YRKESHÖGSKOLAN
NOVIA

Styrsystem för hållfasthetstestanläggning

Karl-Petter Söderman

Examensarbete för ingenjörs (YH)-examen

El- och automationsteknik

Vasa 2024

EXAMENSARBETE

Författare:	Karl-Petter Söderman
Utbildning och ort:	El- och Automationsteknik, Vasa
Inriktning:	Automation
Handledare:	Roger Mäntylä, Steve Ekström

Titel: Styrsystem för hållfasthetstestanläggning

Datum: 24.3.2024 Sidantal: 43 Bilagor: 1

Abstrakt

Syftet med detta examensarbete var att utveckla ett fungerande och pålitligt styrsystem för en anläggning som testar hållfastheten i olika balk- och ramkonstruktioner. Dessa testobjekt belastas upprepade gånger och data loggas av systemet kontinuerligt under testets gång. Målet med den data som samlats in är att det skall vara möjligt att beräkna livslängden på konstruktionen för att få en uppfattning om när materialet blir uttröttat.

Arbetet gjordes åt automationsföretaget Noswing i Jakobstad som en del av en större helhet. Detta arbete omfattar planering och programmering av ett styrsystem enligt slutkundens behov och krav.

Den praktiska delen av arbetet gick ut på att planera styrsystemet och programmera en PLC från Siemens. Programmeringen gjordes i Siemens TIA portal v18. Utöver PLC-programmet skapades också ett användargränssnitt för en operatörspanel och systemet läser data från givare över PROFINET. Arbetet beskriver också konfiguration av en OPC server som samlar data från processen och skickar denna till en PC som kör ett loggningsprogram.

Den teoretiska delen av arbetet beskriver olika tekniker och standarder som använts för att utveckla styrsystemet. Den teoretiska delen beskriver även programvaran som använts.

Resultatet blev en fungerande anläggning som är användbar vid slutkundens produktutveckling. Anläggningen uppfyller de förhandskrav som fanns angående användarvänlighet och tillräcklig noggrannhet på insamlade data.

Språk: svenska
Nyckelord: styrsystem, PLC, hållfasthetstest, automationssystem, loggning

OPINNÄYTETYÖ

Tekijä:	Karl-Petter Söderman
Koulutus ja paikkakunta:	Sähkö- ja automaatiotekniikka, Vaasa
Suuntautumisvaihtoehto:	Automaatiotekniikka
Ohjaaja(t):	Roger Mäntylä, Steve Ekström

Nimike: Ohjausjärjestelmä rasitustestilaitteistolle

Päivämäärä: 24.3.2024 Sivumäärä: 43 Liitteet: 1

Tiivistelmä

Tämän opinnäytetyön tarkoituksena on kehittää toimiva ja luotettava ohjausjärjestelmä laitteistolle, joka testaa erilaisten palkkien ja runkorakenteiden kestävyyttä. Näitä testikohteita kuormitetaan toistuvasti ja järjestelmä kirjaa dataa jatkuvasti testin aikana. Kerätyn datan tavoitteena on mahdollistaa rakenteen käyttöiän laskeminen, jotta voidaan arvioida, milloin materiaali väsyy.

Työ tehdään automaatioyritys Noswingille Pietarsaaressa osana suurempaa kokonaisuutta. Työhön sisältyy ohjausjärjestelmän suunnittelu ja ohjelmointi loppuasiakkaan tarpeiden ja vaatimusten mukaisesti.

Työn käytännön osuus käsittelee ohjausjärjestelmän suunnittelua ja Siemensin PLC:n ohjelmointia. Ohjelmointi suoritetaan Siemensin TIA Portal v18 -ohjelmistolla. PLC-ohjelman lisäksi luodaan käyttöliittymä käyttöpaneelille, ja järjestelmä lukee anturitietoja PROFINETin kautta. Työssä kuvataan myös OPC-palvelimen konfigurointi, joka kerää prosessidataa ja lähettää sen tietokoneelle, jolla ajetaan tiedonkeruuohjelmaa.

Teoreettinen osuus kuvaa tekniikoita ja standardeja, joita on käytetty ohjausjärjestelmän kehittämisessä. Osuus kuvailee myös ohjelmia joita on käytetty projektin toteuttamiseen.

Tuloksena oli toimiva kokonaisuus, joka täyttää asetetut vaatimukset. Sekä käyttäjäystävällisyyden että mitatun datan riittävän tarkkuuden suhteen, jotta sitä voidaan käyttää tuotekehityksessä loppuasiakkaalle.

Kieli: Ruotsi
Avainsanat: ohjausjärjestelmä, PLC, rasitustesti, automaatiojärjestelmä, tiedon keruu

BACHELOR'S THESIS

Author:	Karl-Petter Söderman
Degree Programme:	Electrical engineering and automation, Vaasa
Specialisation:	Automation technology
Supervisor(s):	Roger Mäntylä, Steve Ekström

Title: Control system for durability testing apparatus

Date: 24.3.2024 Number of pages: 43 Appendices: 1

Abstract

The purpose of this thesis is to develop a functional and reliable control system for a machine testing the durability of various beam and frame structures. These test objects are repeatedly put under strain, and data is continuously logged by the system during use. The collected data will then be used to gain an understanding of when the material becomes fatigued and to calculate the lifespan of the structure.

The control system is developed for the automation company Noswing in Jakobstad, Finland as part of a larger project. This thesis includes the planning and programming of a control system according to the needs and requirements of the end customer.

The practical part of this thesis involves planning the control system and programming a PLC from Siemens. The programming is done in Siemens TIA Portal v18. In addition to the PLC-program, a user interface is created for an operator panel, and the system reads data from sensors via PROFINET. The work also describes the configuration of an OPC server that collects data from the process and sends it to a PC running a logging program.

The theoretical part of the thesis describes various technologies and standards used to develop the control system. The theoretical part also describes the software used during the development.

The result was a functional system that meets the specified requirements. Both in terms of user-friendliness and sufficient accuracy of measured data, so that it is usable in product development by the end customer.

Language: Swedish
Key words: control system, PLC, durability test, automationsystem, logging

Innehållsförteckning

Ordlista

PLC	Programmable logic controller
HMI	Human machine interface
LD	Ladder diagram
FBD	Function block diagram
SCL	Structured control language
ST	Structured text
IL	Instruction list
SFC	Sequential function chart
OPC UA	Open Platform Communication Unified Architecture
SCADA	Supervisory control and data acquisition
PC	Personal computer
IOT	Internet of things
PROFINET	Process field net
GSDML	General station description markup language
SQL	Structured query language
FW	Firmware
FC	Function
FB	Function block
OB	Organization block
DB	Database

1 Inledning

För att kunna lita på konstruktioners hållfasthet efter en lång tid av upprepade belastningscykler behöver beräknade värden jämföras med resultatet av verkliga test. Examensarbetet går ut på att skapa en lösning för slutkunden som möjliggör detta.

1.1 Bakgrund och syfte

Syftet med examensarbetet är att planera och utveckla ett PLC-baserat styrsystem för en hållfasthetstestningsanläggning. Anläggningen skall med hjälp av en hydraulpress upprepade gånger belasta en balk konstruktion och under testets gång logga data om balkens böjning samt belastningen den är under. Med hjälp av denna testdata skall det gå att beräkna hur många cykler konstruktionen tål innan materialet blir uttröttat. Slutkunden för projektet har önskat att förbli anonym och därför kommer inte användningen av den insamlade datan beskrivas närmare.

1.2 Uppdragsgivare

Uppdragsgivaren för detta arbete är Noswing AB i Jakobstad. Företagets verksamhetsområde är industriell automation, huvudsakligen med fokus på styrning av industrikranar av traverstyp. Inom företaget utförs också andra arbeten i form av automationsplanering, programmering och helhetslösningar. Företaget har 4 anställda och grundades 2012.

1.3 Avgränsning

Detta projekt gick ut på att planera styrsystemets uppbyggnad, programmera en PLC och hitta en lösning för dataloggning. Noswing har medverkat i valet av hårdvara som används till projektet, men utrustning som hydraulpress och pump, lastcell och signalförstärkare levereras av slutkunden. Planerings- och programmeringsdelen av projektet behandlas i detta arbete liksom loggning av data. Elinstallationen och den mekanisk installationen sköts av kunden eller en tredje part. Kunden ansvarar också för alla säkerhetsaspekter och riskidentifiering för en säker användning av anläggningen. Arbetet tar heller inte ställning till behandling eller användning av loggad data.

2 Teoretisk bakgrund

Detta kapitel beskriver teorin bakom de metoder som använts för att utföra examensarbetet. Kapitlet beskriver vilken typ av hårdvara och mjukvara som använts, samt olika kommunikationsmetoder.

2.1 PLC

PLC står för programmable logic controller och är en form av industriell dator, speciellt anpassad för att styra och övervaka processer eller maskiner inom industrin. En PLC utför logiska operationer enligt hur den är programmerad. Den styr processen genom digital eller analoga in – och utgångar, alternativt någon fältbuss. De huvudsakliga delarna i en PLC är nätaggregat, CPU och I/O. De allra enklaste modellerna har alla dessa delar integrerade medan större och mer modullära enheter kan konfigureras enligt behov. I de flesta fallen körs samma program kontinuerligt på en PLC. De är designade att göra detta så pålitligt som möjligt, även i mycket krävande förhållanden var det förekommer vibrationer, elektromagnetiska störningar, höga temperaturer eller damm.

Olika tillverkare har utvecklat egna programmeringsverktyg och miljöer för konfiguration av dess hårdvara. De vanligaste förekommande verktygen för programmering av PLC-system är Siemens STEP7 (TIA portal), Rockwell (Allen Bradley) Studio 5000 och CodeSys baserade verktyg som används av bland annat Omron, ABB, Schneider electric, Phoenix contact och Mitsubishi. Alla dessa verktyg baseras på den internationella standarden IEC 61131-3 som definierar bland annat fem stycken standardiserade programmeringsspråk. [1] [2]

2.1.1 Ladder diagram (LD)

Är ett grafiskt programmeringsspråk som efterliknar ett elektriskt kopplingsschema för relälogik. Programmeringsspråket var speciellt framtaget för att underlätta skiftet från relälogik till programmerbara system och tanken var att ingenjörer eller elektriker enkelt skulle kunna byta till dessa modernare system utan att vara bekanta med programmering sedan tidigare. I programspråket finns kontaktsymboler som representerar in och utgångar eller interna minnesadresser, dessa kontakter kan vara slutande eller öppnande (NC/NO). Spolar representerar utgångar eller minnesadresser. En programrad i Ladder Diagram

kallas för en rung. I en rung kan dessa symboler kombineras för att skapa logiska operationer som AND, OR eller NOT. I dessa rungar kan också funktionsblock som timers och räknare läggas till för mer avancerade funktioner. En av de största fördelarna med programspråket är att det är lätt att felsöka. [3]

2.1.2 Function block diagram (FBD)

Är ett grafiskt programmeringsspråk där varje funktion representeras av ett block. Blocket kan representera exempelvis logiska operationer som AND eller OR, till vänster finns ingångarna och till höger utgångarna. Till dessa kan Konstanter, I/O-taggar eller minnesadresser kopplas för att utföra olika operationer. Det påminner väldigt långt om Ladder diagram och i de flesta editorer kan man ”översätta” program skrivet i något av dessa språk till det andra automatiskt. [4]

2.1.3 Structured text (ST)

Är ett textbaserat högnivåprogrammeringsspråk som använder sig av syntaxen från programspråket PASCAL, och kodens uppbyggnad är lik C. Språket ger möjlighet att bygga väldigt avancerade funktioner. Programmeringen sker liksom namnet antyder strukturerat och programmet körs rad för rad från början till slut. Språket stöder alla de vanliga logiska operationerna som AND, OR, NOT och jämförelseoperatorer som större än och mindre än. Det finns också funktioner för loopar och iterationer (FOR och WHILE) och funktioner för sekvensstyrning (CASE). I många leverantörers programmeringsverktyg kan structured text integreras smidigt i funktionsblock programmerade i Ladder diagram genom att lägga in en rung med structured text mitt i koden. Structured text är mycket användbart i PLC-program då det kommer till avancerade sekvenser, databaser eller filtrering av data. Nackdelar är att det kan vara svårare att felsöka än LD och FBD. [3]

2.1.4 Sequential function chart (SFC)

Är ett grafiskt språk som är anpassat för att styra sekventiella processer. Det bygger på steg och övergångar. Ett steg kan antingen vara aktivt eller inaktivt, då steget är aktivt körs koden som finns i steget tills villkoret i följande övergång uppfylls och då aktiveras nästa steg och det tidigare inaktiveras. Sekvensen kan också förgrena sig och köra två eller flera steg parallellt. SFC passar bra till vissa typer av sekvenser eftersom det är lättare att följa

med processen grafiskt jämfört med sekvenser byggda i Ladder diagram eller Structured Text. [4]

2.1.5 Instruction list (IL)

Är ett textbaserat lågnivåspråk som kan jämföras med assembler eller maskinkod. Varje rad programkod innehåller en instruktion som körs sekventiellt vilket ger mycket exakt kontroll över programmet. Fördelarna är att programkoden körs väldigt effektivt och kräver mindre resurser av CPU:n. Nackdelen är att den blir väldigt svårt att felsöka, speciellt i längre program. Med dagens snabba hårdvara är språket sällan använt, men kan förekomma i speciella applikationer. [4]

2.2 HMI

HMI står för Human machine interface och är oftast en skärm som erbjuder operatören ett användargränssnitt för att styra en maskin eller process. I dagsläget är de flesta HMI-paneler en pekskärm, men även andra typer förekommer. HMI-panelen kommunicerar med PLC-systemet med kommunikationsprotokoll som PROFINET eller MODBUS. Användning av en HMI-panel i ett PLC-system minskar avsevärt antalet kablar och komponenter jämfört med en kontrollpanel gjord med fysiska knappar och mätare. Den möjliggör också stor flexibilitet om funktioner skall läggas till eller ändras under projektets gång. [5] [6]

2.3 TIA-portal

TIA-portal står för Totally integrated automation portal och är programvara från Siemens som innehåller en mängd olika programmeringsverktyg för programmering och konfiguration av automationsutrustning. Inkluderat i programvaran finns bland annat Simatic STEP7 som används för programmering av Siemens PLC-system. Programvaran baseras på standarden IEC 61131-3, dock stöder inte alla PLC modeller alla i standarden definierade programmeringsspråk. Alla programmeringsspråk är heller inte tillgängliga i alla typer av programblock.

Programuppbyggnaden i en Siemens PLC baseras på fyra olika typer av programblock:

- Organisationsblock (OB) är länken mellan PLC-programmet och operativsystemet och minimikravet för ett PLC-program är att det måste innehålla ett organisationsblock. OB1 skapas automatiskt då man skapar ett nytt projekt och körs cykliskt. Cykeltiden beror på hur stort programmet är, och börjar om från början efter att programmet körts klart. Det är möjligt att använda sig av flera organisationsblock och då körs de i ordning efter sitt nummer. Det finns flera typer av organisationsblock som kan vara användbara i olika tillämpningar, exempelvis vid tidskritiska tillämpningar kan organisationsblock köras efter olika systemavbrott (interrupts) enligt en viss tid (cykliska) eller då en ingång blir aktiv (hårdvara).

- Datablock (DB) lagrar och organiserar variabler och kan vara av typen globala, som man kommer åt var som helst i programmet, eller lokala. Lokala block (instansdatablock) är ofta kopplade till en viss funktion och innehåller endast sådana variabler som används i den funktionen.

- Funktionsblock (FB) används för att bygga olika funktioner i PLC-programmet som behöver dedikerat minne. Alla funktionsblock har ett eget instansdatablock där variabler kan sparas. Samma funktionsblock kan kallas flera gånger i programmet med olika instansdatablock, vilket gör att programkod inte behöver upprepas exempelvis om flera likadana enheter skall styras av programmet eller många värden skall skalas på samma sätt.

- Funktioner (FC) används för funktioner som inte behöver dedikerat minne, fungerar i övrigt på samma sätt som funktionsblock.

En annan del av TIA-portal som används i detta projekt är Simatic WinCC som används för att programmera Siemens HMI-paneler. Det används också för andra typer av projekt-visualisering från Siemens, exempelvis ”View of things” som är en enkel virtuell HMI som kan köras direkt i webbservern på S7-1500 seriens PLC och också för stora SCADA system som körs på en PC i form av WinCC runtime advanced. [7]

Andra viktiga delar av TIA-portal som inte används i detta projekt är: [8]

- Simatic STEP7 Safety som används för programmering av Siemens Safety-klassade (failsafe) PLC-system.
- SINAMICS Startdrive som används för konfiguration av Siemens frekvensomriktare och servodrifter.
- SIMATIC motion control som används för positionering av axlar.
- SIRIUS Simocode används för styrning och monitorering av konstanthastighetsmotorstyrningar via mjukstarter mm.

Integrationen av alla dessa verktyg i en och samma programvara underlättar programmeringen av både stora och små automationsprojekt avsevärt. Alla enheters konfiguration och program kan vara sparade i ett och samma projekt. Dessutom blir konfigurationen av kommunikation mellan de olika enheterna mycket smidigare, speciellt vid användning av någon fältbuss. Eftersom de alla finns i samma projekt och man inte behöver byta mellan olika program hela tiden.

2.4 PROFINET

PROFINET som står för ”Process Field Net” är en industriell kommunikationsstandard som är utvecklat av Profibus & Profinet international (PI). Det är i princip en vidareutveckling av PROFIBUS med fördelen att PROFINET är baserat på Ethernet, och kan användas över vanliga Ethernet-nätverk utan speciell hårdvara, vilket gör det mycket flexibelt. Användarvänlighet och kostnadseffektivitet gör att PROFINET lönar sig att användas i både små och stora automationssystem, speciellt då enheter eller styrskåp är utspridda. Då kopplas dessa enkelt ihop med varandra med endast en kabel och snabbar på ibruktagningen av systemet avsevärt.

Ett problem med Ethernet baserad kommunikation inom automationsindustrin var länge att TCP/IP baserad kommunikation inte är deterministisk, dvs. oberäkneliga fördröjningar kan uppstå speciellt om nätverket har många enheter anslutna. Detta problem löser PROFINET genom att använda TCP/IP ”data link layers” 3 och 4, enligt OSI-modellen

definierade i Ethernet standarden IEEE802.3 endast för konfiguration och diagnostik. Själva processdatat överförs direkt på data link layer 2 (OSI-modellen) i realtid. Detta orsakar i vissa fall problem om Ethernet nätverket också används till annat, exempelvis kan inte PROFINET nätverk delas upp i olika undernätverk (subnets) utan speciella nätverksswitchar. Trådlös kommunikation kräver speciell hårdvara eftersom normalt endast IP-adresser forwardas i Wifi nätverk.

PROFINET gör det enkelt att integrera olika tillverkares enheter i ett PLC-system, eftersom det använder Device Description (GSDML – General station description markup language) filer för att beskriva och parametrisera enheter som läggs till i nätverket. Då man lägger till en enhet definieras in och utgångar automatiskt också behov av minne.

Vid extra krävande processer där precision och hastighet är mycket viktiga, exempelvis vid styrning av synkrona axlar eller exakt positionsmätning kan en variant som heter PROFINET IRT (”Isochronous real time”) användas. I det fallet synkroniserar mastern i nätverket kommunikationen till underenheter enligt en inställd tid och det finns möjlighet att köra prioriterad programkod i PLC-enheten varje gång det finns ny data.

PROFINET innehåller också en rad diagnostiska funktioner som underlättar felsökning och drift. Bland annat genereras felmeddelanden från enheter automatiskt. [9] [2]

2.5 OPC UA

OPC UA som står för Open Platform Communication Unified Architecture är en öppen standard för kommunikation mellan olika enheter och system inom industrin. Bakom standarden står organisationen OPC Foundation. OPC UA är en vidareutveckling av äldre OPC standarder som DA och HDA, som inte tas upp i detta arbete.

OPC UA är ett av de viktigaste och mest använda kommunikationsstandarderna inom Industri 4.0 och IOT. Industri 4.0 är en standard som började implementeras kring 2010 och går i grund och botten ut på att alla enheter som på något vis har att göra med produktionen i en fabrik skall vara uppkopplade till ett överliggande system. Detta för att optimera och effektivera alla steg från beställning till tillverkning och lagerhantering.

Kommunikationen fungerar så att en klient ansluter till en eller flera servrar. Beroende på användarnivå kan klienten ha rättigheter att endast läsa data eller också skriva data. En OPC UA server finns integrerad i många tillverkares PLC-modeller och utgör ett smidigt sätt att skicka vidare data från bland annat givare till ett överliggande system. I serverdefinitionen kan objekt i OPC servern fritt konfigureras att läsa processdata från I/O eller Datablock. För system som inte har en OPC UA server integrerade kan en fristående server användas (t.ex Kepware) som körs på en PC.

Klienten är liksom servern plattformsoberoende, allt som krävs är att enheten stöder TCP/IP, HTTPS eller MQTT protokollbaserad kommunikation. Detta stöd för olika nätverksstrukturer möjliggör kommunikation över lokala nätverk, internet eller molnbaserade system. Klienten kan exempelvis vara ett SCADA system som övervakar och styr en hel fabrik, en databas som loggar processdata, eller ett MES-system. Flera klienter kan ansluta till en och samma server eller till många olika servrar.

Eftersom serverns struktur är standardiserad kan klienten automatiskt läsa in datatyper och serverns struktur då den ansluter. Varje objekt i servern innehåller följande information:

- DisplayName
- Tag Value
- Quality or status
- Timestamp of OPC UA server and Client
- Data type
- Subscription ID

Klienten kan “polla” servern med ett inställt tidsintervall, men det finns också möjlighet att använda en prenumerationsmodell där klienten endast läser data vid förändringar. Detta minskar avsevärt på nätverkets belastning, eftersom all data inte behöver läsas varje gång.

OPC UA är väldigt skalbart och lämpar sig att använda till allt från små slutna system till enorma uppkopplade system. Detta till stor del på grund av avancerade säkerhetsfunktioner som kryptering, användarcertifikat och inloggning med användarnamn/lösenord är en del av standarden. Användare kan delas upp i olika åtkomstklasser där exempelvis administratören har skrivrättigheter medan operatörer endast har läsrättigheter. Det är också möjligt att övervaka vilken användare som gjort ändringar och tidpunkt.

Sammanfattat är OPC UA en mycket flexibel och robust standard för kommunikation inom industriell automation, och ger en bra grund för säkra och flexibla lösningar i en tid då datainsamling och uppkopplade system blir en allt viktigare del av industrin.

I detta examensarbete används OPC UA i en väldigt grundläggande form med endast en server och klient i ett lokalt nätverk. [10] [11]

2.6 Advanced OPC datalogger

Advanced OPC data logger är ett program från AGG Software som används för att skapa en länk mellan en OPC server och olika databaser, Excel eller andra program. Programmet är gjort för att köras på en Windows PC.

Programmet kan anslutas till vilken OPC server som helst och stöder DA, HDA och UA gränssnitt med flera. Det kan ansluta till flera servrar samtidigt och kan exempelvis användas för att samla data från många PLC-system och exportera den till ett SCADA system eller till en databas. Databastyper som stöds är bland annat MSSQL, MySQL, MS Access och Oracle kompatibla databaser.

Programmet kan testas fritt men man behöver köpa en licens för obegränsad användning. [12]

3 Praktiskt utförande

I detta kapitel beskrivs metoderna som använts för att utföra projektets praktiska del. Kapitlet beskriver val av utrustning samt programmering och konfiguration av anläggningens olika delar.

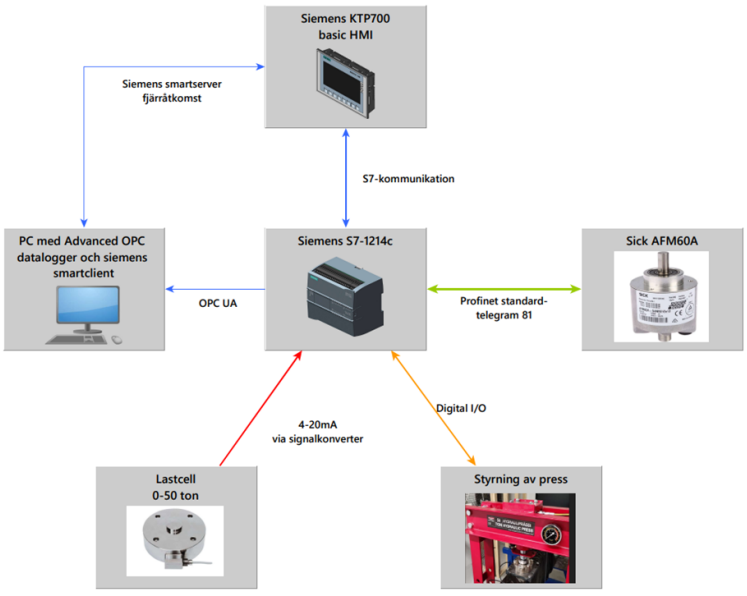
3.1 Planering av styrsystem

Valet av hårdvara till projektet görs tillsammans med kunden och strävar efter att hitta en bra balans mellan kostnadseffektivitet och samtidigt uppfylla de krav på mätprecision och pålitlighet som satts.

PLC och HMI är båda från Siemens medan övrig fältutrustning kommer från en rad olika tillverkare och leverantörer. Programvaran som används i projektet för programmering är Siemens TIA-portal, för loggning av data Advanced OPC data logger från AGG Software och för analys av loggad data Microsoft Excel.

3.2 Val av utrustning

Anläggningen skall vara relativt lätt att flytta på. Därmed behövde mängden kablar och komponenter hållas så liten som möjligt utan att kompromissa på funktioner. Givare och enheter med PROFINET kommunikation användes så långt det var möjligt.



Figur 1. Anläggningens uppbyggnad.

3.2.1 PLC

Systemet baseras på en Siemens S7-1214c PLC som erbjuder en bra balans mellan kostnadseffektivitet och funktioner. Den har inbyggt 14 stycken digitalingångar, 10 stycken digitala transistorutgångar och två stycken analogingångar. Den har också en inbyggd web-server och stöd för OPC server från och med firmware version 4.5. Till den ansluts också ett expansionskort av modell SM1234 vilket har fyra analogingångar och två analogutgångar. In och utgångarna på detta kort kan programmeras till antingen spänning 0-10V eller ström 0(4)-20mA med en upplösning på 14 respektive 13 bitar. Mätvärdet från lastcellen fås via 4-20mA och detta ger oss en upplösning på ungefär 2kg.

Siemens s7-1200 serien stöder de vanligaste programmeringsspråken definierade av IEC 61131-3 standarden med undantag från IL (instruction list) och SFC eller som Siemens kallar det GRAPH.

3.2.2 HMI

Operatören av anläggningen styr processen via en operatörspanel (HMI) också från Siemens av modell KTP700 basic. Den har en pekskärm på 7 tum med en upplösning på 800x480 pixlar samt 8 stycken programmerbara fysiska knappar. Kunden vill ha möjlighet att övervaka processen på distans, detta möjliggörs eftersom HMI-panelen har stöd för fjärråtkomst via Siemens smartserver, detta kräver dock en tilläggslicens.

Då smartserver aktiveras kan man med en PC i samma nätverk som har programmet SmartClient installerat spegla HMI panelen till en annan enhet exempelvis en PC. Kommunikationen kan vid behov skyddas med lösenord, man kan också välja om man ger enheten lov att styra HMI-panelen eller endast spegla skärmen för visning. I detta fall hålls PLC nätet skilt, datorn som kör loggningen och speglingen av HMI-panelen är utrustad med dubbla nätverkskort var endast det ena är anslutet till internet. Vill man styra HMI-panelen utifrån måste det alltså ske via fjärrskrivbord till loggningsdatorn.

3.2.3 Lastcell

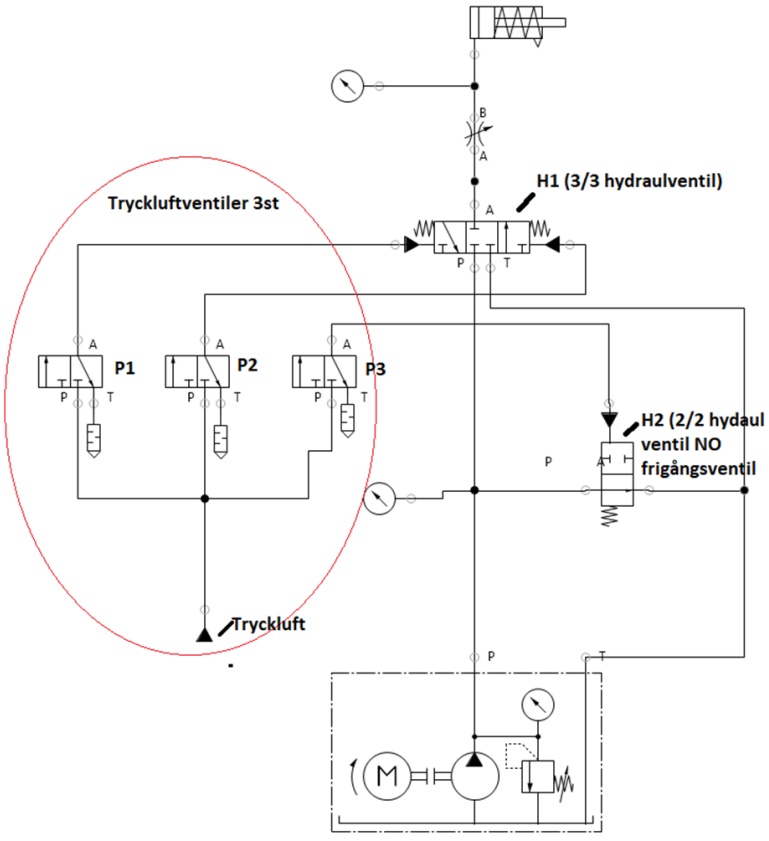
Lastcellen som används har kunden själv valt ut och är ett färdigt paket från företaget Vetek som är specialiserat på våg-system för bland annat industrin. Paketet består av själva lastcellen samt en signalomvandlare från tillverkaren Scaime. Denna kan ge ut en hel del

olika signaltyper, i detta projekt används 4-20mA för viktdata till PLC:n. Signalomvandlaren och lastcellen levereras färdigt kalibrerade från tillverkaren och området 4-20mA är färdigt skalat till att motsvara en belastning på 0-50 ton. Lastcellen ger i sig själv ut en signal på 3mV/V med en noggrannhet på +-0.1%.

En möjlighet var också att ansluta lastcellen direkt till vår PLC genom ett expansionskort. Detta säljer Siemens under namnet Siwarex WP231 men då hade kalibreringen behövts utföras efter installationen och dessutom var leveranstiden mycket lång. [13]

3.2.4 Hydraulisk press

Pressen är enkelverkande och drivs av ett hydraul-aggregat. Aggregatets hydraulventiler styrs med tryckluftsventiler som styrs via reläer från digitala utgångar i vår PLC. Hydraul-aggregatet har ett maxtryck på 800 bar och ungefär 700 bar behövs för att uppnå pressens maxtryck på 50 ton. Detta höga tryck gör att det inte finns hydraulikventiler som är direktstyrda med spole. I funktionsbeskrivningen från aggregatets tillverkare beskrivs tydligt användningen.



Figur 2. Hydraul-aggregatets schema.

Ventilen H2 är en frigångsventil som är normalt öppen från pump till tank. Den styrs av pneumatikventil P3. Så länge P3 är opåverkad (ingen styrsignal) så är H2 öppen och inget tryck kan byggas upp i systemet.

Ventilen H1 är en riktningsventil med stängt mittläge. Den styrs av pneumatikventilerna P1 och P2. Så länge P1 och P2 är opåverkade är H1 i mittläget. H1 är fjädercentrerad, utan pneumatiskt styrtryck returnerar fjädrarna den till mittläget. Cylindern hålls då stilla.

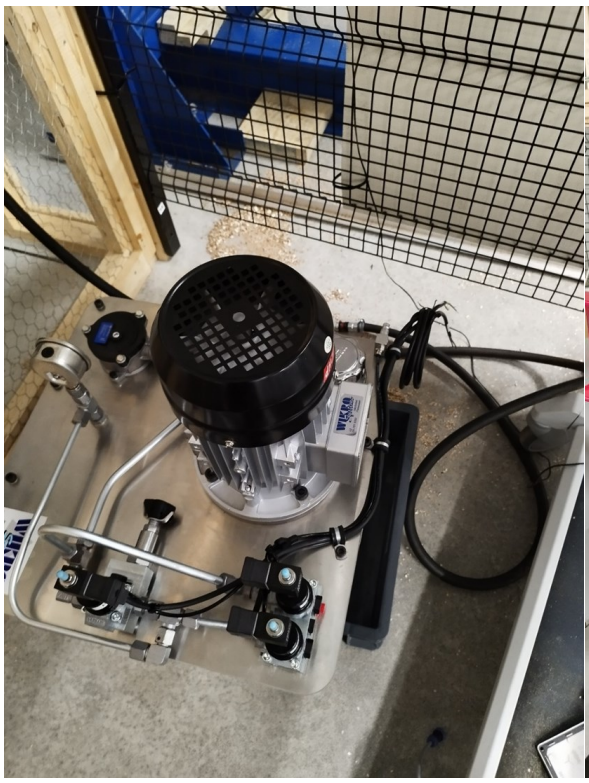
Vid aktivering av P2 ändrar H1 läge så att trycklinjen från pumpen kopplas till cylindern. Vid aktivering av P1 ändrar H1 läge så att linjen från cylindern kopplas till tank.

För att köra ut cylindern skall P2 och P3 aktiveras. P3 kan förslagsvis aktiveras med en liten fördröjning för att undvika tryckstötar (ex 100ms). Om cylindern redan är delvis utkörd och fördröjning används, så kan cylindern hinna backa en aning innan den kör framåt. Ifall detta är ett problem så kan man bli tvungen att ta bort fördröjningen

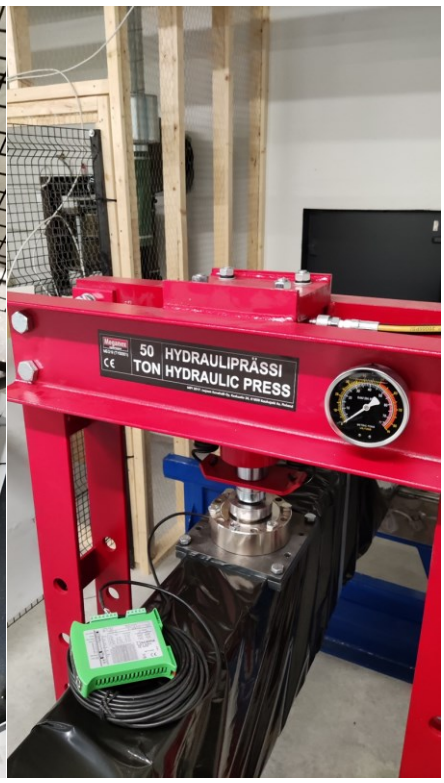
För att hålla cylindern på plats skall P1, P2 och P3 avaktiveras. Ingen fördröjning behövs.

För att köra in cylindern skall P1 aktiveras. P2 och P3 avaktiveras.

P1 och P2 får aldrig aktiveras samtidigt, om man gör det så försöker man ändra läge på H1 till både ”vänster” och ”höger” samtidigt. [14]



Figur 3. Hydraulaggregat.



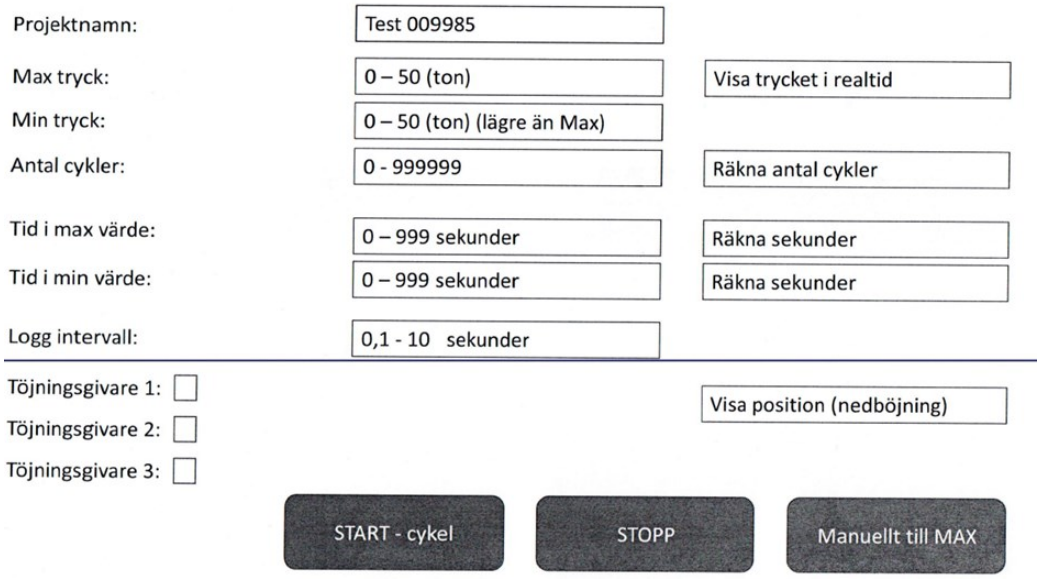
Figur 4. Hydraulpress.

3.2.5 Absolutpulsgivare

För mätning av balkens position med hög noggrannhet används en absolutpulsgivare från Sick av modell AFM60A. Pulsgivaren är utrustad med PROFINET anslutning för enkel dataöverföring. Givaren har en enormt hög resolution på 262144 pulser per varv, i denna tillämpning är den monterad till en mätvajer-mekanism som konverterar den linjära rörelsen till rotation. Målet är att nå en noggrannhet på 0,1mm för att få så exakt data som möjligt. Givaren måste monteras på en skild ställning, eller i taket på utrymmet så att den alltid har en fast punkt som inte påverkas av pressens tryck. Eftersom absolutvärdet inte är viktigt utan endast förändringen, fästs den andra ändan av vajern i test-stycket med en kraftig magnet. Mätvärdet nollas av operatören via HMI-panelen före start av test. [15] [16]

3.3 Programmets funktionsspecifikation

Kunden levererade en funktionsbeskrivning som beskrev grundfunktionerna och ett exempel på vilken data man vill se på HMI-panelens huvudsida.



Figur 5. Förslag på HMI-panelens huvudsida.

3.3.1 Manuellt test

Under en manuell körning mäts och loggas följande data:

- Tryck
- Avstånd (position)
- Spänning (Hållfasthet)

En manuell körning utförs enligt följande steg:

1. Användaren skriver in max tryck som vill uppnås
2. Trycker på manuellt till max för att köra upp trycket till maxtryck.
3. Trycket hålls på maximi.
4. I HMI-panelen syns alla mätresultat som är aktiverade
5. Testet avslutas genom att trycka på stopp och trycket släpps.

3.3.2 Cykeltest

Under ett cykeltest mäts och loggas följande data:

- Tryck
- Avstånd (position)
- Spänning (Hållfasthet)
- Antal genomförda cykler

Ett cykeltest utförs på följande sätt:

1. Användaren skriver in data i HMI-panelen
2. Trycker på start – cykel för att starta testet, testet kan avbrytas när som helst genom att trycka på stopp
3. PLC mäter trycket och höjer trycket tills maxvärde uppnås.
4. Trycket hålls så många sekunder som användaren matat in
5. Trycket släpps till miniminivån och hålls där den inställda tiden.
6. Detta upprepas tills antal cykler uppnåtts, då avbyter testet automatiskt.
7. Nedböjning/position samt tryck loggas hela tiden under testets gång.
8. Ifall töjningsgivare är aktiverade loggas även dess data.
9. Ifall maxtryck inte uppnås inom inställd tid indikeras fel och testet avbryts.
10. Ifall positionsmätningen går över en gräns för maxböjning indikeras fel och testet avbryts.

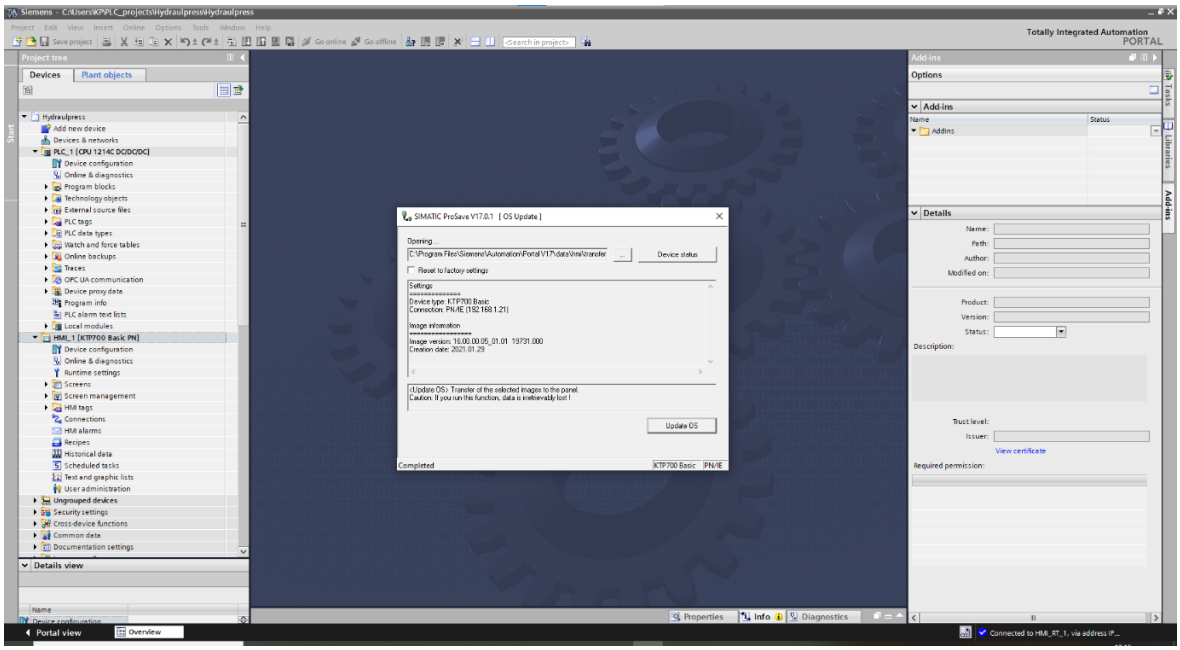
3.4 PLC-programmet

I denna del följer en beskrivning av PLC-programmet

Programmet är byggt i Siemens TIA portal v18 med en kombination av programspråken LAD och ST (eller SCL som det kallas i TIA). Valet av programmeringsspråk är helt baserat på att de är dessa jag behärskar bäst.

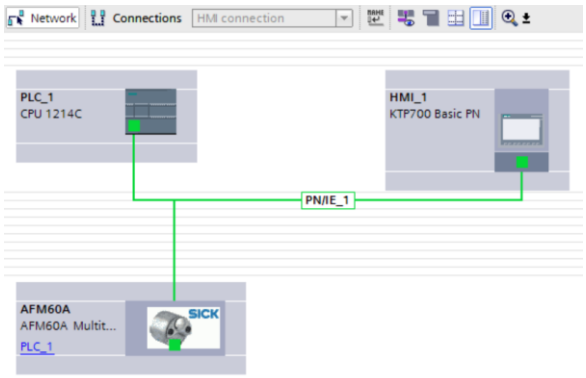
3.4.1 Hårdvarukonfiguration

Projektet i TIA portal påbörjas genom att definiera hårdvarukonfigurationen. PLC och analog-expansionskortet läggs till först och därefter HMI-panelen. Båda dessa är Siemens egna produkter och de hittas i hårdvarukatalogen. Eftersom PLC nätverket kommer vara ett eget nätverk, ställs IP-adresserna in direkt i projektet. Under device configuration fönstret hittas PLC I/O-adresser och kan vid behov ändras. Redan i detta skede definierades ingångs och utgångstaggar baserat på el-ritningarna för automationsskåpet. Hårdvaran fanns redan tillgänglig i detta skede och ett kommunikationstest följde. Under testet framkom det att HMI-panelen hade en gammal firmware version. Nyaste versionen hittades på Siemens hemsida och uppdaterades via TIA portal.



Figur 6. HMI-panelens firmware uppdatering.

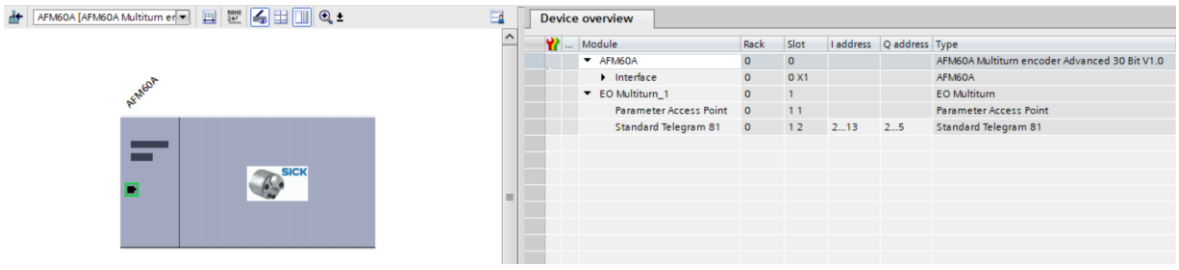
För att lägga till absolutpulsgivaren i projektet behövs en GSD-fil (General Station Description). Denna fil skall finnas tillgänglig från tillverkaren, i detta fall Sick. Den är en standardiserad beskrivning av enhetens I/O gränssnitt/adresser för PLC:n. Efter att filen importerats i TIA-portal hittas enheten i hårdvarukatalogen på samma sätt som Siemens egna förinstallerade enheter.



Figur 7. Hårdvarukonfigurationen i TIA-portal.

Enheten läggs till i nätverket genom ”drag and drop” från pulsgivarens nätverks-port till porten på PLC:n. IP-adressen ställs då automatiskt in till nästa lediga och likaså ställs PROFINET namnet för enheten in automatiskt.

Flexibla enheter som stöder lite olika datastrukturer kräver ännu att man väljer vilken data från enheten man vill läsa. I vårt fall behöver vi endast veta positionen som hittas redan i den kortaste som kan väljas, Standard telegram 81.



Figur 8. Konfiguration av absolutpulsgivarens I/O-adresser.

Vid val av telegram får enheten automatiskt nästa lediga ingångs- och utgångsadresser tilldelade. En snabbtitt i enhetens manual berättar vilken data som finns i Standard telegram 81 och enligt den definieras taggar till respektive adresser. Standard telegram 81 innehåller endast positionsdata och status samt konfigurationsdata. Men om behov finns kan man istället välja exempelvis standard telegram 83, som utöver position också ger hastigheten.

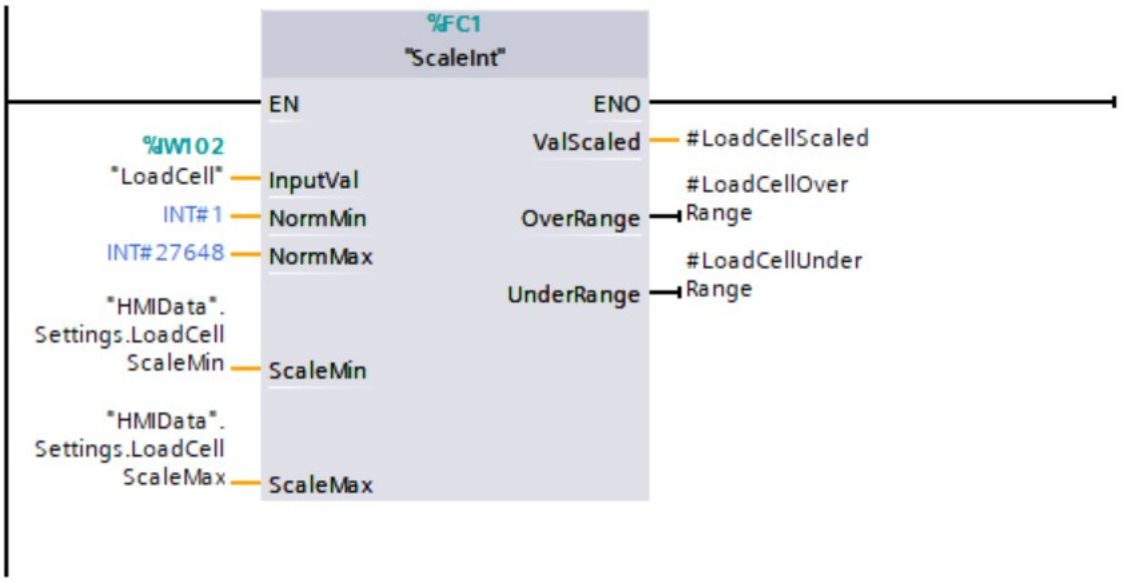
Då hårdvarukonfigurationen är slutförd, enheterna kommunicerar med varandra och alla I/O-taggar är definierade blir det dags att börja med själva PLC-programmet. Programmet består av en blandning funktioner och funktionsblock enligt användningsområde, i detta fall kallas alla block cykliskt från OB1.

3.4.2 OB1-main

I PLC-programmet används ett organisationsblock, OB1 för att köra alla funktioner. OB1 är programmerat med språket ladder diagram (LD). Programmet är uppbyggt så att funktioner och funktionsblock i programmet används på flera ställen för att minska på mängden programkod.

3.4.3 FC1-skalning av analogingångar

Början på OB1 innehåller skalning av analogingångar via funktionen FC1, via denna funktion skalas lastcellens och de tre töjningsremsornas mätvärden. Samma funktion används för alla analogskalningar och kallas sammanlagt fyra gånger.

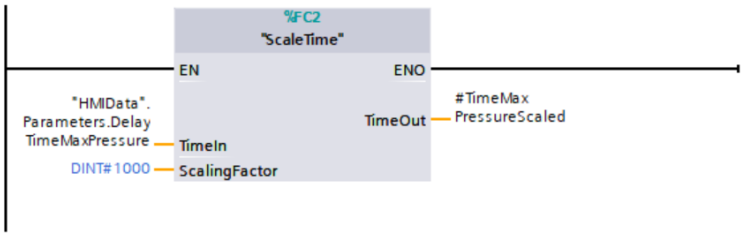


Figur 9. Funktionen FC1, ScaleInt.

Funktionen läser in mätvärdet från analogingången, mätvärdet rör sig mellan 1 och 27648 (motsvarar 4-20mA) om allt fungerar som det ska. Detta område skalas om enligt nya min- och maxgränser som användaren matar in via HMI-panelen. Lastcellen skalas så att utgångsvärdet från funktionen direkt motsvarar belastning i kg, och töjningsgivarna kan skalas så att utgångsvärdet motsvarar exempelvis hundradels millimeter. Funktionen ger ut alarm om analogsignalen avviker från området 4-20mA.

3.4.4 FC2-skalning av tider

Som följande i programmet skalas olika tider från HMI panelen med hjälp av funktion FC2. Dessa tider används av olika timer-funktioner i programmet som kan behöva justeras av användaren. Funktionen ScaleTime läser in ett heltal av datatypen DINT (32bit) från HMI-panelen som motsvarar sekunder och konverterar detta till datatypen Time. En variabel av datatypen Time kan sedan användas vid ingången på en timer, för att ändra tiden för tillslags eller frånslagsfördröjning.



Figur 10. Funktionen FC2, ScaleTime.

För att skala om den inmatade tiden till sekunder behövs en skalningsfaktor på 1000. Ifall man direkt konverterar ett heltal till datatypen Time i TIA-portal blir tiden i millisekunder. Funktionen kallas två gånger i OB1, för att skala tider som används för fördröjningar i huvudsekvensen vid uppnått minimi eller maxtryck.

3.4.5 FB4-mätning av position

Till näst i OB1 följer skalning av mätvärde från absolutpulsgivaren, detta görs via funktionsblock FB4. Positionsdata tas rakt från PROFINET telegram 81 datastrukturen.

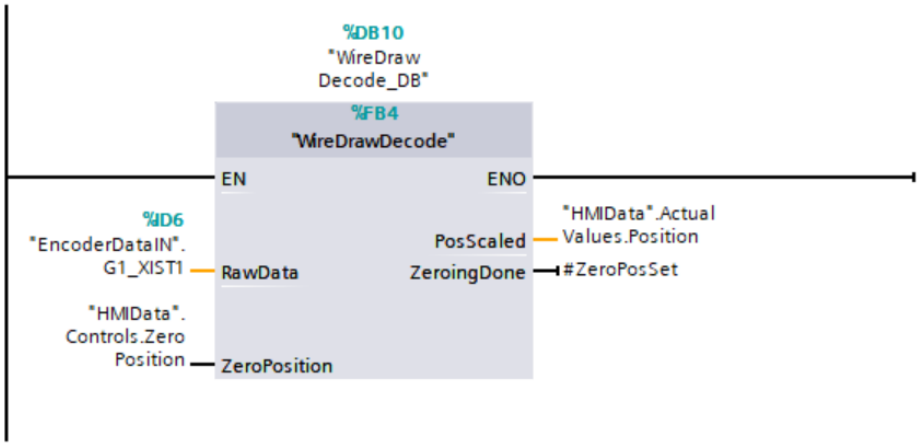
Data IN	1	2
Name	STW2-ENC	G1-STW
Length	16 bit	16 bit
Explanation	Encoder control word 2	Sensor control word 1

Tabell 1. Telegram 81 ingångsdata.

Data OUT	1	2	3	4	5	6
Name	ZSW2-ENC	G1-ZSW	G1_XIST1 MSW	G1_XIST1 LSW	G1_XIST2 MSW	G1_XIST2 LSW
Length	16 bit	16 bit	32 bit		32 bit	
Explanation	Encoder status word 2	Sensor status word 1	Position value 1		Position value 2	

Tabell 2. Telegram 81 utgångsdata.

I utgångsdatat hittas två adresser G1_XIST1 och G1_XIST2 som innehåller positionsdata med formatet DINT (32 bitars heltal). Eftersom rörelseområdet är ganska kort i denna tillämpning räcker det att använda endast den första.



Figur 11. Funktionsblocket FB4, WireDrawDecode.

Funktionsblocket läser in rådata från positionsmätningen och skalar den till millimeter. Ingångsvärdets bitar är förskjutna två steg till vänster vilket framkommer i tillverkarens manual för givaren. Detta betyder att alla bitar först måste flyttas två steg till höger innan ytterligare skalning kan genomföras. Då vajern dras ut 555mm motsvarar det ett helt varv i rotation för absolutpulsgivaren. Ett varv för pulsgivaren motsvarar 262144 pulser. Tack vare denna data blir det möjligt att räkna ut att 1mm motsvarar 472.33 pulser. Detta ger en teoretisk mätnoggrannhet på ca 0.002mm. Alla uträkningar i blocket utförs med flyttal, för att utföra en så noggrann skalning som möjligt.

Mätvärdet skall nollas före start av test, detta görs genom att ge en signal till ingången ZeroPosition. Då detta utförs blir det aktuella rådatat från givaren en ny nollpunkt som sparas i funktionens datablock DB10, detta är orsaken till att FB4 är ett funktionsblock och inte en funktion. Funktionsblocket ger ut en statusbit då detta har blivit utfört.

3.4.6 FB1-huvudsekvensen

Den viktigaste delen av programmet är FB1 MainSequence. Detta funktionsblock sköter styrning av själva pressen både i automat och manuellt läge. Det övervakar också processen och avbryter den om gränsvärden överskrids.

Funktionsblockets ingångar är alla utom nödstopp och motorskyddsövervakning för pumpen kopplade till signaler från HMI-panelen. Till funktionsblocket kommer startsignaler för automat och två typer av manuell körning. Ifall manuellt till max ingången aktiveras, körs samma program som för automatsekvensen men med endast en cykel. Trycket hålls vid maxnivån ända tills det släpps manuellt genom att aktivera stopp. Man har också möjligheten att öka eller släppa trycket helt manuellt via HMI-panelen och då förbikopplas alla säkerhets- och alarmgränser.

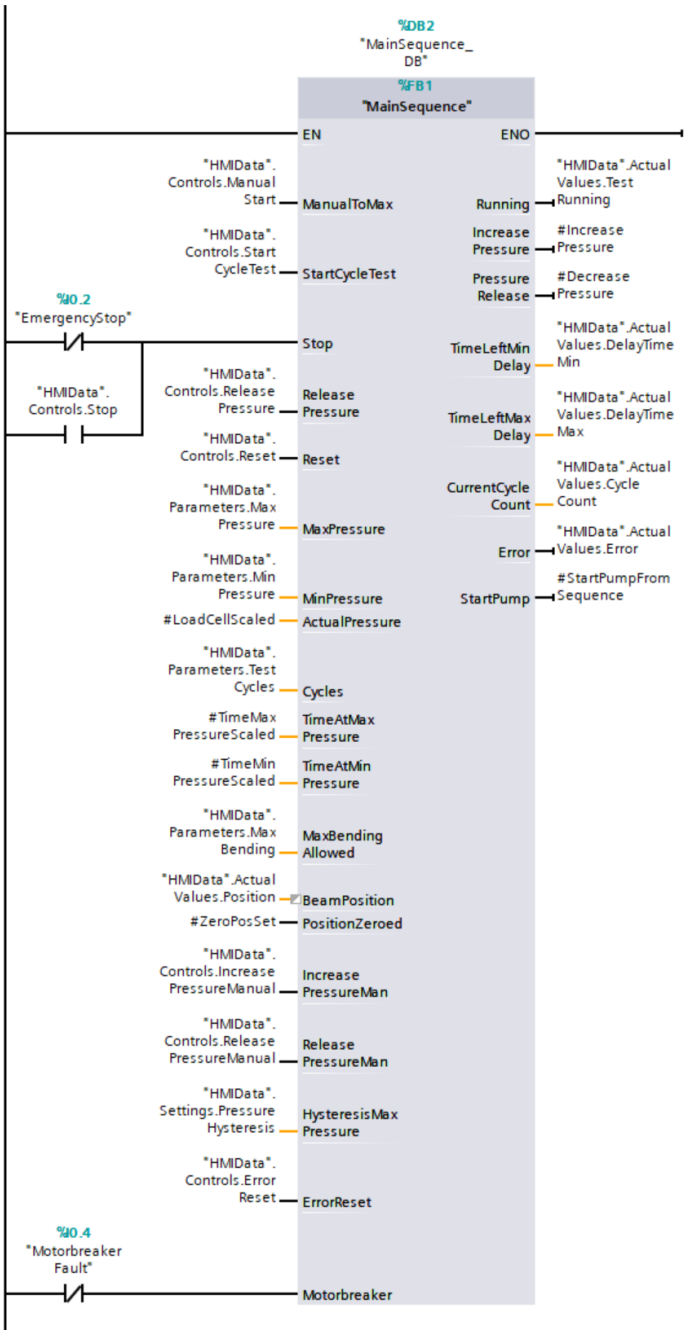
Själva sekvensen i programmet är väldigt kort och består endast av två steg, ett som släpper trycket i pressen och ett som bygger tryck. Startas cykeltest så upprepas dessa steg enligt antal cykler som blivit inställda, vid start manuellt till max lämnar programmet på det andra steget tills det stoppas med stoppknappen.

Alla parametrar i FB1 kan ställas om av användaren via HMI-panelen vid behov. Parametrarna och alarmgränserna är sparade i datablocket HMIData, DB1 och ”retain” funktionen är aktiverad för dessa variabler. Detta betyder att dessa variabler är sparade i permanentminne i PLC:n, vid strömavbrott behåller variablerna sitt värde. Många av parametrarna har gränsvärden satta i HMI-panelen som gör att operatören inte kan mata in värden utanför det tänkta området så någon indatakontroll behöver inte göras i PLC-programmet.

Utgångar från funktionsblocket är endast två signaler för styrning av pressen, en för att öka trycket och en för att släppa trycket. Utöver dessa styrsignaler fås några statusbitar ut som visas i HMI-panelen, som att sekvensen kör eller att ett fel är aktivt. Aktuellt antal cykler som körts fås också ut, detta värde används i loggen för att sortera data, men visas också i HMI-panelen för att följa med hur processen framskrider.

FB1 har ett eget datablock, DB2 i vilket interna variabler sparas. Också alla funktioner, exempelvis timers som kallas inne i FB1 har sina så kallade instans-datablock sparade i DB2. Detta gör att mängden globala datablock i TIA-portal minskar avsevärt och gör listan

på programblock kortare. En annan fördel är att om funktionsblocket kopieras till ett annat program skapas alla interna instans-datablock automatiskt och behöver inte skilt definieras av användaren.

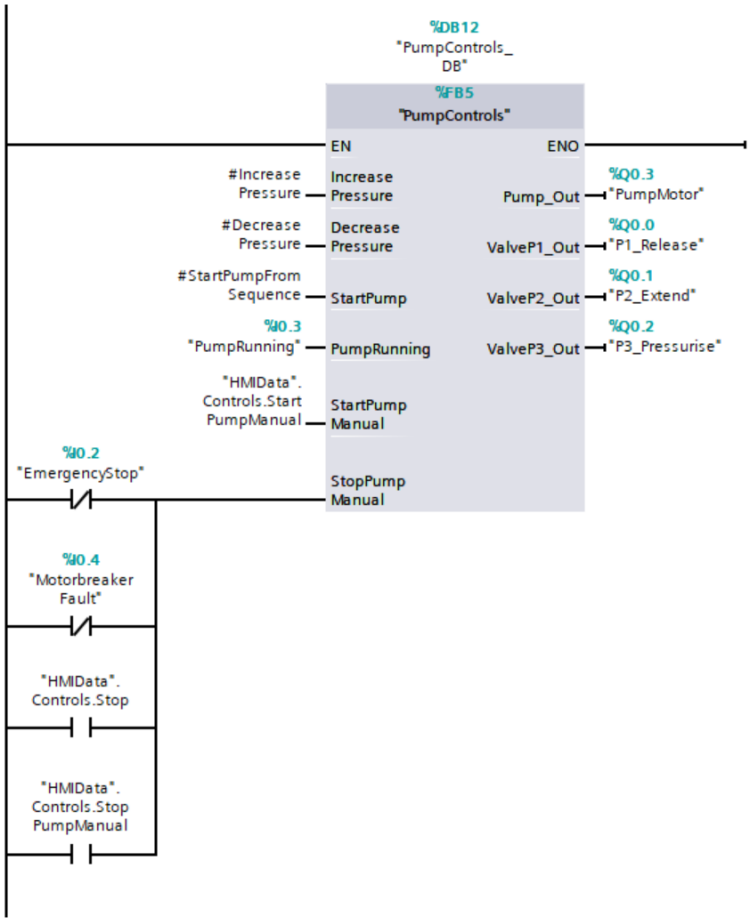


Figur 12. Funktionsblocket FB1, MainSequence.

Se bilaga 1 för FB1 funktionsblockets fullständiga programkod.

3.4.7 FB5-styrning av press

Variablerna från FB1 för att öka eller släppa trycket kopplas till ingångar på FB5 PumpControls. Detta funktionsblock sköter om start av hydraul-pumpen och styr ventilerna P1-P3 på hydraulikaggregatet. Denna del av programmet gjordes som ett skilt funktionsblock, eftersom det varit en del problem med pumparna. Behöver pumpen bytas ut igen till en annan modell räcker det troligtvis att ändra endast detta block i programmet.



Figur 13. Funktionsblock FB5, PumpControls.

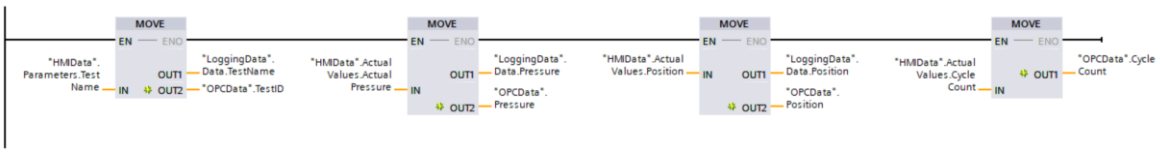
På funktionsblocket finns också en insignal till vilken en tillbakakoppling från pumpens kontaktor kopplas. Ventilerna som ökar trycket skall vara opåverkade vid start av pumpen för att minska på belastningen och därmed startströmmen. Ventilen för att släppa trycket kan manövreras också om pumpen står stilla.

Pumpen stannas omgående om något fel inträffar eller nödstoppet trycks in och alla ventiler lämnar i ett opåverkat läge.

3.4.8 Styrning av datalogg

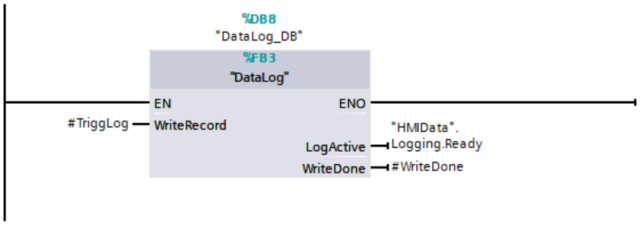
I slutet av OB1 utförs flytt av parametrar som skall loggas till OPC serverns datablock och också till den interna loggen som finns som en färdig funktion i PLC:n. Först testades det att använda endast PLC loggen, men internminnet tar slut efter ca 10h vid önskad loggningsfrekvens.

Loggen är konfigurerad att skriva över den äldsta datan och fick vara kvar som redundans ifall OPC-kommunikationen skulle fallera under exempelvis en natt. Den interna loggen sparas på ett SD-kort i PLC:n och kan laddas ner till loggningsdatorn via webbservern som är aktiverad i PLC:n. På detta vis kan man få med data i analysen som annars skulle gått förlorad.



Figur 14. Flytt av data som skall loggas.

Eftersom det inte ännu i detta skede finns några töjningsgivare flyttas inte deras mätvärden till loggen för att spara på resurser. Ifall de i framtiden skall läggas till behöver man utöka OPC-servern och flytta också denna data.



Figur 15. Funktionsblock FB3 Datalog.

Funktionsblocket FB3 innehåller funktioner för att använda den interna loggen i PLC:n. Detta görs genom färdiga funktionerna DataLogCreate, DataLogOpen och DataLogWrite.

Ny data skrivs varje gång ingången WriteRecord aktiveras. Utgången LogActive ger ut status på att loggfilen skapats eller öppnats och att SD kortet på vilken loggen sparas är inmatat i PLC:n. Utgången write done ger status på att en ny rad skrivits i loggen.

3.4.9 HMI kommunikation

HMI-panelen kommunicerar med PLC:n över PROFINET. All kommunikation sker genom ett och samma datablock, DB1 HMIData. I Datablocket har variablerna kategoriserats med hjälp av datastrukturer enligt användning, detta görs genom att skapa en variabel med data typ Struct. Inne i strukturen kan sedan andra variabler av valfri data-typ läggas till. Fördelar med detta är bland annat om alla parametrar som finns under en struktur skall läggas till i permanentminnet behöver inte varje variabel väljas skilt utan hela strukturen delar huvudsakliga egenskaper.

HMIData

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	Static								
2	Test	Int	123						
3	Parameters	Struct							
4	TestName	UInt	0						
5	MaxPressure	UInt	50000						kg
6	MinPressure	UInt	500						kg
7	TestCycles	UDInt	100						
8	DelayTimeMaxPressure	DInt	2						s
9	DelayTimeMinPressure	DInt	2						s
10	LoggingInterval	DInt	2						s
11	MaxBending	UInt	20						mm
12	ActualValues	Struct							
13	Controls	Struct							
14	Settings	Struct							
15	Logging	Struct							

Figur 16. Globala datablocket DB1, HMIData.

Strukturerna kopieras sedan över till HMI-panelen där variablerna fortsätter vara uppdelade genom att skapa skilda tagg-listor för varje struktur. Vid kopieringen får de nya HMI taggarna automatiskt namn genererade enligt variablerna i PLC datablocket. Kommentarer följer även med och underlättar programmeringen av HMI då man enkelt ser exempelvis enhet.

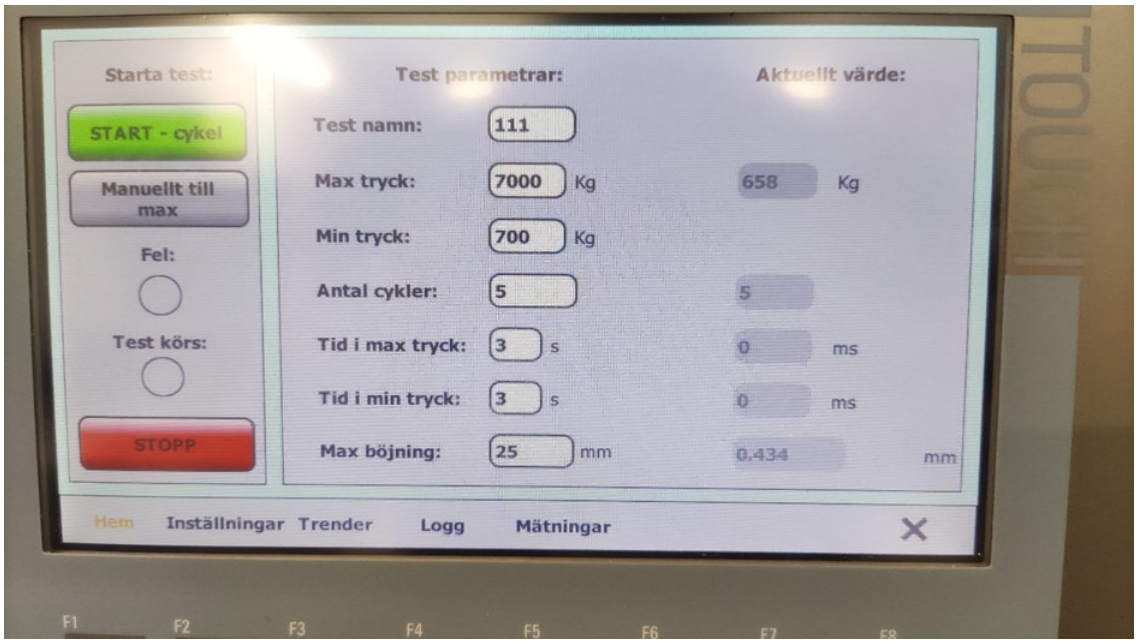
Parameters

Name	Data type	Connection	PLC name	PLC tag	...	...	Acquisition cycle	Source comment
HMIData_Parameters_DelayTimeMaxPressure	DInt	HMI_Connectio...	PLC_1	HMIData.Parameters.DelayTimeMaxPressure		...	1 s	s
HMIData_Parameters_DelayTimeMinPressure	DInt	HMI_Connectio...	PLC_1	HMIData.Parameters.DelayTimeMinPressure		...	1 s	s
HMIData_Parameters_LoggingInterval	DInt	HMI_Connectio...	PLC_1	HMIData.Parameters.LoggingInterval		...	1 s	s
HMIData_Parameters_MaxBending	UInt	HMI_Connectio...	PLC_1	HMIData.Parameters.MaxBending		...	1 s	mm
HMIData_Parameters_MaxPressure	UInt	HMI_Connectio...	PLC_1	HMIData.Parameters.MaxPressure		...	1 s	kg
HMIData_Parameters_MinPressure	UInt	HMI_Connectio...	PLC_1	HMIData.Parameters.MinPressure		...	1 s	kg
HMIData_Parameters_TestCycles	UDInt	HMI_Connectio...	PLC_1	HMIData.Parameters.TestCycles		...	1 s	
HMIData_Parameters_TestName	UInt	HMI_Connectio...	PLC_1	HMIData.Parameters.TestName		...	1 s	

Figur 17. Tagglistan Parameters i HMI-panelen.

3.5 HMI

Det fanns klara direktiv från kundens sida hur användargränssnittet skulle vara uppbyggt i HMI-panelen. De viktigaste parametrarna och driftdata skulle alla vara synliga på huvudsidan, likaså huvudsekvensens styrning och diagnostik.



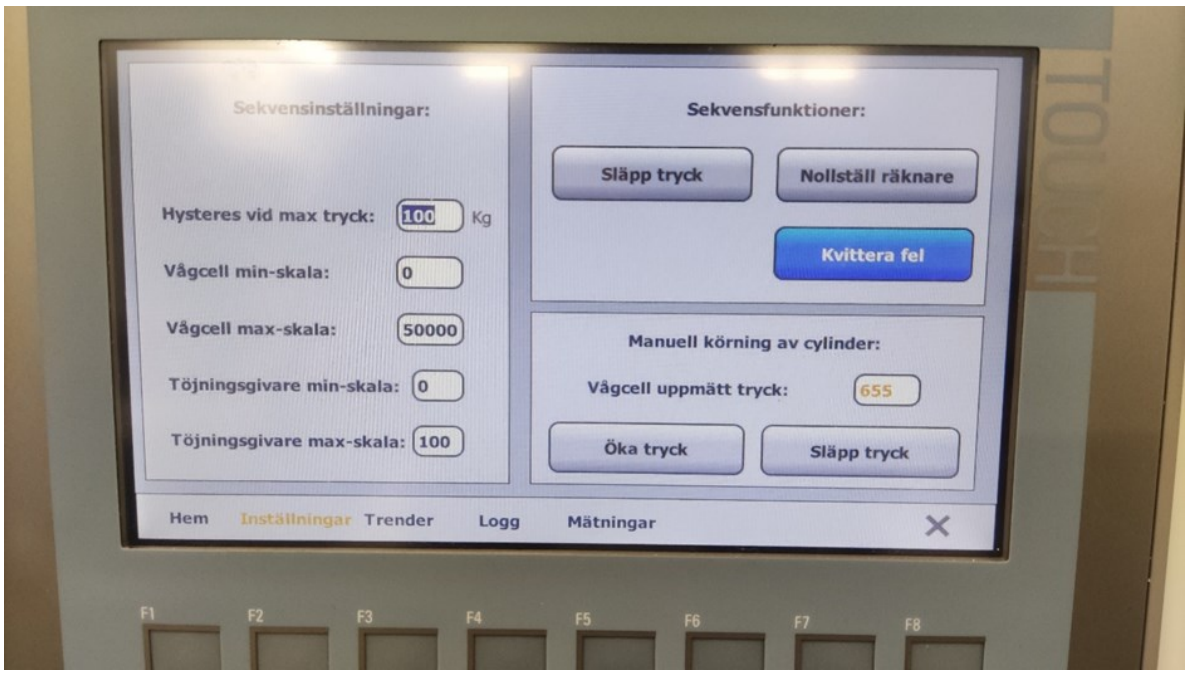
Figur 18. HMI-Panelens huvudsida.

- Till vänster finns Styrning av huvudsekvens med startknapp för cykliskt test eller manuell körning till inställt maxtryck. Stoppknappen stannar alltid pumpen och öppnar ventilerna oavsett om pressen startats manuellt eller från sekvensen. Här syns också indikering på att sekvensen körs och en summafelsindikering som kan betyda några olika saker, bland annat att pumpens motorskydd löst ut eller att inställt tryck inte uppnåtts inom utsatt tid.

- I mitten ställs testets parametrar in, test-namn parametern syns i OPC loggen för att kunna skilja på data.

- Till höger i skärmen syns aktuella mätvärden bland annat uppmätt tryck från vågcellen och testobjektets böjning.

- I balken längst ner på skärmen syns F1 – F8 knapparnas funktioner, F1-F5 används för att byta sida och F8 för att stänga av programmet och gå till HMI-panelens systemmeny.



Figur 19. Sidan Inställningar i HMI-panelen.

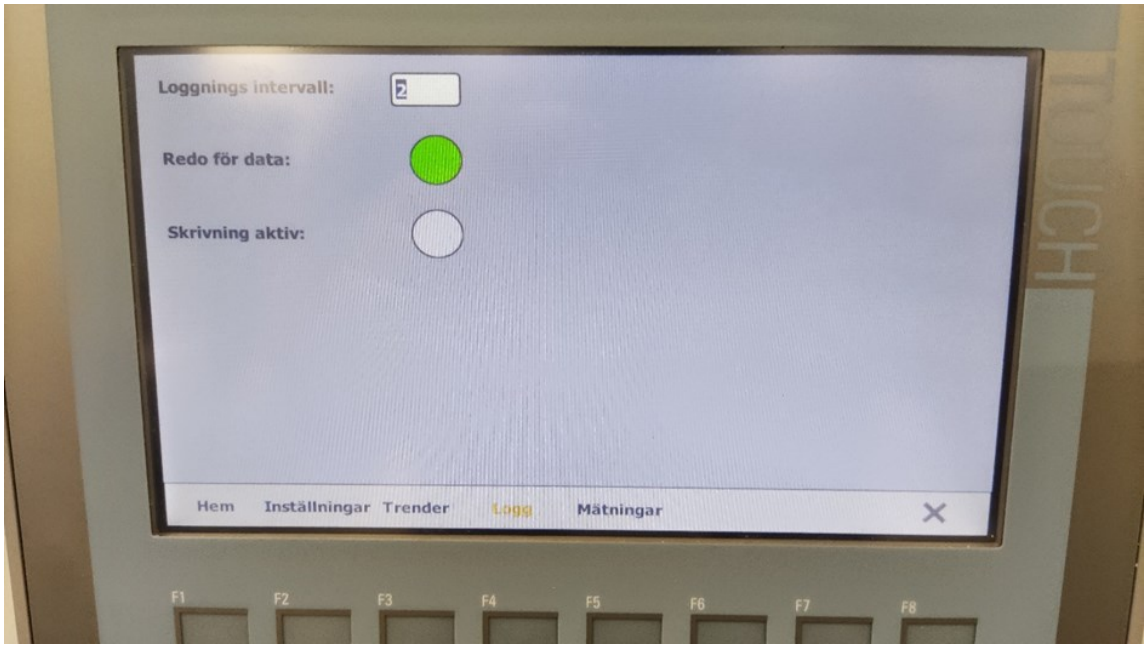
- Till vänster syns sekvensinställningar som inte behöver ändras lika ofta eller inför enskilda test.

- Uppe till höger syns övriga sekvensfunktioner som inte rymdes på hemskärmen, knappen släpp tryck kör pressen tillbaka till inställt minimitryck om manuellt till max funktionen körts och man inte vill starta sekvensen. Nollställ räknare nollar antal testcykler och startar därmed om testet, kvittera fel knappen måste aktiveras före man kan starta sekvensen igen efter ett fel, alla fel som aktiverar indikeringen på hemskärmen stannar också sekvensen i väntan på kvittering som en säkerhetsåtgärd.

- Nere till höger syns helt manuell styrning av press var man antingen kan öka eller släppa trycket manuellt så länge knapparna trycks in. Här visas också aktuellt tryck för säkrare användning.

Figur 20. Sidan Trender i HMI-panelen.

Sidan trender visar uppmätt tryck och position över tid. Samma data visas som loggas via OPC loggern, men här tas den från PLC:ns interna logg vilket betyder att tidsintervallet är ganska begränsat.



Figur 21. Sidan Logg i HMI-panelen.

På denna sida kan loggningsintervallet för PLC loggen ställas in. Dessutom visas status på skrivning och om USB-minnet är anslutet.

3.6 Dataloggning

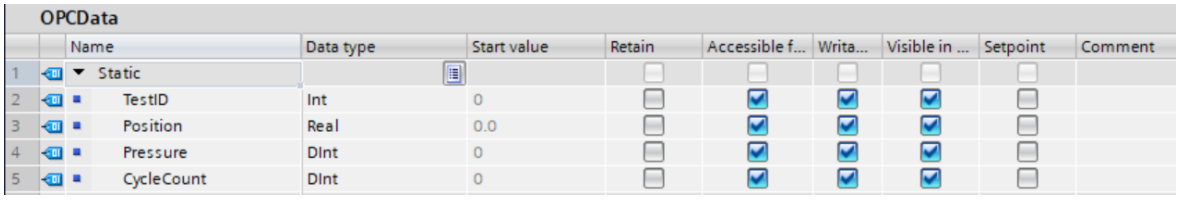
Flera alternativ för dataloggning övervägdes efter att den interna loggen i PLC:n inte klarade av kravet på lagrad mängd data.

OPC UA valdes att användas eftersom en OPC server finns som standard i siemens 1200 serie från och med FW version 4.5.

3.6.1 Aktivering av OPC UA server i PLC:n

Dataloggningsdelen testades i ett skilt projekt innan det integrerades i anläggningens styrsystem. Ett skilt datablock DB11 OPCData skapades för det data som skall loggas, i detta datablock skapas följande variabler:

- Testnamn
- Positionsvärde (Böjning på testobjekt)
- Belastning av testobjektet (skalat värde från lastcell)
- Antal genomförda cykler

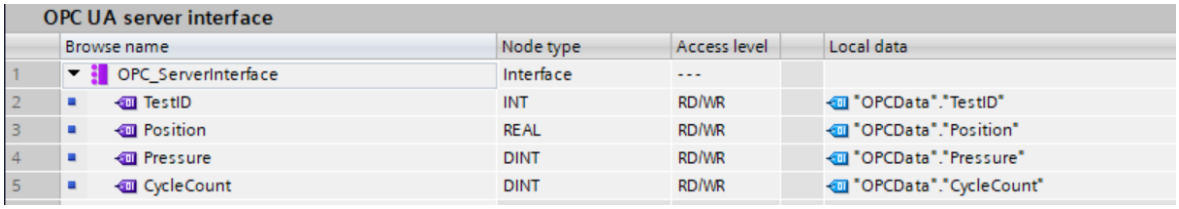


OPCData

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	Static								
2	TestID	Int	0		☑	☑	☑		
3	Position	Real	0.0		☑	☑	☑		
4	Pressure	DInt	0		☑	☑	☑		
5	CycleCount	DInt	0		☑	☑	☑		

Figur 22. Datablocket OPC_Data.

En ny OPC UA server läggs till med namn OPC_ServerInterface, under servern läggs variablerna som nyss skapades i DB11 till.

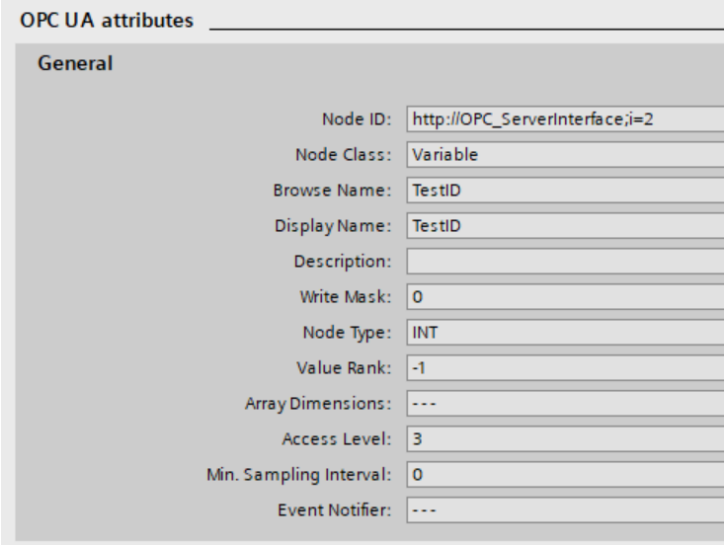


OPC UA server interface

	Browse name	Node type	Access level	Local data
1	OPC_ServerInterface	Interface	---	
2	TestID	INT	RD/WR	"OPCData"."TestID"
3	Position	REAL	RD/WR	"OPCData"."Position"
4	Pressure	DINT	RD/WR	"OPCData"."Pressure"
5	CycleCount	DINT	RD/WR	"OPCData"."CycleCount"

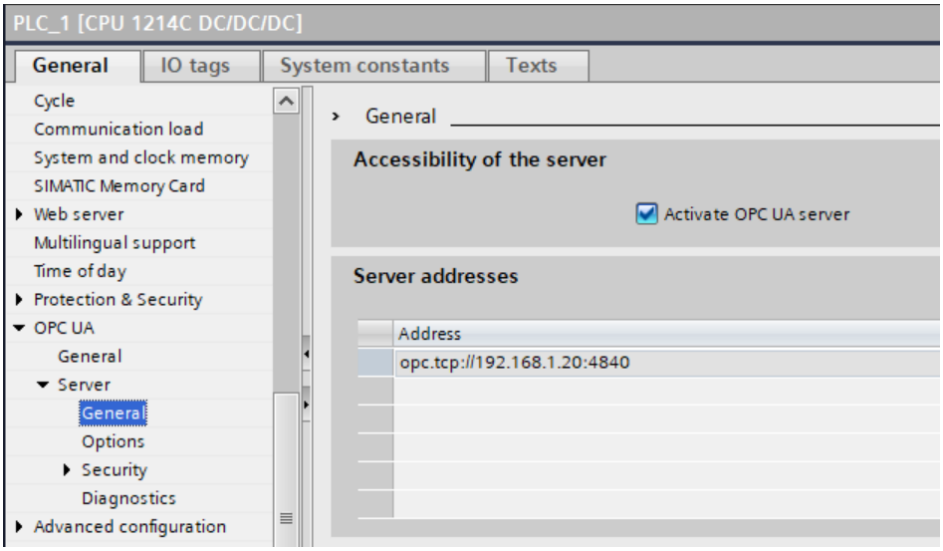
Figur 23. OPC UA servern i PLC:n.

Ifall man öppnar egenskaperna för ett objekt i OPC UA servern kan man se vilka olika attribut det innehåller. Detta är alltså den information som klienten får angående data-typ, namn, skriv och läs rättigheter mm.



Figur 24. OPC UA Attribut för objektet TestID.

Det sista som behöver göras i PLC:n för att förbereda kommunikationen är att söka efter serverns URL adress, den hittas under PLC egenskaperna i undermenyn OPC UA. På samma sida aktiveras också åtkomst till servern utifrån. Här finns också omfattande säkerhetsinställningar för kryptering, användarnivåer och certifikat men eftersom detta är ett isolerat nätverk används i detta fall inga säkerhetsfunktioner.



Figur 25. OPC UA URL och egenskaper.

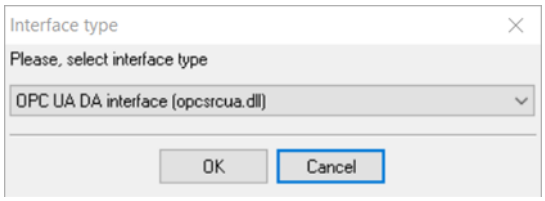
3.6.2 Konfiguration av OPC UA klienten

Det finns många alternativ att välja mellan då det kommer till PC baserade verktyg för att logga data från en OPC server. Flera alternativ övervägdes, men slutligen föll valet på ett program som heter Advanced OPC data logger från AGG software.

Valet föll på detta program eftersom de har en gratisversion med full funktionalitet som man kan testa mot sitt system innan man köper en licens. Enda begränsningen i gratisversionen är att maximalt 100 värden kan loggas under en session. Då denna mängd uppnås måste programmet startas om för att fortsätta användningen.

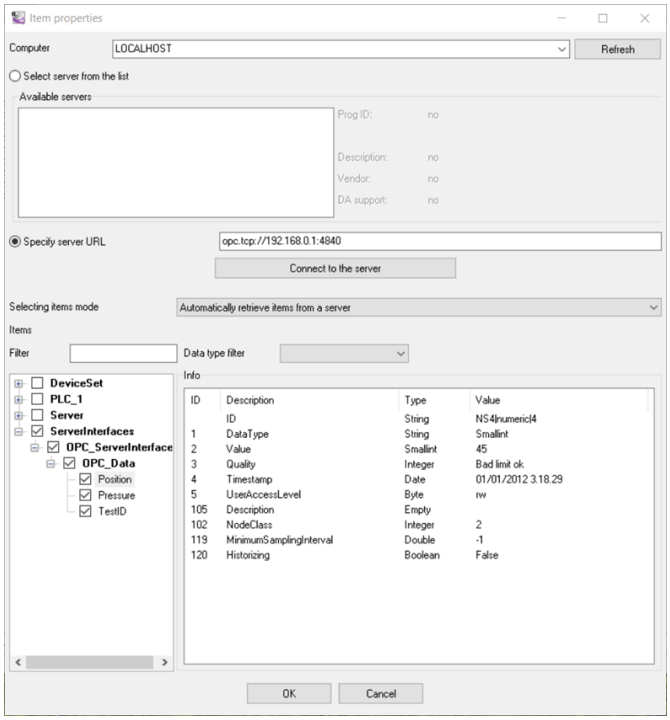
Den fullständiga versionen av programmet är relativt förmånlig och en obegränsad licens erbjuds som inte slutar gälla efter en viss tid. Kunden godkänner inga licenser av prenumerationstyp som måste förnyas regelbundet.

OPC UA klientens konfiguration påbörjas genom att välja typ av gränssnitt för kommunikationen.



Figur 26. Val av OPC gränssnitt i klienten.

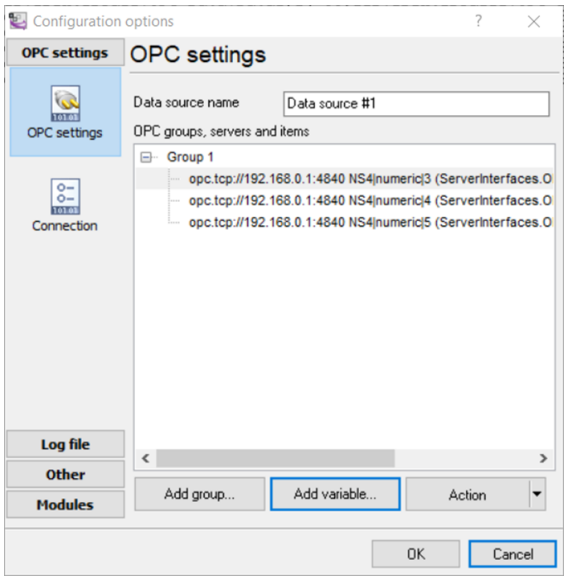
Nästa steg är att lägga in serverns URL adress i programmet och testa anslutningen. Direkt servern ansluter visas de objekt som hittas. Här kan man välja om man vill läsa alla objekt som finns i servern eller endast plocka ut vissa data.



Figur 27. Serverns egenskaper i programmet Advanced OPC datalogger

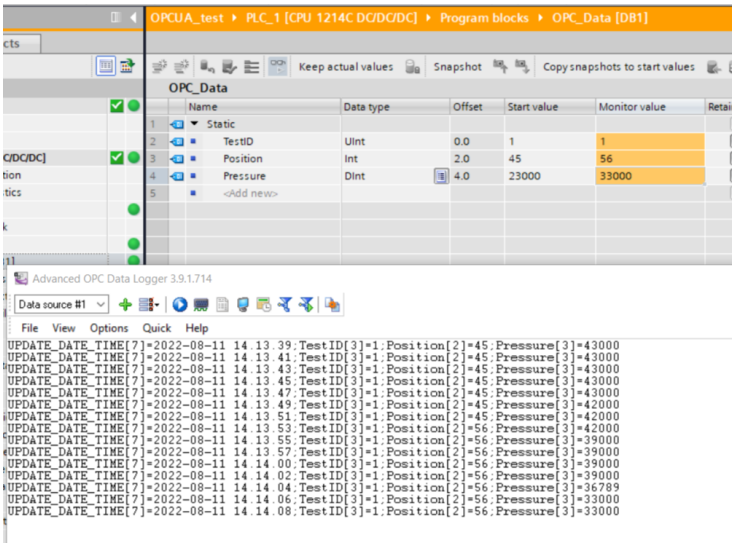
I figur 27 syns vilka objekt som klienten hittar i OPC UA servern, där syns också attributen som anger data-typ, namn och tidsstämpel mm.

Då serven lagts till visas alternativ för att lägga till flera servrar (figur 28). Eftersom detta alternativ finns kunde programmet Advanced OPC datalogger också användas för betydligt större anläggningar och projekt, exempelvis för att logga data från en hel produktionslinje.



Figur 28. Konfiguration av servern.

I och med detta är serverkonfigurationen färdig och det har blivit dags att testa kommunikationen mellan OPC UA servern i PLC:n och OPC UA klienten på PC:n. I figur 29 syns testet, värden matas in i datablocket DB11 OPC_Data i PLC:n och det kontrolleras att samma värden också syns i klienten.



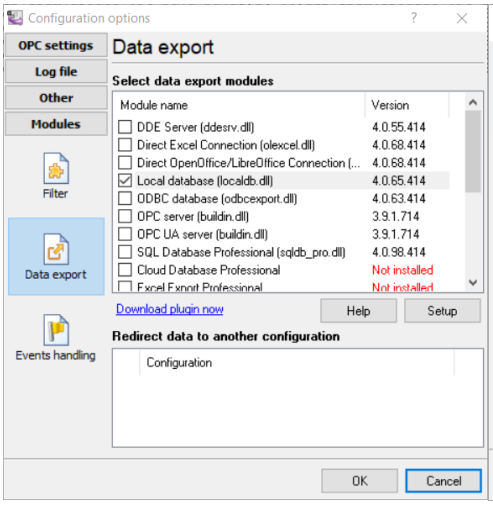
Figur 29. Test av OPC UA kommunikationen

Efter denna konfiguration testades att kommunikationen fungerade. Än så länge visas endast data, ingen logg sparas.

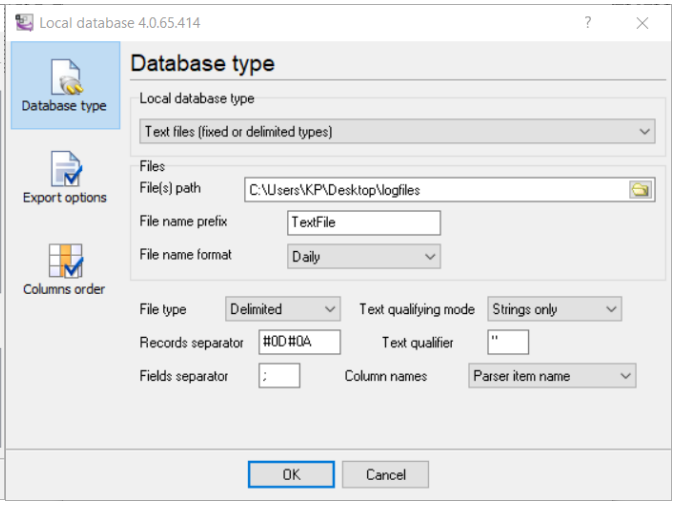
Klienten konfigurerades så att den cykliskt läser data från servern med en cykeltid på 1s.

3.6.3 Inställning av dataexport från klienten

För att få data loggat till en databas eller fil måste ytterligare konfiguration utföras i Advanced OP Data logger. Programmet stöder många olika standarder för dataexport. I detta fall anses en text eller CSV-fil tillräcklig för ändamålet, men olika databassystem exempelvis SQL hade också varit möjliga alternativ.



Figur 30. Inställning av dataexport



Figur 31. Inställning av textfilens uppbyggnad.

För att få data exporterat till en CSV-fil behöver ett tillägg installeras i programmet som heter Local database. Då det är installerad kan tillägget väljas från Dataexport konfigurationsfönstret enligt figur 30. Öppnas egenskaperna för tillägget hittas olika alternativ för filens uppbyggnad var bland annat separatorer kan ändras. Här anges också var i filsystemet loggfilerna skall sparas och hur namnet på filen skall byggas upp.

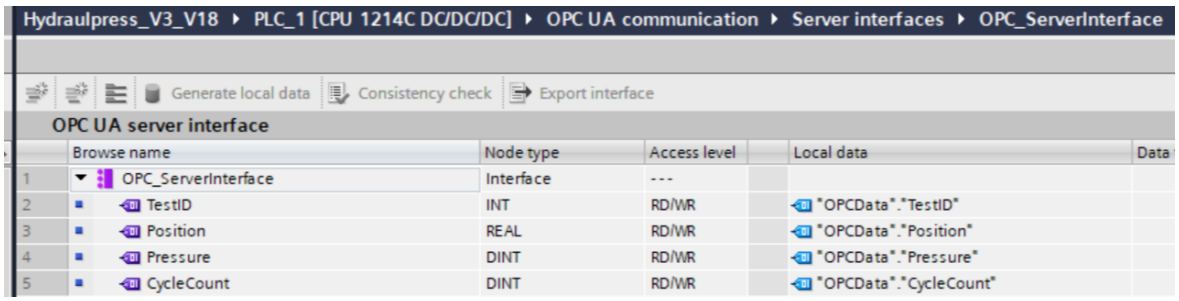
Programmet ställs in så att en ny loggfil skapas automatiskt om ingen finns sedan tidigare, finns en befintlig fil öppnas den och loggen fortsätter där den slutade.

Då inställningarna är gjorda testas formatet på filen genom att importera innehållet till Microsoft Excel enligt figur 32.

	A	B	C	D	E
1	DATE_TIME_STAMP	Position	Pressure	TestID	
2	11/08/2022 14.13	45	43000	1	
3	11/08/2022 14.13	45	43000	1	
4	11/08/2022 14.13	45	43000	1	
5	11/08/2022 14.13	45	43000	1	
6	11/08/2022 14.13	45	43000	1	
7	11/08/2022 14.13	45	43000	1	
8	11/08/2022 14.13	45	42000	1	
9	11/08/2022 14.13	45	42000	1	
10	11/08/2022 14.13	56	42000	1	
11	11/08/2022 14.13	56	39000	1	
12	11/08/2022 14.13	56	39000	1	
13	11/08/2022 14.14	56	39000	1	
14	11/08/2022 14.14	56	39000	1	
15	11/08/2022 14.14	56	36789	1	
16	11/08/2022 14.14	56	33000	1	
17	11/08/2022 14.14	56	33000	1	
18					
19					
20					
21					
22					

Figur 32. Loggfilen importerad till Microsoft Excel.

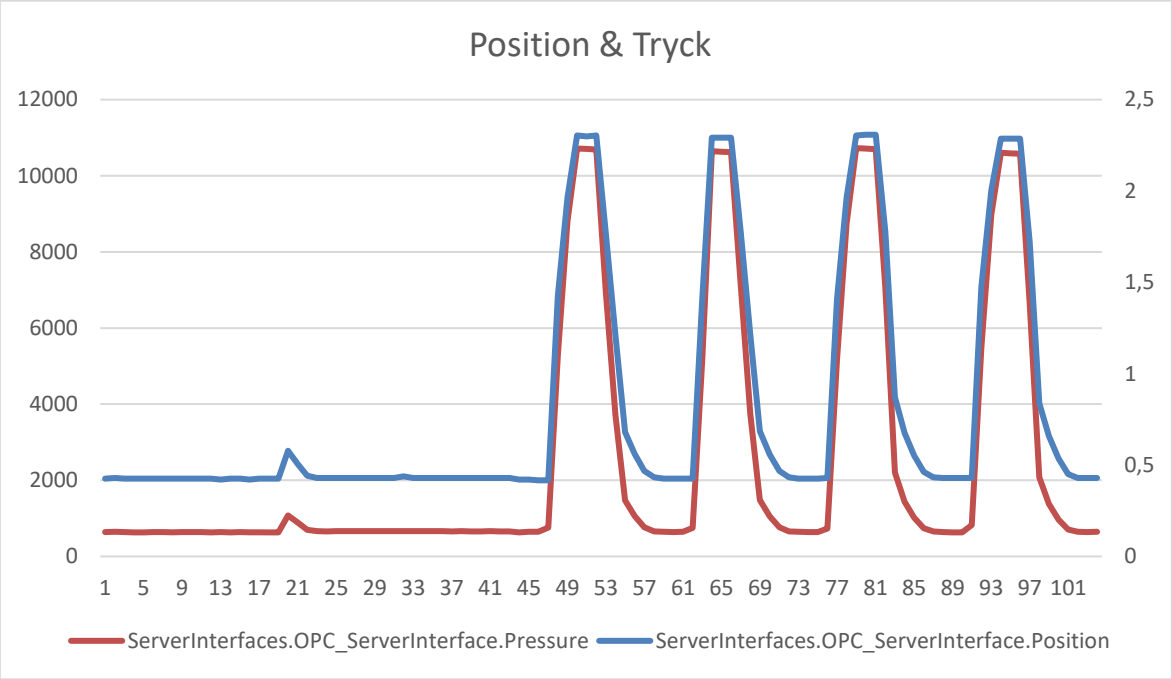
Rubrikerna på kolumnerna tas automatiskt från OPC UA klienten. Den lägger också till en tidsstämpel för när nya värden tagits emot. Efter att detta test genomförts och konfigurationerna är gjorda kopierades testprogrammet till det slutliga PLC-programmet. I samband med detta lades en kolumn till i OPC servern och loggen för att få med antalet genomförda cykler.



Figur 33. OPC UA Servern efter tillagd rad för testcykel.

3.7 Testning av anläggning

Under första testet hittades några buggar i programmet och HMI-panelen som åtgärdades, vissa fördröjningar kortades av vid mån av möjlighet för att uppnå en så kort cykeltid som möjligt. Ett kort första test med endast 4 cykler kördes. Testdata importerad till Excel såg ut enligt följande:



Figur 34. Resultat från första testkörningen.

På X-axeln syns tiden i sekunder, på Y-axeln till vänster tryck i kg och till höger böjningen i mm.

Den orangea kurvan motsvarar trycket från lastcellen och den blåa positionsmätningen från absolutpulsgivaren.

Från grafen kan avläsas att anläggningen hinner med ungefär 4 cykler per minut. Detta innebär att ca 5700 cykler kan genomföras och loggas under ett dygn.

4 Resultat

Resultatet blev en fungerande helhet som förhoppningsvis kommer ge slutkunden viktiga data för framtida produktutveckling. Längre tester kördes under hösten 2023 och anläggningen har visat sig fungera pålitligt.

Ett normalt test har många tusentals cykler och tar ca en vecka att genomföra, längre tester är också planerade. Detta sätter stora krav på mekaniken i anläggningen som troligtvis kommer vara den delen som kräver mest förbättringar.

Figur 35. Systemet efter genomfört test med 5 cykler.

Under projektets gång gjordes en del ändringar på anläggningen, de flesta av ändringarna gällde hydrauliken. Det visade sig vara svårt att hitta en kompakt hydraulpump som klarade det höga trycket i anläggningen utan att överhetta oljan. Detta resulterade i att flera pumpar gick sönder under testkörningen. Innan pumpen som användes i den slutliga installationen installerades, testades tre andra modeller. Varav en gjorde att pressen blev alldeles för långsam, de två andra var snabba men gick sönder efter ungefär två veckor var. Detta gjorde att styrningen av pumparna behövde ändras lite varje gång modellen blev bytt vilket drog ut på de slutliga testen.

I detta skede har inte några externa töjningsgivare fästa på testobjektet tagits i bruk eller testats, men möjligheten finns att ta i bruk upp till 3 stycken med nuvarande konfiguration om kunden anser att data från dessa kunde vara användbar i produktutvecklingen.

Både jag och kunden är nöjda med resultatet, vi uppnådde en tillräckligt hög precision för att data som samlas skall vara användbar i utvecklingen av företagets produkter. Längre tester kommer att visa om förbättringar behövs inom något område.

5 Slutdiskussion

Slutresultatet med fokus på styrsystemet blev ganska långt som det var planerat från början med endast små ändringar i styrning av de olika pumparna. Möjliga kommande ändringar i anläggningens styrsystem kommer troligtvis mest gälla loggningen av data.

Datalogen kunde vid behov ändras till ett annat format än CSV, till exempel användning av någon databas som SQL. En begränsning nu är faktiskt Excels radantal på lite över 1 miljon rader, denna datamängd uppnås efter lite över en vecka med ett loggningsintervall på 2 sekunder och försvårar importen av data en aning.

Datamängden blir ju också nästan dubbelt större ifall töjningsgivare tas i bruk i framtiden och data från dessa också skall loggas.

En annan sak som kunde förbättras är kalibreringen av absolutpulsgivaren. I ett projekt som har utförts senare med samma typ av givare hittades en funktion som möjliggör kalibrering av givaren direkt i enheten i stället för endast i PLC-programmet. PLC-programmet skickar

ett kommando över PROFINET som gör att givaren använder den aktuella positionen som en ny nollpunkt. Denna nollpunkt sparas också efter strömavbrott.

Den mest utmanade delen med projektet och som var nytt för mig var att hitta en passande lösning för dataloggningen som uppfyllde kundens kriterier på användarvänlighet, kostnad och funktioner.

Slutligen vill jag tacka företaget Noswing som gett mig möjligheten att genomföra detta projekt. Det har gett både mig och företaget ny kunskap speciellt inom OPC UA och dataloggning som kommer att komma till användning också i framtida projekt. Allt eftersom standarden industri 4.0 implementeras växer kraven på att samla in data från olika system på ett robust och flexibelt sätt.

6 Källor

[1] PLCopen, ”IEC 61131-3: a standard programming resource,” 2013. [Online]. Available: https://plcopen.org/sites/default/files/downloads/intro_iec_oct2016.pdf. [Använd Mars 2024].

[2] Siemens, ”SIMATIC S7-1200 Programmable controller,” 4 2012. [Online]. Available: https://cache.industry.siemens.com/dl/files/465/36932465/att_106119/v1/s71200_system_manual_en-US_en-US.pdf. [Använd 8 Februari 2024].

[3] G. L. Pratt, ”WHICH IEC 61131-3 PROGRAMMING LANGUAGE IS BEST? PART 1,” *Control engineering,* 2020.

[4] G. L. Pratt, ”WHICH IEC 61131-3 PROGRAMMING LANGUAGE IS BEST? PART 2,” *Control engineering,* 2020.

[5] Inductive automation, ”What is HMI, Common uses, Trends and the future of HMI,” Inductive automation, 10 Augusti 2018. [Online]. Available: https://inductiveautomation.com/resources/article/what-is-hmi. [Använd 13 Mars 2024].

[6] Siemens, ”Learn-/Training document,” 2018. [Online]. Available: https://www.automation.siemens.com/sce-static/learning-training-documents/tia-portal/visualization-s7-1200/sce-041-101-wincc-basic-ktp700-s7-1200-r1709-en.pdf. [Använd 24 Februari 2024].

[7] Siemens, ”Programming guidline for S7-1200/1500,” December 2018. [Online]. Available: https://cache.industry.siemens.com/dl/files/040/90885040/att_970576/v1/81318674_Programming_guideline_DOC_v16_en.pdf. [Använd 12 Mars 2024].

[8] Siemens, ”TIA Portal Options,” 2024. [Online]. Available: https://contentpath.siemens.com/l/tia-portal-options. [Använd 26 Februari 2024].

[9] PI International, ”PROFINET Explained,” 2024. [Online]. Available: https://www.profinet.com/profinet-explained. [Använd 3 Mars 2024].

[10] OPC Foundation, ”Unified Architecture,” 2024. [Online]. Available: https://opcfoundation.org/about/opc-technologies/opc-ua/. [Använd 26 Februari 2024].

[11] OPC Router, ”What is OPC UA?,” OPC Router, 2024. [Online]. Available: https://www.opc-router.com/what-is-opc-ua/. [Använd 7 Mars 2024].

[12] AGG Software, ”Advanced OPC data logger,” AGG Software, 2024. [Online]. Available: https://www.aggsoft.com/opc-data-logger.htm. [Använd 9 Februari 2024].

[13] Vetek, ”Data sheet PA6181,” VETEK, [Online]. Available: https://www.vetek.com/en/dynamics/WebFiles/document/1c1df659-5616-48f7-8a5b-64f59c91fc1d/Datasheet_PA6181_V3.pdf. [Använd Mars 2024].

[14] Wikro Systems, *Hydraul aggregat funktionsprincip,* Jakobstad, 2023.

[15] SICK, ”Absolute encoders: AFS/AFM60 Ethernet,” SICK, 2024. [Online]. Available: https://www.sick.com/us/en/catalog/products/motion-control-sensors/absolute-encoders/afsafm60-ethernet/afm60a-s1nb000s32/p/p675520?tab=detail. [Använd 15 Mars 2024].

[16] SICK, ”AFS/AFM60 PROFINET Operating instructions,” 13 December 2021. [Online]. Available: https://cdn.sick.com/media/docs/9/99/099/operating_instructions_afs_afm60_profinet_absolute_encoder_en_im0047099.pdf. [Använd 15 Februari 2024].

[17] Siemens, ”6ES7214-1AG40-0XB0,” Siemens, 2024. [Online]. Available: https://support.industry.siemens.com/cs/pd/255094?pdti=pi&dl=en&lc=en-US. [Använd 2024].

Bilagor

Bilaga 1 programkoden i FB1 MainSequence

```
//Manual start to max pressure
IF #SequenceRunning = 0 AND #ManualToMax = 1 AND #StopSequenceError = 0 THEN
    #ManualRunning := 1;
    #SequenceStage := 1;
END_IF;

//Start automatic
IF #SequenceRunning = 0 AND #ManualRunning = 0 AND #StartCycleTest = 1 AND #PositionZeroed = 1 AND #PressureReleaseDelay = 0 AND #StopSequenceError = 0 THEN
    #SequenceRunning := 1;
    #SequenceStage := 1;
END_IF;

CASE #SequenceStage OF
    1:  // Statement section case 1
        #Running := 1;
        #IncreasePressure := 0;
        IF #ActualPressure > #MinPressure OR #Unstuck = 1 THEN
            #PressureRelease := 1;
        ELSIF #ActualPressure <= #MinPressure - 50 THEN
            #PressureRelease := 0;
            #StartTimerMin := 1;
        ELSE
            #StartTimerStuck := 1;
        END_IF;
        ;
    2:
        IF #ActualPressure < (#MaxPressure - #HysteresisMaxPressure) THEN
            #IncreasePressure := 1;
            #ErrorTimeOutStart := 1;
            #ErrorTimeOutPressure := 1;
        ELSE
            #IncreasePressure := 0;
            #ErrorTimeOutStart := 0;
            #ErrorTimeOutPressure := 0;
        END_IF;

        IF #SequenceRunning = 1 AND #ActualPressure > (#MaxPressure - #HysteresisMaxPressure) THEN
            #StartTimerMax := 1;
            IF #CycleCounter >= #Cycles THEN
                #SequenceFinished := 1;
            END_IF;
        END_IF;
        ;
    ELSE  // Statement section ELSE
        ;
END_CASE;


//Timer at min pressure
//
#IEC_Timer_0_Instance(IN:=#StartTimerMin,
                      PT:=#TimeAtMinPressure,
                      Q=>#MinPressureDelayDone,
                      ET=>#TimeLeftMinDelay);

IF #MinPressureDelayDone = 1 THEN
    #SequenceStage := 2;
    #StartTimerMin := 0;
END_IF;

```

```
//Time delay at max pressure
#IEC_Timer_0_Instance_1(IN:=#StartTimerMax,
                        PT:=#TimeAtMaxPressure,
                        Q=>#MaxPressureDelayDone,
                        ET=>#TimeLeftMaxDelay);

IF #MaxPressureDelayDone = 1 AND #CycleCounter < #Cycles THEN
    #CycleCounter := #CycleCounter + 1;
    #StartTimerMax := 0;
    #SequenceStage := 1;
END_IF;

//Stop sequence conditions
IF #BeamPosition > #MaxBendingAllowed OR #Stop = 1 OR #StopSequenceError = 1 THEN
    #ManualRunning := 0;
    #SequenceRunning := 0;
    #Running := 0;
    #IncreasePressure := 0;
    #PressureRelease := 0;
    #SequenceStage := 0;
    #StartTimerMax := 0;
    #StartTimerMin := 0;
    #SequenceDone := 0;
    #ErrorTimeOutStart := 0;
    #ErrorTimeOutPressure := 0;
ELSIF #SequenceFinished = 1 THEN
    #ManualRunning := 0;
    #SequenceRunning := 0;
    #Running := 0;
    #IncreasePressure := 0;
    #PressureRelease := 1;
    #SequenceStage := 3;
    #StartTimerMax := 0;
    #StartTimerMin := 0;
    #SequenceFinished := 0;
    #SequenceDone := 1;
    ;
END_IF;

//Start pump when running
#StartPump := #Running;

//Release pressure when sequence is done
IF #SequenceDone = 1 AND #ActualPressure < #MinPressure THEN
    #PressureRelease := 0;
    #SequenceDone := 0;
    #SequenceStage := 0;
END_IF;

//Reset cycle counter
IF #Reset = 1 THEN
    #CycleCounter := 1;
END_IF;

//Manual pressure release
IF #ReleasePressure = 1 AND #PressureReleaseDelay = 0 AND #SequenceRunning = 0 AND #ManualRunning = 0 THEN
    #PressureRelease := 1;
    #PressureReleaseDelay := 1;
END_IF;
```

```
#IEC_Timer_0_Instance_2(IN:=#PressureReleaseDelay,
                        PT:=t#10s,
                        Q=>#pressurereleasedelaydone);

IF #pressurereleasedelaydone = 1 THEN
    #PressureRelease := 0;
    #PressureReleaseDelay := 0;
END_IF;

 #CurrentCycleCount := #CycleCounter;

 //Manual direct control
IF #SequenceStage = 0 AND #PressureReleaseDelay = 0 THEN
    #IncreasePressure := #IncreasePressureMan;
    #PressureRelease := #ReleasePressureMan;

    ;
END_IF;

 //Timeout error
IF #ActualPressure > #MinPressure THEN
    #ErrorTimeOutStart := 0;
END_IF;

#IEC_Timer_0_Instance_3(IN:=#ErrorTimeOutStart,
                        PT:=t#12s,
                        Q=>#PumpFault);

#IEC_Timer_0_Instance_4(IN := #ErrorTimeOutPressure,
                        PT := t#45s,
                        Q => #PressureNotAchived);

IF #PumpFault = 1 OR #PressureNotAchived = 1 OR #Motorbreaker = 1 THEN
    #StopSequenceError := 1;
END_IF;

 #Error := #StopSequenceError;

 //Error reset
IF #ErrorReset = 1 THEN
    #StopSequenceError := 0;
END_IF;

 //IF stuck in stage 1 release pressure after delay
#IEC_Timer_0_Instance_5(IN:=#StartTimerStuck,
                        PT:=t#1s,
                        Q=>#Unstuck);

IF #ActualPressure <= #MinPressure - 50 THEN
    #Unstuck := 0;
END_IF;
```