomoć s7.NET knjižnice

Rijeka, rujan 2023.

Mateo Vozila
0069082583

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Sveučilišni prijediplomski studij računarstva

Završni rad

Dinamička kontrola i monitoring data blokova na Siemens S7-1500 PLC-u uz pomoć s7.NET knjižnice

Mentor: prof. dr. sc. Mladen Tomić

Rijeka, rujan 2023.

Mateo Vozila
0069082583

Umjesto ove stranice umetnuti zadatak za završni ili diplomski rad

Izjava o samostalnoj izradi rada

Izjavljujem da sam samostalno izradio ovaj rad.

Rijeka, rujan 2023.

_____

Mateo Vozila

Zahvala

Zahvaljujem Naidu Haskiću na podršci tijekom pisanja ovoga rada i korisnim raspravama i savjetima. Zahvaljujem Mladenu Tomiću na podršci tijekom studiranja.

Sadržaj

Popis slika

Poglavlje 1

Uvod

Tema ovog rada je izrada sustava za kontrolu i monitoring data blokova na Siemens S7-1500[1] i njemu sličnim PLC-ovima. Program je izrađen u sklopu rada za firmu Danieli Systec d.o.o. i namijenjen korištenju u kontekstu područja razvoja softvera kojim se oni bave. Konkretno, program se koristi kao alat za lakše otklanjanje pogrešaka u radnom procesu koji će detaljnije biti opisan u kasnijim poglavljima. Program je napravljen u Visual Studiu koristeći jezik C#[2] i Microsoftov .NET okvir (eng. .NET framework)[3] za sučelje programa. Za komunikaciju sa Siemens S7 PLC uređajem koristi se nezavisno razvijena S7.NET knjižnica[4] u kojoj su sadržane razne metode koje omogućuju spajanje na PLC uređaj, te čitanje i obradu podataka koji su na tom uređaju.

Poglavlje 2

Opis radnog procesa za koji je razvijen program

Danieli Systec d.o.o. je IT firma koja se bavi razvojem softvera za postrojenja u metalnoj industriji, primarno čeličane. Osim samog razvoja softvera koji upravlja strojevima unutar tih postrojenja, Systec također pruža usluge praćenja puštanja postrojenja u pogon jer je gotovo uvijek nužno napraviti određene promjene u programima koji se koriste i, naravno, testirati radi li sve kako treba u stvarnom okruženju. Programska podrška za svako postrojenje dijeli se na 3 razine: Level 1, Level 2 i Level 3. [5] [6]

2.1 Level 1 automatizacija

Level 1 automatizacija u kontekstu industrijske automatizacije predstavlja najnižu razinu hijerarhijske automatizacije. Glavne značajke Level 1 automatizacije su sljedeće:

- Kontrola procesa: glavna zadaća Level 1 automatizacije je kontrola procesa. To se ostvaruje dobivanjem podataka sa senzora, procesiranjem tih podataka i kontrolom kontrolnih uređaja poput motora.
- Kontrolni uređaji: ovaj pojam obuhvaća sve uređaje koji se mogu upravljati sa strane Level 1 softvera. Ti uređaji mogu biti motori, aktuatori, ventili itd.

- PLC: Programabilni logički kontroler (eng. Programmable Logic Controller) je uređaj na kojem se događa većina logike u pozadini Levela 1. PLC se može usporediti sa stolnim računalom po tome da ima I/O module, CPU, memoriju itd. Glavna razlika je ta da PLC uređaji nisu namijenjeni generalnoj upotrebi, već je svaki programiran da bude koristan specifično za okružje u kojem se koristi.[7]
- HMI: Sučelje između čovjeka i stroja (eng. Human Machine Intrerface) koristi se kako bi se ljudima koji rade na procesu koji Level 1 automatizira dao pregled stanja proces i sve komande koje su im potrebne da kroz Level 1 upravljaju procesom. [8]
- Dobavljanje podataka: Podaci se prikupljaju iz raznih senzora kao što su vage, mjerači tlaka, termoparovi itd. Prikupljene informacije se dalje procesiraju u PLC-u i prikazuju na HMI-u.
- Lokalna kontrola: Operateri na Levelu 1 odgovorni su za lokalnu kontrolu unutar određenog područja ili sekcije proizvodnog ili industrijskog postrojenja. Bitno je imati faktor ljudske kontrole i nadzora kako bi se osigurao siguran i efikasan rad sustava.
- Sustav sigurnosti: Na Levelu 1 se također implementira i sustav za siguran rad. U taj sustav je hitno isključivanje sustava, a neke dodatne značajke mogu biti sigurnosni PLC-ovi koji prate i limitiraju rad određenih komponenti sustava kako bi bio unutar granica sigurnog rada.

Level 1 je neizostavan dio automatizacije svakodnevnog rada industrijskih postrojenja i igra ključnu ulogu u osiguravanju da oprema radi unutar specificiranih parametara, postiže proizvodne ciljeve i održava standarde sigurnosti. Podaci prikupljeni na Levelu 1 mogu se koristiti za nadzor i dijagnostiku, optimizaciju procesa i integraciju s automatizacijskim sustavima više razine, poput Levela 2 i Levela 3.

2.2 Level 2 automatizacija

Level 2 automatizacija predstavlja drugu razinu u hijerarhiji automatizacije. Level 2 se fokusira na nadzor i upravljanje višim razinama procesa u industrijskom postrojenju. Glavne značajke Level 2 automatizacije uključuju:

- Nadzor kvalitete i efikasnosti: Level 2 omogućuje praćenje kvalitete proizvedenih dobara i učinkovitosti procesa proizvodnje. Kroz Level 2 operateri mogu pratiti performanse i optimizirati proces.
- Centralizirano upravljanje: kroz Level 2 se operaterima daje mogućnost da s istog mjesta upravljaju različitim dijelovima postrojenja, kao i mogućnost promjene parametara procesa kako bi se prilagodili promjenama u proizvodnji ili okolišu.
- Arhiviranje podataka: Arhiviranje podataka je ključno za analizu povijesti proizvodnje u svrhu učenja i optimizacije procesa proizvodnje.
- Obavijesti i upozorenja: Na Levelu 2 se također prikazuju razne obavijesti o stanju procesa, kao i upozorenja o potencijalnim problemima u procesu proizvodnje.
- Komunikacija s Levelom 1: Sve podatke o trenutnom stanju procesa Level 2 prikuplja od strane Levela 1. Slično tome, Level 2 može Levelu 1 slati podatke za kontrolu procesa koje Level 1 onda koristi na kontrolnim uređajima.

Razina 2 automatizacije igra ključnu ulogu u upravljanju složenim procesima u industrijskim postrojenjima. Omogućuje operaterima da dobiju širi pogled na postrojenje, donesu informirane odluke i pravilno upravljaju proizvodnjom. Osim toga, podaci prikupljeni na razini 2 često se koriste za izvještavanje i analizu u cilju optimizacije proizvodnje i smanjenja troškova.

2.3 Level 3 automatizacija

Level 3 u kontekstu industrijske automatizacije i kontrolnih sustava označava treću razinu u hijerarhiji upravljanja. Poznata i kao "Razina kontrole proizvodnje" (eng.

production control), fokus Levela 3 je šire upravljanje resursima unutar industrijskog postrojenja. Glavne značajke Levela 3 uključuju:

- Planiranje proizvodnje: Na Levelu 3 koriste se sustavi za planiranje proizvodnje (Manufacturing Execution Systems - MES). MES sustavi omogućuju planiranje proizvodnje, raspored proizvodnje, praćenje narudžbi, i upravljanje resursima kao što su radna snaga, materijali i oprema.[9]
- Koordinacija proizvodnje: Level 3 je odgovoran za koordinaciju proizvodnje na razini cijelog industrijskog postrojenja.
- Optimizacija proizvodnje: Level 3 i, konkretnije, MES sustavi omogućuju optimizaciju proizvodnje kroz praćenje performansi procesa, minimizaciju gubitaka i minimizaciju vremena zastoja.
- Upravljanje inventarom: Na Levelu 3 odvija se i praćenje zaliha resursa potrebnih za rad postrojenja, njihova nabava i trošenje.
- Praćenje kvalitete: Level 3 uključuje sustave za kontrolu kvalitete proizvoda i uspoređivanje istih s kvalitetnim standardima.
- Arhiviranje i izvješćivanje: Level 3 može sadržavati i funkcionalnosti za dugoročno arhiviranje podataka i generiranje izvješća o proizvodnji koji se dalje koriste za analizu napretka učinkovitosti radnog procesa.

Level 3 automatizacija velikim dijelom apstrahira sam proces proizvodnje, te se fokusira organizaciju proizvodnje i na rezultate koji su ostvareni tokom iste. Ti rezultati se kasnije analiziraju u svrhu optimizacije procesa proizvodnje, kako na Levelu 3, tako i na nižim levelima.

2.4 Komunikacija između Levela automatizacije

Kako bi gore navedena 3 levela automatizacije mogli funkcionirati adekvatno, bitno je definirati jasan model koji se koristi za komunikaciju između svakog susjednog levela. Ti modeli znaju biti netrivijalni zbog raznih opcija za implementaciju svakog levela i ograničenja koja se javljaju u toj implementaciji. U daljnjim potpoglavljima ću opisati način na koji leveli automatizacije međusobno komuniciraju u okružju za

koji je napravljen ovaj projekt.

2.4.1 Komunikacija između Levela 1 i Levela 2

Komunikacija između Levela 1 i Levela 2 vrši se preko naznačenih data blokova na PLC uređaju. Svaki PLC koji komunicira sa Levelom 2 ima definirane blokove na kojima je omogućeno čitanje u slučaju podataka o stvarnim vrijednostima procesa ili pisanje u slučaju podataka koje Level 2 šalje za kontrolu procesa. Na strani Levela 2, komunikacija se odvija preko određenog broja servisa, svaki sa svojom zadaćom. Komunikacija je modelirana tako da svaki servis može komunicirati sa svakim PLC uređajem i obratno. Zbog ograničenja mogućnosti komunikacije PLC uređaja, koristi se poseban servis koji upravlja svim porukama za čitanje i pisanje na PLC. Na taj način je samo jedan servis uvijek spojen na PLC umjesto velikog broja servisa koji bi se spajali i prekidali vezu nakon čitanja ili pisanja svake poruke. Ovim se pristupom minimizira opterećenje PLC-a, kao i opterećenje industrijske mreže.

2.4.2 Komunikacija između Levela 2 i Levela 3

U usporedbi s gore navedenom komunikacijom između Levela 1 i Levela 2, komunikacija između Level 2 i Levela 3 je gotovo trivijalna. Naime, svi podaci koji se žele razmijeniti između sustava koji upravljaju tim levelima, zapisuju se u dijeljenu bazu ili više baza podataka. Najčešće se koristi takva implementacija gdje Level 2 piše podatke za razmjenu s Levelom 3 u tablice za razmjenu (eng. transfer tables) u vlastitoj bazi podataka iz koje ih Level 3 čita. Level 3 onda radi obratno; zapisuje podatke za razmjenu sa Levelom 2 u Level 3 bazu podataka iz koje ih Level 2 može čitati. Na ovaj način, može se osigurati da Level 2 servisi ne mogu utjecati na Level 3 bazu podataka i obratno.

Poglavlje 3

Postojeće metode za obavljanje sličnih zadataka

Za dani zadatak postoji veći broj pristupa i metoda koje bi se mogle koristiti za njegovo rješavanje. Nešto što sve of njih imaju zajedničko je da se na jedan ili drugi način spajaju na PLC, no sam način spajanja već varira od metode do metode. Osim toga, PLC mora biti konfiguriran tako da dopušta pristup vanjskim programima na određeni dio svoje memorije.

3.1 Metode čitanja podataka sa Siemens S7 PLC-a

Glavne metode koje se koriste za čitanje podataka sa SIEMENS S7 PLC-a uključuju:

- OPC (OLE for Process Control): OPC je standard koji se koristi za komunikaciju između raznih strojeva za automatizaciju i podržan je na Siemens S7 PLC-ovima. Ti PLC-ovi također imaju opciju ponašanja kao OPC server što dopušta vanjskim OPC klijent aplikacijama spajanje i dohvat podataka s PLC-a. [10]
- Profinet (Process Field Network): Profinet je industrijski Ethernet protokol i jedan od najkorištenijih standarda za komunikaciju s i između PLC uređaja. Komunikacija se vrši preko Ethernet sučelja i koristi unaprijed definirana pra-

vila i postavke za komunikaciju. [11]

- S7 Protokol (S7 komunikacija): Siemens PLC-ovi koriste S7 komunikacijski protokol. Taj je protokol razvijen specifično za Siemens PLC-ove od strane proizvođača. Mogu se koristiti programski jezici poput C#, Pythona ili knjižnica kao što je Snap7 za uspostavljanje veze s PLC-om i čitanje podataka putem ovog protokola. [12]
- HMI/SCADA softver: HMI/SCADA softver je softver za sučelje između čovjeka i stroja (eng. Human Machine Interface) ili softver za nadzor i akviziciju podataka (eng. Supervisory Control and Data Acquisition). Programi koji primjenjuju ovaj pristup koriste knjižnice za komunikaciju s PLC uređajem i podatke s tog uređaja prikazuju na grafičkom sučelju.
- Siemens TIA Portal: TIA (eng. Totally Integrated Automation) je program razvijen od strane proizvođača (Siemens) i koristi se za razvoj PLC softvera. Osim toga, u njemu su dostupni i alati za testiranje komunikacije, kao i alati za čitanje neobrađenih podataka koji se nalaze u data blokovima PLC-ova. [13]
- Modbus TCP/IP: za Siemens PLC-ove koji podržavaju Modbus TCP/IP, mogu se koristiti biblioteke ili softver kompatibilan s Modbusom za čitanje podataka putem TCP/IP mreže. Modbus je često korišten protokol u industrijskoj automatizaciji.

Ostale metode koje se koriste u ovu svrhu uključuju: Siemens Simatic WinCC, razni web servisi i razni načini prilagođeni specifičnim situacijama.

3.2 S7.NET knjižnica

Za ovaj je projekt, uz konzultaciju s kolegama, odabrana S7.NET C# knjižnica koja s PLC-om komunicira preko njegovog Profinet sučelja. Za spajanje je potrebna IP adresa spojenog sučelja, te brojevi rack-a i slot-a za taj PLC. Glavne mogućnosti koje omogućava ta knjižnica su spajanje na PLC i funkcije za čitanje i pisanje u data blokove PLC-a. Za olakšavanje obrade podataka koji su učitani s PLC-a, S7.NET također podržava pretvorbu pročitanih podataka koji se svi dohvaćaju u obliku polja

bajtova. Podržani tipovi podataka uključuju bool, byte, word, dword, int, real, itd. Za potrebe ovog problema dovoljno je bilo koristiti pretvorbu za cijele brojeve (int), decimalne brojeve (float) i nizove znakova (string), dok ostali tipovi podataka nisu korišteni.

Poglavlje 4

Opis razvijenog programa

Projekt je razvijen u Visual Studiju, koristeći C# jezik, Windows Forms[14] i .NET framework. Razvijeni program je koristan u procesu otklanjanja pogrešaka koje nastaju tijekom rada jer omogućava programeru jednostavan pogled direktno u data blokove samih PLC uređaja. Tim se pristupom zaobilazi dispatcher program i tako lakše određuje gdje točno dolazi do pogreške. Svojstva razvijenog sustava uključuju:

- Čitanje varijabli iz odabrane poruke na PLCu
- Kontinuirano praćenje većeg broja varijabli koje se nalaze unutar iste poruke
- Pretvorba .xls datoteke u koju su zapisane informacije o PLC uređajima u .json datoteku koja je znatno više responzivna i sadrži samo one informacije koje su korisne za program
- Dinamičko čitanje strukture poruka na PLCu iz izrađene .json datoteke

4.1 Alati korišteni u izradi programa

U ovom su dijelu dani kratki opisi za glavne alate koji su korišteni u izradi programa.

4.1.1 Visual Studio

Visual Studio je integrirano razvojno okruženje (IDE) koje se koristi za razvoj softvera. Razvijen je od strane Microsofta i podržava različite popularne programske jezike poput C#, C++, JavaScript, Python i druge. [15]Glavne značajke Visual Studia su:

- Integrirani debugger - Debugger je alat koji programeru omogućava jednostavno pronalaženje i uklanjanje grešaka u kodu.
- Integracija s Azure Cloud-om - Microsoft Azure je cloud računalna platforma i set usluga koje pruža Microsoft. Glavne komponente azure-a su virtualne mašine, skladištenje podataka, DevOps sustav za verzioniranje koda, i mnoge druge.
- Alati za analizu koda i produktivnosti - Visual studio pruža i alate koje se mogu koristiti za analizu kvalitete koda i optimizaciju razvojnog procesa.
- Alati za verzioniranje koda - Pruža se i integrirana sučelja za alate za verzioniranje koda poput Git-a, TFS-a ili Azure DevOps-a.

Visual Studio je jedan od najrašireniji razvojnih okruženja u industriji razvoja softvera i standardno razvojno okruženje koje je korišteno pri izradi ovog zadatka.

4.1.2 C#

C# (C-sharp) je programski jezik razvije od strane Microsofta i jedan od glavnih jezika koji se koristi u okviru Microsoftove platforme za razvoj softvera. C# je koristan za razvoj aplikacija namijenjenim raznim platformama poput web aplikacija, kao i desktop ili mobilnih aplikacija. Za razliku od njegovog pretka, C jezika, C# je visokog nivoa (eng. high level), objektno orijentiran i posuđuje neke elemente od

C++ i Java jezika, kao i brojne originalne karakteristike i sintaksu. Glavne značajke C#-a su sljedeće:

- Objektno orijentiran jezik - C# je objektno orijentiran jezik što znači da je kod organiziran u objekte u kojima su podaci i metode za rad s tim podacima.
- Upravljanje memorijom: C# koristi sustav za automatsko prikupljanje smeća (eng. garbage collector). Taj sustav olakšava upravljanje memorijom i oslobađa programera potrebe za ručnim upravljanjem memorijom.
- Velika standardna knjižnica - C# dolazi sa standardnom knjižnicom bogatom gotovim komponentama i funkcionalnostima uz pomoć koje se znatno olakšava razvoj aplikacija.
- Tipiziran jezik - C# je strogo tipiziran jezik. To znači da sve varijable moraju biti deklarirane sa željenim tipom podatka koji se neće mijenjati tijekom izvođenja programa.
- Multithreading - C# podržava multithreading, što omogućava istovremeno izvođenje više dretvi (eng. Threads) kako bi se poboljšale performanse aplikacije.
- Sintaksa - Sintaksa C#-a je relativno jednostavna i lako čitljiva, što smanjuje razinu znanja potrebnu za početak učenja i korištenja jezika.

Za razvoj aplikacija na različitim platformama, C# koristi razne tehnologije kao što su Windows Forms za Windows desktop aplikacije, ASP.NET za Web aplikacije, ili Xamarin za aplikacije namijenjene mobilnim uređajima.

4.1.3 Windows forms i .NET okvir

Windows Forms, poznat kao i WinForms je dio .NET platforme koja se koristi za razvoj desktop aplikacija za Windows operacijski sustav. WinForms omogućava kreiranje aplikacija s grafičkim sučeljem (GUI) i pruža brojne alate za poboljšavanje rada i korisničkog iskustva razvijenog programa.

WinForms se razvija u C# jeziku unutar Visual Studio razvojnog sučelja. Koristi .NET okvir (eng. Framework) kao temeljnu knjižnicu klasa. U tom su okviru sadr-

žane razne komponente poput prozora, polja za unos teksta, padajućih izbornika i slično. To omogućava brz i efikasan razvoj grafičkog korisničkog sučelja. WinForms također podržava rukovanje događajima (eng. event handling) što znači da program može pratiti događaje poput micanja miša, klikanja, unosa teksta ili odabira neke opcije u izborniku i odmah reagirati na te događaje. WinForms je, kako i samo ime nalaže, napravljen za rad u Windows okruženju. Aplikacije razvijene u tom okviru su obično optimizirane tako da zahtijevaju potpuni pristup resursima Windows operativnog sustava.

4.2 Pretvorba excel datoteke s informacijama o PLC-u u JSON format

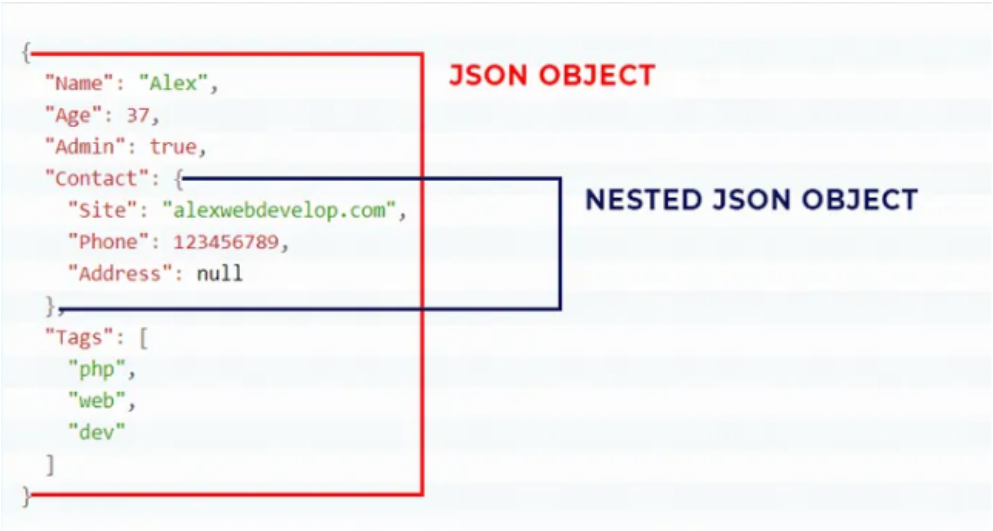
Kao što je ranije rečeno, podaci o rasporedu poruka na PLC-u, kao što su njihovih adresa na PLC-u, tip podatka koji je na toj adresi i slično, su navedeni u excel datoteci koja je dio dokumentacije za svaki PLC uređaj na kojem se radi. Iako je relativno jednostavno čitati podatke direktno iz excel tablice, taj se pristup pokazao lošim jer čitanje iz excel datoteke traje daleko predugo, pogotovo kad je riječ o velikom broju redova i radnih listova koji se čitaju. Kako bi se optimizirao proces dohvaćanja podataka o porukama i varijablama iz Excela, odlučeno je razviti sučelje koje uzima sve važne informacije iz excel datoteke i zapisuje ih u JSON datoteku. JSON format zauzima manji prostor na disku i radnoj memoriji, ali važnije od svega, puno brže se pristupa informacijama u tom formatu. Na ovaj je način dovoljno jednom pročitati Excel datoteku, obraditi podatke u njoj i spremiti ih u JSON datoteku iz koje je čitanje puno jeftinije u pogledu utrošenog vremena.

4.2.1 Što je JSON?

JSON (JavaScript Object Notation) je jedan od standardnih formata za razmjenu podataka između računalnih sustava. [16] Radi se o tekstualnom formatu koji se koriti za pohranjivanje objekata i lista podataka tako da su ti podaci lako čitljivi i ljudima i aplikacijama koje ga koriste. Dvije osnovne strukture podataka koje JSON koristi su:

- Objekti - Objekti su skupine parova ključ - vrijednost u kojem je za svaki ključ definirana vrijednost koju on sadrži. Vrijednost mogu biti gotovo bilo koje vrste standardnih tipova podataka poput cijelih brojeva, znakovnih nizova ili boolean vrijednosti. Osim toga, vrijednost nekog ključa može biti i cijeli JSON objekt s vlastitim ključevima i vrijednostima. Primjer objekta može se vidjeti u slici 4.1.
- Nizovi (eng. arrays) - nizovi su uređene liste podataka koje, poput objekata, mogu sadržavati razne vrste podataka. Ono po čemu se razlikuju od objekata

je to da u nizovima ne postoje parovi ključ - vrijednost, već su sve vrijednosti zapisane u uređenoj listi i, umjesto ključa, pristupaju se koristeći njihov index u listi. Primjer niza može se vidjeti na slici 4.2.



Slika 4.1 Primjer JSON objekta

```
[
    "Niz znakova",
    1234,
    true
]
```

Slika 4.2 Primjer JSON niza

4.2.2 Format Excel datoteke

Prvi radni list u Excelu je Table Of Contents (TOC) na kojem se nalazi velik broj informacija, od kojih je za ovaj projekt bitno samo imena svih poruka, broj data blocka na kojem se nalaze, njihova veličina u bajtovima i brzina sata po kojem se ažuriraju. Osim TOC radnog lista, bitni su radni listovi za svaku poruku koja se šalje. Iz svakog od tih listova, izvlače se imena varijabli, tip podatka, te njihove

adrese, veličine i minimalne/maksimalne dopuštene vrijednosti. Primjer tablice koja sadrži informacije o varijablama unutar poruke može se vidjeti na slici 4.3.

		CIRCULAR BUFFER DATA STRUCTURE		5							124	
DBD 0012		Input Coil L2 Identifier	DI							00	A	4	12
DBB 0016		Input Coil L1 Identifier	MS	20						00	A	20	16
DBD 0036		Pass Number	DI		1	15				00	A	4	36
DBD 0040		Last Pass Flag	DI		0	2		0 = not the last pass 1 = the last pass		00	A	4	40
DBD 0044		Last Rolling Pass Flag	DI		0	1		0 = not the last rolling pass 1 = the last rolling pass		00	A	4	44
DBD 0048		Disabled Stand Flags	DI					bitmask see note #1 0 = enabled 1 = disabled	X	00	A	4	48
DBD 0052		Uncoiler ID	DI						X	00	A	4	52
DBD 0056		Recoiler ID	DI						X	00	A	4	56
DBD 0060		Exit Coil Spool Type	DI		0	1		0 = No 1 = Yes		00	B	4	60
DBD 0064		Entry Operation Mode	DI							00	A	4	64
DBD 0068		Exit Operation Mode	DI							00	A	4	68
DBD 0072		Stand Operation Mode	DI	1						00	A	4	72
DBD 0076		HGC Control Mode	DI	1				0: Undefined 1: Gap Mode 2: Force Control		00	A	4	76
DBD 0080		AGC Primary Regulator Mode	DI	1				1: position 2: tension 3: speed		00	A	4	80
DBD 0084		AGC Secondary Regulator Mode	DI	1				1: position 2: tension 3: speed		00	A	4	84
DBD 0088		AGC Tertiary Regulator Mode	DI	1				1: position 2: tension 3: speed		00	A	4	88
DBD 0092		Rolled Length	FP	1			ft			00	A	4	92
DBD 0096		Spare	FP	10					X	00	A	40	96

Slika 4.3 Primjer tablice koja sadrži informacije o varijablama

4.2.3 Pretvorba i format JSON datoteke

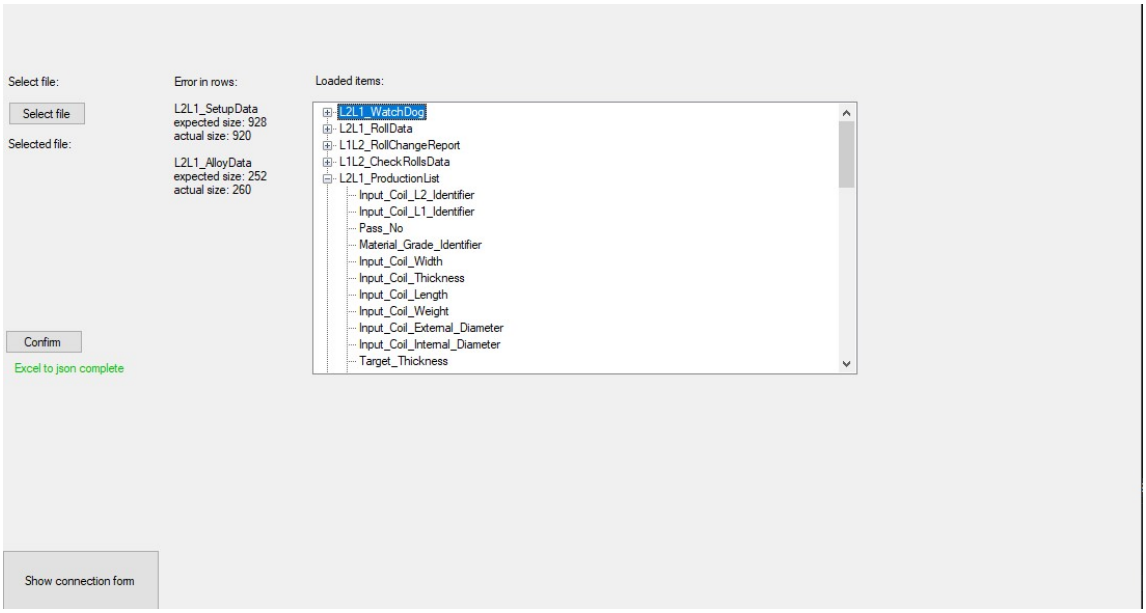
JSON datoteku je, nakon većeg broja isprobanih opcija, odlučeno projektirati kako je opisano na slici 4.4. Ovakav format omogućava jednostavno i intuitivno snalaženje kroz sve evente, njihove karakteristike, kao i karakteristike njihovih varijabli. Koristeći se C# "foreach" petljom, postaje trivijalno prolaziti po svakome od evenata, i nije nužno unaprijed znati koliko ih točno ima niti kakve su strukture.

Slika 4.4 Struktura JSON datoteke

4.2.4 Korisničko sučelje za parsiranje Excel datoteka

Kako je odabir odgovarajuće JSON datoteke jedan od osnovnih koraka za korištenje programa, razvijeno je korisničko sučelje koje krajnjem korisniku olakšava pretvorbu Excel datoteke u JSON. Izgled prozora prikazan je na slici 4.5. Dugme "Select file" otvara prozor za pretraživanje datoteka u kojemu se može odabrati Excel datoteka koju želimo pretvoriti u JSON format. Nakon odabira Excel datoteke, pritiskom na dugme "Confirm", počinje parsiranje podataka iz Excel datoteke u JSON datoteku. Zbog vremenski zahtjevnog procesa pristupa velikom broju ćelija u Excel datoteci, proces parsiranja i zapisa u JSON datoteku traje do par minuta, no to se mijenja s promjenom veličine Excel datoteke. Tako dugo vrijeme čitanja je i primarni razlog pretvorbe, jer kada su podaci sadržani u JSON formatu, vrijeme pristupa istima

je neprimjetno. Nakon parsiranja, na desnoj strani prozora, generira se popis svih pronađenih grešaka u formatu Excel datoteke, kao i popis svih parsiranih poruka i varijabli sadržanih u tim porukama. Pritiskom na "Show connection form" dugme, program se vraća na prozor za spajanje s PLC uređajem.



Slika 4.5 Izgled korisničkog sučelja za parsiranje Excel datoteka

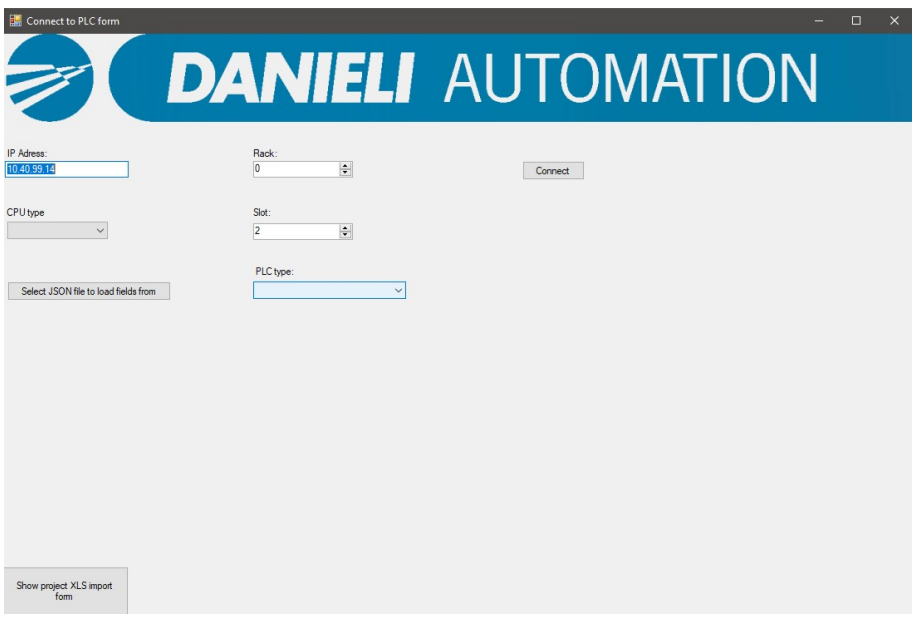
4.3 Sučelja za spajanje na PLC uređaj i čitanje podataka s istog

4.3.1 Korisničko sučelje za spajanje na PLC uređaj

Nakon generiranja JSON datoteke koja sadrži informacije o podacima na PLC uređaju, dolazimo do grafičkog sučelja za spajanje na PLC. Sučelje izgleda kao na slici 4.6.

U polje "IP Address" mora se unijeti IP adresa onog sučelja PLC uređaja na koji je računalo na kojem se pokreće program spojeno. "Rack" i "Slot" polja su također

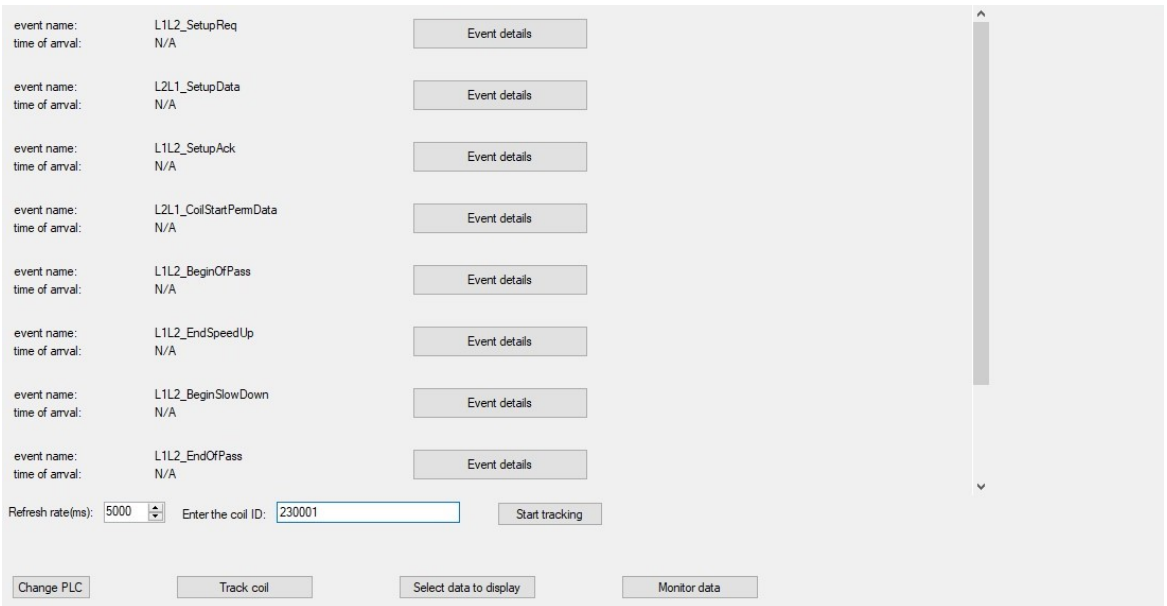
nužna za definiranje točno kojem uređaju se pristupa. Rack i slot su parametri koji se koriste kako bi se označili logički aspekti PLC sustava. Rack predstavlja fizički okvir na kojem se nalazi PLC uređaj na koji se pokušavamo spojiti, a slotovi su fizička mjesta unutar rack-a u koja se umeću moduli za PLC uređaje, kao i sami PLC uređaji. "CPU type" padajući izbornik služi za odabir vrste CPU-a koji odabrani PLC koristi. Trenutno podržani CPU tipovi su: S7 1200, S7 1500, S7 200, S7 200Smart, S7 300, S7 400 i Logo0BA8. Nakon toga, nužno je odabrati JSON datoteku iz koje se žele učitati podaci o memoriji PLC uređaja. Pritiskom na dugme "Select JSON files to load fields from" otvara se pretraživač datoteka iz kojega možemo odabrati JSON datoteku odakle će program iščitati željene podatke. Jedan od podataka su razni tipovi PLC uređaja koji se koriste za ispunjavanje "PLC type" padajućeg izbornika. Kako se u istoj JSON datoteci mogu pohraniti podaci za više od jednog PLC uređaja od koji svaki ima vlastite event-e i varijable, u ovom izborniku se može odabrati koju točno konfiguraciju PLC-a želimo koristiti. Nakon što su sva polja u ovom obrascu ispunjena, moguće je spajanje na PLC uređaj pritiskom na tipku "Connect". Pri uspješnom spajanju, otvara se prozor za sučeljem za čitanje podataka s PLC uređaja, a ako u spajanju dođe do greške, na ovom obrascu se pokaže poruka greške (eng. Error message) i moraju se popraviti parametri za spajanje.



Slika 4.6 Izgled korisničkog sučelja za spajanje na PLC

4.3.2 Korisničko sučelje za čitanje podataka s PLC uređaja

Korisničko grafičko sučelje za čitanje podatak s PLC uređaja vidljivo je na slici 4.7. Sučelje je podijeljeno na 3 načina čitanja podataka s PLC uređaja. Donji red gumba koji se vide na slici služi za navigaciju između načina čitanja, kao i ostatka programa. Prvi gumb, "Change PLC", zatvara vezu s trenutnim PLC uređajem i vraća korisnika na prethodni prozor za spajanje S PLC-om. Ostala 3 gumba svaki otvaraju sučelja za vlastiti način čitanja i obrade podatak koji su detaljnije opisani u sljedećim paragrafima.



Slika 4.7 Izgled korisničkog sučelja za čitanje podataka s PLC uređaja

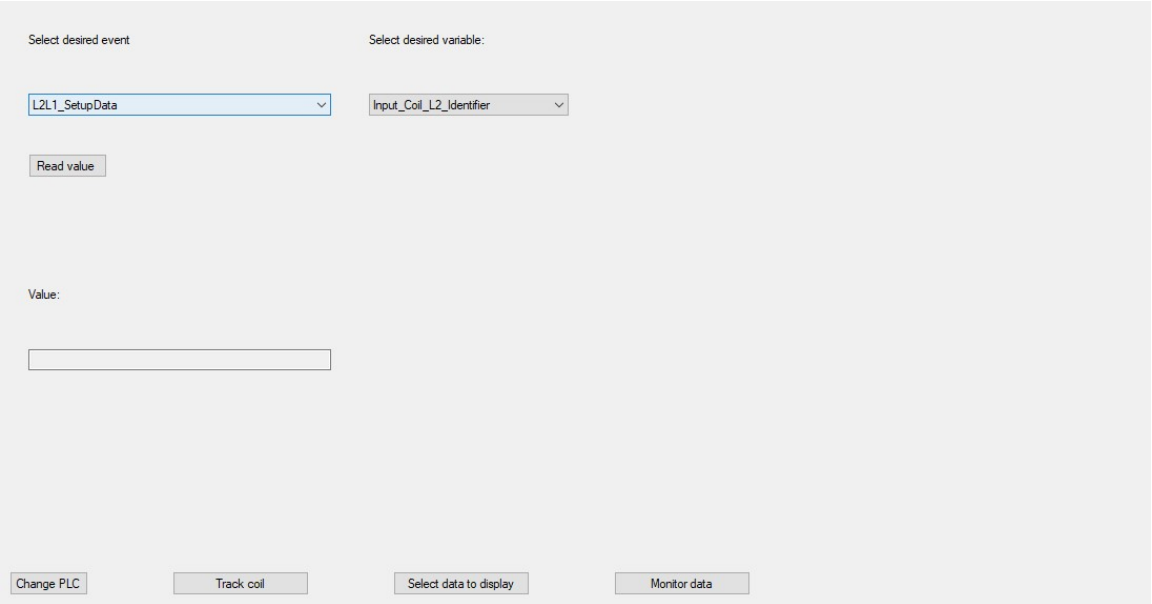
Prvi način, koji se također može vidjeti na slici 4.7, je praćenje coil-a. Coil je proizvod koji se proizvodi na strojevima kojima upravljaju neki od PLC uređaja za koje je dizajniran ovaj program. Sve poruke na PLC-u koje sadrže podatke vezane uz specifičan coil, kao dio svoje strukture imaju coil ID što je jedinstveni identifikator proizvoda. Koristeći taj identifikator, moguće je pratiti u kojem je trenutku došlo do kojeg događaja u proizvodnom procesu. Na ovaj se način jednostavno da vidjeti

ako su podaci zapisani u PLC dobre kvalitete i ako postoji neka greška u servisima Levela 2 koja uzrokuje netočno čitanje ili procesiranje Level 1 podataka. Na dnu slike su vidljivi izbornici za unos stope osvježavanja i odabir identifikatora coil-a. Nakon što su ta polja ispunjena, pritiskom na gumb "Start tracking" počinje kontinuirano čitanje poruka koje sadrže Coil ID varijablu i kada je vrijednost te varijable jednaka vrijednosti koja je upisana u "Enter coil ID" polje, cijeli sadržaj poruke sprema u memoriju i prikazuje u odgovarajućem polju za tu poruku. Za svaku poruku u listi se na korisničkom sučelju bilježi "time of arrival" gdje se prikazuje najnovija pojava traženog coil identifikatora za tu poruku. Pritiskom na gumb "Event detail", otvara se pop-up prozor u kojemu su navedena imena svih varijabli u odabranoj poruci i njihove vrijednosti u zadnjoj spremljenoj instanci.

Drugi način za čitanje podataka je brzo dohvaćanje vrijednosti točno jedne varijable i pristupa mu se pritiskom gumba "Select data to display". Izgled tog prozora vidljiv je na slici: Prozor se sastoji od dva padajuća izbornika za odabir varijable, gumba "Read value" i polja u koje program zapisuje iščitanu vrijednost. Prvi padajući izbornik je popunjen imenima evenata iz JSON datoteke koja je učitana prije spajanja. Nakon odabira vrijednosti za prvi padajući izbornik, program u drugi padajući izbornik dodaje sve varijable koje su definirane za event odabran u prvom izborniku. Pritiskom na "Read value" gumb, program vrši jednokratno čitanje adrese na PLC uređaju koja je povezana s odabranom varijablom. Ideja ovog načina čitanja je da se korisniku omogući najjednostavniji i najbrži način dohvaćanja one vrijednosti koja ga zanima.

Finalni način čitanja podataka zapisanih u PLC-u koji je implementiran u ovom programu je kontinuirano praćenje odabranih varijabli ili elemenata. Pristupa mu se pritiskom na gumb "Monitor data". Korisničko sučelje za taj način čitanja podataka može se vidjeti na slikama 4.9 i 4.10.

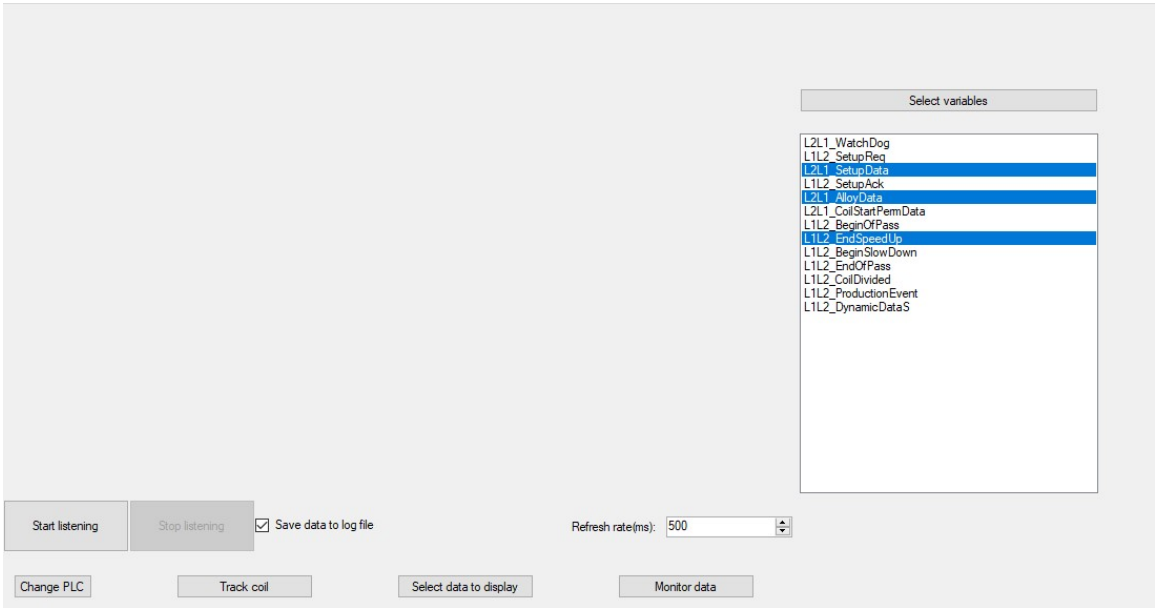
Slično kao u prethodnom načinu, eventi se učitavaju iz JSON datoteke i mogući su za odabir. Za razliku od drop-down izbornika koji je korišten u onoj situaciji, ovdje je korišten izbornik tipa list box u kojem je moguć odabir većeg broja željenih elementa. Nakon odabira poruke iz kojih želimo čitati, pritiskom na "Select variables"



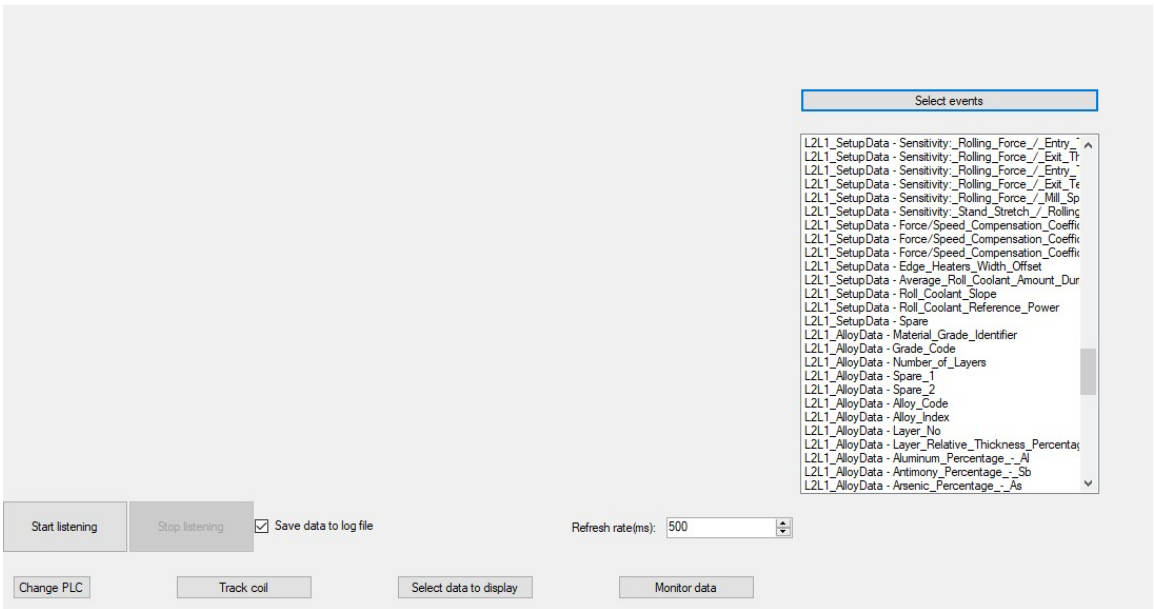
Slika 4.8 Izgled "Select data to display" prozora

gumb u gornjem desnom kutu grafičkog sučelja, generira se novi list box izbornik koji sadrži sve varijable koje sadrže eventi odabrani u prethodnom izborniku. Ovdje je također planiran odabir većeg broja elemenata. Ostale stvari koje utječu na čitanje i spremanje podataka su polje "Refresh rate" koje, isto kao u prvom načinu čitanja određuje učestalost čitanja i potvrdni okvir (eng. checkbox) koji određuje spremaju li se pročitani podaci u log datoteku ili ne. Čitanje počinje pritiskom na "Start listening" gumb i od tog su trenutka sve odabrane varijable, skupa s njihovim vrijednostima koje su pročitane s PLC uređaja, prikazane u srednjem djelu sučelja.

Osim prikaza trenutnih vrijednosti označenih varijabli, ako je opcija "Save data to log file" označena, sve pročitane vrijednosti će se također trajno spremati u log datoteku. Ova je opcija pogotovo korisna kada se čitanje odvija više puta u sekundi jer onda korisnik ne stigne analizirati željene podatke na vrijeme. Struktura log datoteka jednaka je podacima koji se prikazuju na "Monitor data" stranici i može se vidjeti na slici 4.11.

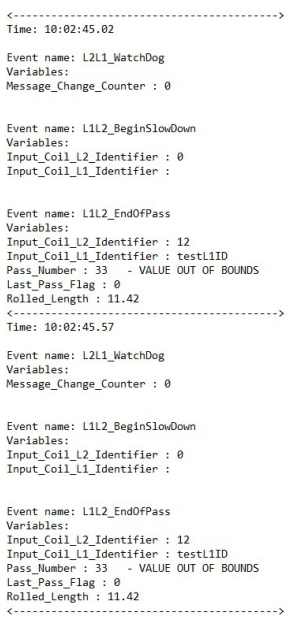


Slika 4.9 Izgled "Monitor data" prozora pri odabiru poruka za čitanje



Slika 4.10 Izgled "Monitor data" prozora pri odabiru varijabli za čitanje

Iz slike se također da primijetiti da program označuje sve vrijednosti varijabla koje su izvan granica definiranih u Excel datoteci za PLC uređaj iz kojeg se čitaju podaci.



Slika 4.11 Primjer log datoteke

Poglavlje 5

Programska pozadina razvijenog programa

Ovo poglavlje služi kao detaljniji opis programske implementacije i procesa izrade razvijenog programa.

5.1 Implementacija korisničkog sučelja

Korisničko sučelje je razvijeno koristeći Windows Forms okvir. Glavne značajke tog okvira koji je korišten u ovom programu su elementi za prikaz podataka poput oznaka (eng. label) i tekstni okviri. Primjer inicijalizacije oznaka koji se može naći u kodu skupa sa tekstnim okvirom je prikazan na slici 5.1. Oznake su korištene za prikaz informacija bitnih za korištenje programa kao na primjer, što znači koje polje u obrascima ili ako je došlo do neke pogreške. Tekstni okviri, s druge strane, uglavnom služe za prikaz podataka koje korisnik čita iz PLC uređaja. Ovisno o namjeni, tekstni okviri se mogu koristiti i za prikaz, ako i za unos informacija, te je u kodu definirano može li korisnik unositi podatke u taj tekstni okvir, ili služi samo za prikaz informacija.

```
private void InitializeComponent()
{
    this.con_failed_label = new System.Windows.Forms.Label();
    this.CPU_type_label = new System.Windows.Forms.Label();
    this.slot_label = new System.Windows.Forms.Label();
    this.rack_label = new System.Windows.Forms.Label();
    this.IP_textbox = new System.Windows.Forms.TextBox();
    this.IP_label = new System.Windows.Forms.Label();
    this.json_file_error = new System.Windows.Forms.Label();
    this.selected_json_file = new System.Windows.Forms.Label();
```

Slika 5.1 Primjer inicijalizacije oznaka i tekstnog okvira

Ostali elementni koji su korišteni u izradi korisničkog sučelja uključuju gumbe, ploče (eng. panel), potvrdne okvire (eng. check box), popisne okvire (eng. list box) i padajući izbornici. Ploče su jedan od osnovnih elemenata grafičkog sučelja i omogućuju jednostavnu podjelu prozora na smislene dijelove. Gumbovi se koriste za potvrdu i slanje informacija unesenih u obrasce, mijenjanje aktivnog načina čitanja i mnoge druge stvari. Potvrdni okviri nisu često korišteni, ali su praktični za jednostavan unos vrijednosti koja se može svesti na točno (eng. true) ili netočno (eng. false). Popisne liste su korisne za odabir više od jednog elementa iz liste definiranih elemenata. One se koriste u sučelju za kontinuirano praćenje gdje je moguć odabir proizvoljnog broja varijabli za čitanje. Zadnji od navedenih elemenata su padajući izbornici koji su i općenito najpopularniji izbor za odabir jednog elementa iz unaprijed definirane liste elemenata.
Za jednostavniji rad s grafičkim elementima, korišten je takozvani "design view" u Visual Studiu uz pomoć kojeg se može na lagan i intuitivan način dodavati i mijenjati svojstva elemenata u grafičkom korisničkom sučelju. Najbitnije svojstvo tog alata je pretpregled (eng. preview) gdje Visual Studio automatski generira izgled sučelja onakav kakav će izgledati u vremenu izvođenja programa. Osim toga, nudi opciju povuci i ispusti (eng. drag and drop) za modificiranje ili dodavanje elemenata unutar grafičkog sučelja. Ovdje se također mogu dodati rukovatelji događajima (eng. event handlers) koji će, kada je njihov uvjet ispunjen, pozvati odgovarajuću funkciju u pozadini programa u kojoj će se taj događaj obraditi.

5.2 Implementacija pozadine aplikacije

Pozadinu ove aplikacije, poznatu i kao backend, se može podijeliti na tri smislene cjeline:

1. Obrada podataka korisničkog unosa
2. Obrada excel datoteke s podacima o strukturi varijabli na PLC uređaju
3. Čitanje podataka s PLC uređaja

Svaka od navedenih cjelina je u više detalja opisana u narednim potpoglavljima.

5.2.1 Obrada podataka korisničkog unosa

Obrada podataka korisničkog unosa odnosi se na sav kod i programsku logiku koja služi za obradu podataka unesenih od strane korisnika. Očekivani tok izvođenja programa je takav da korisnik unese odgovarajuće podatke u za to definirana polja u grafičkom sučelju. Nakon toga dolazi do poziva funkcija za obradu unesenih podataka koristeći ranije navedene rukovatelje događajima. Događaji koji mogu uzrokovati poziv funkcije su uglavnom pritisci na gumb za slanje obrasca, ali u nekim situacijama je to preskočeno i odmah po promjeni sadržaja polja unosa, poziva se funkcija za obradu istog. Na ovaj se način poboljšava korisničko iskustvo i pojednostavljuje korištenje za situacije gdje je takav pristup moguć. U funkcijama za obradu unosa se provjerava valjanost unesenih podataka, i ako postoji neka greška u njima, na korisničkom se sučelju pokazuje poruka koja naznačuje tu grešku. Kada uneseni podaci prođu obradu i provjeru valjanosti, pozivaju se daljnje funkcije koje te podatke zapisuju u njima predviđene objekte i rade odgovarajuće zadaće. Primer rukovatelja događajima može se vidjeti na slici 5.2. Iz slike se da primijetiti da se pritiskom na "Start tracking" gumb pokreće ili zaustavlja mjerač vremena s intervalom unesenim u korisničkom sučelju. Oznaka gumba, kao i određeni elementi u korisničkom sučelju se mijenjaju s obzirom na trenutno stanje mjerača vremena.

```
private void start_tracking_btn_Click_1(object sender, System.EventArgs e)
{
    tracking_timer.Interval = (int)sample_interval_nud.Value;
    if (tracking_timer.Enabled == true)
    {
        tracking_timer.Enabled = false;
        start_tracking_btn.Text = "Start tracking";
    }
    else
    {
        tracking_timer.Enabled = true;
        start_tracking_btn.Text = "Stop tracking";
        foreach (CompactEventDisplay compact_control in panel1.Controls)
        {
            compact_control.time_of_arrival_val.Text = "N/A";
            compact_control.evn_details = String.Empty;
        }
    }
}
```

Slika 5.2 Primjer implementacije rukovatelja događajima

5.2.2 Obrada excel datoteke s podacima o strukturi varijabli na PLC uređaju

Strukture excel i JSON dokumenata su opisane u potpoglavlju 4.2 ovog dokumenta. Ovdje gledam detaljnije opisati programsku implementaciju pretvorbe jednog formata u drugi. Za čitanje excel datoteka korištena je knjižnica "Microsoft.Office.Interop.Excel" [17] razvijena od strane Microsofta. Ona sadrži sve nužne alate koji su korišteni za čitanje podataka sadržanih u excel datotekama. Korištene funkcionalnosti uključuju otvaranje excel datoteke u radnu memoriju računala, dohvaćanje imena radnih listova i dohvaćanje vrijednosti zapisanih u ćelijama. Nakon što korisnik odabere valjanu datoteku putem grafičkog sučelja, program je otvara i počinje parsirati podatke sadržane u njoj. Prva stvar koja se dohvaća je Table Of Contents (TOC) iz prvog radnog lista excel datoteke. Iz njega se učitavaju imena poruka čiji su detalji pohranjeni u daljnjim radnim listovima, jedan za svaku poruku. Po učitavanju svakog reda u TOC, program otvara radni list s naslovom jednakim nazivom poruke u TOC. Iz tih se radnih listova dalje čitaju najprije općenite informacije za tu poruku poput

normalnog intervala čitanja i dužine poruka, a nakon toga se obrađuju podaci o varijablama te poruke. Svi pročitani podaci koji su korisni za kasnije čitanje s PLC uređaja se formatiraju tako da su u skladu s korištenim JSON formatom i zapisuju u privremenu varijablu koja se na kraju čitanja poruke zapisuje u JSON datoteku. Napredak obrade excel datoteke prikazuje se na grafičkom sučelju i po završetku se također prikazuju sve učitane poruke i njihove varijable, kao i moguće pogreške do kojih je došlo tijekom čitanja.

5.2.3 Čitanje podataka s PLC uređaja

Prije početka čitanja podataka s PLC uređaja, program mora učitati valjani JSON dokument i uspostaviti vezu do odabranog uređaja. Koristeći "Plc.Open ()" funkciju program pokušava uspostaviti vezu s PLC uređajem na IP adresi, rack-u i slotu-u koje je korisnik unio u sučelju za spajanje na PLC. Neuspjeh ovog pokušaja uzrokuje prikaz poruke o grešci, a uspješno spajanje otvara prozor za čitanje podataka s PLC-a. Detalji prikaza pročitanih podataka variraju ovisno o načinu čitanja, ali sam proces dobavljanja podataka s PLC uređaja ostaje uglavnom nepromijenjen. Najprije se iz JSON dokumenta čitaju detalji za poruku ili poruke koje se čita. Iz tih detalja se računa duljina poruke koju program želi pročitati s PLC uređaja. Pozivom funkcije "Plc.ReadBytes ()" čiji su argumenti:

- Tip memorije iz koje se čita. Za potrebe ovog programa to je uvijek data blok, ali knjižnica podržava i čitanje iz brojača, memorije PLC uređaja i slično.
- Redni broj data bloka iz kojeg se čita koji je definiran u JSON datoteci.
- Memorijske adrese početka i kraja čitanja unutar tog data bloka. Također zapisane u JSON datoteci, bitno je minimizirati memoriju koja se čita kako bi se poboljšale performanse programa i smanjilo opterećenje mreže.

Funkcija vraća polje bajtova koje se, koristeći podatke o strukturi poruke, parsira u odgovarajući tip podatka, te prikazuje na korisničkom sučelju.

Poglavlje 6

Zaključak

Razvijena aplikacija pokazala se uspješnom u obavljanju zadatka za koji je napravljena, te se koristi u radnom procesu testiranja komunikacije sa PLC uređajima kako je i planirano. Performanse aplikacije su zadovoljavajuće i u mogućnosti je čitati podatke s više data blokova na PLC-u i do 20 puta u sekundi bez osjetnog opterećenja na mrežu ili sam PLC. Testiranje aplikacije moguće je korištenjem datoteka sadržanim u prilogu A. Osim tih dokumenata, za testiranje funkcionalnosti čitanja podataka s PLC uređaja, potreban je i jedan od podržanih PLC uređaja ili njegov simulator.

Bibliografija

[1] Siemens s7-1500 plc dokumentacija. , s Interneta, https://support.industry.siemens.com/cs/document/109742691/documentation-for-the-automation-system-s7-1500-and-the-et-200mp-distributed-i-o-system?dti=0&lc=en-MW

[2] C#. , s Interneta, https://learn.microsoft.com/en-us/dotnet/csharp/

[3] .net framework. , s Interneta, https://learn.microsoft.com/en-us/dotnet/framework/

[4] S7.net knjižnica. , s Interneta, https://github.com/S7NetPlus/s7netplus

[5] Levels of industrial automation. , s Interneta, https://instrumentationtools.com/five-levels-in-industrial-automation/

[6] Automation in steel industry. , s Interneta, https://www.ispatguru.com/automation-in-steel-industry/

[7] What is a plc. , s Interneta, https://www.mroelectric.com/blog/what-is-a-plc/

[8] What is hmi. , s Interneta, https://www.copadata.com/en/product/zenon-software-platform-for-industrial-automation-energy-automation/visualization-control/what-is-hmi/

[9] Manufacturing execution system. , s Interneta, https://en.wikipedia.org/wiki/Manufacturing_execution_system

[10] Opc. , s Interneta, https://www.techopedia.com/definition/1307/ole-for-process-control-opc

[11] Profinet explained. , s Interneta, https://us.profinet.com/profinet-explained/#:~:text=PROFINET%20is%20an%20open%20Industrial,well%2Dadopted%20Industrial%20Ethernet%20solution.

[12] S7 protocol. , s Interneta, https://wiki.wireshark.org/S7comm

[13] Tia portal. , s Interneta, https://support.industry.siemens.com/cs/document/65601780/tia-portal-an-overview-of-the-most-important-documents-and-links-controller?dti=0&lc=en-HR

[14] Windows forms documentation. , s Interneta, https://learn.microsoft.com/en-us/dotnet/desktop/winforms/?view=netdesktop-7.0

[15] visual studio documentation. , s Interneta, https://learn.microsoft.com/en-us/visualstudio/windows/?view=vs-2022

[16] Json. , s Interneta, https://en.wikipedia.org/wiki/JSON

[17] Microsoft office excel knjižnica. , s Interneta, https://learn.microsoft.com/en-us/dotnet/api/microsoft.office.interop.excel?view=excel-pia

Sažetak

Motivacija za izradu projekta bila je razvoj aplikacije koja će se koristiti kao alat za olakšavanje otklanjanja pogrešaka u komunikaciji između Siemens S7 PLC uređaja i Windows server servisa. Aplikacija je razvijena potpuno u C# jeziku koristeći .NET okvir i Windows Forms. Za čitanje podataka sa Siemens S7 PLC-ova koristi se S7.NET knjižnica i ethernet veza s uređajem. Aplikacija automatski, koristeći dokumentaciju o PLC uređaju, pretvara podatke iz data blokova PLC-a u tip podatka naveden u učitanoj dokumentaciji. Mogući su različiti načini čitanja podataka, ovisno o potrebi korisnika u tom trenutku.

Ključne riječi **— C#, PLC, .NET, Windows Forms, S7.NET**

Abstract

The motivation for creating the project was the development of an application that will be used as a tool to facilitate the debugging of communication between Siemens S7 PLC devices and Windows server services. The application was developed entirely in the C# language using the .NET framework and Windows Forms. To read data from Siemens S7 PLCs, the S7.NET library and an ethernet connection to the device are used. The application automatically, using the documentation about the PLC device, converts the data from the data blocks of the PLC into the type of data specified in the loaded documentation. Different ways of reading the data are possible, depending on the needs of the user at that moment.

Keywords **— C#, PLC, .NET, Windows Forms, S7.NET**

Dodatak A

Izvorni kod razvijenog projekta