YRKESHÖGSKOLAN
NOVIA

Uppdatering av PLC-applikation med HMI för styrning av Inteco Multitank-system

Simon Kärkkäinen

Examensarbete för ingenjör (YH)-examen

Utbildningsprogrammet för el- och automationsteknik

Vasa 2025

EXAMENSARBETE

Författare: Simon Kärkkäinen
Utbildning och ort: El- och automationsteknik, Vasa
Inriktning: Automationsteknik
Handledare: Joachim Böling

Titel: Uppdatering av PLC-applikation med HMI-interface för styrning av Inteco Multitank-system

---

Datum: 15.05.2025 Sidantal: 24 Bilagor: 1

Abstrakt

Detta examensarbete behandlar uppdateringen av mjukvaran samt HMI:n för ett befintligt styrsystem som används för styrningen av ett Inteco flertankssystem. Detta för att i framtiden kunna justera och optimera olika processer för experiment och laborationer. Programmeringen är gjord i Siemens TIA Portal och systemet bygger på en Siemens S71215C-kontroller ansluten till en SIMATIC KTP700 Basic HMI-skärm. Syftet med arbetet var att åtgärda funktionella begränsningar i det tidigare programmet och skapa ett mera användarvänligt gränssnitt för kalibreringen och styrningen av nivågivare, pumpen och ventilerna.

Arbetet fokuserade till stor del på att förbättra kalibreringsmöjligheterna genom implementering av nya HMI-skärmar, funktioner samt datablock för lagring av kalibrerade värden. PWM-styrningen av pumpen och ventilerna uppdaterades för mera exakt styrning, vilket även förbättrade PID-styrningen av vattentankarna. Genom dessa ändringar kunde vattennivån i tankarna effektivt styras manuellt eller med PID-styrning.

Projektet resulterade i ett mera användarvänligt program lämpat för utbildningsändamål. Programmet kan justera vätskenivåerna i tankarna antingen manuellt eller med hjälp av PID-styrning via HMI-skärmen. HMI-skärmens användargränssnitt har uppdaterats och gjorts mera intuitivt, med visuell överblick av vätskenivån vid kalibreringen av varje tank samt ett mindre komplicerat sätt att kalibrera dessa. En uppdaterad användarmanual för programmet har även gjorts.

Språk: svenska
Nyckelord: PLC, HMI, PID, kalibrering

OPINNÄYTETYÖ

Tekijä: Simon Kärkkäinen
Koulutus ja paikkakunta: Sähkö – ja automaatiotekniikka, Vaasa
Suuntautumisvaihtoehto: Automaatiotekniikka
Ohjaaja: Joachim Böling

Nimike: PLC:n sekä HMI:n päivitys Inteco-monisäiliöjärjestelmälle

---

Päivämäärä 15.05.2025 Sivumäärä 24 Liitteet 1

Tiivistelmä

Tämä opinnäytetyö käsittelee olemassa olevan järjestelmän ohjelmiston sekä HMI:n päivittämistä Inteco-monisäiliöjärjestelmän ohjausta varten. Tämä mahdollistaa tulevaisuudessa erilaisia laboratoriokokeita. Ohjelmointi on toteutettu Siemens TIA Portaalissa ja järjestelmä perustuu Siemens S71215C säätöyksikköön, joka on kytketty SIMATIC KTP700 Basic kosketusnäyttöön.

Työ keskittyi erityisesti ohjelman kalibrointimahdollisuuksien parantamiseen toteuttamalla uudet HMI-näytöt, toiminnot sekä tietolohkot kalibrointiarvojen tallentamista varten. Pumpun ja venttiilien PWM-ohjaus päivitettiin mahdollistamaan tarkemman säädön, jonka päivitys myös paransi säiliöiden PID-säätöä. Näiden muutoksien avulla säiliöiden nestepinnan ohjauksen pystyy tehdä joko manuaalisesti tai PID-säädöllä.

Tuloksena on käyttäjäystävällisempi ohjelma, joka soveltuu opetuskäyttöön. HMI-näytön kautta pystyy säiliöiden nestetasoa säätää joko manuaalisesti tai PID-säädön avulla. HMI-näytön käyttöliittymä on tehty intuitiivisemmaksi, sisältäen visuaalisen näkymän nestetasosta jokaisen säiliön kalibroinnin aikana sekä yksinkertaisemman kalibrointiprosessin. Ohjelmalle on lisäksi tehty päivitetty käyttöohje.

Kieli: ruotsi
Avainsanat: PLC, PID, HMI, kalibrointi

BACHELOR'S THESIS

Author: Simon Kärkkäinen
Degree Programme: Electrical Engineering and Automation, Vaasa
Specialization: Automation
Supervisor(s): Joachim Böling

Title: Update of PLC control and HMI Interface for the Inteco Multitank System

---

Date 15.05.2025 Number of pages 24 Appendices 1

Abstract

This thesis addresses updating the software and HMI of an existing control system used for an Inteco Multitank System. This is in order to enable future experiments and laboratory tasks, with varying adjustments of various processes. Siemens TIA Portal was used for the programming, and the system consists of a Siemens S71215C controller connected to a SIMATIC KTP700 Basic panel. The goal of this project was to resolve the functional limitations of the previous program and create a more intuitive interface for the calibration and controlling of the pump, valves, and level sensors.

The work mainly focused on improving the calibration capabilities of the system. This is achieved by implementing new HMI screens, functions, and data blocks for storing new calibrated values. The PWM control of the pump and valves was updated to allow for a more precise regulation, which in turn also enhanced the PID control. These improvements allowed for the water tank levels to be controlled either manually, or via PID regulation.

The result is a more user-friendly program suited for educational purposes. The program, through the use of the HMI Interface, now allows for adjustments of water levels either manually or through PID control. The HMI interface has been redesigned and made more intuitive, offering a visual overview of the liquid levels during the calibration of each tank, as well as a simplified calibration method. An updated user manual for the program has also been made.

Language: Swedish
Key words: PLC, HMI, PID, calibration

Ordförklaringar

PLC	Programmable Logic Controller
PID	Proportionell, Integrerande, Deriverande-regulator
HMI	Human Machine Interface
I/O	In-och utgångssignaler
TIA	Totally Integrated Automation
OB	Organisationsblock
FB	Funktionsblock
FC	Funktion
CPU	Central Processing Unit
PWM	Pulse Width Modulation
DB	Data-block
IDB	Instans Data-Block
PIQ	Utgångsinformation i CPU-minnet för PLC
PII	Ingångsinformation i CPU-minnet för PLC

Innehållsförteckning

1 Inledning

Detta examensarbete är gjort på uppdrag av Yrkeshögskolan Novia. Arbetet består av uppdatering av ett existerande styrsystem, var styrsystemet och dess styrenheter programmeras och testkörs. Systemet är programmerat i Siemens TIA portal V17 och baserar sig på en Siemens PLC, en Siemens HMI samt ett styrdon kopplat till ett Inteco multitank system. Projektet är en fortsättning på ett tidigare examensarbete var programmet överfördes från MATLAB till Siemens TIA Portal.

Till en början berättas i detta examensarbete om teori gällande grundläggande funktioner för multitanksystemet, samt organisationsblock och funktioner för Siemens TIA Portal.

En uppdaterad användarhandbok för HMI- gränssnittet kommer även bifogas som bilaga för att underlätta användningen av systemet.

2 Problem

Befintliga programmet för flertankssystemet hade vissa funktionella begränsningar som behövde åtgärdas för att möjliggöra tillförlitlig användning vid framtida laborationer och experiment. Vid PID-styrningen av tankarna observerades att de analoga ingångarna momentant sjönk till noll, vilket påverkade användbarheten av PID-styrningen.

HMI-användargränssnittet för ställningen av min- och maxnivån av tankarna fungerade genom att manuellt skriva in ett värde mellan 0-27648, vilket motsvarar 0–100%. Denna metod upplevdes mindre intuitiv och kunde försvåra användningen. Dessutom sparades inte kalibreringsvärdena i programmet, vilket innebar att kalibreringen av varje tank, pumpen och ventilerna behövde upprepas vid varje omstart.

Vidare noterades att pumpens och ventilernas on/off knappar i HMI:n inte var riktigt responsiva och kunde behöva flera tryck förrän aktivering, vilket påverkade användarupplevelsen.

3 Teori

I detta avsnitt förklaras centrala begrepp relevanta för flertankssystemet som PLC, HMI, PWM och PID-reglering, för att ge läsaren en fördjupad förståelse för systemets uppbyggnad och principer.

3.1 PLC

PLC är en förkortning av Programmable Logic Controller, eller programmerbar logisk styrenhet på svenska. En PLC är en mikroprocessorbaserad enhet som används främst i industriellt bruk för att automatisera, styra samt övervaka maskiner och processer. PLC:n har ett programmerbart minne där instruktioner lagras för att utföra funktioner som logik, sekvenser och beräkningar. Den utvecklades som ett flexibelt alternativ till traditionella reläbaserade styrsystem, vilket minimerade behovet av ompkopplingar vid ändringar av styrlogik.

En typisk PLC består vanligtvis av en central processor (CPU), minnesenheter, en programmeringsenhet, en kommunikationsmodul samt in- och utgångsgränssnitt för anslutning till olika utrustningar som sensorer och motorer. Gränssnitten säkerställer pålitlig funktion för PLC:n i form av signalbehandling och elektrisk isolering.

PLC:er är konstruerade för att vara stabila, användarvänliga och lättprogrammerabara, detta genom användning av standardiserade programmeringsspråk enligt IEC61131-3 standarden som ladder diagram, funktionblocksdiagram, strukturerad text och sekventiella funktionscheman. I detta fall används Siemens TIA Portal, vilket är mycket likt IEC61131-3 standarden, men med egna varianter av kod och funktioner som optimerats för Siemens hårdvara.

PLC:n kan fungera som en ensamtående enhet med ett specifikt antal inbyggda in- och utgångar, dessa PLC:er är lämpade för mindre komplexa applikationer och processer. Alternativt kan ett PLC-system vara modulärt, det vill säga uppbyggt av flera olika komponenter, såsom CPU och en kombination av analoga och digitala in- och utgångsmoduler. Dessa kan monteras i en rack eller chassi, var systemet enkelt kan utökas genom att nya moduler sätts in. Modulära systemet erbjuder en hög grad av flexibilitet vilket möjliggör anpassning av systemet för att möta specifika krav för olika processer inom större och mera komplexa tillämpningar. [1]

3.2 HMI

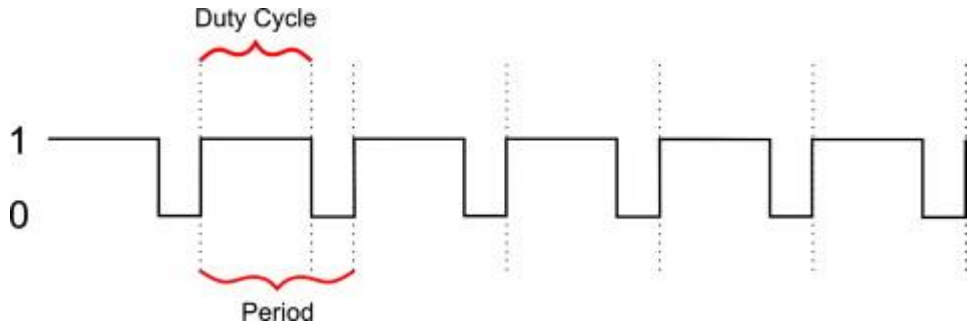
HMI, står för Human Machine Interface, eller människa-maskin-gränssnitt. Begreppet används oftast för en pekskärm, som används för att kommunicera med och övervaka en maskin, dator eller process. För att göra detta möjligt kommunicerar en HMI med en PLC med hjälp av ett kommunikationsprotokoll, exempelvis PROFINET eller Modbus.

HMI:n har många fördelar, bland annat kan ett användargränssnitt designas för pekskärmen, vilket ökar användbarheten av systemet, som man kan styra en process och läsa av sensorer på ett och samma ställe. I och med att en stor frihet ges för designprocessen av visualiseringen, är det möjligt att få fram all väsentlig information och samtliga funktioner på en och samma skärm. Detta istället för att använda fysiska knappar och manuellt läsa av givare. Det designade användargränssnittet överförs till HMI-skärmen, var gränssnittet lagras i HMI:ns interna lagringsminne. Skärmen kan nu anslutas till en PLC och fungera autonomt, utan externa programmeringsenheter.

En pekskärm är ett bra alternativ för funktioner där användaren skall navigera i en meny eller välja ett objekt ur en lista. HMI-skärmar kan variera i storlek och ha olika funktioner, bland annat knappar, pekskärm och trådlös uppkoppling. De har även fördelen att inga rörliga delar behövs, de kan även skapas för användning i hårdare miljöer. [2] [3]

3.3 PWM

Pulse-Width Modulation, eller pulsbreddsmodulering, är en teknik med vilken man kontrollerar en digital signal genom att konstant växla snabbt mellan ett högt och lågt läge, för att utan en analog signal reglera olika komponenter. Genom att variera pulsens bredd för en signal med fast frekvens kan man justera mängden spänning som levereras. Med hjälp av PWM-styrning kan man drastiskt minimera kostnader och energiförbrukning för olika komponenter. Signalen har två viktiga parametrar, frekvens och Duty Cycle.



Figur 1 PWM-signalens Period och Duty Cycle. [6].

Perioden för PWM-signalen anger hur lång tid (i sekunder) det tar för signalen att göra en komplett cykel, som i bilden ovan, från högt till lågt återigen till högt. Frekvensen, som mäts i Hertz visar hur många perioder som sker under en sekund, om PWM-signalen har en frekvens på 500 Hz, körs 500 perioder på en sekund. Perioden och frekvensen är då nära kopplade till varann, har man snabbare frekvens blir perioden mindre, om frekvensen är långsammare blir perioden längre. Man kan se relationen mellan dessa med följande formel.

$$Period(s) = 1 / Frekvensen\ (Hz) \quad (1)$$

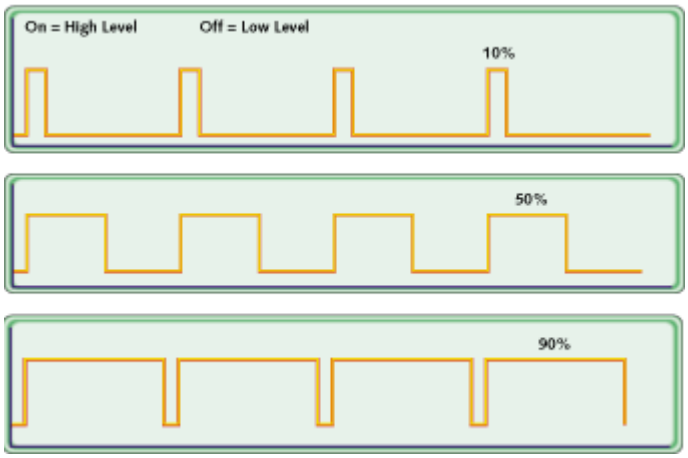
Om vi då hade en 1 kHz signal, skulle perioden av signalen vara 1/ 1 kHz =0.001, eller 1ms.

Duty Cycle, eller pulsbredd, är procenten av tiden som PWM-signalen är hög, eller med andra ord tiden som signalen skickar ut spänning. Formeln för att beräkna Duty Cycle är följande:

$$Duty\ Cycle(\%) = Pulsbredd / Period \cdot 100 \quad (2)$$

Som exempel, för att skapa en 25 % Duty Cycle för den tidigare nämnda 1 kHz signalen med en period på 1 ms, skulle en pulsbredd på 0,25 ms användas i formeln (2), såvis skulle signalen vara hög under 25 % av tiden och låg under de resterande 75 %.

Genom att variera Duty Cycle:n hos en PWM-signal kan man effektivt reglera utgången av signalen, om man hade en ingående spänning på 10 V med 10 % Duty Cycle skulle resultatet vara en utgångssignal med ett effektivvärde på 1,0 V. Observera att det är effektivvärdet som blir 1,0 V, utgångssignalen är i själva verket antingen 0 eller 10 V, se figur 2 för visualisering av signaler med varierande pulsbredd.



Figur 2 Exempel på 10 %, 50 % och 90 % Duty Cycle. [4].

PWM-styrning har flera användningsområden, det används för att justera hastigheten på DC-motorer, inom kraftelektronik för spänningsreglering, effektomvandling och styrning av växelriktare, justering av LED-belysning samt i detta projekt för justering av flödet genom pumpen och ventilerna. [4] [5] [6]

3.4 PID

PID-regulatorn är en av de vanligaste metoderna för återkopplad styrning inom industriella system. Regulatorn skapar en styrsignal genom att använda uppmätta felet mellan ett ärvärde och börvärde, och justerar processen för att minimera felet. Regulatorn består av tre delar: **Proportionell**, **Integrerande** och **Deriverande**, där den proportionella delen ger direkt respons på ett fel, integrerande är proportionell med felet över tid och derivativa reagerar på hur snabbt felet förändras. Genom justering av dessa parametrar kan man justera regulatorns beteende.

Proportionella delen, P, justerar styrsignalen proportionellt mot storleken av felet med följande formel (3).

$$K \cdot e(t) \quad (3)$$

$$e(t) = r(t) - y(t) \quad (4)$$

E(t) står för reglerfelet, var r(t) står för börvärde, y(t) ärvärdet. K står för den proportionella förstärkningen vilken påverkar hur kraftigt regulatorn reagerar.

Integrerande delen, I, hjälper att eliminera kvarstående avvikelser genom att multiplicera summan av tidigare reglerfel med inversen av integrationstiden *Ti*. (5)

$$\frac{1}{Ti}\int_0^t e(t)d\tau \quad (5)$$

Deriverande delen, D, kan uppskatta förändringar genom att beräkna förändringshastigheten på felet. Td står för derivat tiden, den bestämmer hur starkt förändringstakten påverkar signalen. (6)

$$Td \cdot (\frac{d}{dt} \cdot e(t)) \quad (6)$$

Den mest grundläggande strukturen för en PID-regulator är kontinuerlig, den fortsätter kontinuerlig mätning och beräkning av analoga signaler i nutid. Följande formel definierar en kontinuerlig PID-regulator. (7)

$$u(t) = K(e(t) + \frac{1}{Ti}\int_0^t e(t)dt + Td\frac{de(t)}{dt}) \quad (7)$$

Eftersom en PID-regulator konstant försöker nå börvärdet, eller med andra ord den önskade nivån, kan man med justeringen av regulatorns parametrar styra processens beteende för hur regulatorn uppnår samt uppehåller det önskade värdet.

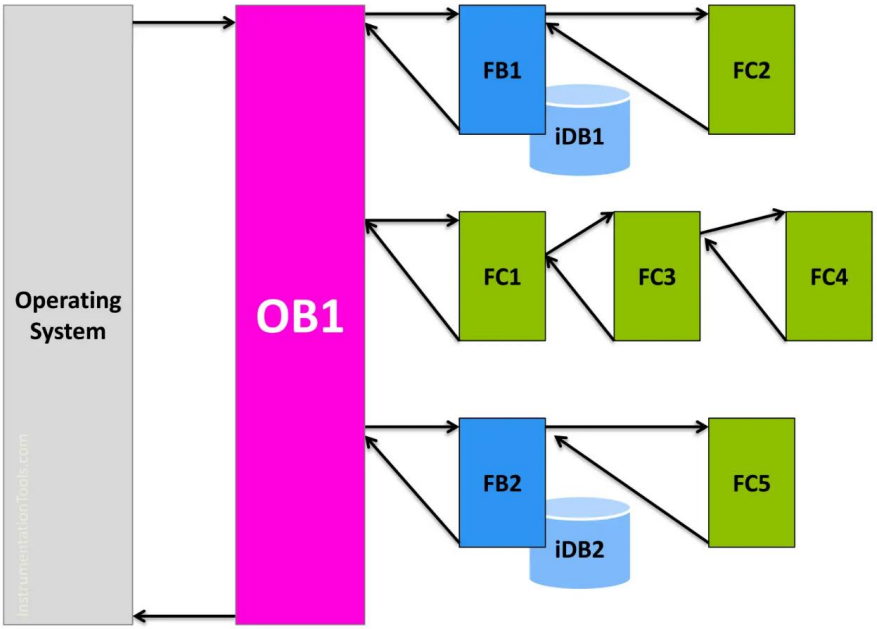
Om P-värdet ställs till ett högre värde kan systemet ge en snabbare respons, men vara ostabilare med risk för mera oscillerande svängningar. Ställningen av I till ett lägre värde kan även orsaka ett mera oscillerande program, men även en snabbare respons för kompensering av störningar i ett system. Högre D ger mera stabilitet åt programmet genom ökad dämpning av en oscillerande signal, men en för hög deriveringstid kan dock orsaka minskad dämpning och göra systemet mindre stabilt. [7]

4 Organisationsblock i Siemens TIA

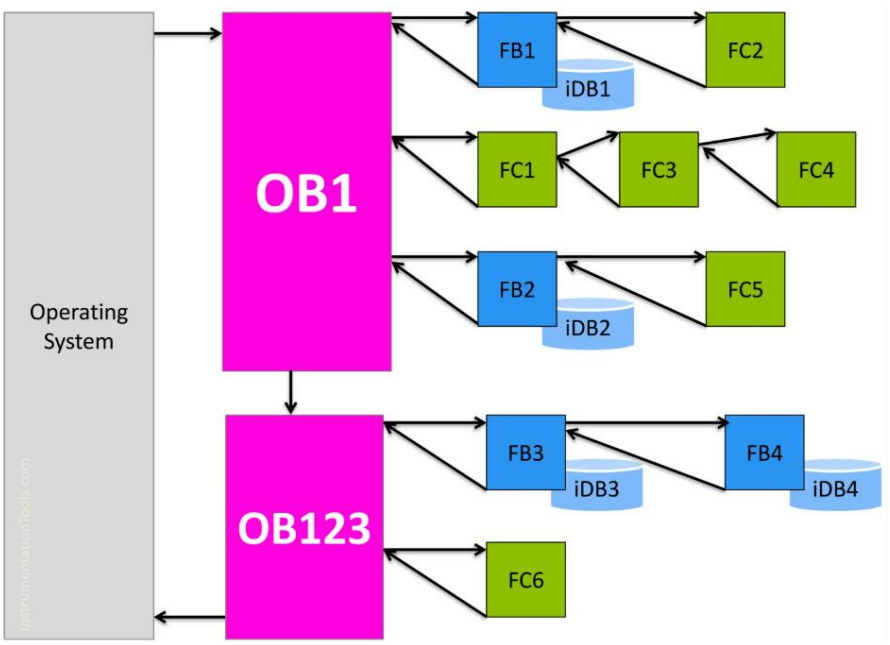
Organisationsblock (OB) i Siemens TIA används för att strukturera och bestämma hur olika delar av programmet i en PLC körs. Dessa block är väsentliga i hanteringen av olika automatiserade system för att säkerställa en effektiv och stabil drift. Genom att korrekt hantera cycliska processer, händelsestyrda exekveringar samt uppstartsrutiner kan man optimera systemets prestenda och även minimera exekveringstider och störningar i den övergripande processen. Tre viktiga OB i Siemens TIA är OB main, OB cyclic interrupt och OB startup, som följande tas deras egenskaper och användningsområden upp.

4.1 OB Main (OB1)

OB Main(OB1) är huvudblocket i ett Siemens PLC-program, blocket är ett programcykel-OB som körs cycliskt, vilket innebär att när sista funktionen i blocket körts börjar programmet om från början. I och med att OB1 är ett huvudblock fungerar det som kärnan i programmets exekveringscykel, vilket kontinuerligt kör all logik och kod i blocket så länge PLC:n är i RUN –läge. Det är här man lägger alla instruktioner för att styra programmet samt anropa på olika definierade block. Det är möjligt att ha flera programcykel-OB, dessa exekveras i stigande ordning var OB1 alltid körs först.



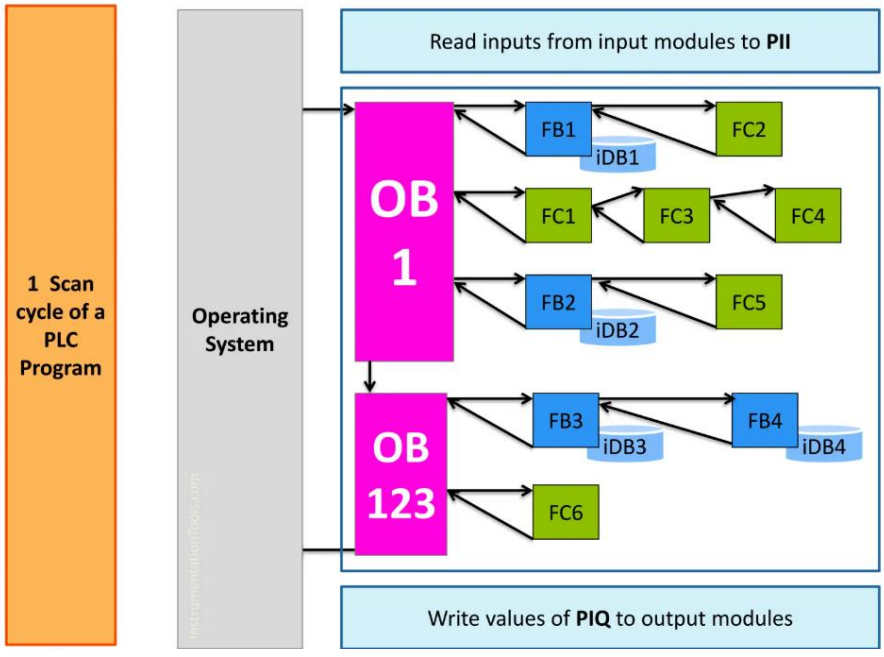
Figur 3 Exekvering av OB1. [9].



Figur 4 Exekvering av programcykel med mer än ett Cyklisk OB. [9].

Under en exeveringscykel även benämnd ”scan cycle”, inträffar en programcykelhändelse varje cykel. CPU:n genomför under denna process ett antal centrala operationer enligt följande:

- Inläsning av ingångsvärden.
- Exekvering av programcykel OB.
- Skriver utgångsvärden.



Figur 5 Scan Cycle av en Siemens PLC. [9].

CPU:ns centrala operationer under exekveringscykeln är alltid aktiverade och körs oavsett om man inte har ett programcykel OB eller om man har flera. Eftersom exekveringen är cyklisk, körs händelsen om igen efter följande händelser:

- Sista OB-startup har exekverats.
- Den sista programcykel - OB:n har exekverats.

För att säkertsälla en snabb respons i styrsystemet är det viktigt att se till att OB1:s scan cycle är så kort som möjligt. OB1:s scan cycle påverkas av mängden kod som blocket måste läsa in under en cykel, där räknas även med alla FC:n, FB:n och högt prioriterade OB, men även ingångs –samt utgångsvärden. [2] [8] [9]

4.2 OB Cyclic Interrupt

OB Cyclic Interrupt är en cyklisk avbrottrutin som används för att exekvera kod inom en bestämd tidsintervall. OB Cyclic Interrupt(OB30-38) har en högre prioriterings klass än OB1, vilket betyder att OB Cyclic Interrupt:s kod kan köras oberoende av OB1:s cykel. Detta är användbart när en process behöver en exakt exekveringstid, vilket är viktigt i realtidsapplikationer.

Cyclic Interrupt körs vid tidsintervallet som man ställt in i PLC-konfigurationen, detta exempelvis 1 ms, 10 ms eller 1 sekund. I och med att Cyclic Interrupt har en högre prioritet än OB1, avbryts huvudprogrammet vid OB Cyclic Interrupts programmerade tidsintervall. Detta är idealiskt för en PID kontroller, eftersom OB main:s exekveringstid kan variera och påverka noggrannheten av en PID:s funktion. För att undvika detta placeras PID instruktionerna i OB Cyclic Interrupt för att exekveras inom en definierad och konstant cykel.

Om flera OB Cyclic Interrupt används samtidigt (t.ex. OB30 och OB31) körs de i ordning baserat på deras prioritet, var OB30 körs först, sedan OB31, varav sedan huvudprogrammet återupptas. På samma sätt som huvudprogrammet kan även Cyclic Interrupt avbrytas av högre prioriterade OB:er, exempelvis diagnostik- och felsöknings OB:er som OB 40-47(Process Alarm) och OB 80-87(Error Handling). [2] [8] [10]

OB No.	OB Type	Priority
OB 1	Cyclic program	1
OB 10	Time-of-day interrupt	2
OB 20	Time-delay interrupt	3
OB 35	Cyclic interrupt	12
OB 40	Hardware interrupt	16
OB 82	Error handling	26 / 28

Figur 6 Prioriteringslista för olika OB, högre siffra = högre prioritet. [10].

4.3 OB Startup

OB Startup(OB100) är ett start-upp-block som körs en gång av PLC:ns CPU när PLC:n startas och programmet övergår från STOP – till RUN-läge. OB100 exekveras en gång vid varje omstart av PLC:n, om det finns flera start-upp-block, körs de i storleksordning från lägst till högst. Om en PLC t.ex. skulle ha blocken OB100 och OB123, skulle OB100 exekveras först.

OB Startup kan användas för att initialisera startvärden för variabler, kontrollera larm, rekalibrera sensorer samt återställa systemmoduler innan huvudprogrammet (OB1) körs. OB1 exekveras inte förrän alla funktioner inom OB100 har gjorts. I och med detta säkerställer man att systemet befinner säg i ett säkert tillstånd vid starten, eller har rätta globala variabler innan de används i projektet.

Förrän operativsystemet startar logiken för huvudprogrammet, har operativsystemet några uppgifter som måste exekveras, oavsett om du skapat ett startup OB eller inte. De är följande:

- Rensa icke retentiva minnen, t.ex. temporära variabler och tillstånd för utgångar.
- Rensa PIQ, Process Image Output, del av PLC-minnet för utgångsinformation.
- Exekvera Startup OB:er.
- Updatera PII, Process Image Input.
- Aktivera utgångar efter övergång till RUN-läge.

[2] [8] [11]

5 Retain-funktioner för block i Siemens TIA

Siemens TIA portal erbjuder flera alternativ för hur variabler i datablock hanteras och bevaras, speciellt vid händelser som systemomstart eller strömavbrott. Dessa är viktiga att förstå eftersom de påverkar hur informationen sparas, återställs och hanteras i PLC:n. Detta avsnitt redogör hur **Retain**, **Non-Retain** och **Set in IDB** fungerar, skillnaderna mellan dessa samt deras tekniska och praktiska fördelar.

5.1 Non-Retain

Variabler i TIA portal är till standard inställda till Non-Retain. Vid varje omstart av PLC:n återställs dessa variabler till sina initialvärden, vilket är lämplig för data som inte behöver bevaras över en längre tid såsom temporär logik eller interna flaggor.

5.2 Retain

Retain-variabler kan ställas direkt i ett globalt datablock eller funktionsblock, dessa behåller sitt värde även efter att PLC:n omstartats eller genomgått ett strömavbrott. Detta är väsentligt för system som behöver bevara viss tillståndsinformation eller viktig data som inte kan gå förlorad.

När PLC:n omstartas eller vid strömavbrott, använder CPU:n intern energi från en superkondensator och skickar variabelvärdet från RAM-minnet till ett internt Non-volatile minne, ett minne som inte raderas vid strömavbrott. Vid nästa uppstart kopierar CPU:n värdet från Non-volatile minnet till RAM-minnet, vilket ger variablerna deras tidigare värde innan omstarten/strömavbrottet. [12]

5.3 Set in IDB

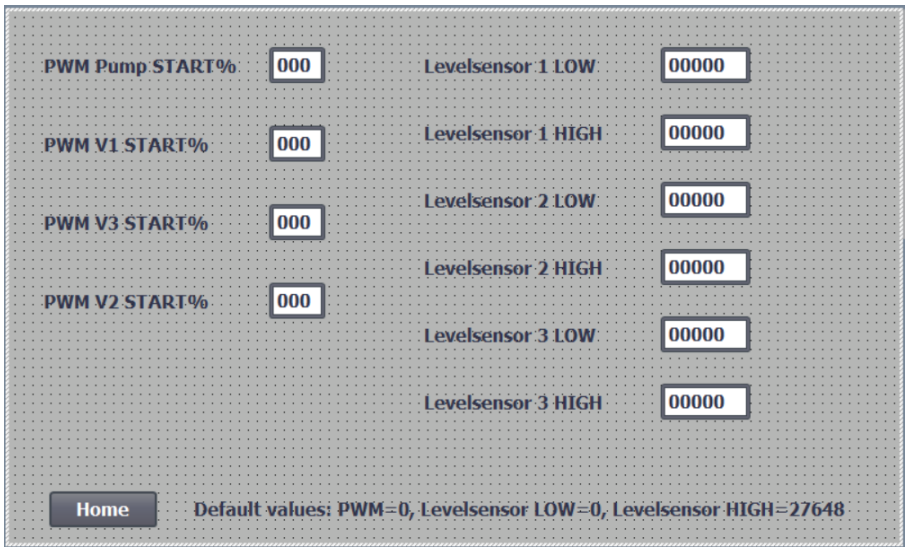
Set in IDB är en flexibel funktion som möjliggör att definiera variabler som Retain/Non-Retain vid varje instans av ett funktionsblock i ett program. Varje gång ett funktionsblock med Set in IDB används i ett program skapas ett nytt Instans Data Block, var inställningen av Retain/Non-Retain görs, istället för i funktionsblocket. På så vis kan man bestämma vilka instanser av blocket som sparar sin data under ett strömabrott/vid omstart, även om det är exakt samma block som används. Med detta kan ett och samma FB användas för flera olika funktioner, även om det har olika minnesbeteenden eller funktioner. [13]

6 Metoder

Projektets första fas gick till att undersöka den tidigare PLC-koden för att förstå hur alla dess funktioner fungerade med varann för att kunna skapa de strävade ändringarna till det nya uppdaterade systemet. Det viktigaste som behövdes uppdatera var funktionerna för alla kalibreringsmöjligheter, PWM-styrningen, PID-styrningen och HMI:ns användargränssnitt.

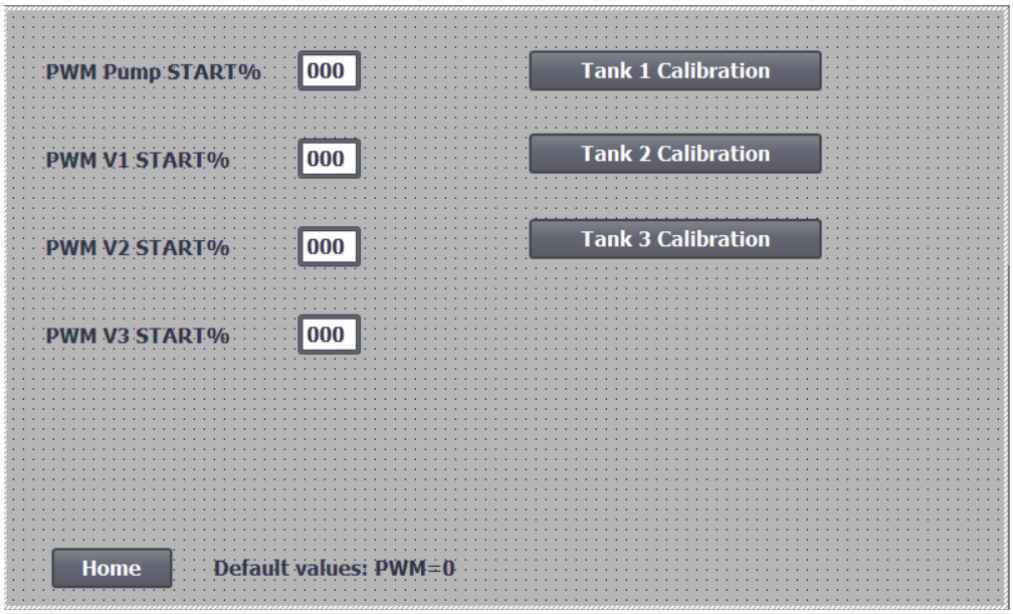
6.1 Uppdatering av kalibreringen för nivågivare

Tidigare fungerade kalibreringen av min- och maxnivån för respektive vattentanks nivågivare genom att manuellt skriva in ett värde mellan 0-27648, som står för 0–100%, i respektiva ruta på HMI:ns kalibreringsskärm (Figur 7). Detta var okonventionellt och krävde att man behövde först räkna ut vilket värde procenten av vattentanken hade. När man ansett att tanken nått sin maxnivå, skulle man kolla på hemskärmens stapeldiagram hur många procent mellan 0–100% som tanken fyllts, sedan ta procentvärdet * 27648. Om tanken nått 80% = 0.8 * 27648 = 22118, detta värde skulle då manuellt skrivas in i levelsensor x high för tanken i fråga.



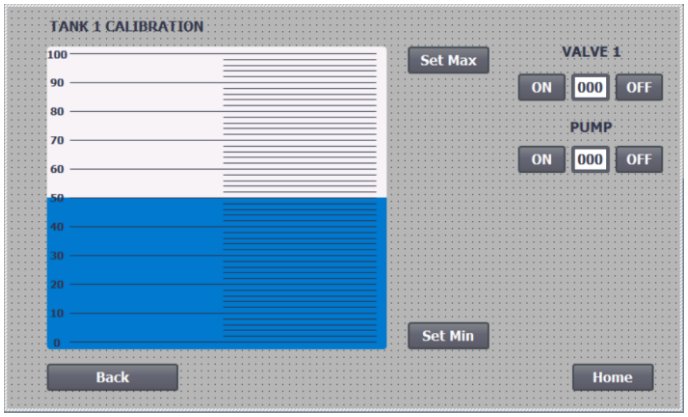
Figur 7 Ursprungliga kalibreringsskärmen.

Till en början omdesignades användargränssnittet för kalibreringen. Istället för rutorna med Levelsensor LOW och HIGH , skapades en tryckknapp för varje vattentank, vilken tar en till en ny skärm för kalibrering av respektive tank.

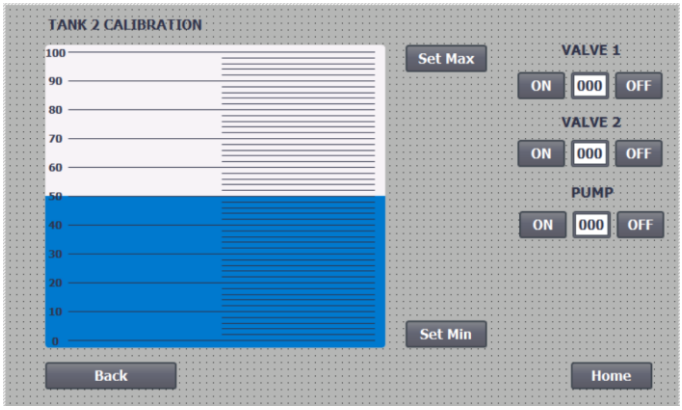


Figur 8 Uppdaterade kalibreringsskärmen.

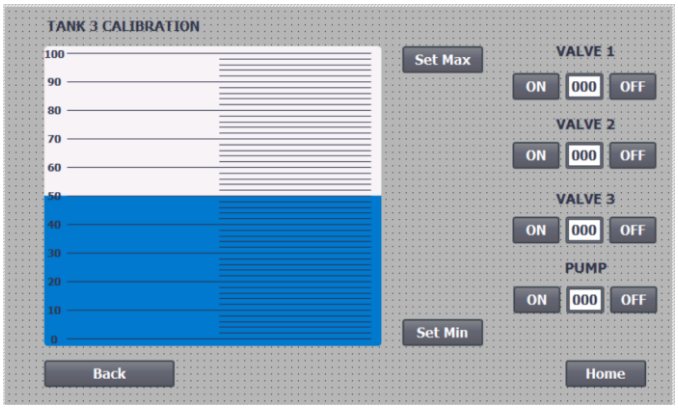
Vattentankarnas nya kalibreringsskärmar är skapade för att göra kalibreringen mera användarvänlig och intuitiv. Skärmen har ett stapeldiagram för att se den nuvarande vattennivån, lämplig styrning av pump/ventiler för respektive tank samt tryckknapparna Set Min och Set Max. Set Min och Set Max fungerar istället för det tidigare Levelsensor x LOW och Levelsensor x HIGH. Snarare än att behöva beräkna vilket värde att manuellt lägga till, tar dessa med ett knapptryck nivågivarens värde och ställer det som tankens kalibrerade min- eller maxvärde.



Figur 9 Kalibrering av tank 1.

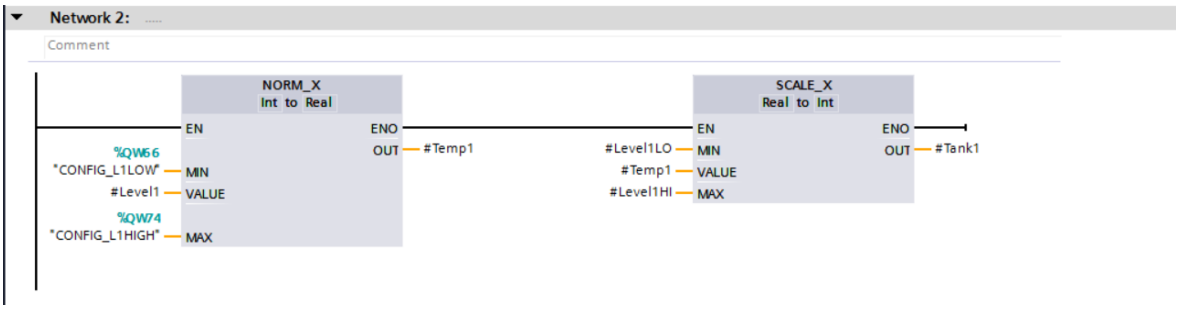


Figur 10 Kalibrering av tank 2.



Figur 11 Kalibrering av tank 3.

I originalversionen av programmet användes tags som Config_L1LOW och Config_L1HIGH för att ställa in min-och maxnivån av i detta fall den första tanken. Dessa tags var av typen Integer(heltal) och ställdes alltid till 0 och 27648 vid varje omstart av programmet m.h.a. ett move block i OB-startup. Ett av målen med projektet var att få programmet att komma ihåg de nya kalibrerade värdena, för att slippa omkalibrera varje värde vid omstart.



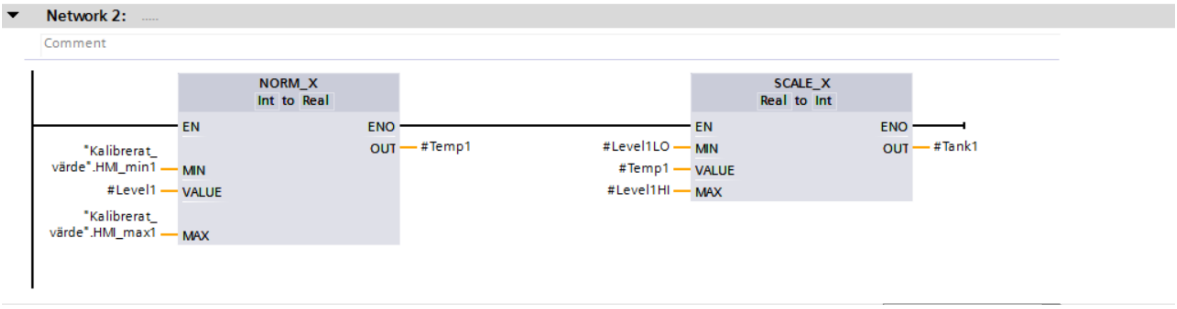
Figur 12 Ursprungliga funktionen för avläsning av nivågivare.

För att göra detta möjligt skapades ett datablock ”Kalibrerat_Värde”, var alla värden för kalibrationen av nivågivare, pumpen och ventilerna lagras. I datablocket kan man ställa startvärdet för varje enskild tag, vilket betyder att funktionerna i OB Startup inte längre var nödvändiga och slopades. Genom att dessutom klicka in retain för varje tag, kommer PLC:n ihåg kalibrerade värdet vid varje omstart.

Kalibrerat_värde

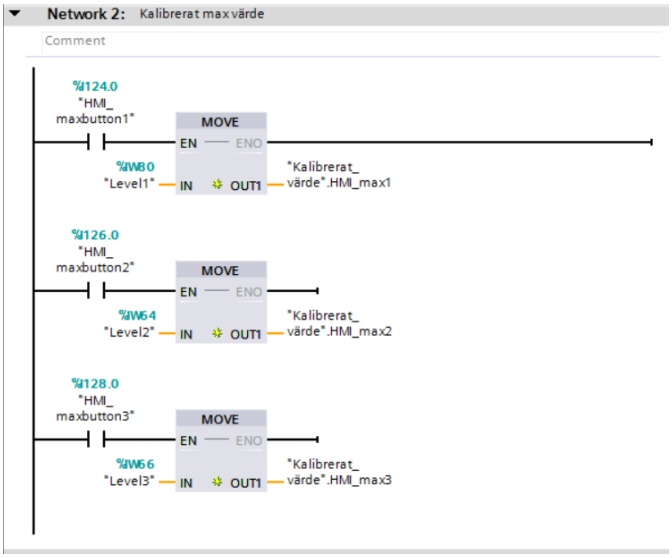
	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint
1	Static							
2	HMI_max1	Int	27648	☑	☑	☑	☑	☐
3	HMI_max2	Int	27648	☑	☑	☑	☑	☐
4	HMI_max3	Int	27648	☑	☑	☑	☑	☐
5	HMI_min1	Int	0	☑	☑	☑	☑	☐
6	HMI_min2	Int	0	☑	☑	☑	☑	☐
7	HMI_min3	Int	0	☑	☑	☑	☑	☐
8	Pump_startvärde	Real	0.0	☑	☑	☑	☑	☐
9	V1_startvärde	Real	0.0	☑	☑	☑	☑	☐
10	V2_startvärde	Real	0.0	☑	☑	☑	☑	☐
11	V3_startvärde	Real	0.0	☑	☑	☑	☑	☐

Figur 13 Datablocket Kalibrerat_Värde.

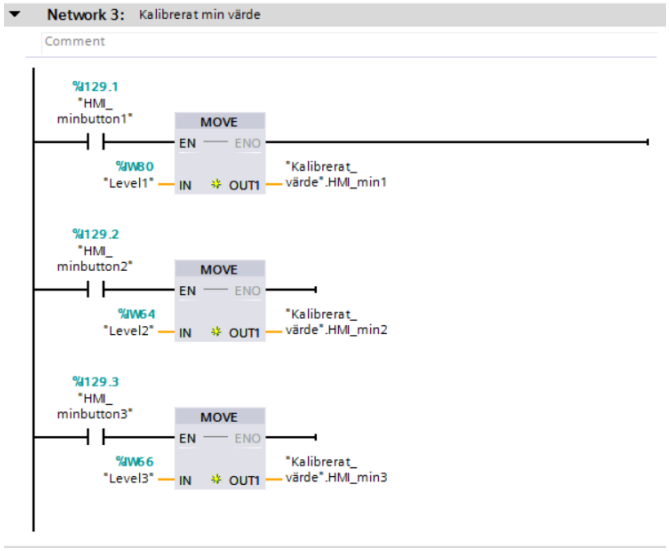


Figur 14 Uppdaterad funktion för avläsning av nivågivare.

I programmets Cyclic Interrupt OB skapades två nätverk för ställningen av de nya kalibrerade mix- och maxvärdena för nivågivarna, kalibrerat minvärde och kalibrerat maxvärde. Som alternativ till att lägga dessa i OB main, ska dessa inte köras kontinuerligt, utan endast vid påkallelse av tryckknapparna, därför placerades nätverken i Cyclic Interrupt för att kunna köras kontrollerat och endast vid behov. De fungerar med en normally open contact, som styrs av HMI-knapptrycket av respektive tanks kalibreringsskärms set min/max, ett move block tar värdet från nivågivaren och skickar det till datablockets tag vid aktivation från HMI-knappen, och slutar skicka när knappen släpps.



Figur 15 Nätverket för Set Max.



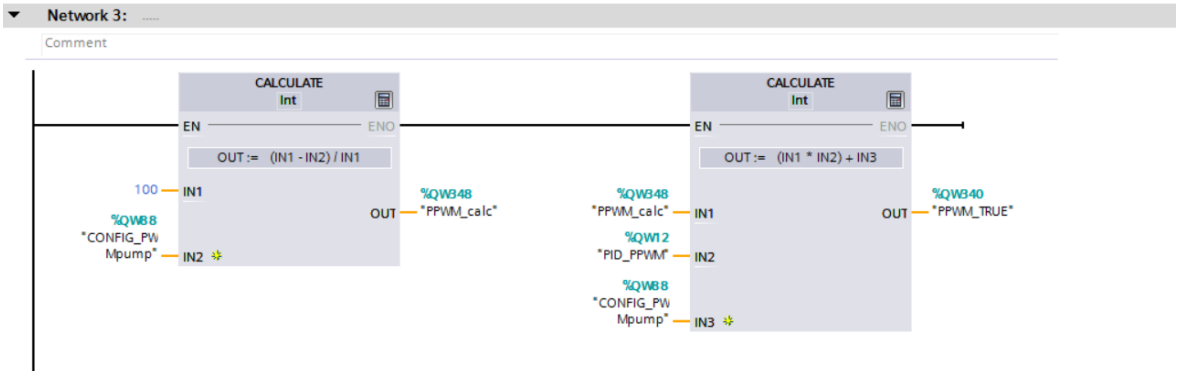
Figur 16 Nätverket för Set Min.

För varje Set Min-och Set Max-knapp i HMI, har i dess properties, Press ställts till setbit och Release till resetbit med HMI-knapptrycket som input/output. På detta sätt får man knappen att endast aktiveras när knappen är nedtryckt, och avaktiveras när knappen släpps. Detta för att inte konstant uppdatera datablockets tag, utan endast när knappen är intryckt.

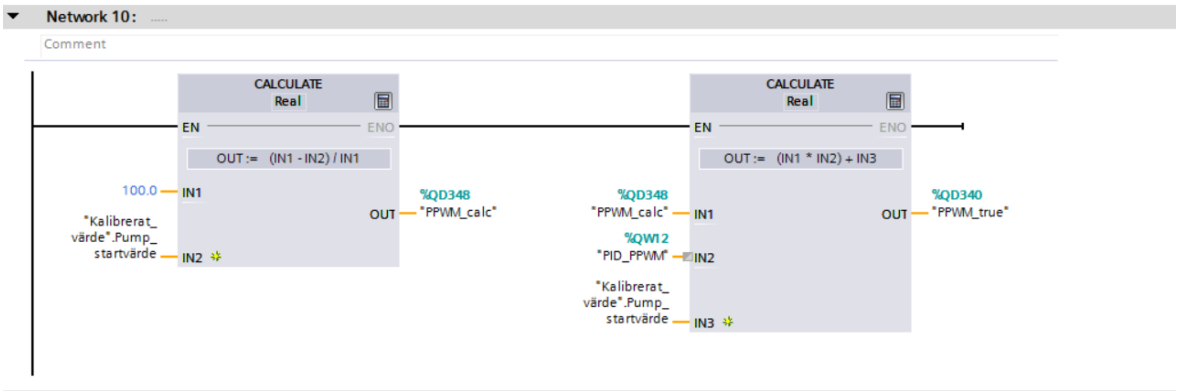
6.2 Uppdatering av kalibrering för PWM-styrning

Även som i uppdateringen av nivågivarna, användes Integer(heltal) tags som ställdes till 0 med ett move block i OB startup för kalibreringen av PWM-styrningen. Dessa togs bort och nya tags till det tidigare nämnda Kalibrerat_Värde Datablocket lades till (figur 13). De nya taggarna för kalibreringen av pumpen samt de tre ventilerna, skapades av typen Real(flyttal) för att få noggrannare värden samt för att få det ursprungliga programmets PWM kalibrering samt PID- styrning att fungera.

I det ursprungliga programmet användes heltals Calculate-block för beräkningen av den verkliga PWM:n för ventilerna och pumpen, med funktionen: 100 – kalibrerat starvärde / 100. Om man inte hade kalibrerat dessa, fungerade funktionen som den ska eftersom då stämmer funktionen och ger ut 100 – 0 /100 = 1. Ifall man kalibrerat PWM-styrningen av pumpen/ventilerna till ett startvärde på till exempel 50, borde resultatet för PWM_calc taggarna bli 0.5, men i.o.m att taggarna och funktionen var i Integer blir dessa tal 0, eftersom Integer inte har desimaler utan är heltal. Kalibreringen påverkades naturligtvis av detta eftersom värdet kunde endast vara 1 utan kalibrering, och 0 om någon kalibrering gjorts, istället för ett tal mellan 0.0 och 1.0.

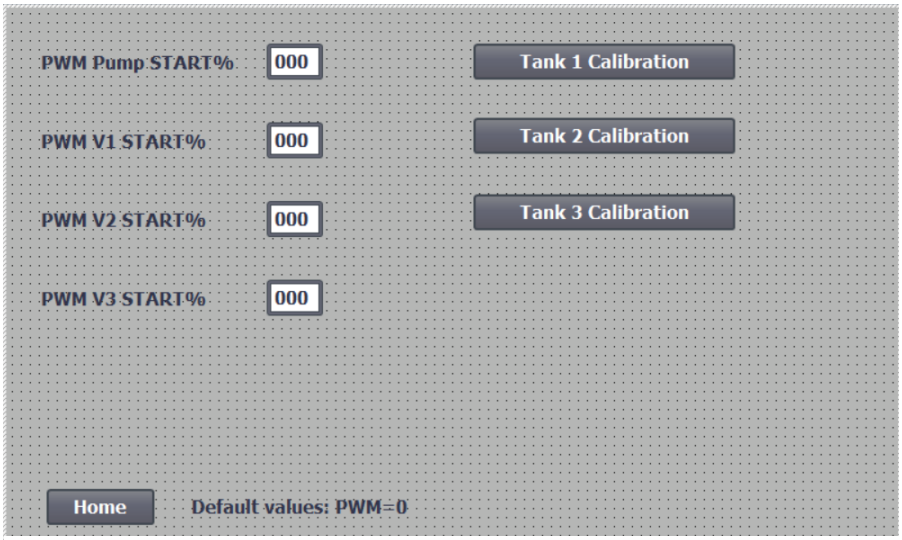


Figur 17 Funktionen för PWM-styrning av pumpen i det ursprungliga programmet.



Figur 18 Uppdaterade funktionen för PWM-styrningen av pumpen med Real.

Till det uppdaterade programmet ändrades alla tags och funktionsblock som användes för PWM-styrningen från heltal till flyttal (Figur 17 och 18). HMI-funktionerna förblev de ursprungliga, eftersom de fungerade som strävat efter justeringen till flyttal. Efter att man öppnat pumpen/ventilerna till sitt eget ansedda startvärde, skriver man in detta värde i textrutan för pumpen/ventilens START% på kalibreringsskärmen(figur 19), detta värde blir det nya startvärdet.

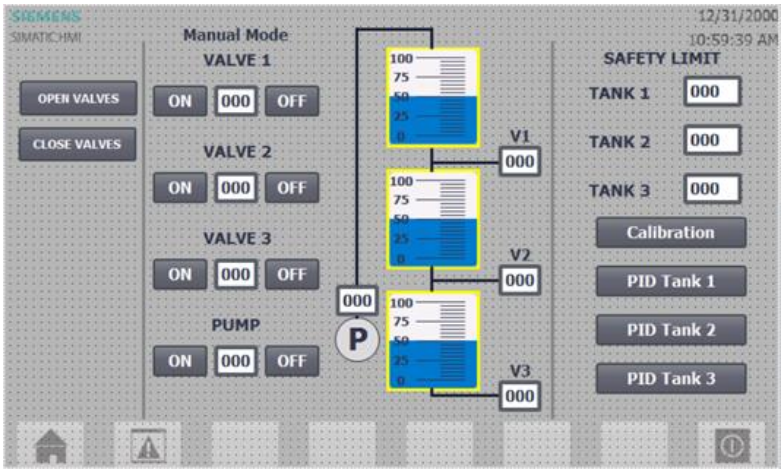


Figur 19 Kalibreringsskärmen.

Om en ventils eller pumpens nya startvärde läggs till 50 %, kommer 0–100 % av PWM styrningen på HMI:ns hemskärm, nu stå för 50 %–100 %. Om man sedan lägger 10 % på hemskärmen och startar ventilen/pumpen, kommer det kalibrerade startvärdet * procenttalet köras, i detta exempel 55 %. Nedan en tabell över PWM-signalen med olika kalibreringsvärden.

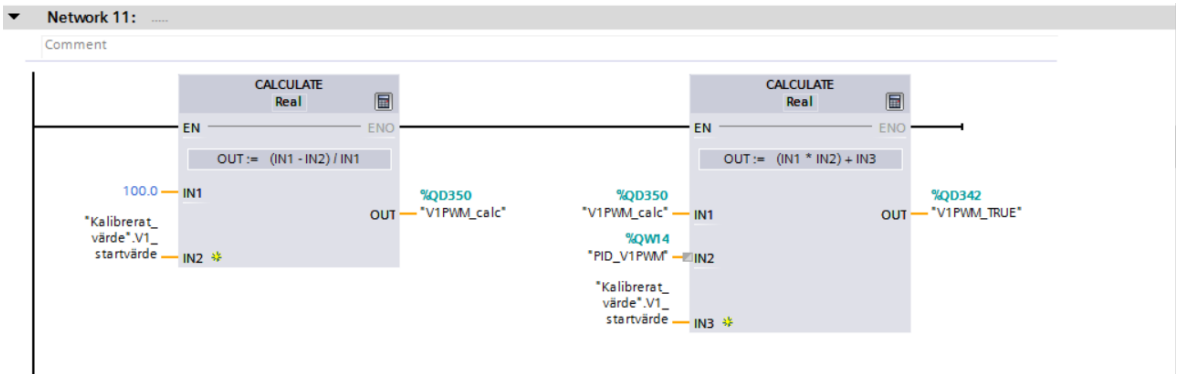
PWM	Okalibrerat	Kalibrerat 10%	Kalibrerat 25%
0	0	10	25
10	10	19	32,5
20	20	28	40
30	30	37	47,5
40	40	46	55
50	50	55	62,5
60	60	64	70
70	70	73	77,5
80	80	82	85
90	90	91	92,5
100	100	100	100

Tabell 1 Kalibrerade PWM-signaler.



Figur 20 Uppdaterade hemskärmen.

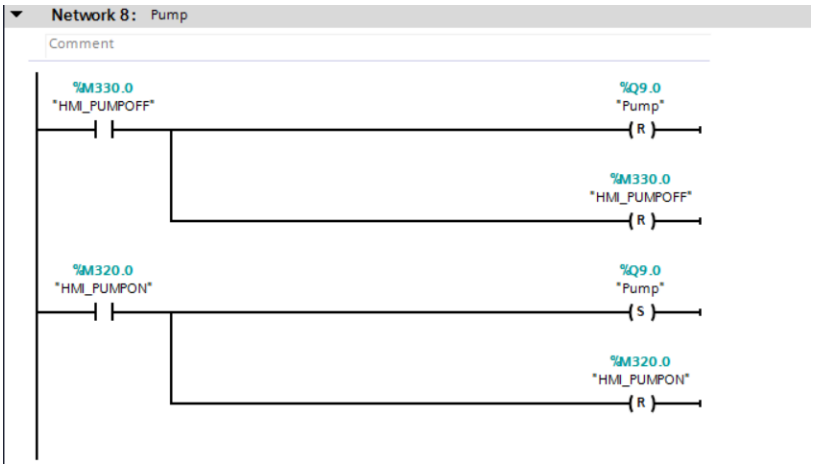
Denna ändring fick inte endast PWM-kalibreringen att fungera, utan även PID-styrningen. Eftersom utgången av PID-regulatorerna användes i samma funktion för att styra och kalkylera PWM:n som PWM_calc. Detta med formeln (PWM_calc * PID_PWM) + kalibrerat _startvärde (se figur 21).



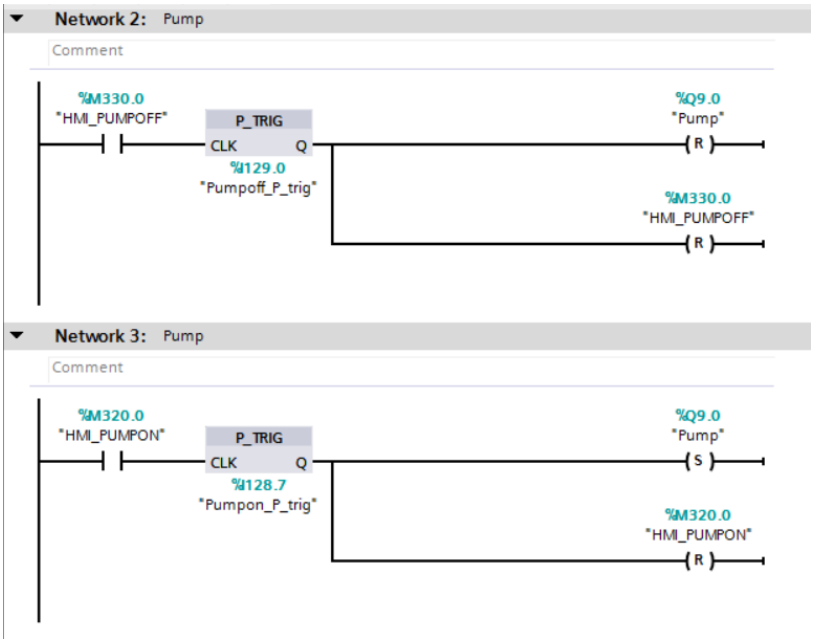
Figur 21 Uppdaterade nätverket för ventil 1.

Om pumpen eller ventilerna hade kalibrerats i det ursprungliga programmet ledde detta till att funktionen blev 0 * PID:ens PWM utgång + kalibrerade startvärdet, vilket renderade PID:ens styrning av PWM:n obsolet, eftersom 0 * PID_PWM = 0. I.o.m ändringen till Real-tal, justerades styrningen korrekt enligt kalibreringen, var PWM_calc var ett tal mellan 0.0 och 1.0, baserat på vad som lagts som startvärde.

Slutligen för att göra on/off-knapparna för styrningen av pumpen/ventilerna mera responsiva lades ett P_trig block till funktionen av knapptrycket. Detta block känner av en positiv flank när en signal ändras från 0 till 1. För varje knapptryck skapades enskilda tags vilka används som instansminne.



Figur 22 Pumpens ursprungliga On/Off-knappar.



Figur 23 Pumpens On/Off-knappar med P_TRIG.

7 Resultat

PLC-programmet har färdigställts med de önskade funktionerna implementerade. HMI-gränssnittet för kalibreringen av nivågivarna har gjorts mera användarvänligt och intuitivt, försatt med uppdaterade funktioner och en mera visuell överblick av processen. Pumpens och ventilernas kalibrering fungerar stabilt, man kan definiera ett startvärde för pumpen eller ventilernas PWM-signal, och programmet använder detta som startpunkten vid manuell styrning samt PID-reglering av dessa.

PID-regleringen av vattentankarna fungerar fullständigt. Genom att lägga setpoint för önskad vattennivå i tanken, kan man genom finjustering av PID-parametrarna upprätthålla vattennivån i tankarna, antingen en i taget, eller alla tre på samma gång. Baserat på ställda setpointen vid en tanks PID-styrning, ger pumpen mera eller mindre vatten samt ventilerna öppnas eller stängs, för att hålla önskade vattennivån.

Utöver detta examensarbete uppdaterades den tidigare skapade användarmanualen för systemet, för att underlätta förståelsen och användningen av programmet. Bilagan Multitank-PLC-Manual är bifogad med examensarbetet. Uppdateringarna som gjorts för manualen går genom kalibreringen av nivågivarna, pumpen och ventilerna samt saker att ta i beaktan angående dessa.

8 Diskussion

Till programmet skulle ännu kunna läggas till någon form av funktion för att visuellt se när tankarna är i PID-styrningsläge, eftersom under PID-styrningen kan inte manuella styrningen av ventilerna och pumpen användas, vilket kan lätt skapa förvirring.

Möjligtvis skulle även varningar kunna göras för funktioner som gör att programmet inte körs, exempelvis när ställda safety limit av en tank har nåtts eller nödstoppknappen är intryckt.

9 Sammanfattning

PLC-programmet och HMI-användargränssnittet uppdaterades för ett befintligt system för styrningen av ett Inteco flertankssystem. Programmet är skapat för en Siemens 1215 DC/DC/DC PLC, kopplad till en SIMATIC KTP700 basic HMI-pekskärm. Den uppdaterade mjukvaran erbjuder ett mera intuitivt HMI-användargränssnitt för kalibreringen av nivågivare samt ventilerna och pumpens PWM utgångar. Med pekskärmen kan man även ställa in min- och maxnivå för vattentankarna, genomföra tester av de olika komponenterna samt styra vattennivån i en eller flera av tankarna antingen manuellt eller genom PID-styrning.

9.1 Skribentens egna anmärkningar

Förhoppningsvis kommer mina justeringar till programmet användas för framtida experiment och laborationer med flertankssystemet. Projektet har varit mycket lärorikt och jag känner att jag lärt mig mycket nytt. Jag hoppas att detta projekt möjliggör att systemet i framtiden kan bli en del av utbildningen på el- och automationslinjen vid Yrkeshögskolan Novia. Jag vill rikta ett stort tack till Joachim Böling som varit till stor hjälp med snabba svar på problem och funderingar. Är även tacksam för hans idéer med en klar bild över funktioner att lägga till samt hur programmet ska fungera.

Källor

[1] W.Bolton. (2015). *Programmable Logic Controllers Sixth Edition.* Elsevier.

[2] Berger, H. (2016). *Automating with SIMATIC 6th revised and enlarged edition.* Erlangen: Publicis Publishing.

[3] (What is HMI?, 2025) Retrieved from Copadata: https://www.copadata.com/en/products/zenon-software-platform-for-industrial-automation-energy-automation/what-is-hmi/

[4] Barr, M. (September 2001). Pulse Width Modulation. *Embedded Systems Programming*, pp.103-104.

[5] *What is Pulse Width Modulation: A Concise Overview*. (2024, November 8). Retrieved from Electronic Manufacturing Service: https://electronicmanufacturingservice.org/what-is-pulse-width-modulation-a-concise-overview

[6] Fernandez, A & Dang, D. (2013). *Getting Started with MSP430 Launchpad.* Elsevier Inc.

[7] Åström, K. J. & Hägglund, T. (1995). *PID Controllers, 2nd edition.* Research triangle Park: Instrument Society of America

[8] *SIMATIC S7 S7-1200 Programmable Controller*. (2014, 04 30). Retrieved from Siemens SiePortal: https://support.industry.siemens.com/cs/mdm/91696622?c=50060266123&dl=sk&lc=en-SE

[9] Salama, M. (n.d.). *OB1 - Main Cyclic Organization Block in TIA portal*. Retrieved from Instrumentationtools: https://instrumentationtools.com/main-cyclic-organization-block/

[10] Ghareb, H. (n.d.). *Configuring and Usage of Cyclic Interrupts TIA Portal* . Retrieved from InstrumentationTools: https://instrumentationtools.com/configuring-and-usage-of-cyclic-interrupts-tia-portal/

[11] Salama, M. (n.d.). *Siemens PLC Tia Portal – OB100 Start-up Organization Block*. Retrieved from InstrumentationTools: https://instrumentationtools.com/ob100-start-up-organization-block/

[12] Pegaia, D. (2013, May 2). *Technical Forum - SiePortal Community*. Retrieved from Siemens SiePortal: https://sieportal.siemens.com/en-fi/support/forum/posts/s7-1200-retain-set-in-idb/91211

[13] Pegaia, D. (2018, April 25). *Technical Forum - Siemens Community*. Retrieved from Siemens SiePortal: https://sieportal.siemens.com/en-fi/support/forum/posts/what-does-it-mean-set-in-idb/189717