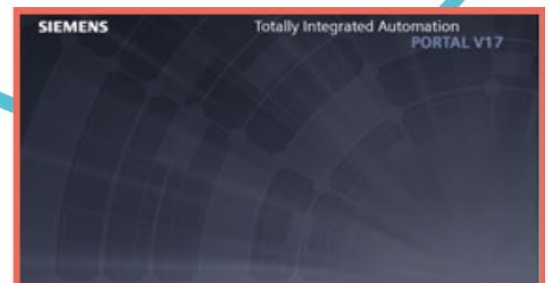


Januar 2023



PLC- Programmierung

Forord

Bogen om PLC-programmering kan anvendes som teoribog på efter- og lærlingeuddannelse inden for PLC-området.

Alle programeksempler er:

- For Siemens lavet i TIA V16 eller V17, og hvis ikke andet er angivet på en S7 1200 CPU
- For Allen-Bradley lavet i Studio 5000 Logix Designer og på en 1769-L30ER CPU

Denne PLC-bog er en revision og samskrivning af to over ti år gamle bøger: PLC 1 og PLC 2. Disse bøger var udviklet for offentlige midler. Men tiden var løbet fra dem, da PLC-området heldigvis er i fortsat udvikling.

Den nye bog er skrevet som et samarbejde mellem Videncentret for Automation og Robotteknolog Nord, Mercantec, TEC, SDE Vejle og EUC Syd. Videncentret for Automation og Robotteknolog Nord har været den drivende og koordinerende kraft på projektet. En stor tak skal gives til forfatterne fra de nævnte skoler for deres uundværlige bidrag.

Bogen udkommer i Word-format. Bogen er ligesom sine forgængere også betalt af offentlige midler, og derfor er der ingen copyright på den. Alle kan derfor frit bruge den i trykt form eller på LMS-platforme i sin helhed eller som enkelte afsnit. Alle er også velkomne til at skrive afsnit om, så de passer bedre til et specifikt undervisningsforløb.

Da videncentrene bliver målt på, i hvor høj grad vi formår at skabe værdi for landets faglærere, EUD-elever og AMU-kursister, vil vi gerne have feedback på vores arbejde. Så skriv endelig til os med ros, fejl der skal rettes eller afsnit, I mener der mangler i bogen. I skal skrive til:

Redaktør og medforfatter januar-2023:
Videncenter for Automation og Robotteknologi Nord
Udviklingskonsulent
Line Goul Stoffer
lgni@mercantec.dk
tlf. 24767456



Indholdsfortegnelse

| | |
|--|-----|
| Forord | 3 |
| Indholdsfortegnelse..... | 5 |
| PLC-styring..... | 7 |
| Programmering..... | 19 |
| Siemens S7-1200 PLC-bestykning..... | 27 |
| Allen-Bradley 1769 PLC-bestykning..... | 35 |
| I/O-kredsløb..... | 43 |
| Timer-kredsløb | 55 |
| Flip-flop..... | 65 |
| Kip-relæ/Toggle flip-flop..... | 71 |
| Tæller..... | 73 |
| Følere..... | 79 |
| PLC-installation..... | 103 |
| EMC (Engelsk)..... | 111 |
| Idriftsætning af PLC-styrede anlæg..... | 125 |
| Reparation og fejlfinding | 129 |
| Sekvens eksempelpå Allen-Bradley..... | 139 |
| Sekvens Eller-delning på Allen-Bradley | 149 |
| Sekvens Og-delning på Allen-Bradley..... | 151 |
| Sekvens Siemens med flip-flop..... | 153 |
| Sekvens med Move og Compare | 163 |
| Sekvens-standarder | 167 |
| Sekvens Siemens Graph 7..... | 181 |
| Talsystemer | 185 |
| Bit, Byte, Word og Dword..... | 189 |
| Datatyper iht. 61131-3 | 193 |
| Siemens adressering, talformat og word-instruktioner | 197 |
| Allen-Bradley adressering, talformat og word-instruktioner | 209 |
| Siemens bloktyper, OB, FC, FB, og DB | 217 |
| Siemens Datablokke | 223 |
| Siemens User Defined Type (UDT) | 241 |
| Siemens funktioner og funktionsblokke..... | 245 |
| Siemens PLC-simulator | 261 |
| Allen-Bradley Add-On-instruktioner..... | 275 |
| Allen-Bradley User Defined Type (UDT) | 281 |
| Analog PLC-signaler generelt..... | 287 |
| Analog signaler Siemens S7-1200..... | 295 |
| Analog universal skaleringsblok | 303 |
| Analog skalering Norm_X og Scale_X Siemens S7-1200..... | 313 |
| Analog signaler Allen-Bradley..... | 317 |
| PID-regulering generelt | 326 |
| PID-regulator S7-1500 CONT_C..... | 332 |
| PID-regulator Compact S7-1200 og 1500 | 336 |
| PID-regulator Allen-Bradley..... | 347 |

PLC-styring

Indledning/historie

På grund af voksende krav til fleksibiliteten i den amerikanske bilindustri opstod der sidst i 60'erne et behov for et nyt styresystem til produktionsmaskinerne, som kunne erstatte de eksisterende, fastfortrådede relæ- og logikstyringer.

Det ønskede styringssystem skulle kunne opfylde følgende hovedkrav:

- Programmerbar styring af hensyn til fleksibiliteten for hurtig omstilling af produktionsudstyret
- Styringen skal fungere uden problemer i et industrimiljø, dvs. at forstyrrelser på grund af temperatur, støv og snavs, forsyningsspændings ændringer, elektrisk støj m.m. skal være minimale
- Ind- og udgangssignalerne skal være digitale, og spændingerne skal ligge inden for det typiske industriområde, dvs. fra 24 VDC til 230 VAC. Endvidere bør styringen kunne behandle analoge ind/udgangssignaler samt være modulopbygget således, at en udvidelse eller udskiftning kan foregå hurtigt og simpelt
- Programmeringen af styringen skal være simpel, hurtig at ændre og generel brugervenlig således, at der kun kræves et kort omskolingsforløb af vedligeholdelsespersonalet

Resultatet blev en PLC, **P**rogrammable **L**ogic **C**ontroller, der var konstrueret til at arbejde i et industrielt miljø, og som var tilsluttet produktionsudstyret via tilsluttede ind- og udgangsmoduler.

PLC'en er designet til at styre/regulere og overvåge en proces. Det kan være en simpel niveaustyring til store komplekse processer, hvor der indgår servodrev til positionering, visionkameraer til kvalitetskontrol eller PLC'en indgår i et samarbejde med eksempelvis robotter.

Den store fordel ved at anvende PLC'en er, at den ikke er statisk, som f.eks. en relæstyring. Når først relæstyringen er lavet, er den svær og tidskrævende at lave om på, hvorimod en PLC, der kan programmeres, er mere fleksibel, både med hensyn til funktion, men også procesoptimering og overvågning.

Det mest opsigtsvækkende var dog programmeringssproget, idet der var udviklet et nyt programmeringssprog, som kunne oversætte simple nøglediagram-lignende koder til mikroprocessorens nødvendige maskinkoder.

PLC

PLC er som tidligere nævnt en forkortelse af de engelske ord **P**rogrammable **L**ogic **C**ontroller, som oversat betyder programmerbart logisk styresystem.

At en PLC er programmerbar betyder, at man ved hjælp af en programmeringsenhed kan programmere styresystemet.

Logisk styresystem betyder, at de logiske grundfunktioner AND, OR og NOT kan programmeres.

Foruden disse findes følgende bit-instruktioner:

- Timer
- Tæller
- Flip-flop
- Skifteregistre

Derudover findes ordinstruktioner som f.eks:

- Compare-funktioner
- Regnefunktioner
- Matematiske funktioner
- Datafunktioner
- Talkonverteringer

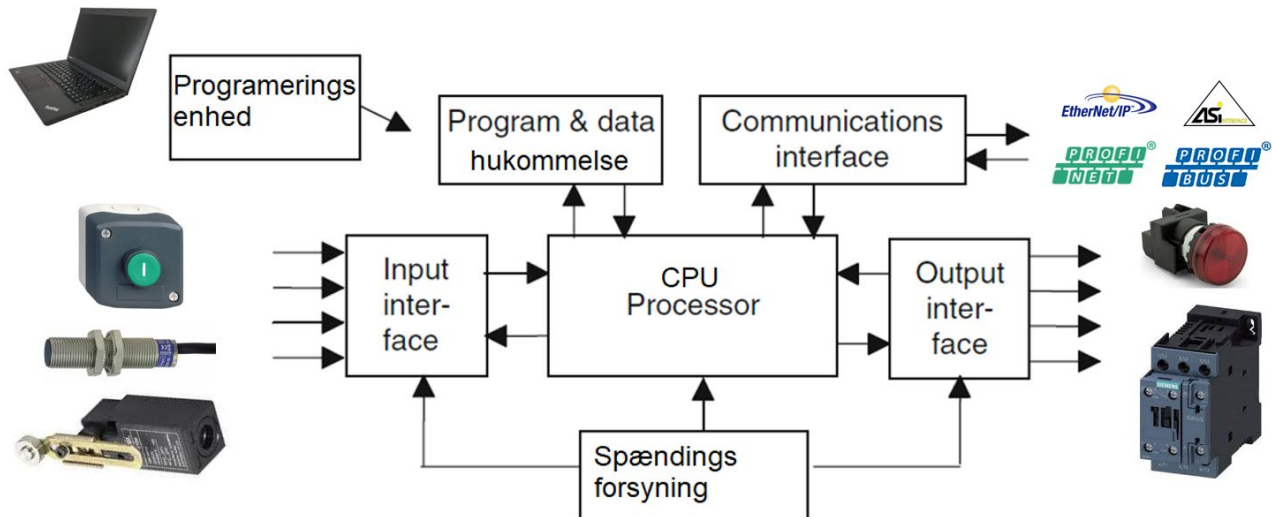
PLC-styresystemer er i dag en fast del af næsten enhver automatisk styring. Siden fremkomsten af de første PLC'er i begyndelsen af halvfjerdserne er der sket en enorm udvikling både teknisk og økonomisk, således at man i dag står med et produkt med en væsentlig større kapacitet end tidligere.

PLC'erne har fået en større ydeevne, hvis der ses på instruktionssæt, lagerkapacitet, hastighed, antal ind/udgange, behandling af analoge signaler, programmeringsenheder og kommunikation mellem forskelligt udstyr.

På samme tid er prisen på det enkelte produkt blevet mindre og derved opnås en økonomisk gevinst selv ved mindre automatiseringsopgaver.

PLC-blokdiagram

PLC'en kan forenklet deles op i følgende blokke.



Spændingsforsyning

Spændingsforsyningen omformer netspændingen på typisk 400 V eller 230 V AC til en 24 V DC, som bruges internt i PLC'en til forsyning af de enkelte blokke, som vist foroven.

CPU – Central Processing Unit

CPU – Central Processing Unit – er den enhed, alle funktioner bliver udført i. CPU'en udfører beregninger, kommunikerer med de andre blokke og styrer programafviklingen.

Program og datahukommelse

Forskellige typer af hukommelsesblokke er nødvendige, for at CPU'en kan afvikle et program. En hukommelse kan betragtes som en kommode med et antal skuffer. I disse skuffer læses/skrives data, som er nødvendige for programafviklingen.

- Systemhukommelse

I denne hukommelse ligger de data, som får processoren i PLC'en til at afvikle/forstå et indtastet program. Denne hukommelse er udviklet af fabrikanten og er en såkaldt statisk hukommelse. Dette medfører, at der kun kan læses i hukommelsen, og at data bibeholdes under spændingssvigt.

- Datahukommelse

Processoren har brug for en hukommelse til at gemme interne data som mellemresultater, timerværdier, hjælperelæer m.m. under programafvikling. Denne hukommelse er en dynamisk hukommelse, som medfører, at der kan både læses og skrives af processoren.

- Programhukommelse

Her er den del af hukommelsen, som brugeren ved hjælp af en programmeringsenhed har adgang til. Her gemmes de instruktioner, som brugeren ønsker PLC'en skal udføre.

Fælles for de tre hukommelsestyper er, at størrelsen af dem afhænger af, hvilken PLC der vælges. Jo dyrere, jo mere hukommelse.

- Memory-kort

På mange PLC'er er der mulighed for at anvende et memory-kort, som programhukommelse. Det kan være et SD-kort. Det kan anvendes til at downloade firmware, PLC-program samt lagre datalogningsfiler.

Input/output

Grænsefladen/snitfladen mellem PLC'en og maskinen/processen benævnes som et interface og har til formål at tilpasse processens spændinger til PLC'en.

Input-interfacet omsætter processens indgangssignaler til signaler, som kan anvendes af PLC'en.

Outputinterfacet omsætter PLC'ens udgangssignaler til signaler, som processen kan anvende.

Typisk anvendes 24 VDC til de digitale input/output og 0-10 V eller 4-20 mA til de analoge input/output.

Interfacet har også til formål at beskytte PLC'en for transienter via en optokobler.

Kommunikations interface

Kommunikations interface opgave er, at kommunikere med forskellige fieldbus og net systemer så som Profibus, Profinet, DeviceNet eller EtherNet IP. På disse bus eller net systemer, bliver der typisk monteret decentrale I/O-moduler, HMI skærme og mange andre enheder. For på den måde at gøre kablingen af styringen enklere.

Programmeringsenhed

De første PLC systemer skulle programmeres ved hjælp af en specialenhed, som PLC fabrikanten havde udviklet. Det kunne være en programloader, som var en lille skærm og et tastatur og en kassettebåndstation, så man kun gemmer en backup af programmet. Sener blev kassettebåndet erstattet af et floppydiskdrev til 5¼" og siden 3½" disketter. Det kunne også være en enhed, der mest af alt mindede om en stor lommeregner med specialfunktioner så som And og Or osv., samt en kabeltilslutning til kommunikation med PLC'en.

I vore dage sker al PLC programmering via en PC, på hvilken der er installeret en programmeringssoftware som fabrikanten har udviklet. Har man en Siemens PLC heder programmet TIA Portal, og til en Allen-Bradley PLC anvendes programmet Studio 5000 Logix Designer.

PLC-styring



Billedet viser en mindre Siemens S7-1200 PLC, der er bestykket med 14 digitale indgange og 10 digitale udgange. Udover de digitale ind/udgange er der 2 analoge indgange (0-10V)

PLC

De fleste PLC'er består af et antal sammenbyggede fysiske blokke eller moduler.

Hvilke slags moduler, der kan tilkobles, afhænger af fabriksmærket. Tager man en Siemens S7-1200, er der her nogle eksempler på kommunikationsmoduler:

I/O-LINK
AS-i
Profibus
RS232/RS485

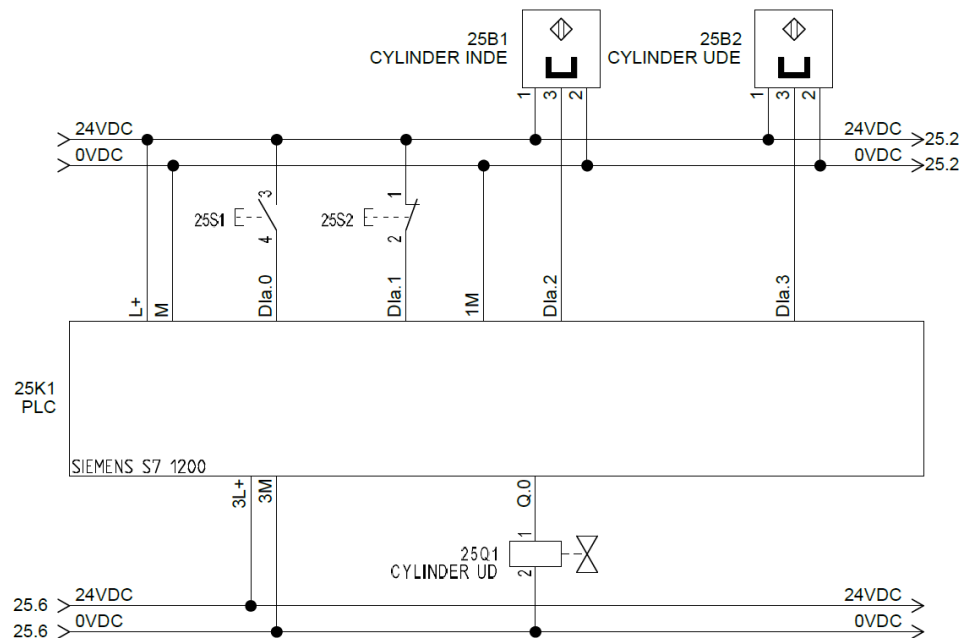
Af input/output kan nævnes:

Digitale
Analoge

Mindre PLC'er består ofte af et basismodul evt. med indbygget strømforsyning til CPU og et antal interfacemoduler.

Indgangssignalerne forsynes evt. fra den indbyggede strømforsyning og føres til indgangsmodulen. Udgangsmoduler er ofte med små relæer, hvor ventiler, kontaktorer m.m. tilsluttes.

Programhukommelsen er indbygget i PLC'en i form af en hukommelseskreds. PLC'en kan skiftes, hvis der er behov for mere hukommelse ved senere udvidelser, f.eks. med et modul for at øge antallet af ind- og udgange.



Forsyningen til den viste PLC er 24 VDC

Ligeledes er PLC'ens ind- og udgange er forsynet med 24 VDC.

CPU-modul

På CPU-modulet indikerer en lysdiode Gul for STOP og for Grøn for RUN når PLC'en afvikler brugerens program.

Input

Inputsignalerne fra processen er tilsluttet input modulet og status indikeres for hver enkelt digitale indgang i form af en lysdiode.

Hver enkelt indgang benævnes med en adresse, i dette eksempel %I0.0 til %I1.7. "I" betyder, at det er et input, mens talværdien angiver byte og bitadresse.

Output

Udgangssignalerne til processen monteres på output-modulet. Også her indikeres hver udgang med en lysdiode.

Udgangen benævnes med en adresse på samme måde som indgange, i dette eksempel %Q0.0 til %Q1.1. "Q" betyder, at det er et output, mens talværdien angiver byte og bitadresse.

Programmering

Ved programmering af en PLC anvendes en PC med et program for udvikling og test af PLC-programmer

Programmeringssprog

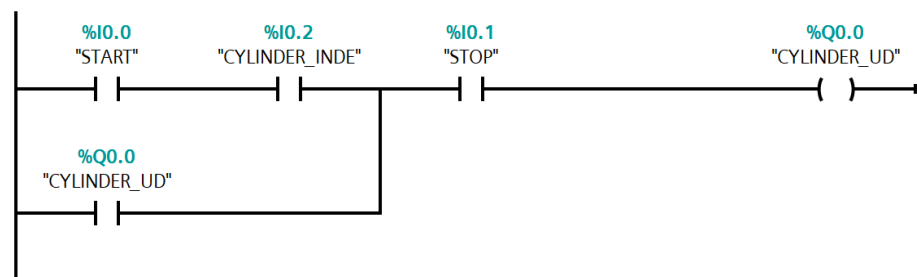
Der findes en række forskellige programmeringssprog til en PLC. I dette kapitel gennemgås de sprog, som er de mest anvendte.

IEC

IEC 61131-3

IEC 61131-3 er en standard som omhandler programmeringssprog. De fleste PLC-mærker overholder denne standard. Standarden beskriver to tekstsprog, Instruction List (IL) og Structured Text (ST) samt to grafiske sprog: Ladder Diagram (LD) og Function Block Diagram (FBD), som vi bedre kender som logikblokke.


Ladder Diagram (LD)




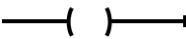
Ladder-diagrammet er det oprindelige programmeringssprog til en PLC. Sproget er inspireret af nøglediagrammer, hvilket gør det lettere, når man ønsker at konvertere et nøgleskema til et PLC-program.

Ladder-diagrammet er det sprog, der anvendes af de fleste, som arbejder med almindelig PLC-programmering.

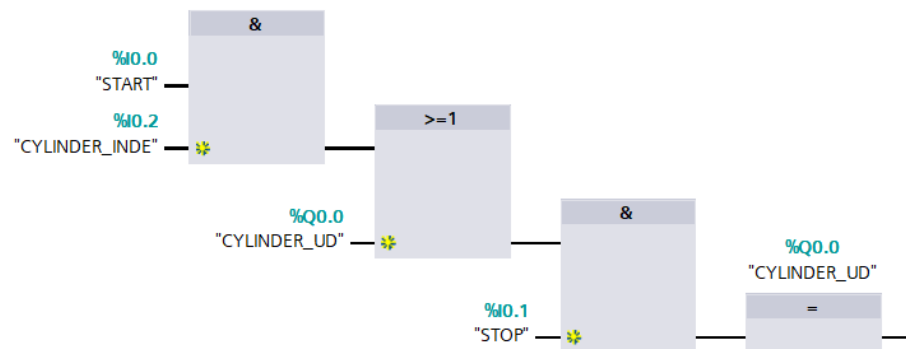
I et Ladder-diagram anvendes følgende symboler for de grundlæggende logiske funktioner:

%I0.0
"START"
 Aktiv på logisk 1

%I0.1
"STOP"
 Aktiv på logisk 0

%Q0.0
"CYLINDER_UD"
 Udgang, aktiv på logisk 1

Function Block Diagram (FBD)



I dette sprog anvendes de logiske grundelementer AND, OR og NOT for at programmere de logiske funktioner. De programmerbare relæer, såsom Siemens LOGO, har FBD som "hovedsprog".

Structured Text (ST)

```
IF ("START" AND "CYLINDER_INDE" OR "CYLINDER_UD")
  AND "STOP" THEN
  "CYLINDER_UD" := 1;
ELSE
  "CYLINDER_UD" := 0;
END_IF;
```

Dette sprog er udviklet, så det ligner de programsprog, man bruger i data-verdenen. Sproget kan med fordel anvendes, når noget bliver for kompleks at lave i Ladder Diagram for eksempel komplekse beregninger.

Instruction List (IL)

```
1  A (
2  A   "START"
3  A   "CYLINDER_INDE"
4  O   "CYLINDER_UD"
5  )
6  A   "STOP"
7  =   "CYLINDER_UD"
```

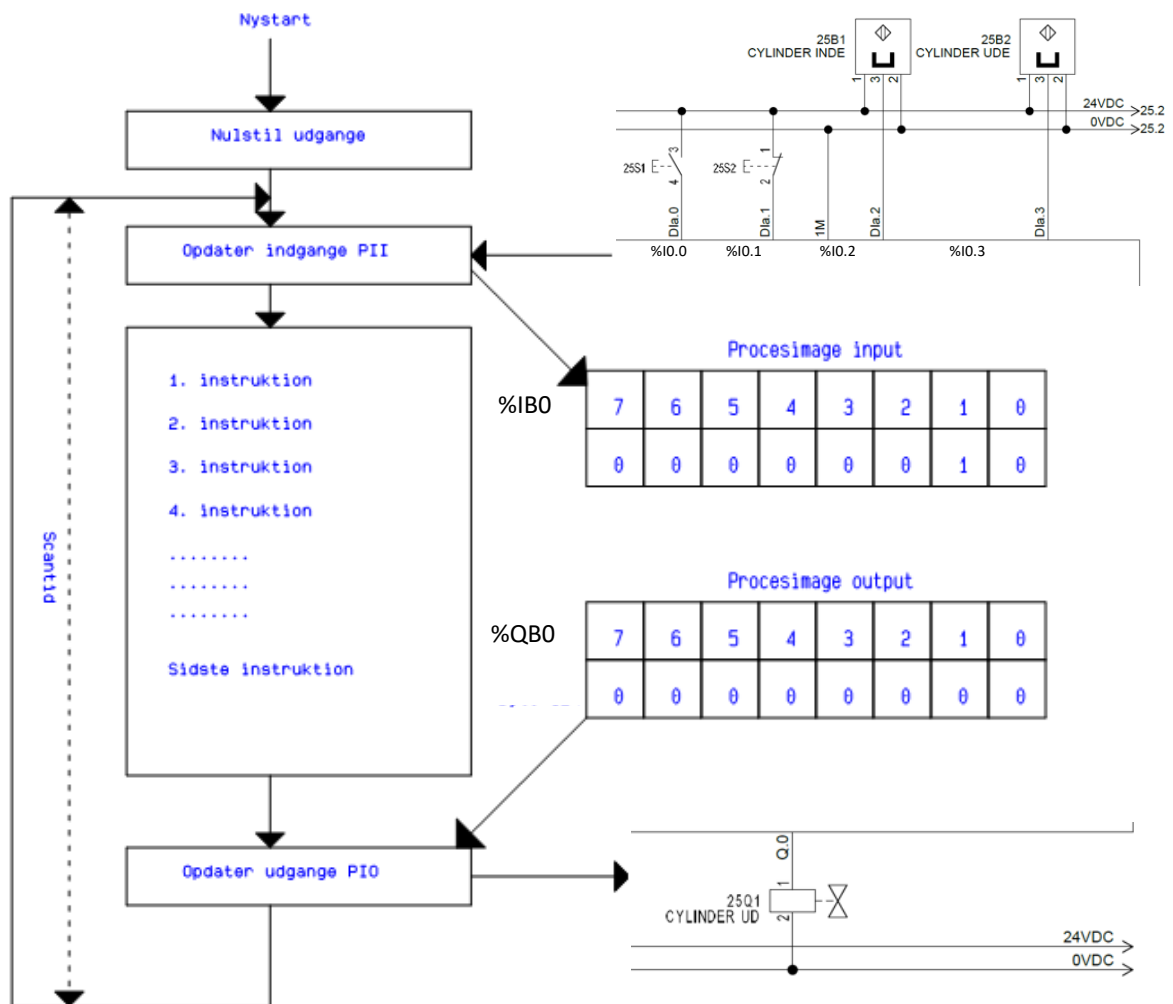
Når man programmerer i Instruction List skrives de logiske funktioner i den rækkefølge, som de skal eksekveres.

Normalt bruger man kun Instruction List til specielle funktioner, som ikke kan afbildes grafisk i programmeringssoftwaren. Programmet bliver hurtigt uoverskueligt for reparatøren. Instruktionslist er et gammelt programsprog og understøttes ikke af S7-1200, men S7-1500 PLC'er kan for bagud-kompatibilitetens skyld forsat programmeres i dette sprog.

Programeksekvering

Generelt

I en PLC bliver instruktionerne behandlet i en forudbestemt rækkefølge. Når alle instruktioner er behandlet, startes forfra i rækkefølgen. Resultatet af de forskellige instruktioner gemmes i en række interne registre, der løbende ændres under programafviklingen.



Ved nystart slettes udgangen procesbillede, hvilket betyder, at alle udgangsbit sættes på logisk 0.

Alle bit på indgange indlæses i PLC'ens statusregister for input. I dette register gemmes værdien, indtil den opdateres igen i næste scan. På engelsk benævnes input statusregisteret også PII, Proces Image Input.

Brugerprogrammet løbes igennem instruktion for instruktion. Når der i programmet "spørges" på en indgang, så hentes værdien fra statusregistret

Proces Image Input, på samme måde hvis der "skrives" til en udgang, så ændres værdien kun i statusregistret Proces Image Output (PIO).

Når programmet er løbet igennem, overføres de logiske værdier fra statusregistret Proces Image Output til output på kortet. Når det er overført, "fryses" værdien på udgangskortet indtil næste scan evt. ændre dem.

Scan

Den samlede programafvikling består af:

- Indlæsning fra indgange
- Programafvikling af brugerprogram
- Udlæsning til udgange

Scantiden er afhængig af antallet af instruktioner samt den CPU, som anvendes. Typisk er tiden mellem 1 og 10 ms for 1000 binære instruktioner.

Programmering

Generelt

I det foregående kapitel blev det nævnt, at en PLC bliver programmeret efter et nøgleskema. Dette er for så vidt også rigtigt bortset fra, at det mere ligner amerikanske nøgleskemaer kaldet Ladder-diagrammer (stigediagrammer), som er lidt anderledes end de danske/europæiske.

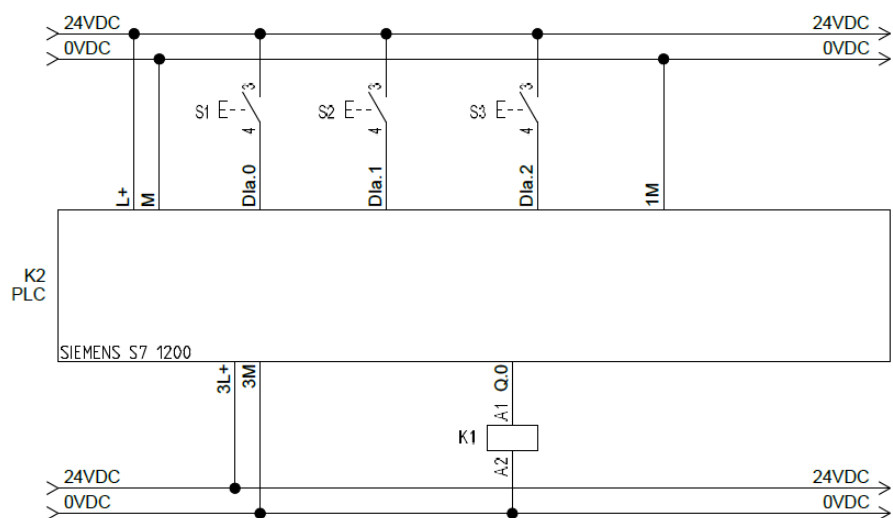
Hver enkelt PLC-fabrikat har deres egen programmeringssoftware. I disse softwarepakker kan man normalt vælge mellem flere forskellige sprog. Disse sprog benævnes Ladder Diagram, Function Block Diagram, Structured Text og Instruction List.

PLC-sproget er som før nævnt standardiseret og beskrevet i IEC 61131-3.

Programmering

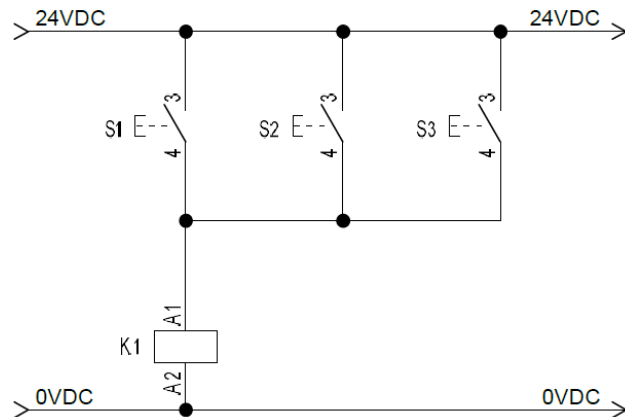
I det efterfølgende gennemgås de enkelte grundkoblinger, det vises hvordan de tegnes i nøgleskema, Ladder Diagram, Function Block Diagram, samt hvordan de skrives i Structured Text og Instruction List.

Tilslutning på PLC



Ovenstående lille styreopgave ønskes løst med en PLC. Derfor tilsluttes ind/udgange, som vist på diagrammet. Tilslutningen anvendes i de andre eksempler i dette kapitel.

ELLER-funktion



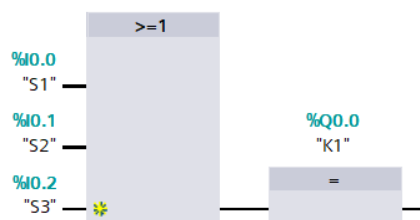
Diagrammet viser tre kontakter, S1, S2 og S3, som er parallelt forbundet. Der er 3 muligheder for at aktivere relæet, som bliver aktiveret, hvis minimum én kontakt bliver aktiveret.

Det boolske udtryk for diagrammet bliver hermed: $K1 = S1 + S2 + S3$.

Ladder Diagram



Function Block Diagram



Instruction List

| | | |
|---|---|------|
| 1 | O | "S1" |
| 2 | O | "S2" |
| 3 | O | "S3" |
| 4 | = | "K1" |

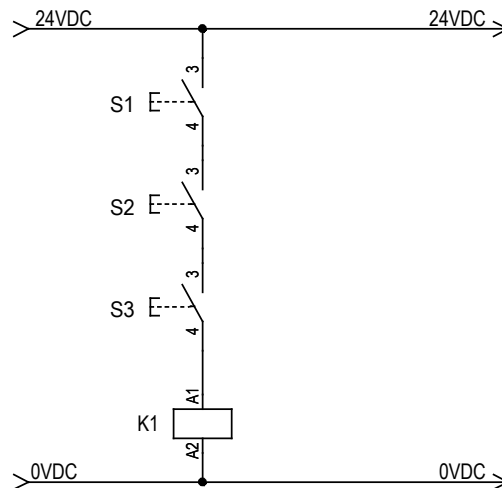
Structured Text

```

IF ("S1" OR "S2" OR "S3")
THEN
    "K1" := 1;
ELSE
    "K1" := 0;
END_IF;

```

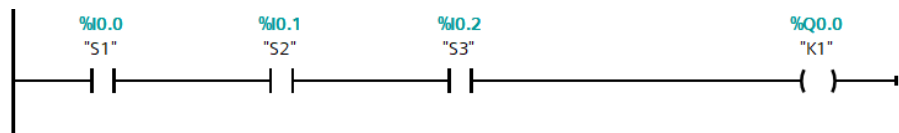
OG-funktion



Diagrammet viser tre kontakter, S1, S2 og S3, i en serieforbindelse. Alle kontakter skal være aktiveret for at få relæet aktiveret.

Det boolske udtryk for diagrammet bliver hermed: $K1 = S1 * S2 * S3$.

Ladder Diagram



Function Block Diagram



Instruction List

| | | |
|---|---|------|
| 1 | A | "S1" |
| 2 | A | "S2" |
| 3 | A | "S3" |
| 4 | = | "K1" |

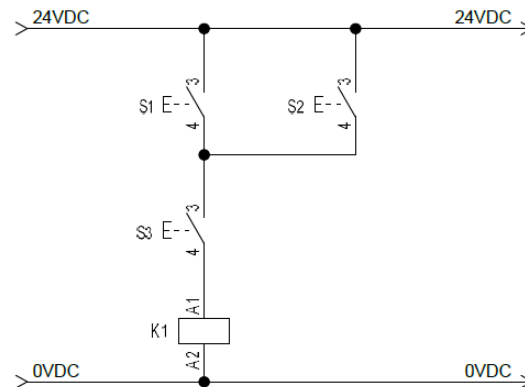
Structured Text

```

IF ("S1" AND "S2" AND "S3")
THEN
    "K1" := 1;
ELSE
    "K1" := 0;
END_IF;

```

Kombinatorisk logik



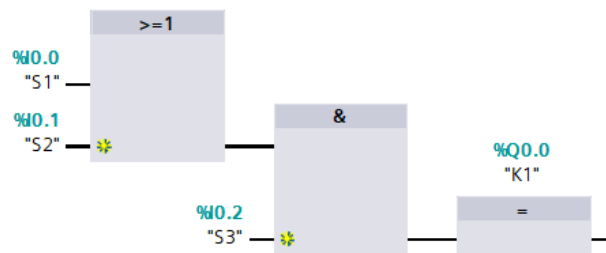
Et kombinatorisk kredsløb er kendetegnet ved, at udgangssignalets status altid svarer til det øjeblikkelige signalniveau på indgangene. Kredsløbet er ikke i stand til at »huske« tilstande.

Analyserer vi diagrammet, ser vi, at det består af en parallellforbindelse af S1 og S2. Det er ensbetydende med, at der skal benyttes en ELLER-funktion. S3 er forbundet i serie med parallellforbindelsen S1 og S2, så her anvendes en OG-funktion.

Ladder Diagram



Function Block Diagram



Instruction List

```

1      A (
2      O      "S1 "
3      O      "S2 "
4      )
5      A      "S3 "
6      =      "K1 "

```

Structured Text

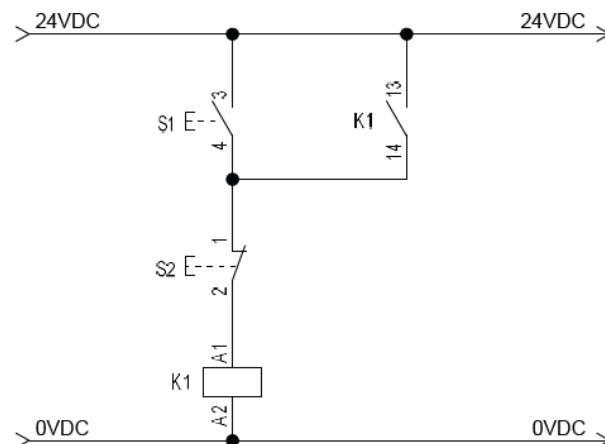
```

IF ("S1" OR "S2") AND "S3"
THEN
    "K1" := 1;
ELSE
    "K1" := 0;
END_IF;

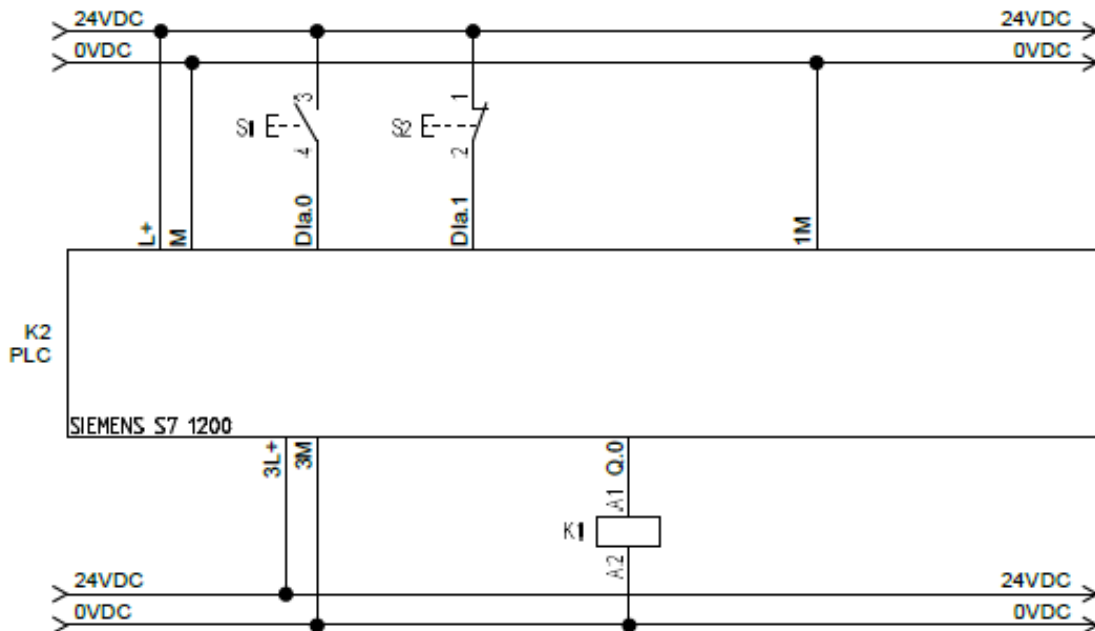
```

Sekventiel logik

Et sekventielt kredsløb er kendetegnet ved, at udgangssignalets status afhænger af signaler, der har været til stede, og som regel også af signaler, der kan tilbagestille disse, samt et »huske« kredsløb. Et meget godt eksempel er et start/stop-kredsløb.



Tilslutning på PLC



Indgangsinterfacet overfører et logisk 1 til PLC'en, hvis der er spændingssignal til stede på indgangsterminalerne. Hvis der ikke er spænding på en indgang, overføres et logisk 0-signal.

Derfor kan der ikke internt i PLC'en konstateres, om det er en slutte- eller en brydekontakt, der er forbundet til indgangen.

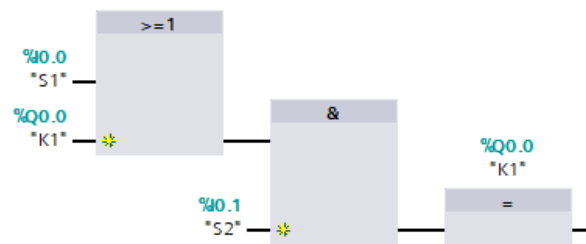
Hvis en brydefunktion tilsluttes en PLC-indgang, vil der internt i PLC'en være et logisk 1-signal, indtil kontakten aktiveres, hvorefter der vil være et logisk 0-signal.

Indgangssignaler i forbindelse med stop og sikkerhedsfunktioner er altid brydekontakter, fordi denne kontaktfunktion sikrer en overvågning af ledningsforbindelsen mellem kontakt og PLC.

Ladder Diagram



Function Block Diagram



Instruction List

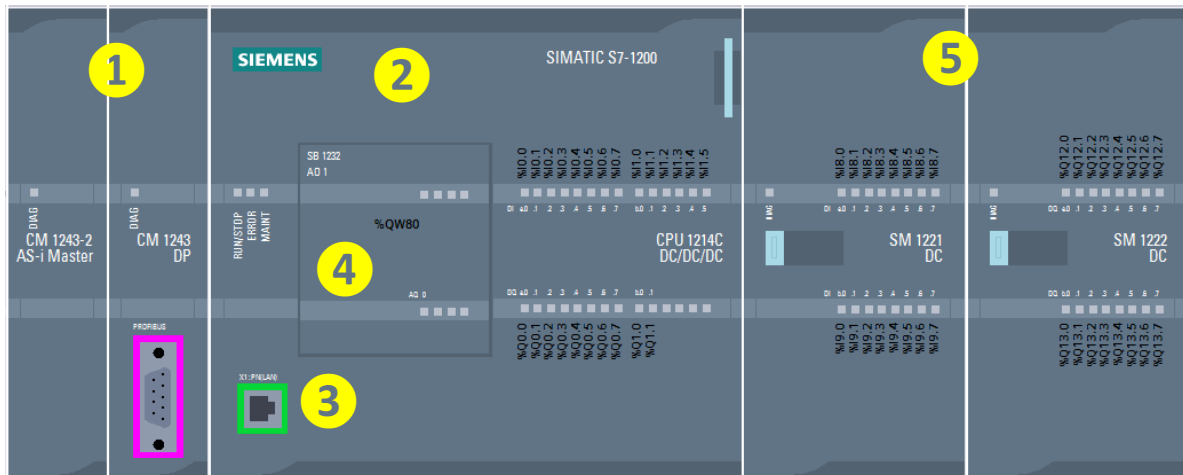
| | | |
|---|-----|------|
| 1 | A (| |
| 2 | O | "S1" |
| 3 | O | "K1" |
| 4 |) | |
| 5 | A | "S2" |
| 6 | = | "K1" |

Structured Text

```

IF "S1" = 1 THEN
    "K1" := 1;
END_IF;
IF "S2" = 0 THEN
    "K1" := 0;
END_IF;
  
```


Siemens S7-1200 PLC-bestykning



S7-1200 PLC-systemet består af følgende moduler:

① Kommunikationsmoduler

Kommunikationsmoduler aflaster CPU'en i forbindelse med diverse kommunikationsopgaver og muliggør kommunikation med andre bussystemer, end CPU'en giver mulighed for. Her er det PROFIBUS og AS-interface.

② Processormodul CPU (Central Processing Unit)

CPU-modulet indeholder processor, hukommelse og operativsystemet, som afvikler brugerprogrammet samt en CPU-bus.

Denne bus er en serial databus, hvor forsyning samt kommunikation overføres mellem PLC'ens moduler. Forbindelserne til CPU-bussen mellem modulerne etableres ved hjælp af busforbindelsstykker.

Afhængig af opgaven kan der vælges mellem flere CPU-typer. CPU'erne, der efter typebetegnelsen har forkortelsen C, indeholder enkelte specielle digitale indgange og udgange, samt normale digitale ind- og udgange og enkelte analoge ind- og udgangskanaler.

CPU'er med F i typebetegnelsen understøtter failsafe sikkerhedsfunktioner og PROFIsafe.

③ CPU-kommunikationsporte

CPU'en har typisk en til to PROFINET-porte, der kan bruges til kommunikation mellem CPU og programmerings-PC samt til netværkskommunikation med decentrale enheder.

④ Signalboard SB (Signal Boards)

Signalboard er små digitale eller analoge ind- og udgangskort. Disse kan indeholde specialfunktioner, såsom high speed puls in- og output. På analogsiden kan det være standart analog kort eller specialkort, der gør det muligt at tilslutte PT100-føler eller termokoblere direkte.

⑤ Signalmoduler SM (Signal Moduls)

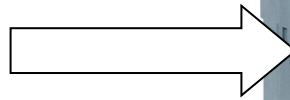
Signalmodulerne danner grænsefladen mellem anlæggets føler og handleorganer og PLC-systemet. Dvs. digitale og analoge ind- og udgangsmoduler.

Strømforsyning PM (Power Moduls)



PM-modulet forsynes med netspændingen (230 VAC) og leverer en 24 VDC driftsspænding til forsyningen af S7-1200 CPU'en. (PM- modulet er ikke vist på billedet ovenfor, da den ikke er en del af TIA's device configuration)

LED-statusvisninger



- RUN/ STOP** Fast gult lys angiver STOP-mode, Fast grønt angiver RUN-mode.
- ERROR** Rødt blink angiver fejl f.eks. en intern fejl i CPU'en, fejl på memory card eller konfigurationsfejl (modul mismatch)
- MAINT** Blinker, når et memory card indsættes. CPU'en går i STOP-mode. Når CPU'en er i STOP-mode kan man foretage en af følgende ting for at behandle memory card'et:
- Gå i RUN-mode
 - Foretage memory reset (MRES via TIA portal)
 - Genstart CPU

| Beskrivelse | STOP/RUN Gul/Grøn | ERROR Rød | MAINT Gul |
|---|----------------------------------|--------------|--------------|
| Power er off | Off | Off | Off |
| Startup, selftest, firmware update | Blink (skift mellem gul og grøn) | - | Off |
| Stop mode | On (gul) | - | - |
| Run mode | On (grøn) | - | - |
| Fjern memory card | On (gul) | - | Blink |
| Error | On (gul eller grøn) | Blink | - |
| Maintenance req. (vedligehold påkrævet) | On (gul eller grøn) | - | On |
| Defekt hardware | On (gul) | On | Off |
| LED-test eller defekt firmware | Blink | Blink | Blink |

Reset PLC'en til fabriksindstillinger

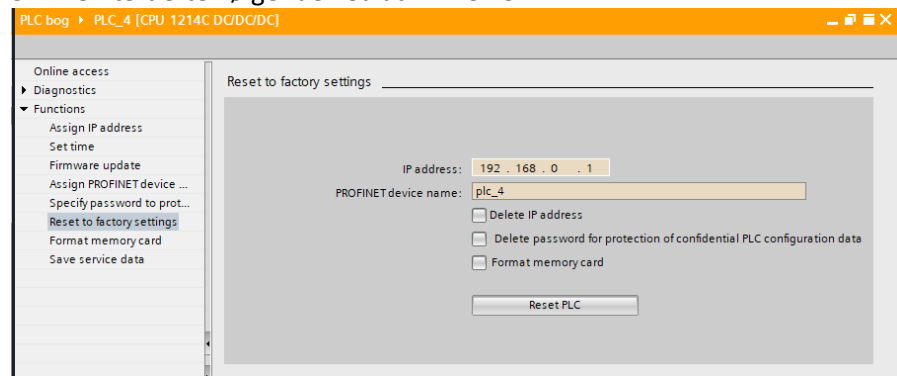
Forudsætninger:

- Der må ikke være isat memory card
- CPU'en skal have en online-forbindelse til TIA portal
- CPU'en skal være i STOP

Procedure:

Følg nedenstående steps for at resette til fabriksindstillinger:

1. Åben Online and Diagnostics-vinduet (TIA)
2. I "Functions" vælges "Reset to factory settings"
3. Vælg om du vil bevare IP-adressen, fjerne password, formatere memory card (som så skal side i PLC'en) eller ej i de respektive tjekbokse
4. Klik på Reset PLC-knappen
5. Bekræft efterfølgende ved at klikke "OK"



Resultat:

CPU'en skifter om nødvendigt til STOP-mode og resettes til fabriksindstillinger:

- Arbejdshukommelsen og den interne programhukommelse og alle variabel-områder slettes
- Alle parametre resettes til deres default-værdier
- Diagnosebufferen slettes
- Time of Day resettes
- IP-adressen og password slettes eller bevares i overensstemmelse med valgte indstillinger

Hukommelsesområder og deres funktioner:

TIA understøtter symbolsk adressering, dvs. man opretter en tag-tabel med relevante navne for projektets variabler og tildeler herefter fysisk I/O-absolutte adresser.

For overblikkets skyld skelner vi mellem følgende hukommelsestyper:

Global hukommelse: Det vil sige inputs (%I), outputs(%Q), og memorybits (%M). Disse typer kan tilgås af programmet kun begrænset af det installerede antal.

Tag-tabel: I tag-tabellen knyttes et symbolsk navn til den fysiske hukommelse. De symbolske navne anvendes under programmeringen, og gør således programmet lettere at læse.

Datablok (DB): I programmet kan der oprettes datablokke, som "indkapsler" definerede dele af hukommelsen. Du kan selv oprette datablokke til at indeholde globale variabler, eller de kan være tilknyttet en FB som instansdatablok. Sener i bogen er der beskrevet mere om dette.

Lokal hukommelse (L): Temporær. Når en programblok kaldes, tildeler CPU'en den fornødne hukommelse til brug under afviklingen. Efter afvikling frigives hukommelsen til brug for andre programblokke.

Hver hukommelses-bit har sin egen plads i lageret. Programmet tilgår disse adresser og læser deres værdi. Adresseringer af input (eks. %I0.7) eller output (eks. %Q1,2) tilgår procesimage. Hvis man ønsker "immediate read eller write" tilføjes ":P" efter adresseringen.

Table 4- 8 Memory areas

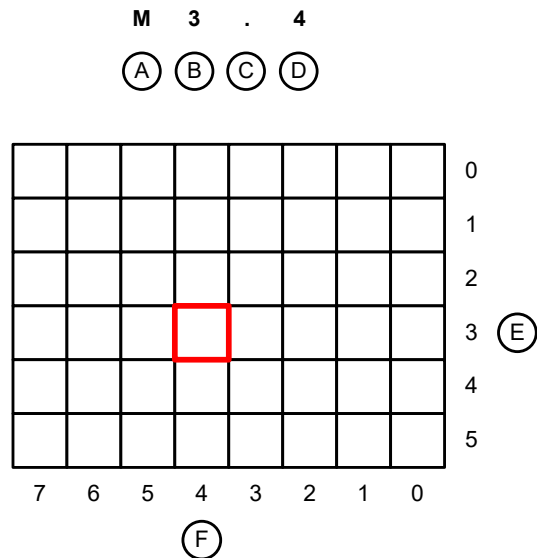
| Memory area | Description | Force | Retentive |
|---|--|-------|----------------|
| I Process image input I_:P ¹ (Physical input) | Copied from physical inputs at the beginning of the scan cycle | No | No |
| | Immediate read of the physical input points on the CPU, SB, and SM | Yes | No |
| Q Process image output Q_:P ¹ (Physical output) | Copied to physical outputs at the beginning of the scan cycle | No | No |
| | Immediate write to the physical output points on the CPU, SB, and SM | Yes | No |
| M Bit memory | Control and data memory | No | Yes (optional) |
| L Temp memory | Temporary data for a block local to that block | No | No |
| DB Data block | Data memory and also parameter memory for FBs | No | Yes (optional) |

¹ To immediately access (read or write) the physical inputs and physical outputs, append a ":P" to the address or tag (such as I0.3:P, Q1.7:P, or "Stop:P").

De absolutte eller fysiske adresser identificeres som følger:

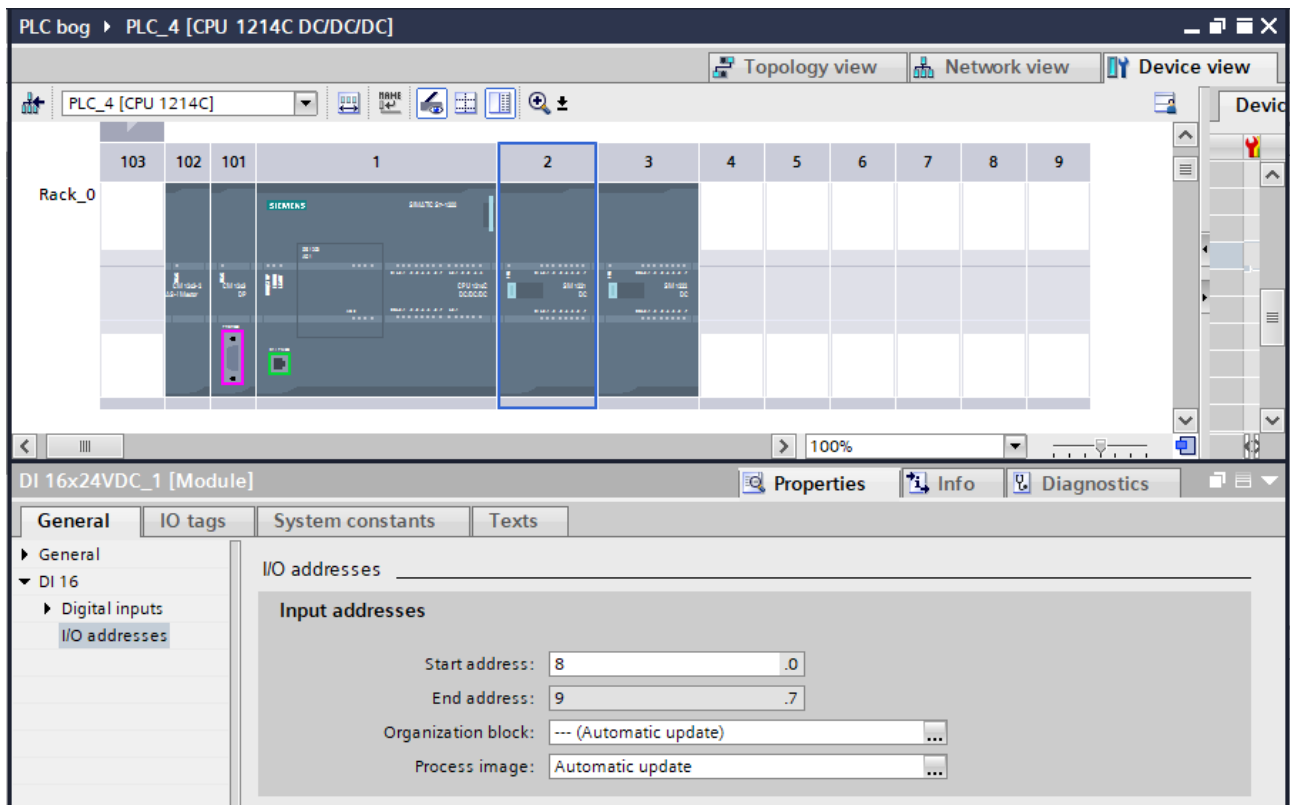
- Memory identifier (f. eks. I, Q eller M)
- Størrelse af den hukommelse, der ønskes adresseret (eks. B-byte, W-word eller D-doubleword)
- Startadressen som adresseres

Ved boolske variable opgives størrelsen ikke, den er underforstået:



- | | |
|---|--------------------------|
| A Memory area identifier | E Bytes of the memory |
| B Byte address: byte 3 | F Bits of the selected |
| C Separator ("byte.bit") | |
| D Bit location of the byte (bit 4 of 8) | |

Adresser



Adressering af moduler

I TIA kan man se, hvilke adresser de enkle kort automatisk får tildelt. Ønsker man ikke disse adresser, kan de ændres i TIA.

Det viste kort har 16 indgange og disse har følgende adresser.

Den første indgangsbyte får adresse 8 og den næste adresse 9. Det medfører, at de første 8 indgange benævnes:

I8.0, I8.1, I8.2, I8.3, I8.4, I8.5, I8.6 og I8.7

De næste 8 indgange benævnes:

I9.0, I9.1, I9.2, I9.3, I9.4, I9.5, I9.6 og I9.7

Byteadresse 10 og 11 anvendes ikke, da de er reserveret til, hvis man senere udskifter kortet til et 32-punkters kort.

Allen-Bradley 1769 PLC-bestykning



Allen-Bradley 1769 PLC-systemet består af følgende moduler:

① Kommunikationsmoduler

Kommunikationsmoduler, der muliggør kommunikation med andre bussystemer, end CPU'en giver mulighed for. Her er det DeviceNet.

② Processormodul CPU (**C**entral **P**rocessing **U**nit)

CPU-modulet indeholder processor, hukommelse og operativsystemet, som afvikler brugerprogrammet samt en CPU-bus. Denne bus er en serieel databus, hvor forsyning samt kommunikation overføres mellem PLC'ens moduler. Forbindelserne til CPU-bussen mellem modulerne etableres ved hjælp af busforbindelsstykker. Afhængig af opgaven kan der vælges mellem flere CPU-typer.

③ CPU-kommunikationsporte

USB-port bruges kun til kommunikation mellem CPU og programmerings-PC. Derudover har CPU'en typisk en til to Ethernet/IP-porte, der kan bruges til såvel PC som netværkskommunikation til decentrale enheder.

④ Digitale I/O-moduler

Digitale I/O-moduler danner grænsefladen mellem anlæggets føler og handleorganer og PLC-systemet. Dvs. digitale ind- og udgangsmoduler.

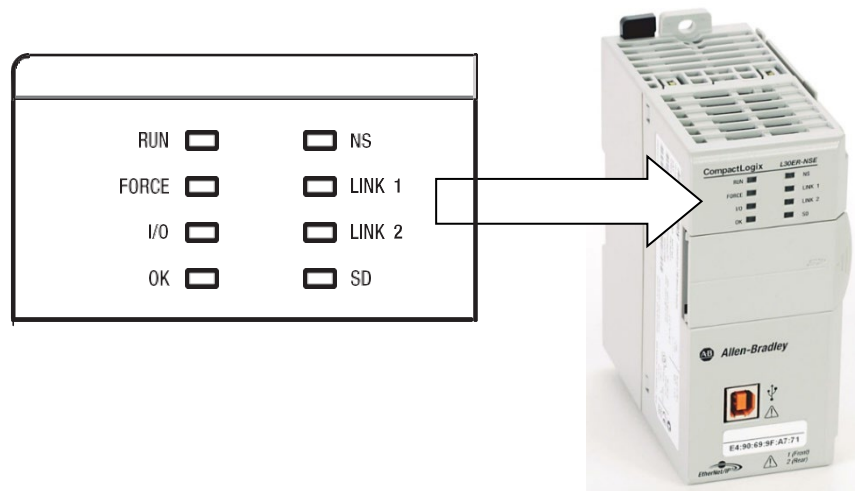
⑤ Analog I/O-moduler

Analog I/O-moduler danner grænsefladen mellem anlæggets analoge måleværdigiver og handleorganer og PLC-systemet. Dvs. analoge ind- og udgangsmoduler.

⑥ Strømforsyning PA (Power supply)

Power Supply-modulet forsynes med 230 VAC eller 24 VDC og leverer ved hjælp af CPU-bussen forsyning til de enkelte PLC-kort. Er den forsynet med 230 VAC, kan den også lever 24 VDC driftsspænding til forsyning af, de fra CPU-bussen galvanisk adskilte, ind- og udgangsdele på PLC-kortene.

LED-statusvisninger



På 1769-CPU'erne er der en række statusdioder, der kan anvendes i forbindelse med fejlfinding. Man kan også anvende programmeringssoftwaren til fejlfinding.

| Status Indicator | Description | Status |
|------------------|---|-----------------|
| RUN | Indicates the operating mode of the controller. | |
| | The controller is in Program or Test mode. | Off |
| | The controller is in Run mode. | Green |
| FORCE | Indicates the force state. | |
| | No tags contain I/O force values. I/O forces are inactive (disabled). | Off |
| | I/O forces are active (enabled). I/O force values can exist. | Yellow |
| | One or more input or output addresses have been forced to an On or Off condition, but the forces have not been enabled. | Flashing yellow |
| I/O | Indicates the current state of communication between the controller and I/O modules. | |
| | One of the following conditions exists: <ul style="list-style-type: none"> There are no devices in the I/O configuration of the controller. - Applies to only CompactLogix 5370 L3 controllers. The controller does not contain a project. | Off |
| | The controller is communicating with all devices in its I/O configuration. | Green |
| | One or more devices in the I/O configuration of the controller are not responding. | Flashing green |
| | One of the following conditions exists: <ul style="list-style-type: none"> The controller is not communicating with any devices. A fault has occurred on the controller. - Only CompactLogix 5370 L1 and L2 controllers. | Flashing red |
| | | |
| OK | Indicates the state of the controller. | |
| | No power is applied. | Off |
| | The controller is OK. | Green |

| | | |
|--------------------------|---|--------------------------------|
| | The controller is storing a project to or loading a project from the SD card. | Flashing green |
| | The controller detected a nonrecoverable major fault and cleared the project from memory. | Red |
| | One of the following: <ul style="list-style-type: none"> The controller requires a firmware update. A major recoverable fault occurred on the controller. A nonrecoverable major fault occurred on the controller and cleared the program from memory. A controller firmware update is in process. An embedded I/O module firmware update is in process. - Only CompactLogix 5370 L1 controllers. | Flashing red |
| | Save to Flash at power-down. | Dim green to red |
| | NS Indicates the EtherNet/IP network status regarding the controller operating on the network. | |
| | The port is not initialized; it does not have an IP address and is operating in BOOTP or DHCP mode. | Off |
| | The port has an IP address and CIP connections are established. | Green |
| | The port has an IP address, but no CIP connections are established. | Flashing green |
| | The port has detected that the assigned IP address is in use. | Red |
| | The port is performing its power-up self test. | Flashing red/green |
| LINK 1 and Link 2 | Indicates the EtherNet/IP link status for port 1 and 2 if the controller. | |
| | One of the following conditions exists: <ul style="list-style-type: none"> No link. Port administratively disabled. Port disabled because rapid ring fault condition was detected (LINK2). | Off |
| | One of the following conditions exists: <ul style="list-style-type: none"> A 100 Mbps link (half- or full-duplex) exists, no activity. A 10 Mbps link (half- or full-duplex) exists, no activity. Ring network is operating normally and the controller is the active supervisor. Ring network has encountered a rare partial network fault and the controller is the active supervisor. | Green |
| | One of the following conditions exists: <ul style="list-style-type: none"> A 100 Mbps link exists and there is activity. A 10 Mbps link exists and there is activity. | Flashing green |
| SD | Indicates if there is activity on the SD card. | |
| | There is no activity to the SD card. | Off |
| | The controller is reading from or writing to the SD card. | Flashing green |
| | The SD card does not have a file system. | Flashing red |

Adresser

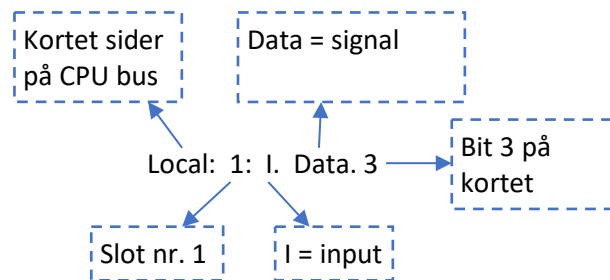


Adressering af moduler

Allen-Bradley 1769 PLC-familien adresser sine I/O-kort efter slot nr. og bit nr. Ovenfor kan man se, hvordan PLC'ens slot er nummereret. Bemærk at spændingsforsyningen ingen nummer har.

Digitale input-adresser

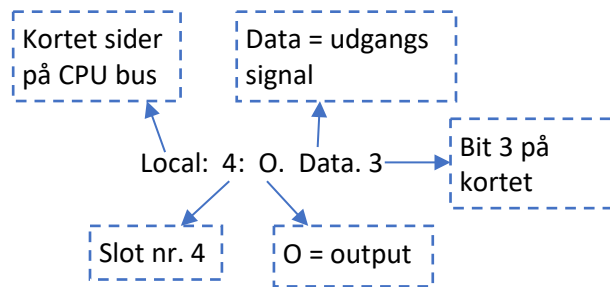
Input 3 på et digitalt input kort i "Slot 1" adresseres som vist herunder.



I programmeringssoftwaren ser det således ud, når der bruges et inputkort med 16 input.

| Controller Tags - PLC_bog(controller) | | | | | | |
|---------------------------------------|-----------|----------|-----------------|-------------|-----------------|--|
| Name | Alias For | Base Tag | Data Type | Description | External Access | |
| Local:1:I | | | AB:1769_DI16:k0 | | Read/Write | |
| Local:1:I.Fault | | | DINT | | Read/Write | |
| Local:1:I.Data | | | INT | | Read/Write | |
| Local:1:I.Data.0 | | | BOOL | | Read/Write | |
| Local:1:I.Data.1 | | | BOOL | | Read/Write | |
| Local:1:I.Data.2 | | | BOOL | | Read/Write | |
| Local:1:I.Data.3 | | | BOOL | | Read/Write | |
| Local:1:I.Data.4 | | | BOOL | | Read/Write | |
| Local:1:I.Data.5 | | | BOOL | | Read/Write | |
| Local:1:I.Data.6 | | | BOOL | | Read/Write | |
| Local:1:I.Data.7 | | | BOOL | | Read/Write | |
| Local:1:I.Data.8 | | | BOOL | | Read/Write | |
| Local:1:I.Data.9 | | | BOOL | | Read/Write | |
| Local:1:I.Data.10 | | | BOOL | | Read/Write | |
| Local:1:I.Data.11 | | | BOOL | | Read/Write | |
| Local:1:I.Data.12 | | | BOOL | | Read/Write | |
| Local:1:I.Data.13 | | | BOOL | | Read/Write | |
| Local:1:I.Data.14 | | | BOOL | | Read/Write | |
| Local:1:I.Data.15 | | | BOOL | | Read/Write | |

Digitale output-adresser

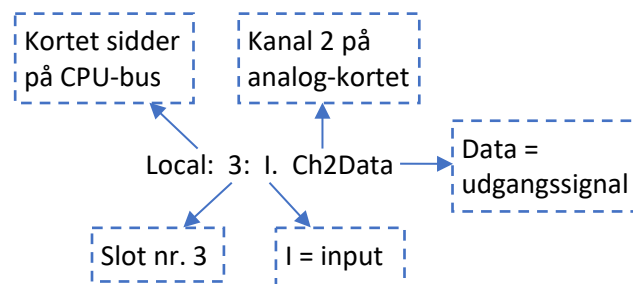


I programmeringssoftware ser det således ud, når der bruges et outputkort med 16 output.

| Controller Tags - PLC_bog(controller) | | | | | | |
|---------------------------------------|-----------|----------------|------------------|-----------------------------|-----------------|--|
| Scope: PLC_bog | | Show: All Tags | | Enter Description Filter... | | |
| Name | Alias For | Base Tag | Data Type | Description | External Access | |
| Local:4:C | | | AB:1769_DO16:C:0 | | Read/Write | |
| Local:4:I | | | AB:1769_DO16:I:0 | | Read/Write | |
| Local:4:O | | | AB:1769_DO16:O:0 | | Read/Write | |
| Local:4:O.Data | | | INT | | Read/Write | |
| Local:4:O.Data.0 | | | BOOL | | Read/Write | |
| Local:4:O.Data.1 | | | BOOL | | Read/Write | |
| Local:4:O.Data.2 | | | BOOL | | Read/Write | |
| Local:4:O.Data.3 | | | BOOL | | Read/Write | |
| Local:4:O.Data.4 | | | BOOL | | Read/Write | |
| Local:4:O.Data.5 | | | BOOL | | Read/Write | |
| Local:4:O.Data.6 | | | BOOL | | Read/Write | |
| Local:4:O.Data.7 | | | BOOL | | Read/Write | |
| Local:4:O.Data.8 | | | BOOL | | Read/Write | |
| Local:4:O.Data.9 | | | BOOL | | Read/Write | |
| Local:4:O.Data.10 | | | BOOL | | Read/Write | |
| Local:4:O.Data.11 | | | BOOL | | Read/Write | |
| Local:4:O.Data.12 | | | BOOL | | Read/Write | |
| Local:4:O.Data.13 | | | BOOL | | Read/Write | |
| Local:4:O.Data.14 | | | BOOL | | Read/Write | |
| Local:4:O.Data.15 | | | BOOL | | Read/Write | |

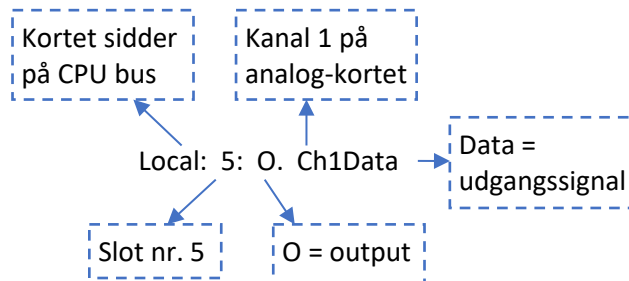
Analoge input-adresser

Et analogt input indeholder en variabel talværdi. Derfor skal en analog indgang bruge mere end 1 bit til at gemme denne værdi. På 1769 PLC'erne bruges der 16 bit til en analog indgang. Disse 16 bit benævnes som en kanal (channel).



Analoge output-adresser

Et analogt output indeholder en variabel talværdi. Derfor skal en analog udgang bruge mere end 1 bit til at gemme denne værdi. På 1769 PLC'erne bruges der 16 bit til en analog udgang. Disse 16 bit benævnes som en kanal (Channel).



Brug af Alias

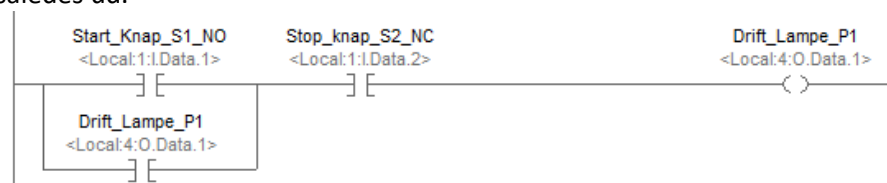
Når man skal programmer en PLC, er det en god ide at give ind- og udgangen nogle navne, så man kan huske, hvad de laver. I 1769 PLC'erne kan dette med fordel gøres ved at oprette en række "Alias"-tag for de I/O, man bruger.

Herunder kan man se, at der er oprette "Alias"-tag for to indgange og en udgang:

- Input "Local:1:I.Data.1", som har fået "Start_Knap_S1_NO" som Alias
- Input "Local:1:I.Data.2", som har fået "Stop_knap_S2_NC" som Alias
- Output "Local:4:O.Data.1", som har fået "Drift_Lampe_P1" som Alias

| Controller Tags - PLC_bog(controller) | | | | |
|---------------------------------------|------------------|------------------|------------------|-------|
| MainProgram - MainRoutine* | | | | |
| Scope: | PLC_bog | Show: | All Tags | Enter |
| Name | Alias For | Base Tag | Data Type | |
| Local:1:I | | | AB:1769_DI16:I:0 | |
| Local:2:I | | | AB:1769_DI16:I:0 | |
| Local:3:C | | | AB:1769_IF4:C:0 | |
| Local:3:I | | | AB:1769_IF4:I:0 | |
| Local:4:C | | | AB:1769_DO16:C:0 | |
| Local:4:I | | | AB:1769_DO16:I:0 | |
| Local:4:O | | | AB:1769_DO16:O:0 | |
| Local:5:C | | | AB:1769_OF2:C:0 | |
| Local:5:I | | | AB:1769_OF2:I:0 | |
| Local:5:O | | | AB:1769_OF2:O:0 | |
| Start_Knap_S1_NO | Local:1:I.Data.1 | Local:1:I.Data.1 | BOOL | |
| Stop_knap_S2_NC | Local:1:I.Data.2 | Local:1:I.Data.2 | BOOL | |
| Drift_Lampe_P1 | Local:4:O.Data.1 | Local:4:O.Data.1 | BOOL | |

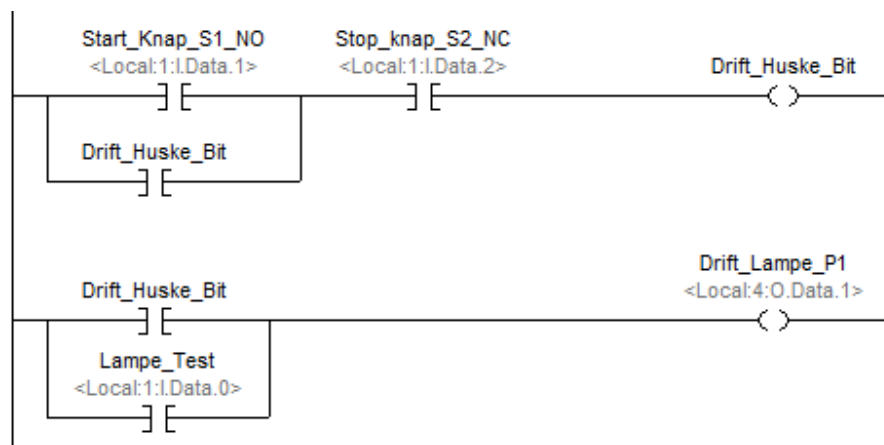
Når vi efterfølgende programmerer med disse adresser, kunne det se således ud.



Interne hukommelses-bit (BOOL)

Når man programmer en PLC, er der brug for at huske-bit for eksempel til en selvholdefunktion. Disse huskefunktioner skal anvende PLC'ens interne hukommelse. For at bruge denne interne hukommelse skal den adresseres. I 1769 PLC'er sker denne adressering ved at oprette et tag med et navn.

Herunder er programmet udvidet med en mulighed for at udføre lampetest ved hjælp af input "Local:1:I.Data.0". Vi ønsker at kunne udføre lampetesten uden at starte maskinen, derfor kan start/stop-kredsen ikke længere tænde driftslampen. Vi har derfor brug for et internt hukommelse bit, der kan huske, at vi har trykket start eller stop. Bittet er her kaldt "Drift_Huske_Bit".



"Drift_Huske_Bit" er som vist herunder oprette som en BOOL. Det betyder, at det kun er et bit, man adresserer. Og den er ikke alias for noget. Man kan lave lige så mange af disse bit, som man har brug for. Dog kan CPU-størrelsen begrænse dette ved meget store programmer.

| MainProgram - MainRout... | | Controller Tags - PLC_bog(controller) | |
|---------------------------|------------------|---------------------------------------|------------------|
| Scope: | PLC_bog | Show: | All Tags |
| Name | Alias For | Base Tag | Data Type |
| Drift_Huske_Bit | | | BOOL |
| Drift_Lampe_P1 | Local:4:O.Data.1 | Local:4:O.Data.1 | BOOL |
| Lampe_Test | Local:1:I.Data.0 | Local:1:I.Data.0 | BOOL |
| Local:1:I | | | AB:1769_DI16:I:0 |
| Local:2:I | | | AB:1769_DI16:I:0 |

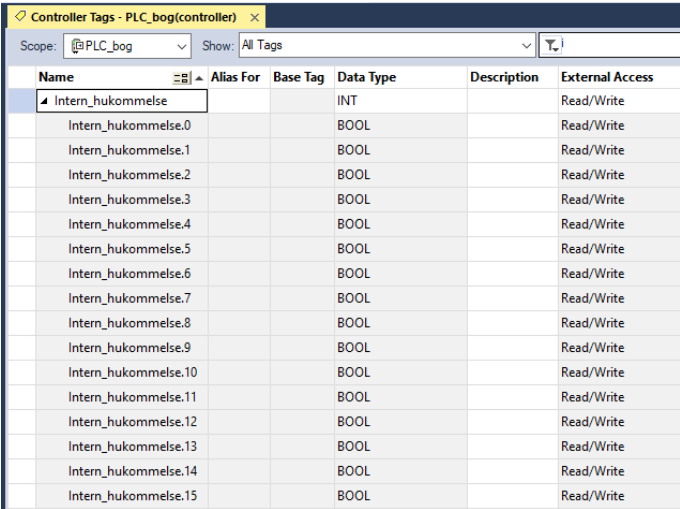
Interne hukommelsesord (INT, DINT og REAL)

Der kan også være behov for at huske talværdier i forbindelse med beregninger og håndtering af analoge ind- og outputs. Disse huskefunktioner anvender også PLC'ens interne hukommelse. For at bruge denne interne hukommelse skal den adresseres. I 1769 PLC'er sker denne adressering ved at oprette et tag med et navn. Er tagget oprettet som en INT, bruger den 16 bit, og disse er adresseret som vist nedenfor.

Navn på intern hukommelsesord

Bit nr.

Intern_hukommelse.6



| Name | Alias For | Base Tag | Data Type | Description | External Access |
|----------------------|-----------|----------|-----------|-------------|-----------------|
| Intern_hukommelse | | | INT | | Read/Write |
| Intern_hukommelse.0 | | | BOOL | | Read/Write |
| Intern_hukommelse.1 | | | BOOL | | Read/Write |
| Intern_hukommelse.2 | | | BOOL | | Read/Write |
| Intern_hukommelse.3 | | | BOOL | | Read/Write |
| Intern_hukommelse.4 | | | BOOL | | Read/Write |
| Intern_hukommelse.5 | | | BOOL | | Read/Write |
| Intern_hukommelse.6 | | | BOOL | | Read/Write |
| Intern_hukommelse.7 | | | BOOL | | Read/Write |
| Intern_hukommelse.8 | | | BOOL | | Read/Write |
| Intern_hukommelse.9 | | | BOOL | | Read/Write |
| Intern_hukommelse.10 | | | BOOL | | Read/Write |
| Intern_hukommelse.11 | | | BOOL | | Read/Write |
| Intern_hukommelse.12 | | | BOOL | | Read/Write |
| Intern_hukommelse.13 | | | BOOL | | Read/Write |
| Intern_hukommelse.14 | | | BOOL | | Read/Write |
| Intern_hukommelse.15 | | | BOOL | | Read/Write |

I eksemplet ovenfor er der anvendt "INT" som "Data Type". Det betyder, at der bruges 16 bit, som er det samme som et ord.

Man kan også oprette tag med andre "Data Types", for eksempel "DINT" og "REAL". Disse bruger 32 bit og er et dobbeltord. Når man programmerer, kan man både bruge hele ordet på engang eller et enkelt bit i ordet. De forskellige Data Type eller dataformater, som de også kaldes, bliver grundigere beskrevet i afsnittet "Allen-Bradley adressering, talformat og Word-instruktioner" i denne bog.

I/O-kredsløb

Signaltilpasning Interface

Ved interface forstås de kredsløb, som befinder sig mellem de ydre enheder og selve styringen. Interfacen er en »omsætter« af de signaler, der kommer fra ydre enheder, således at de er forståelige for f.eks. en PLC.

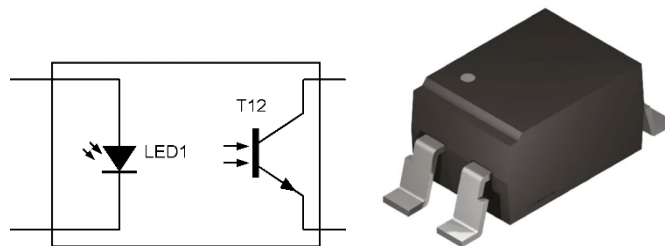
Interfaceudstyrets opgave er:

- At tilpasse processens styrespændinger til PLC'en
- At udføre galvanisk adskillelse mellem processen og PLC'en
- At undertrykke prel fra processen

Dette kapitel gennemgår henholdsvis input- og outputinterface.

Galvanisk adskillelse

Til galvanisk adskillelse anvendes ofte en optokobler.



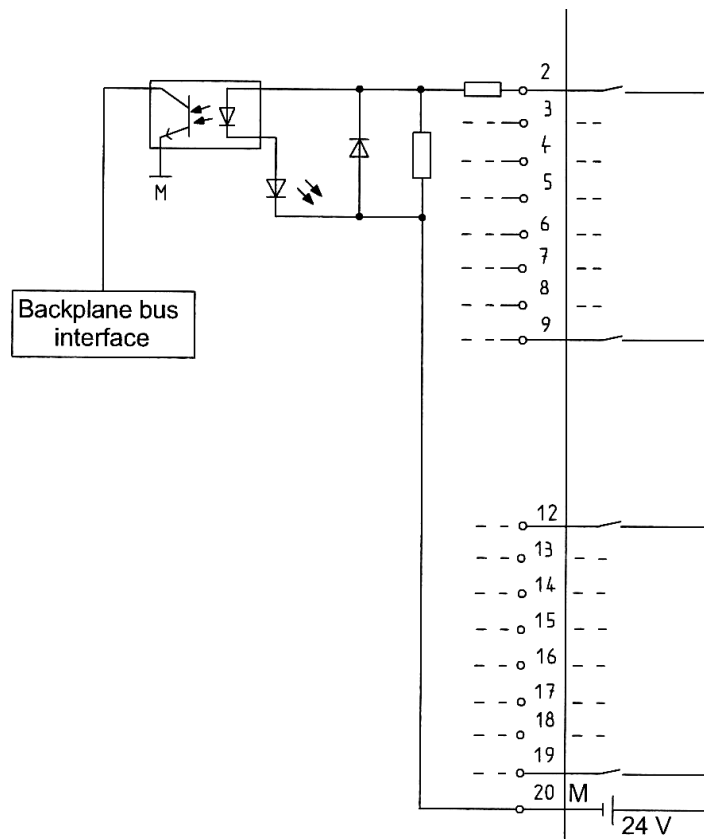
Optokobleren består af en lysdiode og en fototransistor i samme hus. Signalet overføres via lys, og derved opnås en galvanisk adskillelse.

Input Interface

Sink eller PNP input-interface

Sink eller PNP betyder, at der anvendes positive spændingssignaler på modulet.

Kortet har til formål at tilpasse indgangssignaler til PLC'ens spændingsniveau.



Diagrammet viser tilslutningen af et digitalt input på en Siemens PLC.

De to modstande danner en spændingsdel, der nedsætter spændingen til 5 volt.

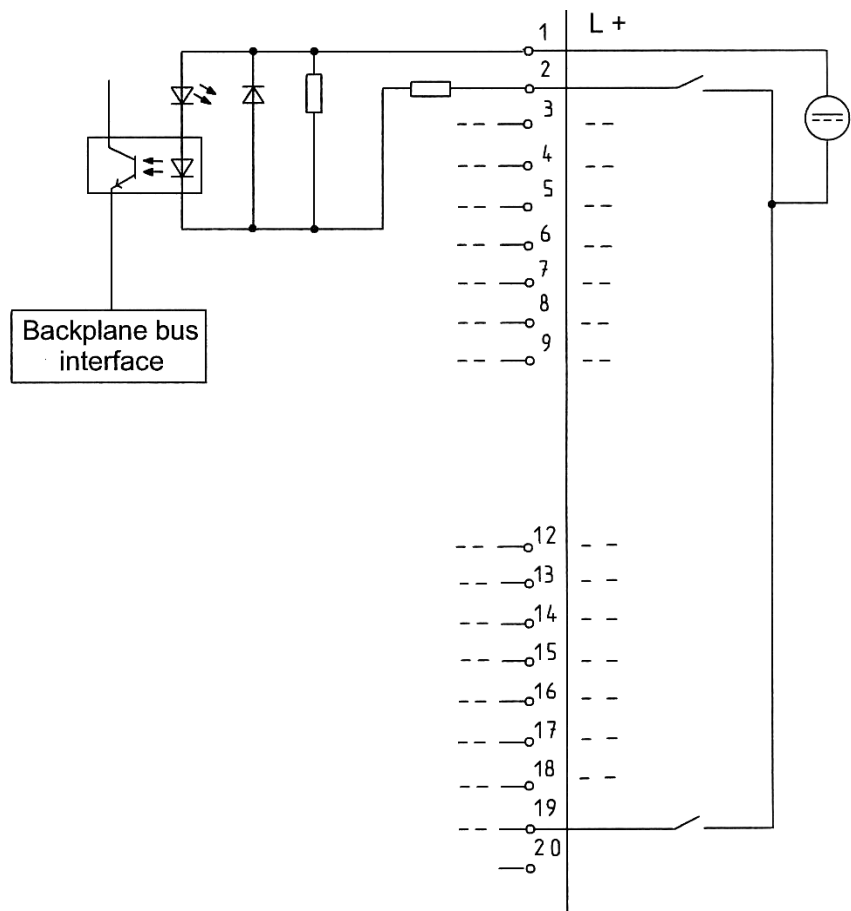
Dioden beskytter mod forkert polaritet på indgangen.

Strømmen løber fra indgangen og igennem optokobler og lysdiode for indikering af signal. Transistoren i optokobleren afleverer signalet videre til CPU'en, når den scannes.

Det viste inputmodul anvendes kun, når vi har positive spændingssignaler på indgangen. Et sådant modul benævnes også som PNP-indgang, hvilket også medfører, at der kun kan tilsluttes PNP-følere.

Source eller NPN-inputinterface

Source eller NPN betyder, at der anvendes negative spændingssignaler på modulet.



Diagrammet viser tilslutningen af et digitalt input på en Siemens PLC, hvor der anvendes NPN-signaler.

Kortet fungerer på samme måde som det foregående.

I Europa er det sjældent, at der anvendes NPN-indgange på en PLC.

Et sådant modul benævnes også som NPN-indgang, hvilket også medfører, at der kun kan tilsluttes NPN-følere.

Datablad CPU 1214C DC/DC/DC



Dette eksempel tager udgangspunkt i en Siemens CPU 1214C DC/DC/DC

Table A- 52 Digital inputs

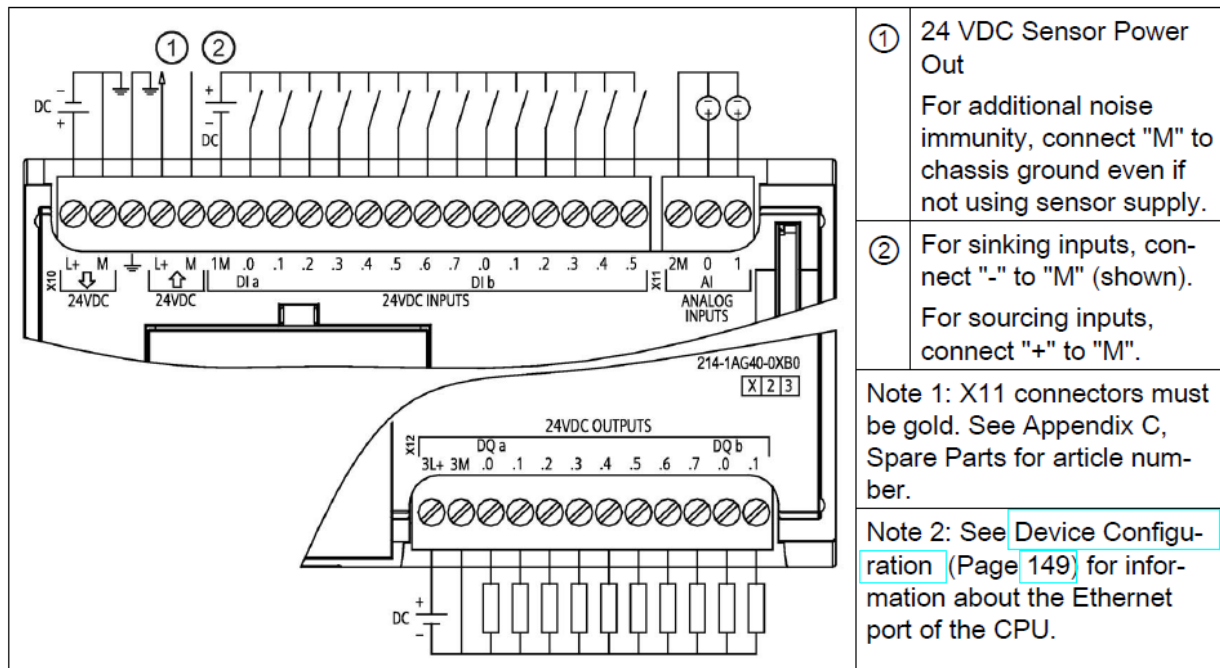
| Technical data | CPU 1214C AC/DC/Relay | CPU 1214C DC/DC/Relay | CPU 1214C DC/DC/DC |
|--|--|--------------------------|-----------------------|
| Number of inputs | 14 | | |
| Type | Sink/Source (IEC Type 1 sink) | | |
| Rated voltage | 24 VDC at 4 mA, nominal | | |
| Continuous permissible voltage | 30 VDC, max. | | |
| Surge voltage | 35 VDC for 0.5 sec. | | |
| Logic 1 signal (min.) | 15 VDC at 2.5 mA | | |
| Logic 0 signal (max.) | 5 VDC at 1 mA | | |
| Isolation (field side to logic) | 500 VAC for 1 minute | | |
| Isolation groups | 1 | | |
| Filter times | us settings: 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0 ms settings: 0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0 | | |
| HSC clock input rates (max.) (Logic 1 Level = 15 to 26 VDC) | 100/80 kHz (Ia.0 to Ia.5) 30/20 kHz (Ia.6 to Ib.5) | | |
| Number of inputs on simultaneously | <ul style="list-style-type: none"> 7 (no adjacent points) at 60 °C horizontal or 50 °C vertical 14 at 55 °C horizontal or 45 °C vertical | | |
| Cable length (meters) | 500 m shielded, 300 m unshielded, 50 m shielded for HSC inputs | | |

Ud fra databladet kan vi se, at de 14 indgange både kan anvende Sink eller Source. Det vil være mest typisk at anvende Sink.

Kortet er forsynet fra en 24 VDC-strømforsyning

Specielt skal man lægge mærke til, at spændingen på indgangen skal være minimum 15 V for logisk 1 og skal være mindre end 5 V for logisk 0.

Table A- 62 CPU 1214C DC/DC/DC (6ES7 214-1AG40-0XB0)



Diagrammet viser tilslutningen af PLC'en.

Sink-input

Hvis indgangene skal anvendes som PNP-indgange (Sink), tilsluttes strømforsyningen – pol på terminalen 1M.

Strømforsyningens + tilsluttes til indgangssignalerne.

Det betyder, når en indgang aktiveres, så kan der måles + 24 V på indgangen.

Dette er den mest anvendte tilslutning, og det er denne tilslutning, som er vist på diagrammet.

Source-input

Det samme indgangskort kan også bruges som en NPN-indgang (Source).

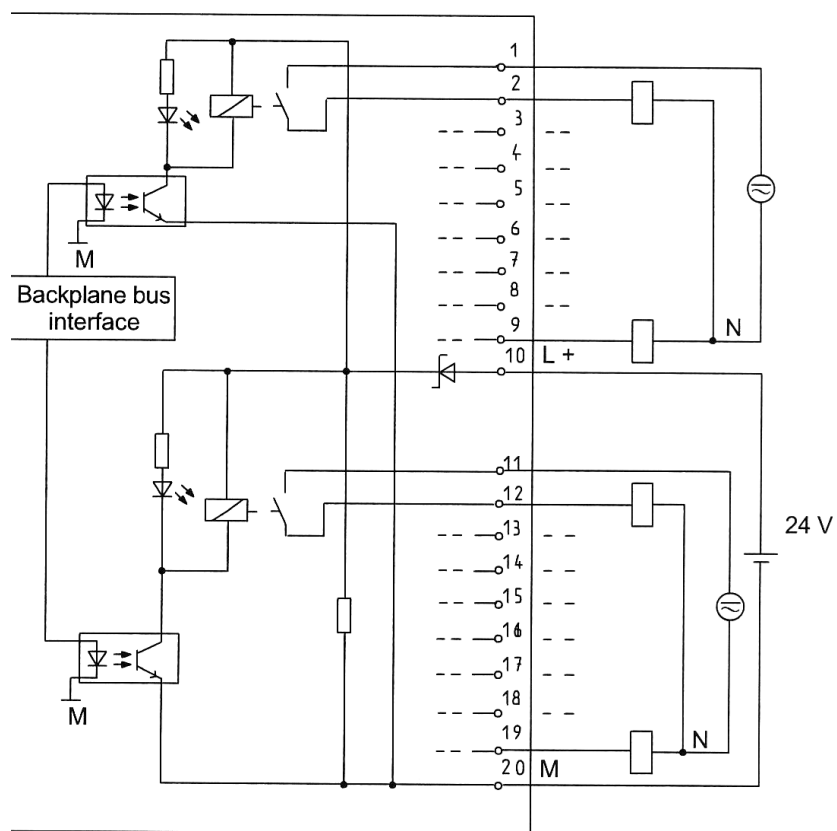
Her tilsluttes strømforsyningen + pol til 1M, Strømforsyningens - pol tilsluttes til indgangssignalerne.

Output interface

Formålet med et udgangsinterface er at tilpasse PLC'ens signal til de ydre enheder.

Her er en galvanisk adskillelse som regel nødvendig, da der ofte er tale om ind/udkobling af større strømme samt induktive belastninger fra relæ- og ventilspoler. En eventuel fejl eller overspænding må ikke forårsage ødelæggelse af PLC-CPU'en. Den galvaniske adskillelse mellem PLC'ens bussystem og de ydre enheder er enten udført med et relæ eller en optokobler.

Relæ udgang



Relæudgange ses ofte på mindre PLC'er. Fordelen ved et sådant modul er, at det uden videre kan tilpasse sig den ønskede udgangsspænding.

Der er mulighed for både AC- og DC-tilslutninger.

Der er ofte tale om fysisk meget små relæer, som kun kan klare en mindre belastning. Ulempen er foruden den lille belastning strøm, en større fejlhyppighed end ved elektroniske udgangssignaler. Da relæer slides ganske lidt hver gang de tænder og slukker.

Datablad CPU 1214C DC/DC/RLY



Dette eksempel tager udgangspunkt i en Siemens CPU 1214C DC/DC/RLY

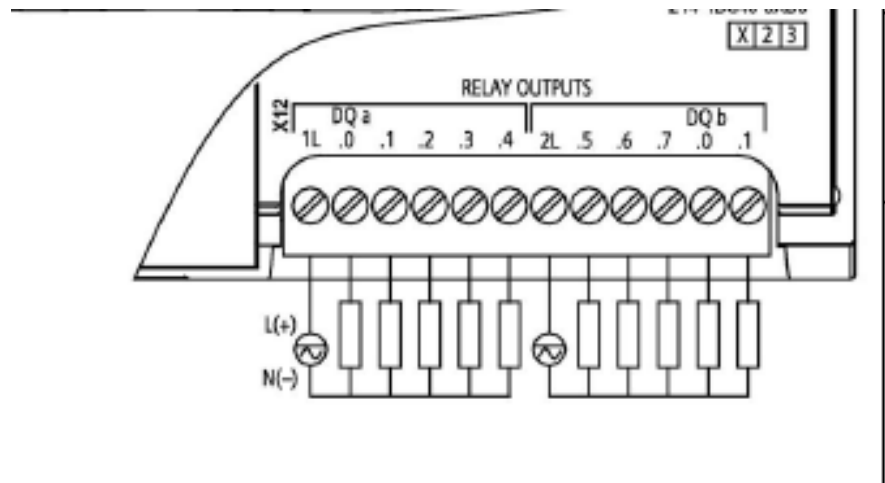
PLC'en forsynes med en 24 VDC, og den har 24 VDC-indgange og AC/DC relæudgange.

Table A- 53 Digital outputs

| Technical data | CPU 1214C AC/DC/Relay and DC/DC/Relay | CPU 1214C DC/DC/DC |
|--|---|---------------------------------|
| Number of outputs | 10 | |
| Type | Relay, mechanical | Solid state - MOSFET (sourcing) |
| Voltage range | 5 to 30 VDC or 5 to 250 VAC | 20.4 to 28.8 VDC |
| Logic 1 signal at max. current | -- | 20 VDC min. |
| Logic 0 signal with 10 K Ω load | -- | 0.1 VDC max. |
| Current (max.) | 2.0 A | 0.5 A |
| Lamp load | 30 W DC / 200 W AC | 5 W |
| ON state resistance | 0.2 Ω max. when new | 0.6 Ω max. |
| Leakage current per point | -- | 10 μ A max. |
| Surge current | 7 A with contacts closed | 8 A for 100 ms max. |
| Overload protection | No | |
| Isolation (field side to logic) | 1500 VAC for 1 minute (coil to contact) None (coil to logic) | 500 VAC for 1 minute |
| Isolation resistance | 100 M Ω min. when new | -- |
| Isolation between open contacts | 750 VAC for 1 minute | -- |
| Isolation groups | 2 | 1 |

Ud fra databladet kan vi se, at PLC'en har 10 relæudgange.

Hvis der anvendes DC, så er spændingsområdet fra 5 til 30 V, ved AC er spændingsområdet fra 5 til 250 V.

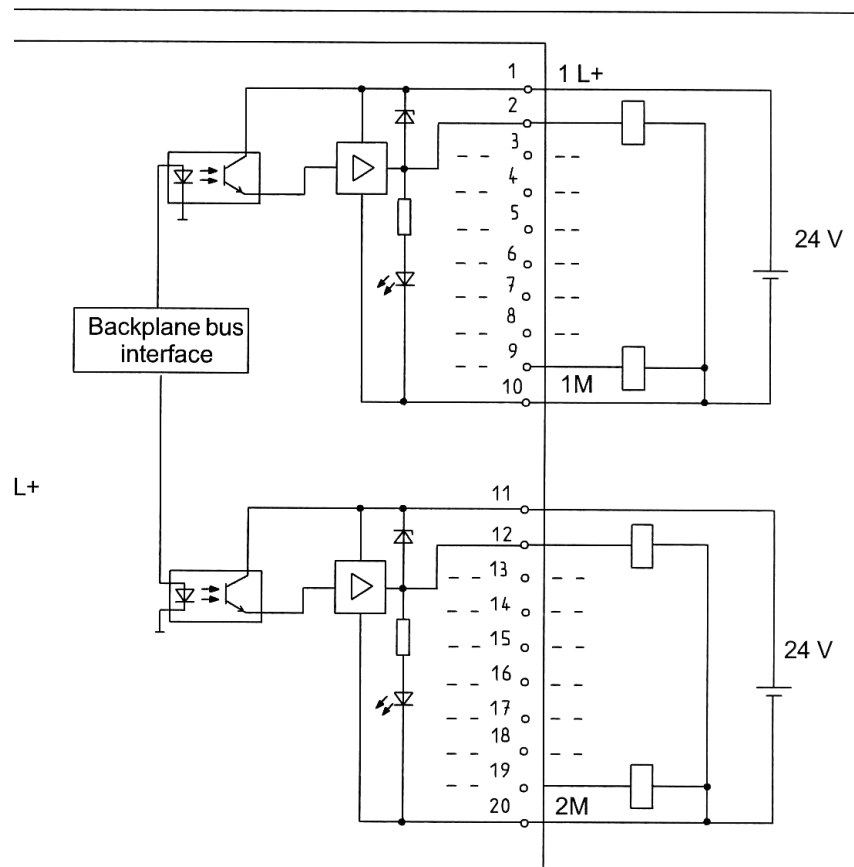


Diagrammet viser tilslutning af relæudgang.

Hvis udgangssignalet skal være er f.eks. 230 VAC, så tilsluttes fase på terminalen 1L og nulledningen på hver enkelt forbruger. Når udgangen er aktiv, så trækker relæet i udgangen, og der kommer en 230 V-fase på udgangen.

Det er sjældent, at man bruger 230 VAC-udgange, men muligheden findes.

Transistor source-udgang



Det mest benyttede udgangskort er et transistormodul, hvor der er mulighed for hurtige reaktioner til f.eks. displaymoduler. Et sådant modul kan kun anvendes til DC, som regel 24 VDC.

Hvis der er behov for andre spændinger eller større strømme, end modulet kan klare, indkobles der et interface-/overdragsrelæ imellem udgang og handleorganer.

Sådanne relæer er efterhånden så små og kompakte, at de er indbygget i klemrækker. Fordelen ved et eksternt relæ frem for et indbygget relæ i et relæmodul er, at en udskiftning er simpel og hurtigt udført.

Datablad CPU 1214C DC/DC/DC



Dette eksempel tager udgangspunkt i en Siemens CPU 1214C DC/DC/DC

PLC'en har en 24 VDC-forsyning, både ind- og udgange anvender 24 VDC.

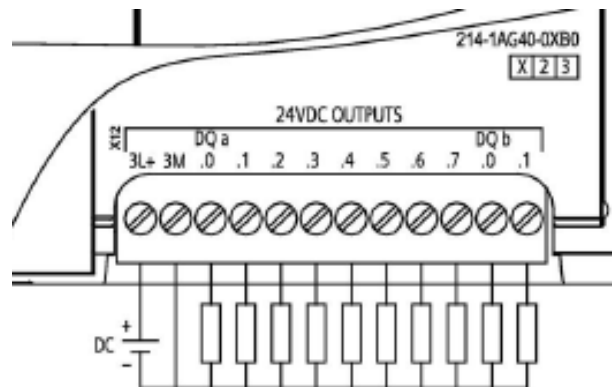
Table A- 53 Digital outputs

| Technical data | CPU 1214C AC/DC/Relay and DC/DC/Relay | CPU 1214C DC/DC/DC |
|--|---|---------------------------------|
| Number of outputs | 10 | |
| Type | Relay, mechanical | Solid state - MOSFET (sourcing) |
| Voltage range | 5 to 30 VDC or 5 to 250 VAC | 20.4 to 28.8 VDC |
| Logic 1 signal at max. current | -- | 20 VDC min. |
| Logic 0 signal with 10 K Ω load | -- | 0.1 VDC max. |
| Current (max.) | 2.0 A | 0.5 A |
| Lamp load | 30 W DC / 200 W AC | 5 W |
| ON state resistance | 0.2 Ω max. when new | 0.6 Ω max. |
| Leakage current per point | -- | 10 μ A max. |
| Surge current | 7 A with contacts closed | 8 A for 100 ms max. |
| Overload protection | No | |
| Isolation (field side to logic) | 1500 VAC for 1 minute (coil to contact) None (coil to logic) | 500 VAC for 1 minute |
| Isolation resistance | 100 M Ω min. when new | -- |
| Isolation between open contacts | 750 VAC for 1 minute | -- |
| Isolation groups | 2 | 1 |

Databladet for de digitale udgange viser, at der er 10 udgange på PLC'en af PNP-typen (Source).

Udgangene forsynes normalt med en 24 VDC-forsyning, minimum forsyning er 20,4 V, og maksimum er 28,8 VDC.

Den strøm, der maks. kan trækkes ud af en udgang, er 0,5 A.



Diagrammet viser tilslutning af DC-udgang.

Spændingsforsyningens + pol tilsluttes på terminalen 3L.

Spændingsforsyningens – pol tilsluttes på hver enkelt forbruger.

En forbruger kunne f.eks. være et overdragsrelæ, en lampe eller en magnetventil.

Når udgangen er aktiv, så sender PLC'en et 24 V + signal ud på udgangen.

Timer-kredsløb

PLC-program Siemens timer

I en Siemens S7 PLC findes der 4 forskellige typer af timere. I dette kapitel er en kort gennemgang af de to mest brugte typer: On-delay og off-delay.

En Siemens timer kan programmeres i to forskellige standarder. Den ene benævnes boks timer, mens den anden benævnes bit timer.

Tidsområdet for en S7 TIME (IEC time) er i millisekunder. Tiden kan indeholde oplysninger for dage (d), timer (h), minutter (m), sekunder (s) og millisekunder (ms). Værdien må ikke overstige 24 dage, 23 timer, 59 minutter, 59 sekunder eller 999 millisekunder

Konstanten for en timerværdi angives med T#, hvilket medfører, at f.eks. en tidsforsinkelse på 3 sekunder angives T#3s.

Tiden kan angives i dage, timer, minutter, sekunder eller millisekunder.

T#10d_20h_30m_20s_630ms

Det er ikke nødvendigt at angive alle tidsenheder.

Timeren er tilknyttet en datablok med følgende struktur.

Parameters

The following table shows the parameters of the "Generate on-delay" instruction:

| Parameters | Declaration | Data type | | Memory area | | Description |
|------------|-------------|-----------|-------------|---------------------------|-------------------------------|---|
| | | S7-1200 | S7-1500 | S7-1200 | S7-1500 | |
| IN | Input | BOOL | BOOL | I, Q, M, D, L or constant | I, Q, M, D, L, P, or constant | Start input |
| PT | Input | TIME | TIME, LTIME | I, Q, M, D, L or constant | I, Q, M, D, L, P or constant | Duration of the on-delay The value of the PT parameter must be positive. |
| Q | Output | BOOL | BOOL | I, Q, M, D, L | I, Q, M, D, L, P | Output that is set when the time PT expires. |
| ET | Output | TIME | TIME, LTIME | I, Q, M, D, L | I, Q, M, D, L, P | Current time value |

Allen-Bradley timer

I en Allen-Bradley PLC findes der 6 forskellige typer af timere. I dette kapitel er en kort gennemgang af de to mest brugte typer: On-delay og off-delay.

Tidsenheden er altid 1 millisekund. For en 2 sekunders timer skal du f.eks. indtaste 2000 i Preset-værdien.

Dataformatet for tidsenheden er double integer "DINT" 32 bit, hvilket giver mulighed en maksimal tid på 2.147.483,0 sekunder eller 1.491,0 timer.

Timer skal tilknyttes et tag, der har følgende datastruktur:

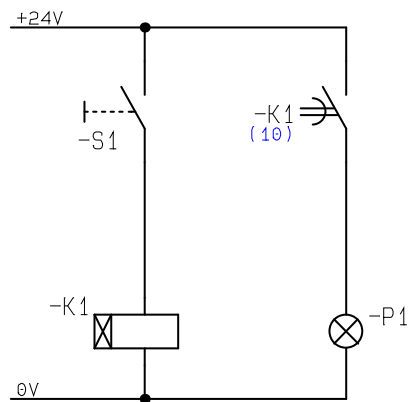
TIMER Structure

| Mnemonic | Data Type | Description |
|----------|-----------|--|
| .EN | BOOL | The enable bit contains rung-condition-in when the instruction was last executed. |
| .TT | BOOL | The timing bit when set indicates the timing operation is in process. |
| .DN | BOOL | The done bit when cleared indicates the timing operation is complete (or paused). |
| .PRE | DINT | The preset value specifies the value (1 millisecond units) which the accumulated value must reach before the instruction indicates it is done. |
| .ACC | DINT | The accumulated value specifies the number of milliseconds that have elapsed since the TOF instruction was enabled. |

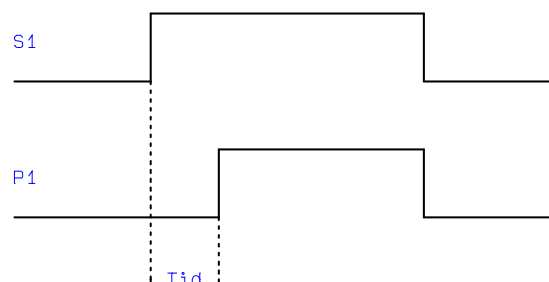
Forsinket indkobling (On-delay)

Alle PLC'er har en indkoblingsforsinket timer som standard. Rent programmeringsmæssigt kan der være en del forskel fra system til system.

Nøgleskema



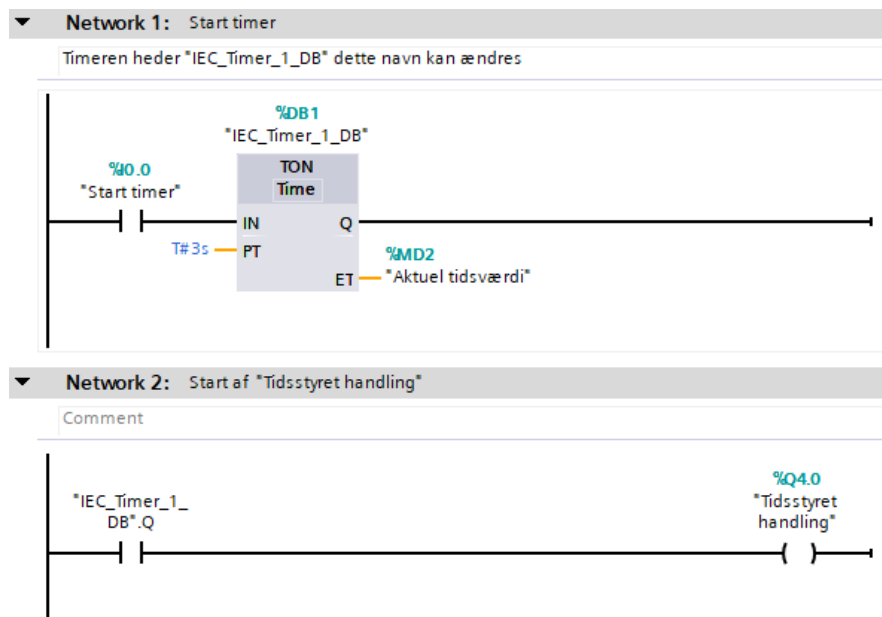
Funktionsdiagram



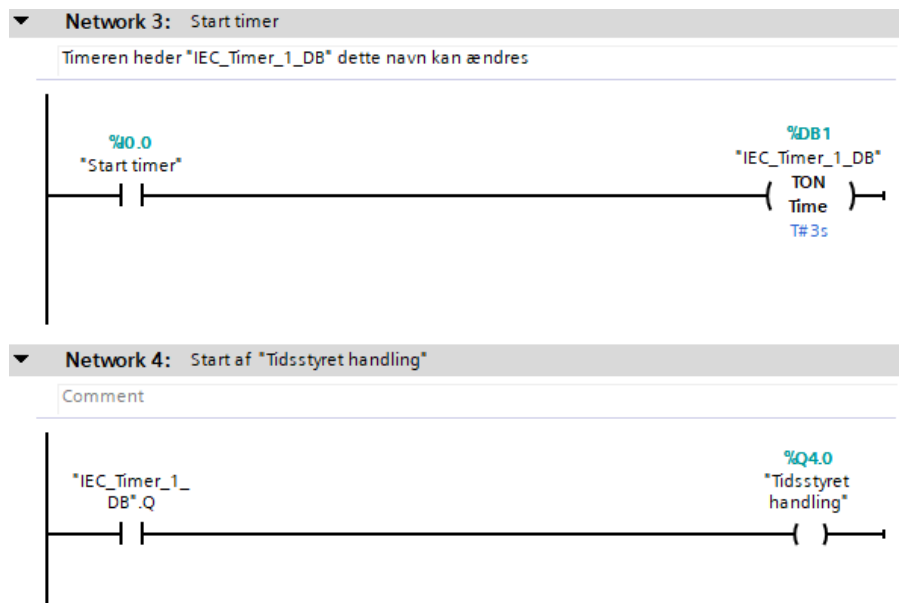
Når S1 er aktiv starter timeren K1. Når tiden er udløbet, trækker kontakten, og lampen P1 lyser. Hvis signalet fjernes på K1, falder kontakten fra med det samme, og lampen P1 slukker igen.

On-delay timer Siemens

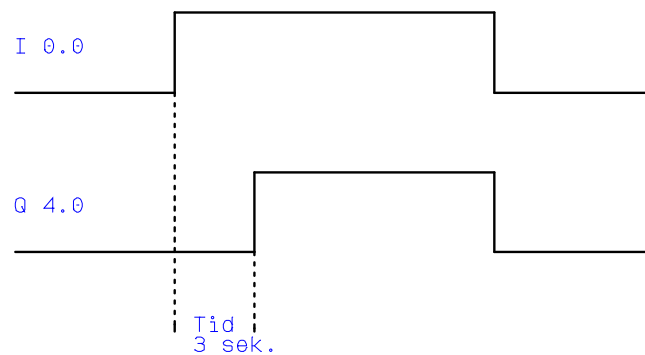
Boks timer



Bit-timer



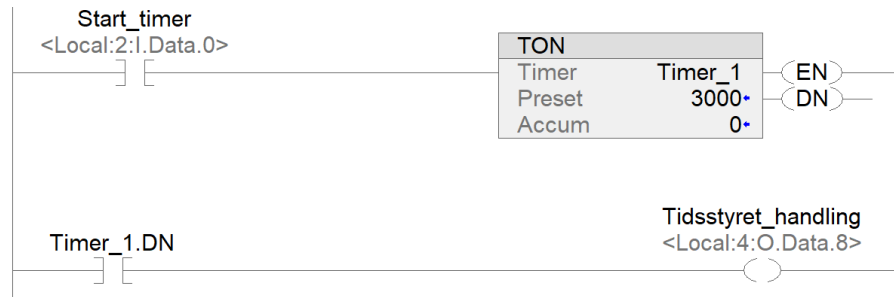
Funktionsdiagram



Når I 0.0 er aktiv, starter timeren, og efter 3 sekunder trækker udgangen Q 4.0.

På udgangen ET er en udlæsning af, hvor lang tid der er gået, siden timeren begyndte sin tidsudmåling. Tiden udlæses TIME format.

On-delay timer Allen-Bradley



Funktionsbeskrivelse

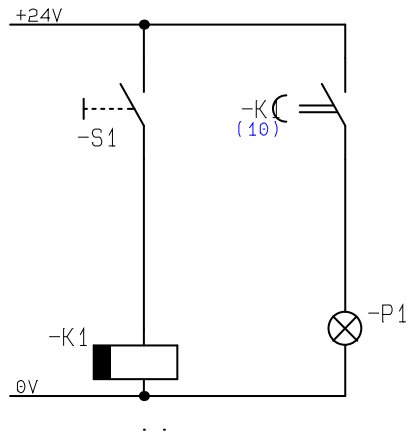
Når indgang Local:2:I.Data.0 er aktiv, starter timeren. Efter Preset-værdien på 3 sekunder er gået, er den done .DN (færdig) og trækker udgang Local:4:O.Data.8.

Accum (akkumuleret tid) er en udlæsning af hvor lang tid, der er gået, siden timeren begyndte sin tidsudmåling.

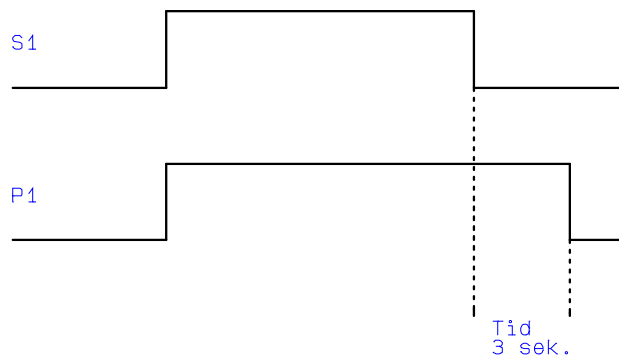
Timerens tag hedder "Timer_1".

Forsinket udkobling (Off-delay)

Nøgleskema



Funktionsdiagram



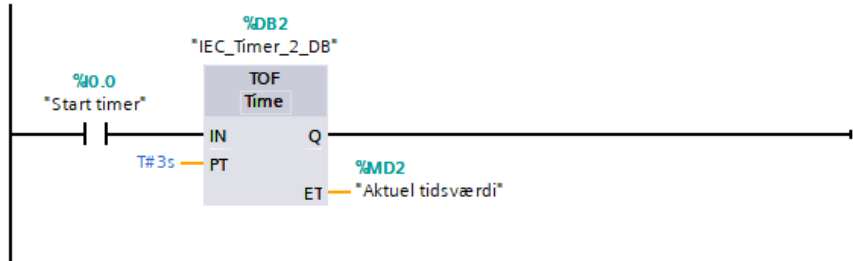
Når S1 er aktiv, trækker timeren, kontakten lukker og lampen P1 lyser. Hvis signalet fjernes på K1, starter timeren, og kontakten falder fra efter 3 sekunder, hvorefter lampen P1 slukker igen.

Off-delay timer Siemens

Boks timer

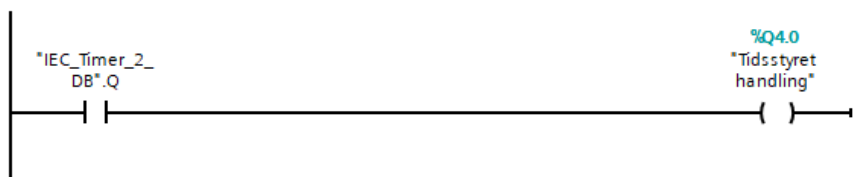
Network 5: Start timer

Timeren hedder "IEC_Timer_2_DB" dette navn kan ændres



Network 6: Start af "Tidsstyret handling"

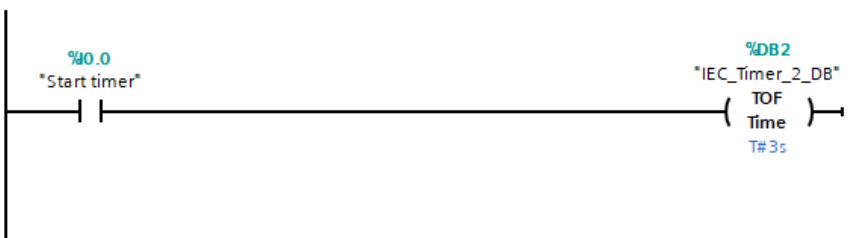
Comment



Bit timer

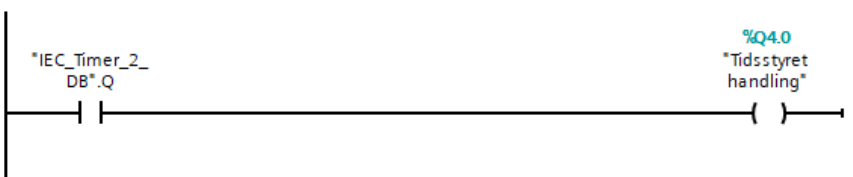
Network 7: Start timer

Timeren hedder "IEC_Timer_2_DB" dette navn kan ændres

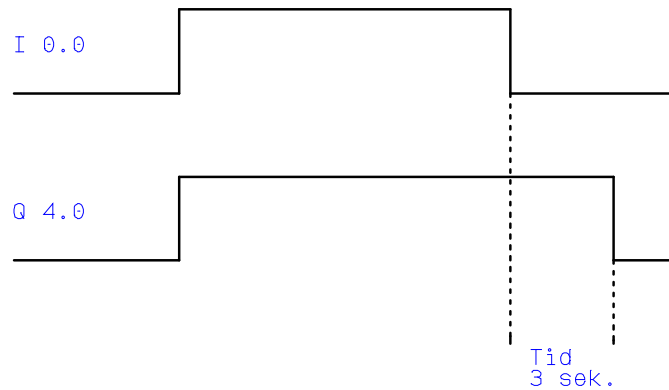


Network 8: Start af "Tidsstyret handling"

Comment



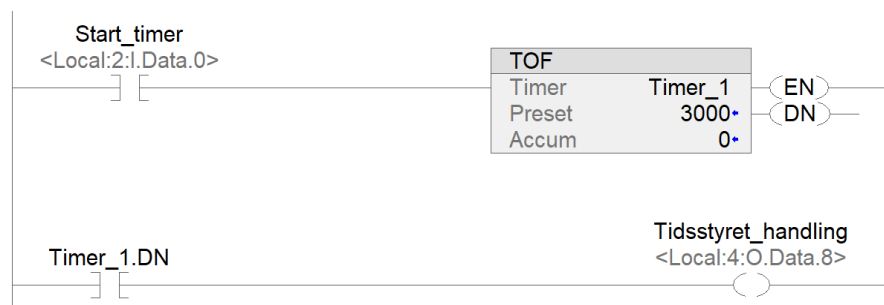
Funktionsdiagram



Når I 0.0 er aktiv trækker udgang Q 4.0 med det samme. Når signalet på I 0.0 forsvinder igen, starter timeren, efter 3 sekunder falder udgang Q 4.0 igen.

På udgangen ET er en udlæsning af hvor lang tid, der er gået siden timeren begyndte sin tidsudmåling. Tiden udlæses TIME format.

Off-delay Allen-Bradley



Funktionsbeskrivelse

Når Local:2:I.Data.0 er aktiv, trækker udgang Local:4:O.Data.8 med det samme. Når signalet på Local:2:I.Data.0 forsvinder igen, starter timeren. Efter Preset-værdien på 3 sekunder er gået, er den done .DN (færdig), og udgang Local:4:O.Data.8 falder igen.

Accum (akkumuleret tid) er en udlæsning af hvor lang tid, der er gået siden timeren begyndte sin tidsudmåling.

Timerens tag hedder "Timer_1".

On-delay timer Structured Text Siemens SCL

```

2 // ON-delay timer med tidsstyret handling
3 IEC_Timer_1_DB.TON(IN:="Start timer 1",
4
5         PT:=T#3s,
6         Q=>"Tidsstyret handling 1",
7         ET=>"Aktuel tidsværdi");

```

| | | |
|---|-------------------------|-------|
| ▼ | "IEC_Timer_1_DB" | %DB2 |
| | "Start timer 1" | %I4.0 |
| | "Tidsstyret handling 1" | %Q4.0 |
| | "Aktuel tidsværdi" | %MD2 |

Funktionsbeskrivelse

Når I 4.0 er aktiv starter timeren, efter 3 sekunder trækker udgangen Q 4.0.

På udgangen ET er en udlæsning af hvor lang tid, der er gået siden timeren begyndte sin tidsudmåling. Tiden udlæses TIME format. Ønsker man ikke at bruge denne funktion, kan linjen slettes.

Ønsker man at bruge timerens udgang flere forskellige steder i sit program, kan det gøres således.

```

8 // En anden måde at lave en tidsstyret handling på
9 IF "IEC_Timer_1_DB".Q THEN
10     "Tidsstyret handling 1.1" := 1;
11 ELSE
12     "Tidsstyret handling 1.1" := 0;
13 END_IF;

```

| | | |
|---|---------------------------|-------|
| ▼ | "IEC_Timer_1_DB" | %DB2 |
| | "IEC_Timer_1_DB".Q | |
| | "Tidsstyret handling 1.1" | %Q4.1 |
| | "Tidsstyret handling 1.1" | %Q4.1 |

Off-delay timer Structured Text Siemens SCL

```

17 // OFF-delay timer med tidsstyret handling
18 IEC_Timer_2_DB.TOF(IN:="Start timer 2",
19
20         PT:=T#3s,
21         Q=>"Tidsstyret handling 2",
22         ET=>"Aktuel tidsværdi");

```

| | | |
|---|-------------------------|-------|
| ▼ | "IEC_Timer_2_DB" | %DB3 |
| | "Start timer 2" | %I4.1 |
| | "Tidsstyret handling 2" | %Q4.2 |
| | "Aktuel tidsværdi" | %MD2 |

Funktionsbeskrivelse

Når I 4.1 er aktiv, trækker udgangen Q4.2 med det samme. Når I4.1 deaktiveres, starter timeren, og der går 3 sekunder før udgangen Q4.2 deaktiveres.

På udgangen ET er en udlæsning af hvor lang tid, der er gået, siden timeren begyndte sin tidsudmåling. Tiden udlæses TIME format.

Ønsker man ikke at bruge denne funktion, kan linjen slettes.

Ønsker man at bruge timerens udgang flere forskellige steder i sit program, se da eksemplet ovenfor.

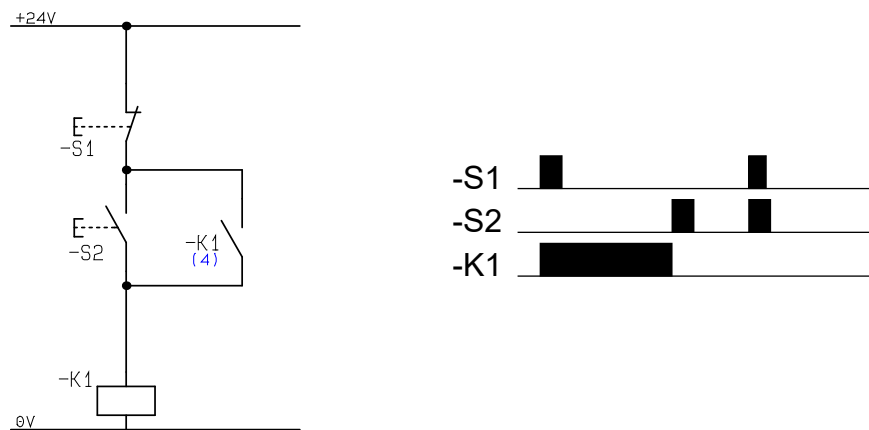
Flip-flop

Generelt

I PLC-styringer anvendes ofte forskellige type af hukommelsesfunktioner; dels som grundinstruktioner, dels som anvendte standardkoblinger. Disse funktioner betegnes som flip-flop-funktioner eller set-reset-funktioner.

Hukommelse

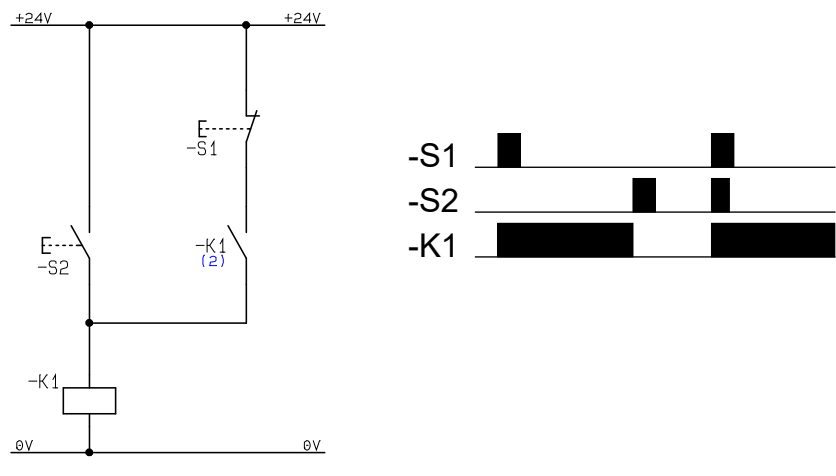
Relæteknik stop-prioritet



Diagrammet viser et start/stop-kredsløb, der er bygget op vha. relæteknik. Hvis startkontakten S2 aktiveres, trækker K1, og pga. selvholdet forbliver K1 aktiveret, indtil stopknappen S1 aktiveres. Hvis både S1 og S2 aktiveres, så kobles K1 ud. Dette betyder, at der er prioritet for stop.

Relæteknik start-prioritet

Systemet herunder har startprioritet



PLC flip-flop

Generelt

I en Siemens PLC findes der to typer af flip-flop. Den ene er boks-typen, mens den anden er bit-typen.

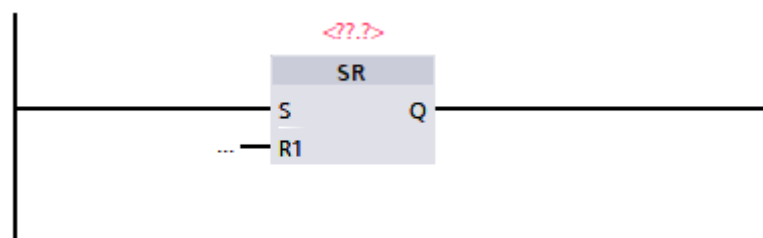
En flip-flop kan programmeres med reset- (stop) eller set- (start) prioritet.

I et Siemens-program kaldes en flip-flop med reset-prioritet for en SR flip-flop, og en med set-prioritet kaldes RS flip-flop. Prioriteten bestemmes af, hvilken rækkefølge den programmeres i. Det, som står sidst, har prioriteten.

En flip-flop benævnes som en Latch / Unlatch-funktion i en Allen-Bradley PLC. Og her er prioriteten på samme måde som i Siemens.

SR flip-flop boks-type Siemens

Symbol

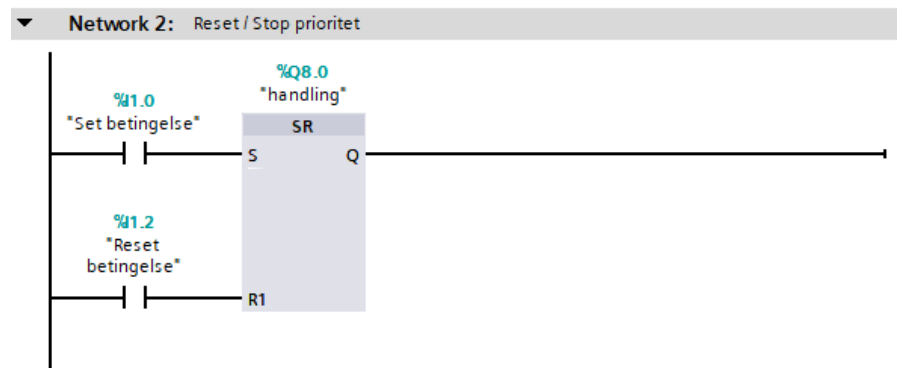


På S-indgangen angives betingelser for Set. På R1-indgangen angives betingelser for Reset.

På "<???.?>" angives adresse for flip-flop-funktionen.

På Q-udgangen angives, hvilken udgang som "address" overføres til.

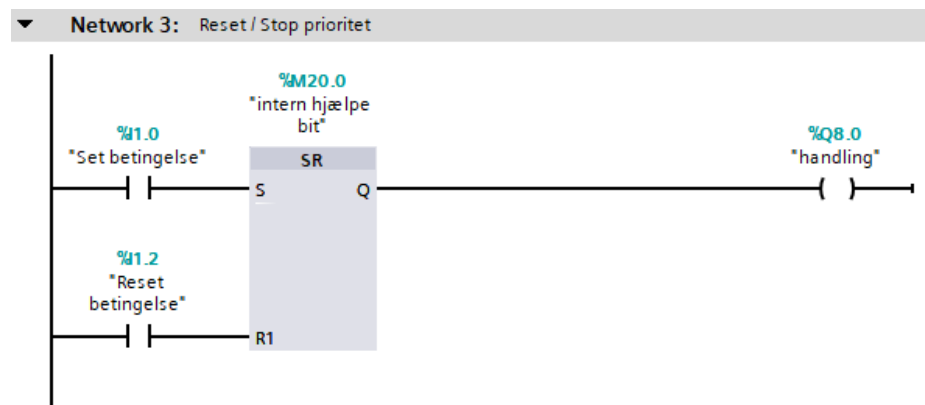
Når der er signal på S, så bliver udgangen aktiv. Udgangen forbliver aktiv, indtil der kommer signal på R1.



I det viste eksempel settes udgang Q 8.0 af indgang 1.0 og resettes igen af I 1.2.

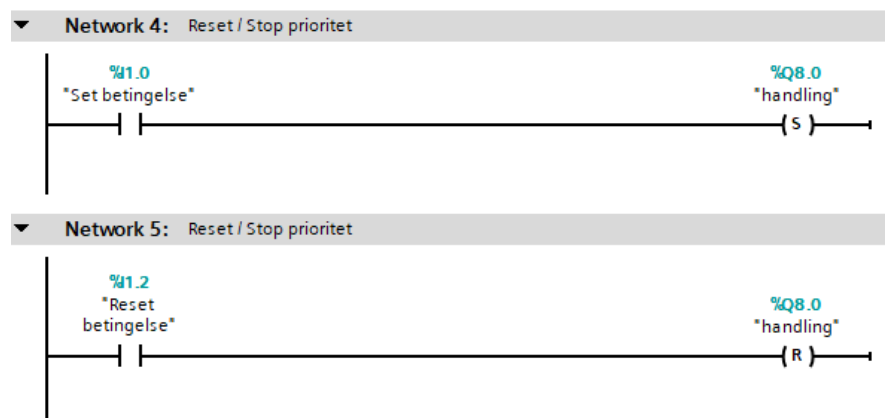
I "address" feltet er der angivet en udgangsadresse. Det er derfor ikke nødvendigt at angive en værdi på Q.

Hvis både S og R er aktive, så bliver Q 8.0 lig med logisk 0, fordi den viste flip-flop har reset-prioritet.



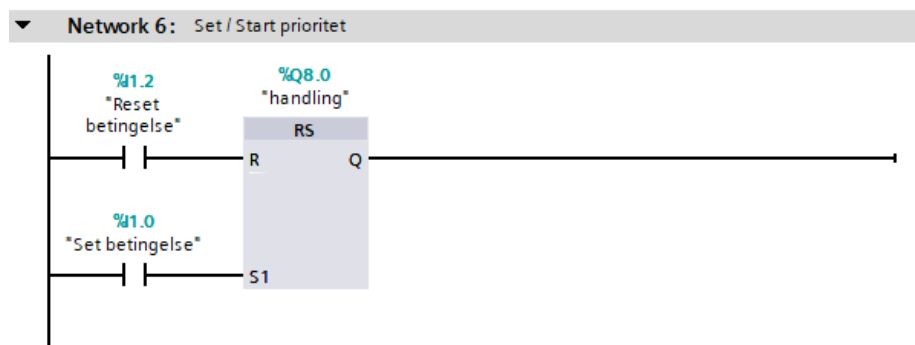
Den viste flip-flop har samme funktion som den foregående. Den eneste forskel er, at der bruges en M-adresse til at overføre status til Q 8.0.

SR flip-flop bit-type Siemens



Eksemplet viser den samme flip-flop-funktion som i det foregående. Forskellen er, at i stedet for bokstypen er der nu anvendt en bit flip-flop. Når der anvendes bit flip-flop, er det ikke et krav, at de to linjer programmeres lige efter hinanden. Dog skal reset "R" altid komme efter set "S", hvis man ønsker stop-prioritet.

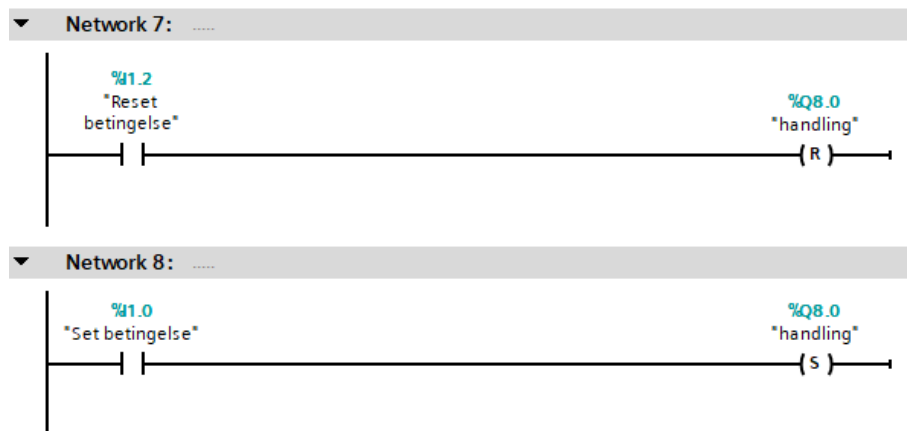
RS flip-flop boks-type Siemens



Udgang Q 8.0 settes af I 1.2 og resettes igen af I 1.0.

Hvis både R og S er logisk 1, bliver Q 8.0 lig med logisk 1. Den viste flip-flop har set-prioritet.

RS flip-flop bit-type Siemens



Eksemplet viser den samme flip-flop-funktion som i det foregående. Forskellen er, at i stedet for bokstypen er der nu anvendt en bit flip-flop. Når der anvendes bit flip-flop, er det ikke et krav, at de to linjer programmeres lige efter hinanden. Dog skal set "S" altid komme efter reset "R", hvis man ønsker start-prioritet.

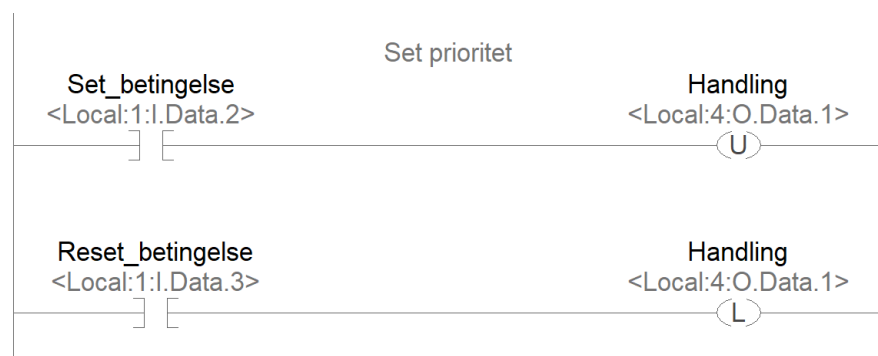
Flip-flop Allen-Bradley



Den første indgang er set-indgangen (Latch), mens den nederste er reset-indgangen (Unlatch).

Når der er signal på Local:1:I.data.2, så settes udgang Local:4:O.Data.1 og forbliver aktiv, indtil der kommer signal på indgang Local:1:I.data.3.

I en Allen-Bradley PLC kan der ændres på rækkefølgen af set og reset, hvilket medfører, at man kan vælge, om man vil have reset-prioritet som i eksemplet ovenfor eller set-prioritet som herunder.

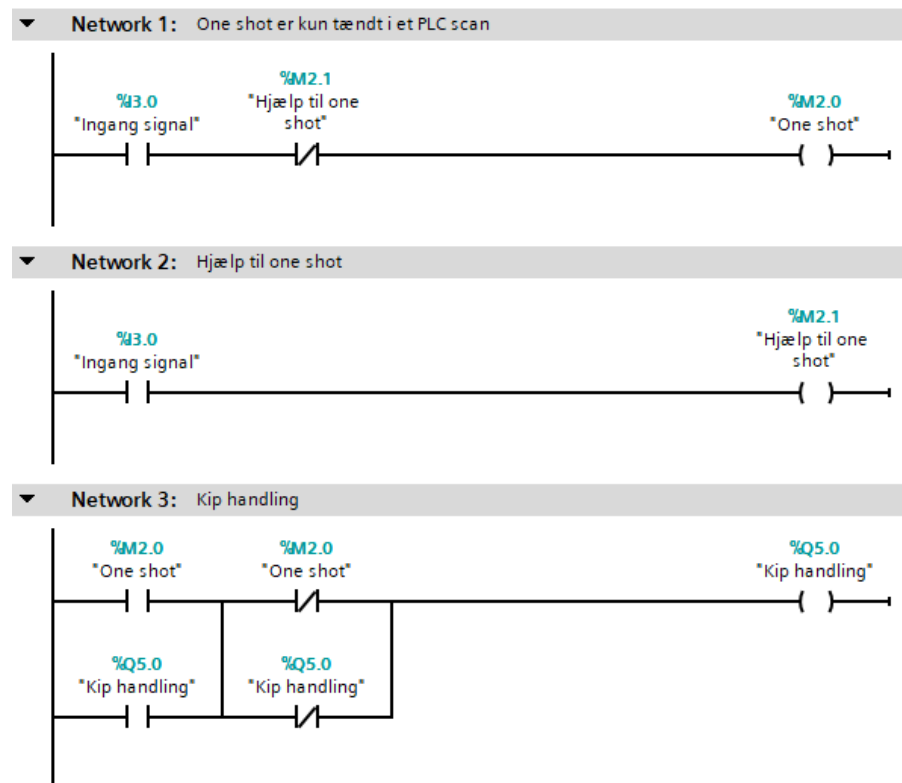


Kip-relæ/Toggle flip-flop

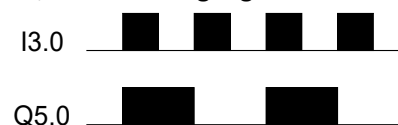
Generelt

I forbindelse med PLC-styringer har man ofte brug for en kip-relæfunktion, f.eks. til vekseldrift/omskiftning mellem to motorer. En kip-relæfunktion benævnes også som en Toggle flip-flop.

Programeksempel



Hver gang I 3.0 aktiveres, så skifter udgang Q5.0.



- Netværk 1 indeholder en one-shot, hvilket betyder, at hvis I 3.0 er aktiv, så kommer der kun en puls i et PLC-scan på M 2.1.
- Netværk 2 indeholder et hjælpe-bit M2.1, der er nødvendig for at lave one-shot-funktionen
- Netværk 3 indeholder selve Toggle flip-flop-funktionen. Hver gang I 0.0 aktiveres, så skifter Q 8.1 til modsat stilling.

OBS. En Toggle eller kip-funktion kan programmeres på mange andre måder.

Tæller

Generelt

De fleste PLC-systemer er i dag forsynet med en række tællere. Tælleren anvendes til at tælle impulser, f.eks. fra et indgangskort.

Siemens Tæller

I Siemens S7 PLC'er findes der en optæller (CTU) og en nedtæller (CTD), og en tæller som både kan tælle op og ned (CTUD).

Konstanten for en tæller-værdi PV angives med tal, f.eks. 5 i integer "INT"-format, og tællerområdet er fra -32768 til 32767.

Tæller er tilknyttet en datablok med følgende struktur:

Parameters

The following table shows the parameters of the "Count up" instruction:

| Parameter | Declaration | Data type | Memory area | | Description |
|-----------|-------------|-----------------------------|------------------------------|------------------------------------|-------------------------------------|
| | | | S7-1200 | S7-1500 | |
| CU | Input | BOOL | I, Q, M, D, L or constant | I, Q, M, D, L or constant | Count input |
| R | Input | BOOL | I, Q, M, D, L, P or constant | I, Q, M, T, C, D, L, P or constant | Reset input |
| PV | Input | Integers | I, Q, M, D, L, P or constant | I, Q, M, D, L, P or constant | Value at which the output Q is set. |
| Q | Output | BOOL | I, Q, M, D, L | I, Q, M, D, L | Counter status |
| CV | Output | Integers, CHAR, WCHAR, DATE | I, Q, M, D, L, P | I, Q, M, D, L, P | Current counter value |

Allen-Bradley Tæller

I Allen-Bradley PLC'er findes der en optæller (CTU) og en nedtæller (CTD).

En tæller, som både kan tælle op og ned, kan også laves i Allen-Bradley ved, at lade en optæller og en nedtæller deles om det samme tag.

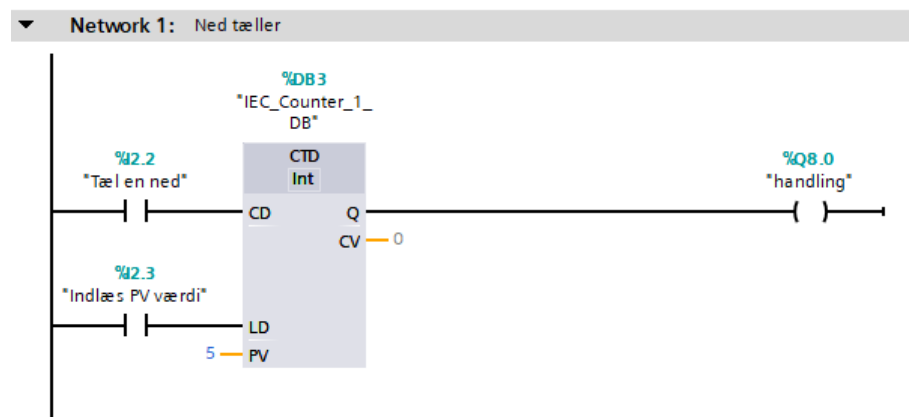
Konstanten for en tæller preset-værdi .PRE angives med tal, f.eks. 5 i dobbelt integer "DINT"-format, og tællerområdet er fra -2.147.483.648 til 2.147.483.647.

Tæller skal tilknyttes et tag, der har følgende datastruktur:

COUNTER Structure

| Mnemonic | Data Type | Description |
|----------|-----------|--|
| .CU | BOOL | The count up enable contains rung-condition-in when the instruction was last executed. |
| .DN | BOOL | The done bit when set indicates the counting operation is complete. |
| .OV | BOOL | The overflow bit when set indicates the counter incremented past the upper limit of 2,147,483,647. |
| .UN | BOOL | The underflow when set indicates the counter decremented past the lower limit of -2,147,483,648. |
| .PRE | DINT | The preset value specifies the value which the accumulated value must reach before the instruction indicates it is done. |
| .ACC | DINT | The accumulated value specifies the number of transitions the instruction has counted. |
| .CD | BOOL | The countdown enable bit contains rung-condition-in when the instruction was last executed. |

Siemens Nedtæller Boks-tæller



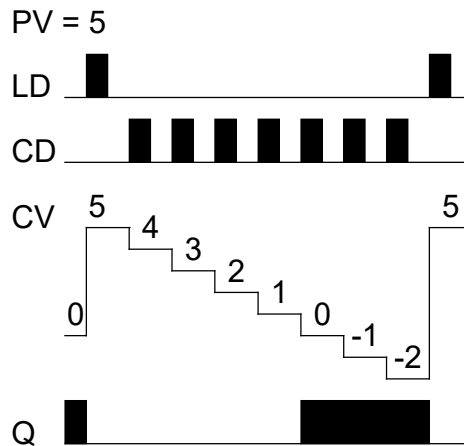
Funktionsbeskrivelse

Når I 2.3 aktiverer indgangen LD "load data" settes tælleren til den værdi, der er angivet på PV, altså 5. Når I 2.2 aktiveres, tælles tællerværdien ned med én, det vil sige fra 5 til 4. Hver gang I 2.2 aktiveres, tælles en ned.

Udgangen Q er logisk 1, når tællerens værdi er lig med eller mindre end 0. Dette betyder, at Q 8.0 tænder, når tællerens værdi når ned til 0.

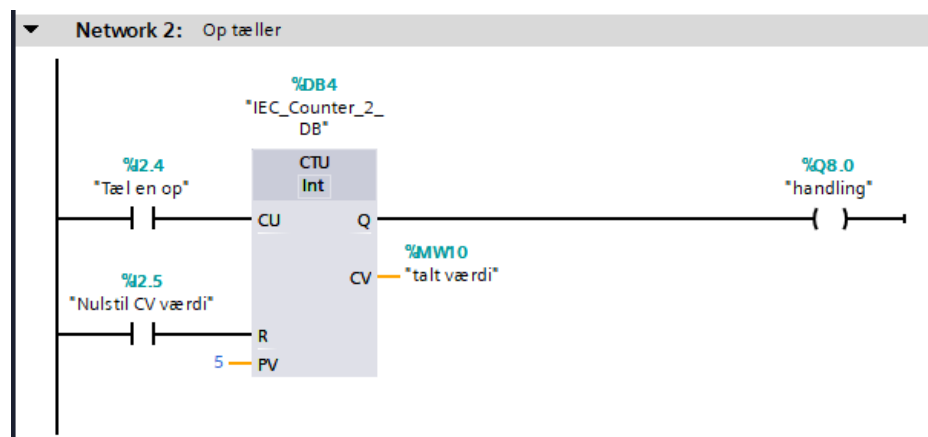
Udgangene CV er statusudgange, hvor man i "INT"-format kan udlæse, hvor langt ned tælleren har talt. Denne funktion finder kun sted, hvis der er tilknyttet et dataområde til dette ben. Som her vist, virker tælleren fint uden tilknytning.

Funktionsdiagram



Siemens Optæller

Boks-tæller



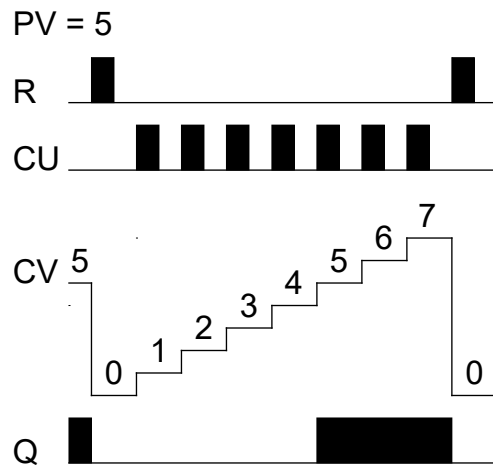
Funktionsbeskrivelse

Når indgang I 2.5 aktiverer R, resettes tælleren til værdien 0. Når I 2.4 aktiverer CU, tælles tællerværdien op med en. Hver gang I 2.4 aktiveres, tælles en op.

Udgangen Q skifter til logisk 1, når tællerværdien CV er lig med PV eller større end. Dette betyder, at Q 8.0 er tænder, når tællerens værdi når op til 5.

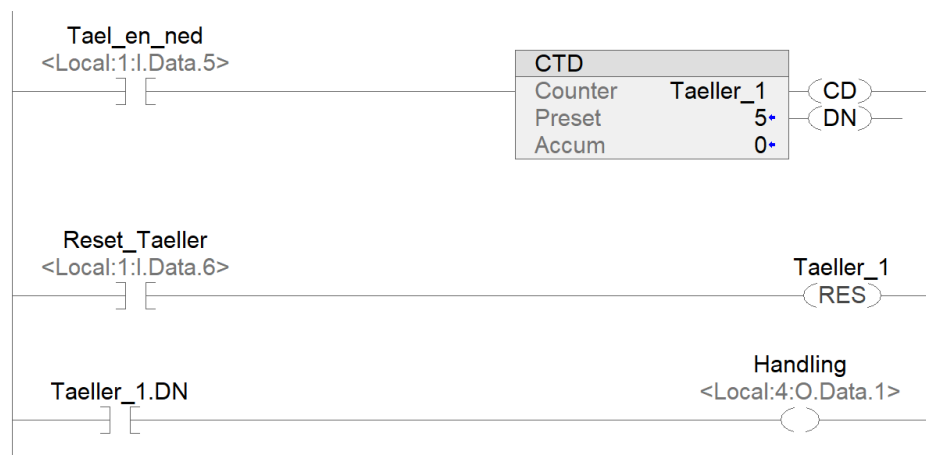
Udgangene CV er statusudgange, hvor man i "INT"-format kan udlæse, hvor langt op tælleren har talt. Denne funktion finder kun sted, hvis der som her er tilknyttet et dataområde til dette ben. Tælleren virker også fint uden tilknytning.

Funktionsdiagram



Allen-Bradley Nedtæller

Boks-tæller



Funktionsbeskrivelse

Når indgang <Local:1:I.Data.5> "Tael_en_ned" aktiveres, tælles tællerværdien Accum ned med en. Det vil sige fra preset = 5 til 4. Hver gang <Local:1:I.Data.5> "Tael_en_ned" aktiveres, tælles en ned.

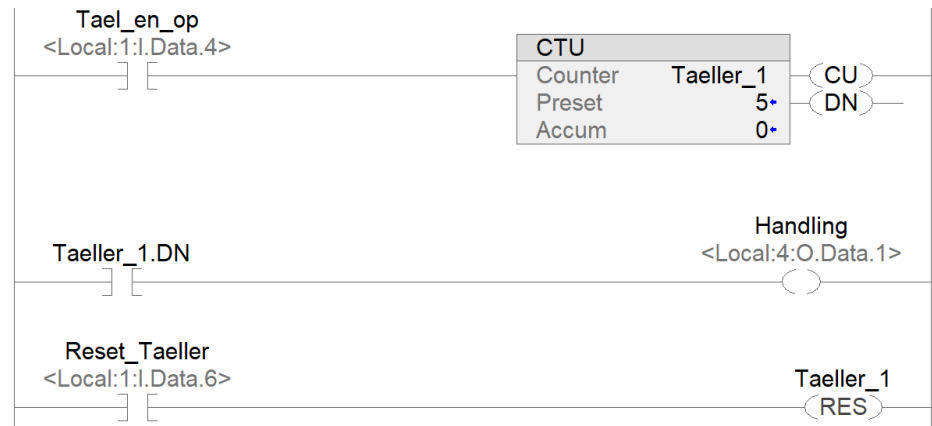
Udgangs-bittet "Taeller_1.DN" i taget "Taeller_1" er logisk 1, når tællerens værdi er lig med eller mindre end 0.

Dette betyder, at <Local:4:O.Data.1> "Handling" tænder, når tællerens værdi når ned til 0.

Når <Local:1:I.Data.6> "Reset_Taeller" tændes, bliver Accum = Preset. Det vil sige 5, og tælleren er nu klar til at tælle forfra.

Allen-Bradley Optæller

Boks-tæller



Funktionsbeskrivelse

Når indgang <Local:1:I.Data.4> "Tael_en_op" aktiveres, tælles tællerværdien Accum op med en. Det vil sige fra 0 til 1. Hver gang <Local:1:I.Data.4> "Tael_en_op" aktiveres, tælles en op.

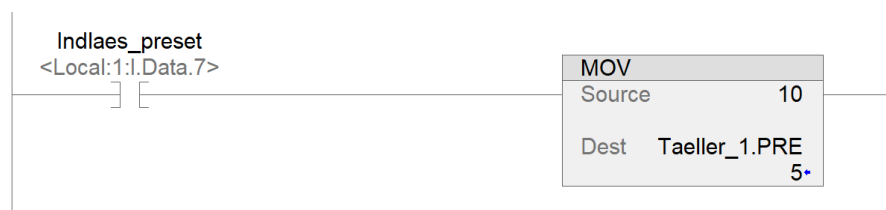
Udgangs-bittet "Taeller_1.DN" i tagget "Taelle_1" er logisk 1, når tællerens værdi er lig med eller større end Preset = 5.

Dette betyder, at <Local:4:O.Data.1> "Handling" tænder, når tællerens værdi når op til 5.

Når <Local:1:I.Data.6> "Reset_Taeller" tændes, bliver Accum = 0, og tælleren er nu klar til at tælle forfra.

Behandling af tællerværdier

Ønsker man at indlæse en ny preset værdi til en tæller, kan det gøres på følgende måde. Når <Local:1:I.Data.7> "indlaes_preset" aktiveres, vil move-instruktionen overskrive den gamle preset-værdi på 5 med 10.



Ønsker man, at udlæse den værdi tælleren har talt til, kan det gøres på følgende måde.



OBS. Disse metoder kan også bruges på timere.

Optæller Structured Text Siemens SCL

```

23 // Count Up tæller styret handling
24 □ "IEC_Counter_1_DB".CTU(CU:="Tæl en op",
25
26           R:="Nulstil CV værdi",
27           PV:=5,
28           Q=>"Tæller handling 1",
29           CV=>"talt værdi");

```

| | | |
|---|---------------------|-------|
| ▼ | "IEC_Counter_1_DB" | %DB4 |
| | "Tæl en op" | %I5.4 |
| | "Nulstil CV værdi" | %I5.5 |
| | | |
| | "Tæller handling 1" | %Q4.3 |
| | "talt værdi" | %MW10 |

Funktionsbeskrivelse

Når indgang I 5.5 aktiverer R, resettes tælleren til værdien 0. Når I 5.4 aktiverer CU, tælles tællerværdien op med en. Hver gang I 5.4 aktiveres, tælles en op.

Udgangen Q skifter til logisk 1, når tællerværdien CV er lig med PV eller større end. Dette betyder, at Q 4.3 er tænder, når tællerens værdi når op til 5.

Udgangene CV er statusudgange, hvor man i "INT"-format kan udlæse, hvor langt optælleren har talt. Ønsker man ikke denne funktion, slettes linjen.

Ønsker man at bruge tællerudgang flere forskellige steder i sit program, kan det gøres således.

```

31 // En anden måde at en lave tæller styret handling på
32 □ IF "IEC_Counter_1_DB".QU THEN
33     "Tæller handling 1.1" := 1;
34     // Statement section IF
35 ELSE
36     "Tæller handling 1.1" := 0;
37 END_IF;

```

| | | |
|---|-----------------------|-------|
| ▼ | "IEC_Counter_1_DB" | %DB4 |
| | "IEC_Counter_1_DB".QU | |
| | "Tæller handling 1.1" | %Q4.4 |
| | | |
| | "Tæller handling 1.1" | %Q4.4 |

Nedtæller Structured Text Siemens SCL

```

39 // count down med tæller styret handling
40 □ "IEC_Counter_2_DB".CTD(CD := "Tæl en ned",
41
42           LD := "Indlæs PV værdi",
43           PV := 5,
44           Q => "Tæller handling 2",
45           CV => "talt værdi 2");

```

| | | |
|---|---------------------|-------|
| ▼ | "IEC_Counter_2_DB" | %DB5 |
| | "Tæl en ned" | %I5.2 |
| | "Indlæs PV værdi" | %I5.3 |
| | | |
| | "Tæller handling 2" | %Q4.5 |
| | "talt værdi 2" | %MW0 |

Funktionsbeskrivelse

Når I 5.3 aktiverer indgangen LD load data, sættes tælleren til den værdi, der er angivet på PV altså 5. Når I 5.2 aktiveres tælles tællerværdien ned med en. Det vil sige fra 5 til 4. Hver gang I 5.2 aktiveres, tælles en ned. Udgangen Q er logisk 1, når tællerens værdi er lig med eller mindre end 0. Dette betyder, at Q 4.5 er tænder, når tællerens værdi når ned til 0.

Udgangene CV er statusudgange, hvor man i "INT"-format kan udlæse, hvor langt nedtælleren har talt. Ønsker man ikke denne funktion, slettes linjen.

Følere

Beskrivelse

Følere er et væsentligt led mellem mekaniske og elektriske styresystemer.

Følernes opgave er at registrere en mekanisk bevægelse, og omsætte den til et elektrisk signal, som derved erstatter den menneskelige overvågning ved maskinen eller i processen.

De følere, som gennemgås i dette afsnit, omhandler kun følere, som ikke berøres ved aktivering. Disse følere kaldes også nærhedsfølere, berøringsløsefølere eller på engelsk sensorer.

Til dette formål findes der forskellige typer af følere, bl.a.:

- Induktive følere
- Fotoaftastere
- Kapacitive følere
- Magnetiske (reed-kontakt)
- Specialfølere

Forskellen ligger i, hvordan den enkelte føler skal aftastes for at aktivere.

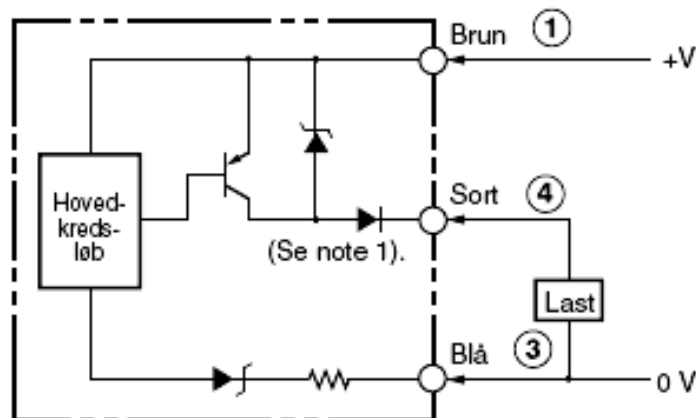


Billedet viser en række følere fra Omron. De fleste følere, som anvendes i dag, har indbygget elektronik i føleren. Deres montage og udgangssignaler er ens, derfor behandles de fælles i det efterfølgende afsnit.

Udgangssignal og montage

Udgangssignalet fra følere deles op i to forskellige typer, henholdsvis en PNP- og en NPN-type. Udgangssignalet er henholdsvis positiv og negativ fra de to udgangstyper.

PNP-udgang



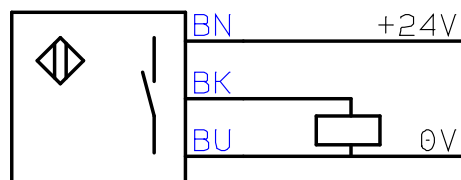
Diagrammet viser den elektriske tilslutning af en PNP-føler. Forsyningsspændingen tilsluttes med plus på den brune ledning, mens 0 V tilsluttes den blå.

Hvis forsyningsspændingen er 24 V, giver føleren + 24 V ud, når den aktiveres.

Farvekodningen på ledninger overholder en standard, hvor der anvendes følgende kode:

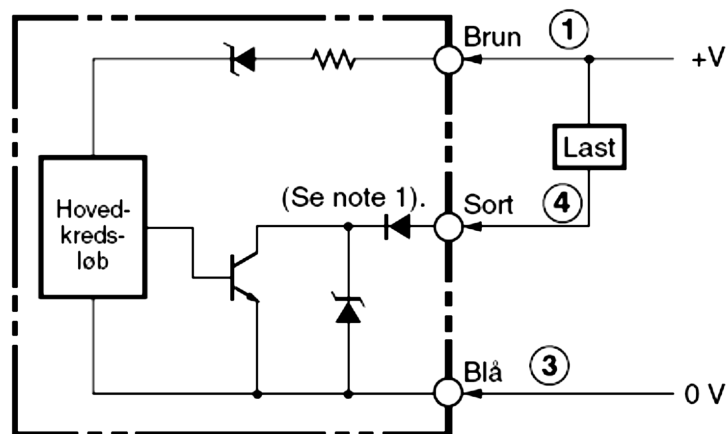
- Brun, positiv forsyningsspænding (**BrowN**)
- Blå, 0 V forsyningsspænding (**BlUe**)
- Sort, signalledning (**BlacK**)

Symbol for PNP-udgang



Diagrammet viser symboler for en føler med PNP-udgang. Kontakten er en NO-kontakt, hvilket betyder, at føleren giver signal ud, når den aktiveres.

NPN-udgang



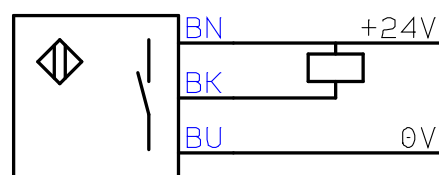
Diagrammet viser den elektriske tilslutning af en NPN-føler. Forsyningsspændingen tilsluttes med plus på den brune ledning, mens 0 V tilsluttes den blå.

Lasten tilsluttes mellem den positive forsyningsspænding og signalledningen. Når føleren aktiveres, kobles 0 V til signalledningen, hvilket medfører, at "lasten" får spænding.

Farvekodningen på ledninger overholder en standard, hvor der anvendes følgende kode:

- Brun, positiv forsyningsspænding
- Blå, 0 V forsyningsspænding
- Sort, signalledning

Symbol for NPN-udgang



Diagrammet viser symboler for en føler med NPN-udgang. Kontakten er en NO-kontakt, hvilket betyder, at føleren giver signal ud, når den aktiveres.

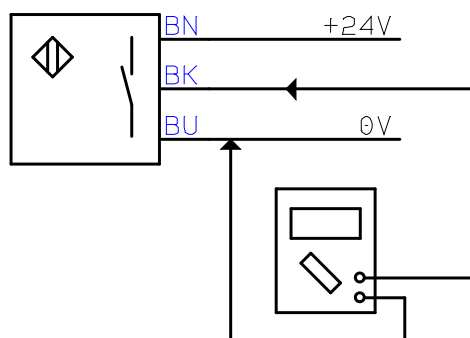
Typiske data for føler med transistorudgang

De følgende data er hentet fra en Omron-induktivføler. Disse data er stort set fælles for alle følere af samme fabrikat.

| | |
|------------------------|---|
| Spændingsforsyning: | 12 til 24 VDC maks. 10 % ripple |
| Driftsspændingsområde: | 10 til 32 VDC |
| Strømforbrug: | 10 mA |
| Belastning: | 200 mA |
| Spændingsfald: | maks. 2V |
| Omgivelsestemperatur: | -40 °C til +70 °C |
| Beskyttelsesgrad: | IP 67 |
| Beskyttelse mod: | Omvendt polaritet Høje transienter Kortslutning |

Husk altid at kontrollere datablad for de eksakte data for den enkelte føler.

Kontrol af følere



Den viste PNP-føler skal kontrolleres. Monter et voltmeter mellem den blå (0 V) og den sorte (signal) ledning. Når føleren aktiveres, skal den give et signal på 24 V DC, mens den uaktiveret skal give et signal på 0 V.



Billedet viser et testinstrument fra firmaet Sick. Med dette instrument er det muligt at teste både NPN- og PNP-følere.

Induktive nærhedsføler

Evne til at kunne detektere metalobjekter er uundværlig i en automationsproces. Induktive nærhedssensorer kommer i mange forskellige udformninger. Det er en vigtig komponent i en automationsproces, hvor der bl.a. stilles krav til hygiejne, høj temperatur og aggressive medier.

Overordnet typer:

- Industrielle sensorer
- Fuld-metal sensorer
- ATEX-sensorer
- Analoge sensorer
- Slot-sensorer
- Konstant rækkevidde-sensorer
- Fail-safe-sensorer
- Ring- og rørsensorer



Figur 1 - Induktiv Sensorer: Ifm's hjemmeside.

Hoveddele i en industriel induktive nærhedssensorer.
På billede nedenfor kan ses en "klassisk" induktiv nærhedssensor.

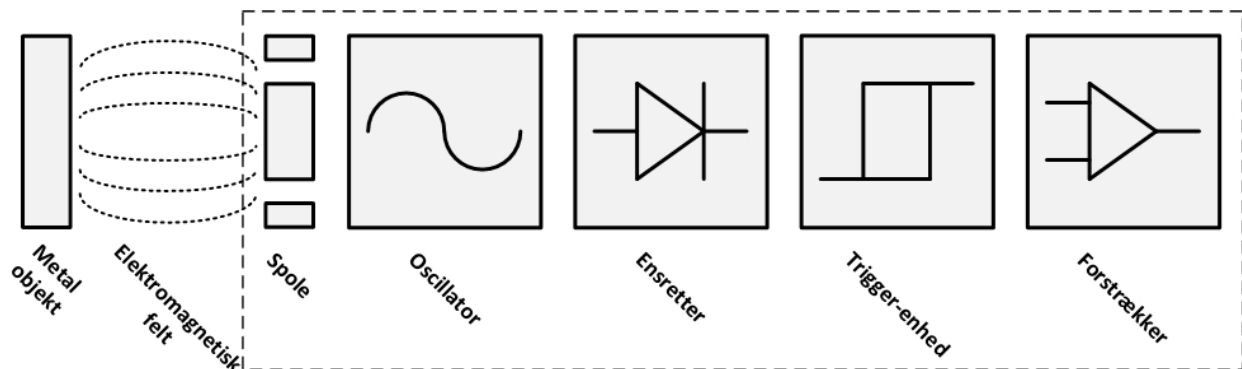


Figur 2 - IGS232 Induktiv Nærhedssensor: Ifm's hjemmeside.

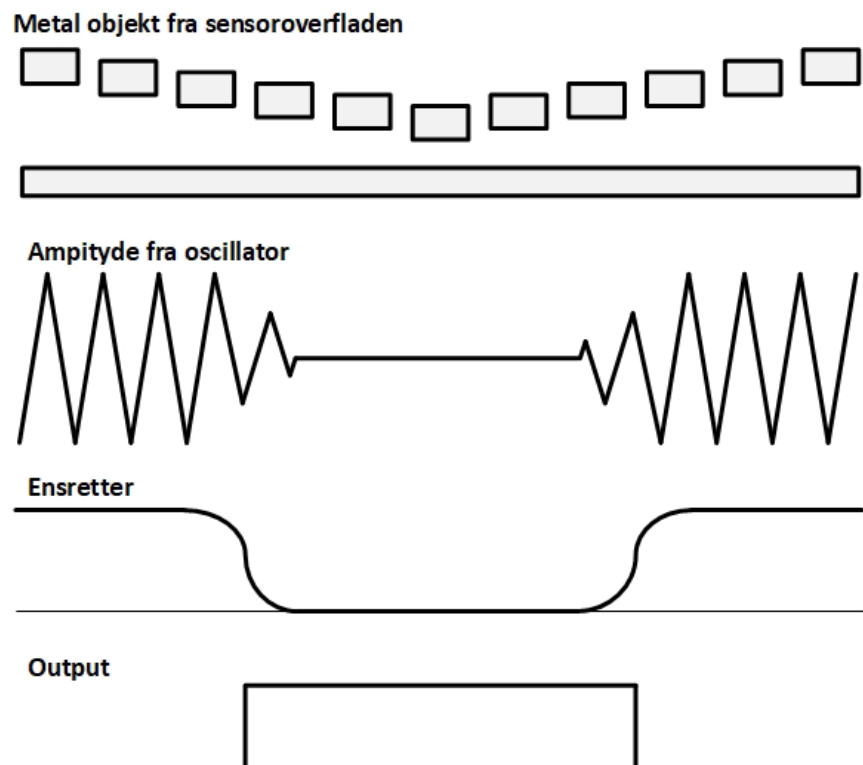
Virkemåde og opbygning

En induktiv nærhedssensor har grundlæggende fem hoveddele:

- En spole, der danner et vekslende elektromagnetisk felt
- En oscillator, der for det meste er et LC-kredsløb
- En ensretter, der forsyner oscillatoren
- En trigger-enhed, som ved hjælp af en hysteresis giver et klart defineret skifte mellem aktiveret og uaktiveret, når et metalobjekt kommer indenfor rækkevidde
- En forstærkning til at kunne drive en last



Figur 3 - Induktiv Nærhedssensors Hoveddel: Forfatterens Egen Illustration.



Figur 4 – Funktionsdiagram af Induktiv Nærhedssensors: Forfatterens Egen Illustration.

Når et metalobjekt kommer ind i sensorens vekslende elektromagnetiske felt (EMF), dannes der en hvirvelstrøm i det strømførende metalobjekt (Eddy current). Det betyder, at der også opstår et magnetfelt fra den strøm, der løber i metalobjektet.

Magnetfeltet, der bliver skabt fra Eddy current, modvirker sensorens vekslende EMF.

Sensorens vekslende EMF bliver svækket, når metalobjektet kommer tættere på sensoren. Tilsvarende bliver Eddy current-magnetfeltet strakt, når metalobjektet nærmer sig sensoren.

Amplituden fra strømme, der løber gennem spolen, er proportional med den EMF, der bliver dannet gennem spolen. Det medfører, at strømmen bliver mindre, når sensoren bliver påvirket.

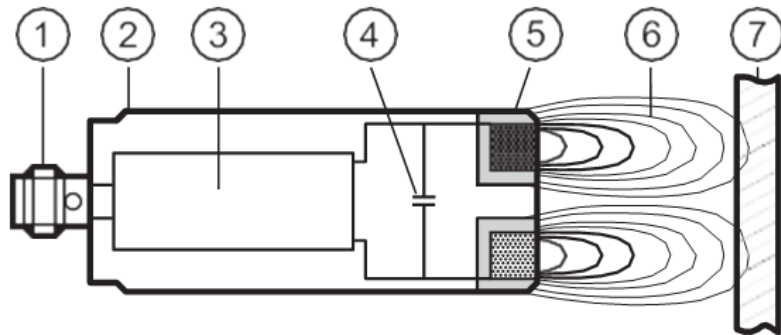
Når strømmen bliver en prædefineret størrelse, bliver sensorens udgang sat.

Fjernes metalobjektet fra sensorens rækkevidde, tiltager amplituden og strøm tilsvarende.

Tasteafstande på metaltyper

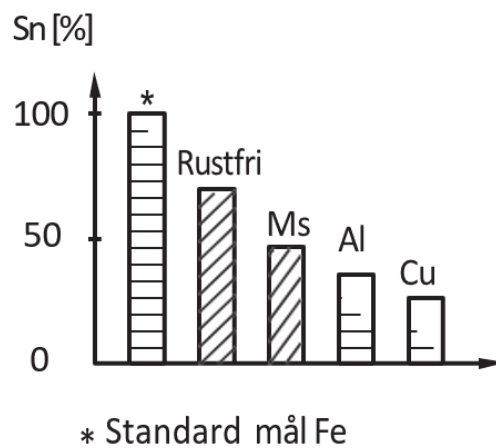
Spole og kondensator danner en LC-oscillatorkreds, også kaldet en basissensor.

Hvirvelstrømstab i elektrisk ledende materialer bruges til et koblingssignal.



- ① Stik-tilkobling
- ② Hus
- ③ Hjelpeelektronik
- ④ Kondensator
- ⑤ Spole
- ⑥ Vekslede elektromagnetisk felt = aftastningszone
- ⑦ Aftastningsemne = elektrisk ledende metal

Ved installation af induktive nærhedsføler skal der tages højde for tasteafstand S_n , da denne ændrer sig afhængigt af, hvilket metalmateriale føleren skal taste på.



Figur 5 - Uddrag af IGS232 Datablad: Ifm's hjemmeside.

Nedenfor er et uddrag af en IGS232 Induktiv Nærhedssensor fra ifm. Her kan man aflæse korrektionsfaktoren på messingobjektet til 0,5. Når den nominelle tæstefstand S_n er 8[mm] på jern/stål, vil den på messing være:

$$S_{n-messing} = S_{n-stål} \times K_{messing} = 8[mm] \times 0,5 = 4[mm]$$

Det vil sige en halvering af S_n .

| Nøjagtighed / afvigelser | |
|--|--|
| Korrektionsfaktor | Stål: 1 / Rustfrit stål: 0,7 / Messing: 0,5 / Aluminium: 0,4 / kobber: 0,3 |
| Hysteresis [% af S_n] | 3...15 |
| Tolerance, koblingspunkt [% af S_n] | -10...10 |

Figur 6 - Uddrag af IGS232 Datablad: ifm's hjemmeside.

En induktiv følers tæstefstand vil stige med føleren størrelse, da en større føler har plads til en større tæsteflade, som så også kan indeholde en større spole.

Repeterbarhed

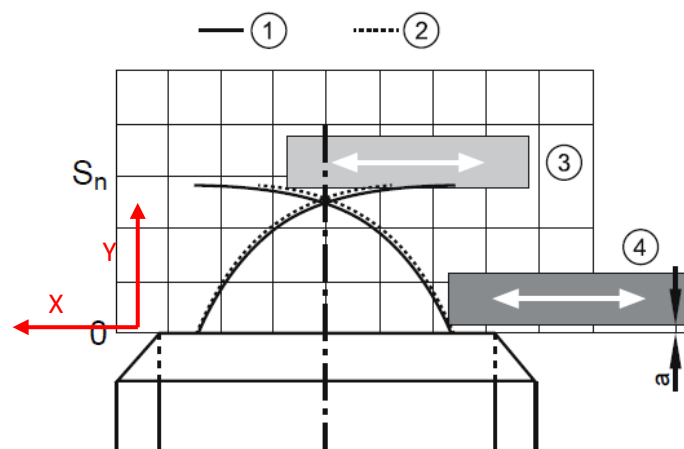
Det vekslende elektromagnetisk felt, der udstråler fra sensoren, giver et resulterende aftastningsområde, som er vist nedenunder.

Sidefølsomheden i X-tasteretning betyder, at S_n mindskes i forhold til Y-tasteretning. Diagram over tæstefstand ved et standard metalobjekt, forefindes normalt i databladet på sensoren.

- ① Typisk indkoblingskurve (for langsom tilnærmelse)
- ② Typisk udkoblingskurve (for langsom tilnærmelse)
- ③ Dårlig repeterbarhed
- ④ God repeterbarhed

For god repeterbarhed af koblingspunktet: Jo nærmere emnet er placeret på følefladen, jo bedre.

Generelle anbefaling: $a = 10\%$ af nominel følerækkevidde.

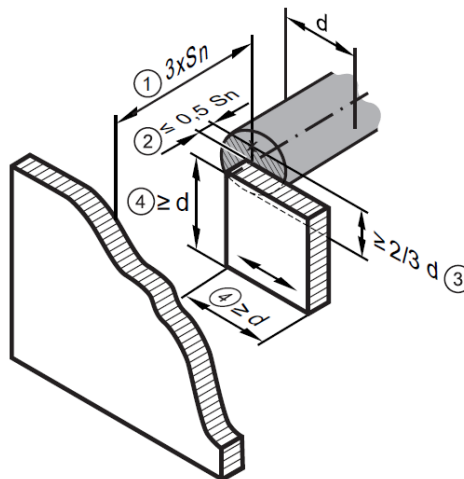


Figur 7 - Uddrag af Info card Inductive Sensors: ifm's hjemmeside.

Installationsanbefalinger

I info-kortet på denne sensor finder vi også de anbefalinger, producenten har til installationen.

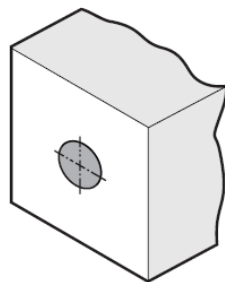
- ① Afstand til baggrunden
- ② Anbefalet tastestand
- ③ Anbefalet grad af dækning af følefladen
- ④ Anbefalet emnestørrelse



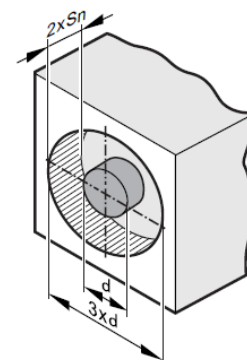
Figur 8 - Uddrag af Info card Inductive Sensors: Ifm's hjemmeside

Induktive følere, der er cylinderformet, fås i to typer. En skærmet, som ikke er sidefølsom, men som typisk har kortere tasteafstand og en uskærmet, som er sidefølsom, men har længere tasteafstand.

Skærmet føler



Uskærmet føler

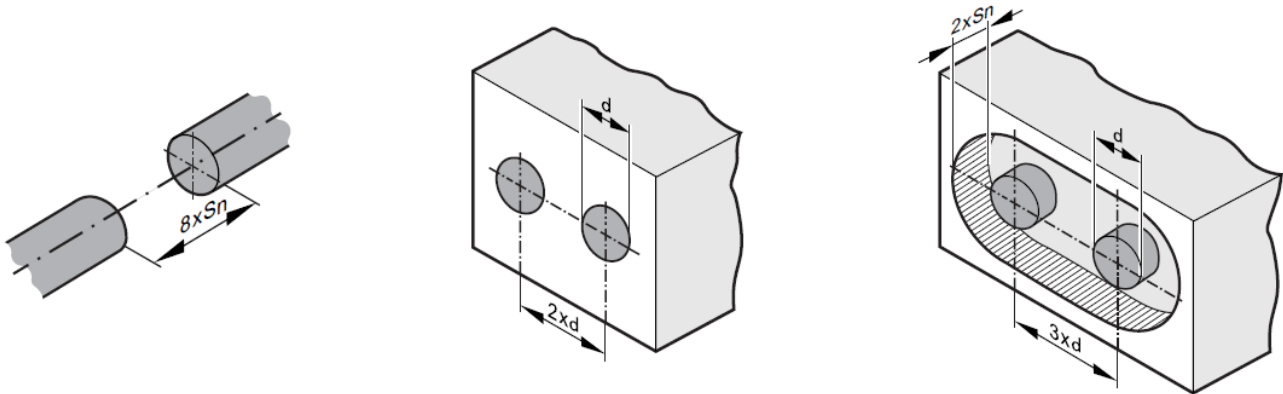


Figur 9 - Uddrag af Info card Inductive Sensors: Ifm's hjemmeside.

Når en cylinderformet induktiv føler skal monteres plant i et metalmateriale, skal det udføres med respektafstande, hvis den er uskærmet, da føleren ellers vil taste på det metal, man monterer den i.

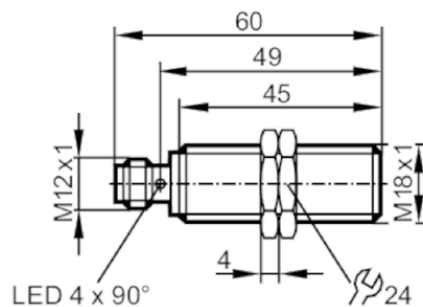
Er den skærmet, kan den uden problemer planforsænkes i et metalmateriale.

I situationer, hvor man monterer flere induktive følere ved siden af hinanden, skal der også være respektafstande mellem disse. Kravene til disse afstande er vist på figur 9 nedenunder.



Figur 10 - Uddrag af Info card Inductive Sensors: Ifm's hjemmeside.

På dimensionstegningen over sensoren vil man kunne aflæse de fysiske størrelser og møtrikker.



Figur 11 - Uddrag af IGS232 Datablad: IBM's hjemmeside.

Fortrådning

I databladet for sensoren kan man aflæse driftsspændingen for sensoren. Man skal være opmærksom på, at det kan være VAC, VDC eller begge to.

| Elektriske Data | |
|-----------------------|------------|
| Driftsspænding [V] | 10...30 DC |
| Strømforbrug [mA] | < 10 |
| Kapslingsklasse | III |
| Polaritetsbeskyttelse | ja |

Figur 12 - Uddrag af IGS232 Datablad: Ifm's hjemmeside.

Nedenunder er vist fra venstre mod højre 2, 3 og 4 tråds-forbindelser.

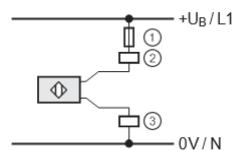
① Brug en miniaturesikring i henhold til det tekniske databladet hvis specificeret. Anbefaling: Kontroller, at enheden fungerer sikkert efter en kortslutning.

② Negativ kobling NPN

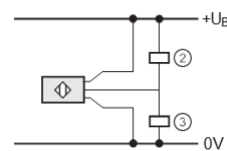
③ Positiv negativ PNP

④ Sensor 1

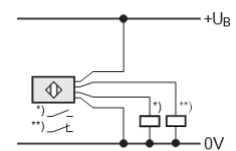
⑤ Sensor n



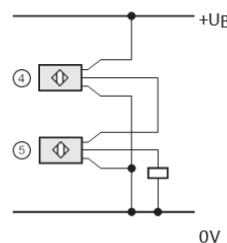
To-leders teknologi
(negativ **eller** positiv kobling)



3-leders teknologi
(negativ **eller** positiv kobling)

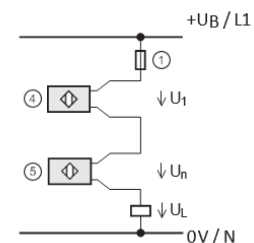


4-leders teknologi
(positiv kobling, normalt lukket og normalt åben)



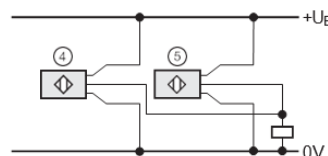
Seriekobling af 3-leder føler

Maks. 4 føler. Indkoblingsforsinkelsestider, spændingsfald og strømforbrug summerer sig. $U_{B \min}$ (sensor) og $U_{HIGH \min}$ (belastning) må forblive uændret.



Seriekobling af 2-leders føler

Ikke anbefalet på grund af udefineret drift når blokker! Brug specielle typer som kan kobles i serie (maks. 2 enheder). Spændingsfald summerer sig.



Parallelkobling 3-leders føler

Strømforbruget til alle tilkoblede føler lægges sammen. føleren kan bruges i kombination med mekaniske brydere.

Parallelkobling 2-leders føler

Ikke mulig.


Serie og parallelkobling af føler bør kun bruges, hvor der ikke er PLC-indgange til rådighed.

Som standard er de 4 ledere i et M12-stik fordelt således:

| | |
|---------------|---|
| Ben 1 / Brun: | Driftspænding |
| Ben 3 / Blå: | 0V |
| Ben4 / Sort: | Primær signal NO |
| Ben2 / Hvid: | Sekundært signal NC eller evt. teach-funktion |

Colours: BK: black, BN brown, BU: blue, WH: white

Standard configuration for 3-wire DC:

| | | Cable | Terminal chamber | US-100 plug |
|--------|---|-------|------------------|--------------------------|
| L+ | | BN | 1 / 3 | Pin 1 / BN |
| L- | | BU | 2 / 4 | Pin 3 / BU |
| Output |  | BK | X | Pin 2 / WH Pin 4 / BK |

Pin 4: BK  Pin 3: BU
Pin 1: BN Pin 2: WH

For the cable and the pin configuration as well as the unit data of special versions please refer to the wiring diagrams in our main catalogue for position sensors.

Figur 13 - Uddrag af Info card Inductive Sensors: Ifm's hjemmeside.

Fotoaftastere

Generelt

Fotoaftastere er benævnt med mange forskellige navne. De benævnes f.eks som: Fotoceller, optiske følere, optiske aftastere, fotofølere og fotoaftastere.

I dette kapitel er der en gennemgang af nogle enkelte typer.

For alle fotoaftastere gælder det samme princip: Der udsendes lys som på en eller anden måde brydes/reflekteres og derved fås en aftastning.

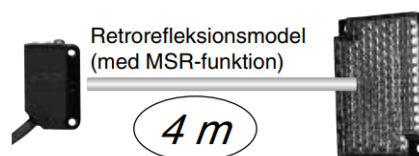
Man anvender infrarødt moduleret lys, som har den fordel, at det ikke påvirkes af det omgivende lys. At lyset er moduleret betyder, at det udsendes med en bestemt frekvens, der er afstemt med modtageren.

Aftastningstyper

Det efterfølgende er en beskrivelse af de forskellige muligheder for aftastning ved hjælp af en fotoaftaster.



Den ovenfor viste fotoaftaster består af en sender og en modtager. Der udsendes lys fra sender, som modtages på receiver. Hvis lysstrålen brydes, sker der en aftastning, og der gives signal på følerens udgang. De 30 meters tasteafstand gælder specielt for den viste type, som er en Omron E3Z.




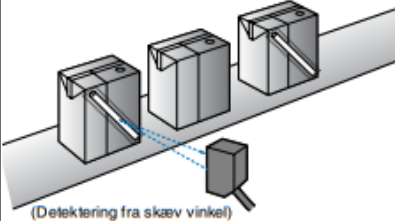
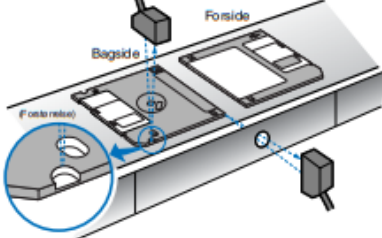
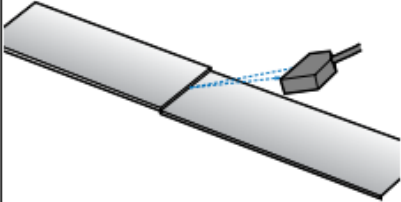
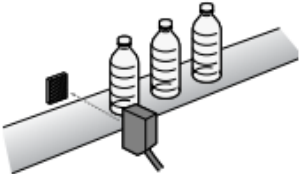
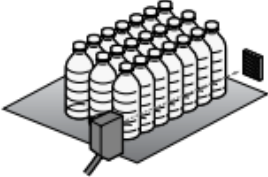
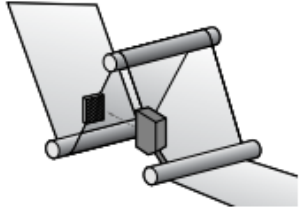
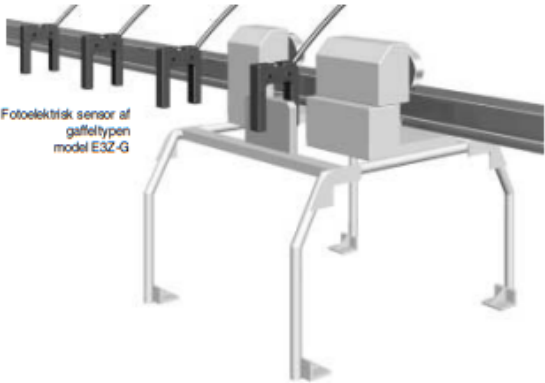


Den ovenfor viste føler har sender og modtager indbygget i samme hus. Føleren benævnes normalt refleksionsføler. I det viste eksempel anvendes en reflektor for at sende lyset retur.



Denne føler er også en refleksionstype: Den aftaster direkte på emnet, og derved "sparer" man en reflektor. Til gengæld bliver afstanden også mindre.

Eksempler på aftastning:

| E3Z-LS-modeller med bag- og forgrundsundertrykkelse | | |
|---|---|--|
| Detektering af låg på kosmetikprodukter  | Detektering af kager på samlebånd  | Detektering af færdigpakket tyggegummi eller slik  |
| E3Z-L-modeller med smal stråle | | |
| Kontrol af sugerør  <p>(Detektering fra skævt vinkel)</p> | Bestemmelse af forside/bagside eller orientering af disketter  <p>Forside Bagside (Fotorelater)</p> | Kontrol af ujævne sammenføjninger  |
| E3Z-B-modeller til gennemsigtige objekter | | |
| Detektering af gennemsigtige PET-flasker – enkelt flaske  | Detektering af gennemsigtige PET-flasker – flere flasker (bundet sammen)  | Detektering på film og glasplader  |
| E3Z-G-gaffelmodel | | |
| Inspektion og positionering til kraner og automatiske transportbånd på lager.  <p>Fotoelektrisk sensor af gaffeltypen model E3Z-G</p> | | |

Billedet viser nogle eksempler på, hvordan en fotoaftaster kan anvendes til forskellige opgaver.

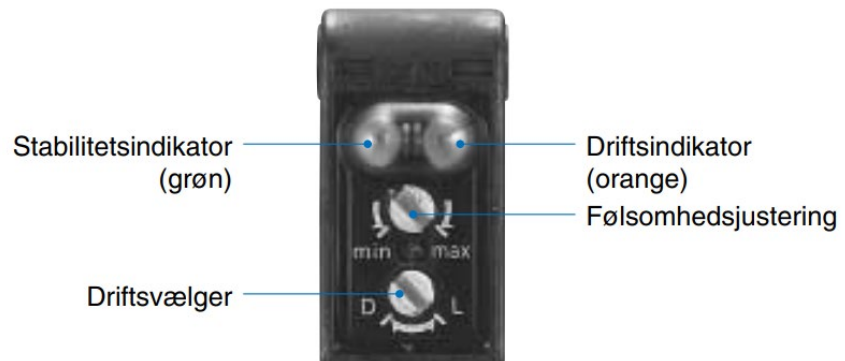
Light on / dark on

På føleren har man mulighed for skifte mellem funktionerne light on og dark on.

Light on betyder, at føleren giver signal, når den belyses, mens dark on betyder, at føleren giver signal, når den ikke belyses.

| Status for udgangstransistor | Sekvensdiagram | Tilstandsvalg | Udgangskredsløb |
|------------------------------|--|--------------------|--|
| Light ON | Indfald Afbrudt Drifts-indikator (orange) ON OFF Udgangstransistor ON OFF Belastning (relæ) Drift Nulstilling (Mellem brun og sort) | L•ON (LIGHT ON) | Modtager Retrorefleksionsmodel Alm. diffus model |
| Dark ON | Indfald Afbrudt Drifts-indikator (orange) ON OFF Udgangstransistor ON OFF Belastning (relæ) Drift Nulstilling (Mellem brun og sort) | D•ON (DARK ON) | Benenes placering i stikket Bemærk: Klemme 2 anvendes ikke. |

Billedet viser funktionen for føleren, når den henholdsvis er koblet til light on og dark on.



På driftsvælgeren vælges mellem light on og dark on.

På følsomhedsjustering indstilles afstanden til det emne, der skal aftastes på.

I føleren er der også indbygget 2 lysdioder: Henholdsvis en driftsindikator, som er orange, og en stabilitetsindikator, som er grøn. Stabilitetsindikatoren bruges til at fortælle, om tasteafstanden er blevet for stor.

Kapacitive nærhedsfølere

Generelt

Automationsopgaver, hvor induktive nærhedssensorer og fotosensorer ikke er anvendelige, ser vi kapacitive nærhedssensorer. Kapacitive nærhedssensorer taster på næsten alle objekter. Anvendelsesområder er i væske, træ-, papir-, glas-, kunststof-, fødevare-, kemi- og halvlederindustrien.



Figur 14 - Capacitive Sensorer: Ifm's hjemmeside.

Det betyder, at den kapacitive føler kan anvendes som føler i en tank med f.eks. olie, vand, mælk, øl, plastgranulat og lignende.

Til gengæld er den også følsom overfor snavs, som kan hænge på føleren. Det kan give problemer i en styring, da der herved opstår falske signaler.

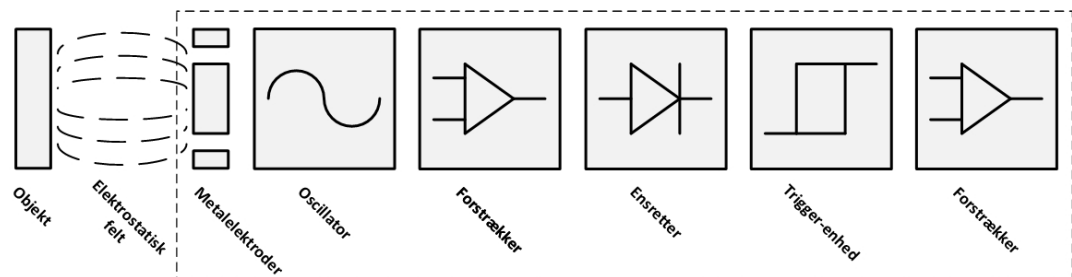
Hoveddele i en kapacitiv nærhedsføler

På billede nedenfor ses en "klassisk" kapacitiv nærhedssensor.



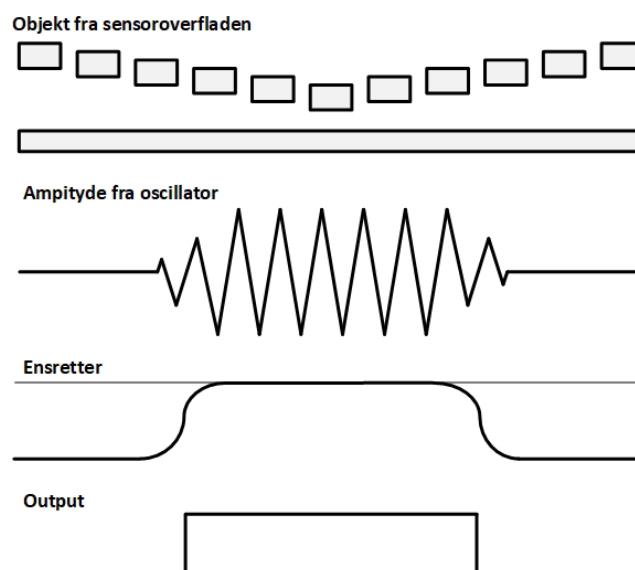
Figur 15 - KG6000 Kapacitiv Nærhedssensor: Ifm's hjemmeside.

En kapacitiv nærhedssensorer har grundlæggende seks hoveddele. En metalelektrode, der danner et vekslende elektrisk felt. En oscillator, der for det meste er et LC-kredsløb, som er forbundet til en forstærker, der sender signalet til en ensretter, en trigger-enhed til at tæste, når et objekt kommer indenfor rækkevidde og så en forstærkning til at drive en last.



Figur 16 - Kapacitiv Nærhedssensors Hoveddel: Forfatterens Egen Illustration.

Når et objekt kommer ind i sensorens elektrostatiske felt, vil kapacitansen ændre sig, og oscillatoren begynder at svinge. Svingningerne bliver forstærket, og strømmen begynder at stige proportionalt med amplituden i oscillatoren. Når strømmen bliver en prædefineret størrelse, bliver sensorens udgang tændt. Fjernes objektet fra sensorens rækkevidde, aftager amplituden og strøm tilsvarende.



Figur 17 – Funktionsdiagram af Kapacitive Nærhedssensors: Forfatterens Egen Illustration.

Typer af Kapacitive Nærhedsfølere

Der er to typer af capacitive nærhedsfølere: Dielektrisk¹ og ledende capacitive nærhedssensorer.

De dielektriske kan detektere objekter, der har en dielektricitetskonstant², der er større end luft. Sensorhovedet består af to parallelle plader, der virker som en kondensator. Kapacitansen vil være lav, når der ikke er noget objekt indenfor sensorens tasteafstand. Når et objekt kommer ind i tasteafstanden stiger kapacitansen, fordi dielektricitetskonstanten i kondensatoren nu bliver ændret. Denne type kan også taste på metaller.

I ledende capacitive nærhedssensorer er der kun en plade, og objektet bliver den anden plade. Objektet er nødsaget til at være elektrisk ledende materiale. Dielektricitetskonstanten mellem de to plader vil ikke ændre sig, men afstanden mellem de to plader vil mindskes. Når afstanden mellem de to plader mindskes, øges kapacitansen.

¹ En betegnelse for materialer, som ikke leder elektrisk strøm

² Betegnelse på virkningsgraden af et materiale, som ikke leder elektrisk strøm

Installation

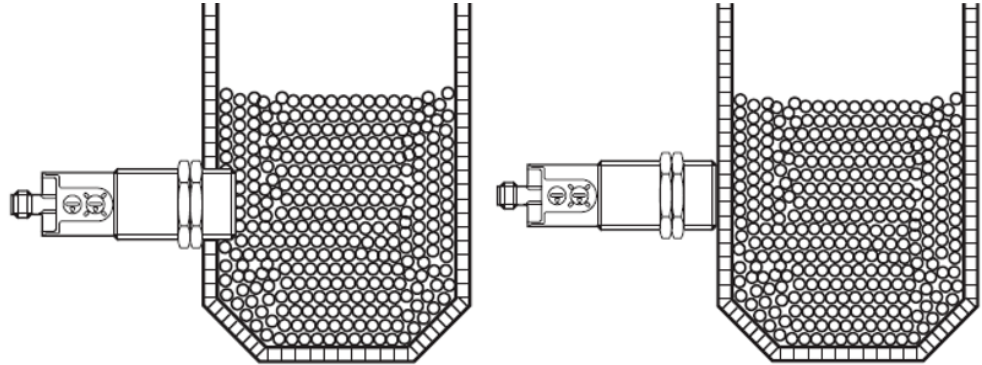
Installationen af capacitive nærhedssensorer minder om de induktive nærhedssensorer. Der er dog visse forskelle, man skal tage højde for.

Nedenfor er et uddrag af en KG6000 kapacitiv nærhedssensor fra ifm. Her kan man aflæse tasteafstanden, som kan justeres.

| Produktegenskaber | |
|-------------------------|--|
| Elektrisk design | PNP |
| Udgangsfunktion | normally open / normally closed; (valgbar) |
| Tasteafstand [mm] | 0,5...30 |
| Kommunikationsinterface | IO-Link |
| Kapsling | gevindtype |
| Dimensioner [mm] | M18 x 1 / L = 92,5 |

Figur 18 - Uddrag af KG6000 Datablad: Ifm's hjemmeside.

Kapacitive nærhedsføler har muligheden for at taste gennem materialer. Dette er kun gennem materialer, der ikke er metaller, som vist nedenfor til højre. Hvis det er en metaltank, skal sensoren være i kontakt med materialet, som vist nedenfor til venstre.



Figur 19 - Uddrag af Operating Instructions Capacitive Sensors: Ifm's hjemmeside.

Kapacitive sensorer, der er cylinderformet og monteres plant i materialer, skal udføres med respektafstande.

Af de to udtryk Flush (skærmet) og Non-flush (uskærmet) er det nødvendigt at benytte non-flush til montering i metaller.

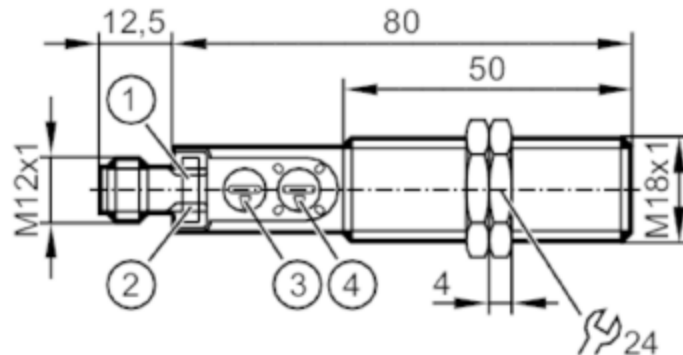
I situationer, hvor man anvender flere capacitive sensorer af samme type, vil der også være respektafstande mellem disse. Modstående, side by side-installationer, vist nedenunder for en ifm capacitive nærhedssensor.

| Flush / skærmet | Non-flush / uskærmet |
|---|----------------------|
| | |
| Minimum distances when mounting several switches of the same type | |
| | |

Figur 20 - Uddrag af Operating Instructions Capacitive Sensors: Ifm's hjemmeside.

På dimensionstegningen over sensoren vil man kunne aflæse de fysiske størrelser og møtrikker. Fra tidligere er der mulighed for at kunne indstille tasteafstand og sensorens koblingsfunktion (NO/NC).

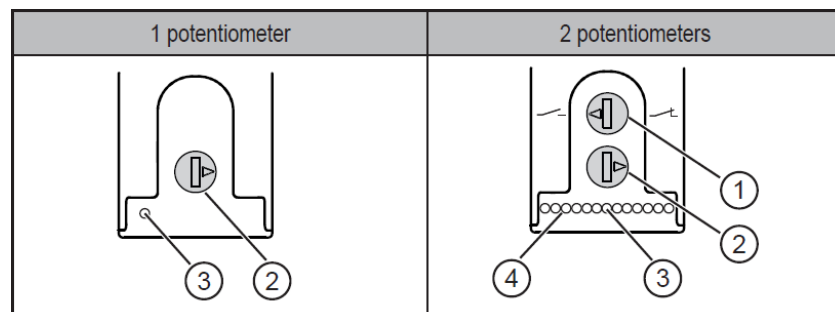
- ① LED gul udgangsstatus
- ② LED grønt signal
- ③ Potentiometer Tasteafstand
- ④ Potentiometer Koblingsfunktion



Figur 21 - Uddrag af KG6000 Datablad: Ifm's hjemmeside.

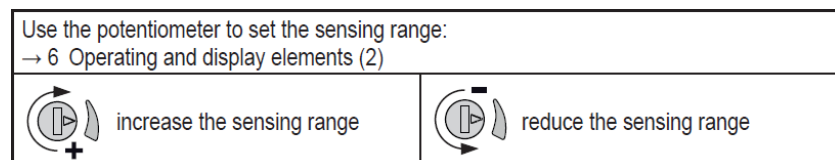
Denne ifm-sensor kan fås med mulighed for at skifte mellem NO/NC, og så har den også mulighed for at indstille tasteafstanden.

Typen med 2 potentiometer har også en LED-række, der viser, om objektet er for langt væk fra sensorens tastepunkt eller har passeret det.



Figur 22 - Uddrag af Operating Instructions Capacitive Sensors: Ifm's hjemmeside.

Disse potentiometre er meget følsomme, så indstilling skal foregå roligt.

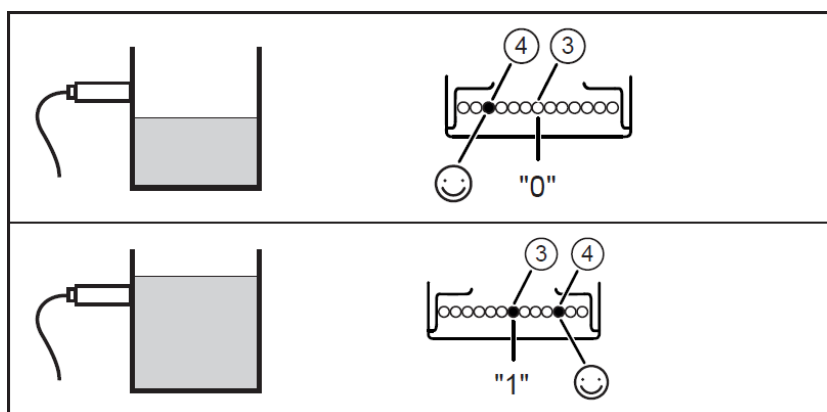


Figur 23 - Uddrag af Operating Instructions Capacitive Sensors: Ifm's hjemmeside.

Hvis man vil anvende en kapacitiv nærhedsfølter til at detektere et medie i en beholder, skal man starte med at tømme tanken, så sensoren ikke kan blive påvirket. Hvis der er en LED-lampe (indikator), kan man benytte denne for at se, om sensoren er påvirket.

Skrue op for tasteafstanden, til sensoren bliver påvirket. Derefter skrues ned for tasteafstanden, til indikatoren skifter. Fyld tanken derefter og kontroller, om indikatoren skifter.

Denne funktion kræver selvfølgelig, at det er muligt at indstille tasteafstanden. Typen KG6000 har LED-rækken til at indstille denne funktion, som vist nedenfor.



Figur 24 - Uddrag af Operating Instructions Capacitive Sensors: Ifm's hjemmeside.

Fortrådning

Fortrådning af 3 tråds kapacitive nærhedssensorer er den samme fremgangsmåde som de induktive. De kan komme med støbt stik, M8/12-stik, som vist nedenfor. Har man en hvid ledning i et 4-lederstik, er det normalt et inverteret digitalt signal eller en teach-funktion.

| | Cable | Connector |
|-----|---------------------------------------|-----------|
| PNP | | |
| NPN | | |
| | BN = brown BK = black BU = blue | |

Figur 25 - Uddrag af Operating Instructions Capacitive Sensors: Ifm's hjemmeside.

Magnetiske

Generelt



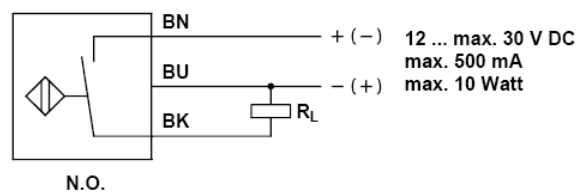
Billedet viser en magnetisk føler fra ifm. En magnetisk føler benævnes også som reedkontakt. En reedkontakt aftastes af en magnet.



Reedkontakter anvendes typisk i forbindelse med cylindre, hvor de bruges til at aftaste cylinderens position. Billedet viser, hvorledes føleren monteres vha. et specielt beslag, som passer til cylinderen.

For at føleren kan anvendes på en cylinder, kræves det, at cylinderen er forberedt til at aftaste på en reedkontakt. Dette medfører, at der skal være en magnet på cylinderens stempel, som aftaster føleren ved at sende magnetfelt ud igennem cylinderhuset.

Elektrisk tilslutning



Diagrammet viser tilslutning af en reedkontakt med en tretrådsforbindelse.



Diagrammet viser tilslutning af en reedkontakt med en totrådsforbindelse.

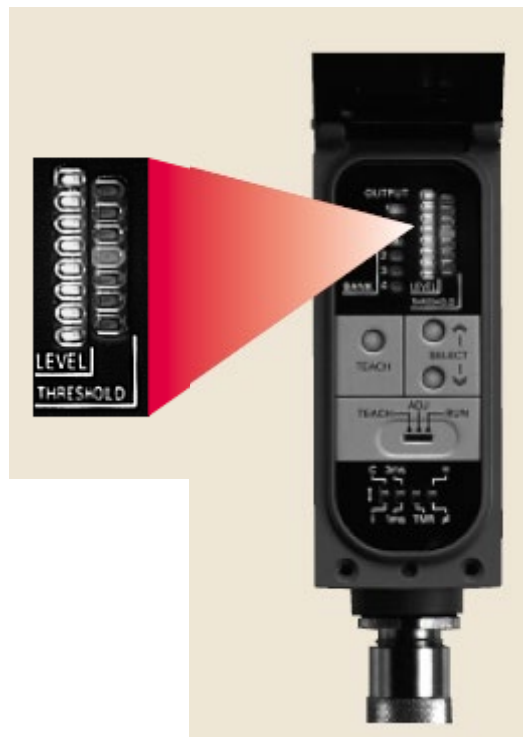
Specialfølere

Generelt

Foruden standardfølerne findes der en række specialfølere, som f.eks. kan aftaste på farve, emneoverflade, emnestruktur og lignende.

I det efterfølgende afsnit vises et enkelt eksempel på en føler, som aftaster på farver.

Farveføler



Billedet viser en Omron RGB farveføler. Denne føler kan bruges til at sortere emner efter farve.

Den viste føler kan detektere 4 forskellige farver. Hver farve har deres egen signaludgang på føleren.

Når føleren skal justeres, kobles den i "teach". Herefter lægges et emne med den pågældende farve under føleren. For at aktivere udgangen med den pågældende farve trykkes på knappen "teach".

PLC-installation

Generelt

Installation af PLC'en, ledningsfremføring og ledningstilslutning skal altid udføres omhyggeligt og i overensstemmelse med de anvisninger, der følger udstyret. Der skal også tages hensyn til EMC (Elektro Magnetic Compatibility, der kan udtrykkes som elektromagnetisk sameksistens).

I den efterfølgende tekst er nævnt nogle konkrete data, som er hentet fra installationsmanualen til et Siemens 300-system.

Ved installation bør der tages hensyn til følgende:

- Omgivelsestemperatur:
 - 0 – 40 °C ved vertikal montage
 - 0 – 60 °C ved horisontal montage
- Fugtighed
- Tåler ikke større mængder støv, snavs, salt eller små jernpartikler
- Tåler ikke direkte sol
- Tåler ikke stød og vibrationer
- Tåler ikke korroderende eller brændbare gasser

Støjfyldt miljø

Skal PLC'en sidde i et elektrisk støjfyldt miljø, monteres den i en metaltavle eller tavleudsnit. Ved montering er det af største vigtighed, at PLC'en har en god elektrisk kontakt med grundpladen; skrab evt. maling bort og skær gevind i bundpladen. Grundpladen skal igen have god elektrisk kontakt til metalkabinettet. Jordledning forbindes til PLC'en, og der udlignes mellem PLC og maskinudstyr.

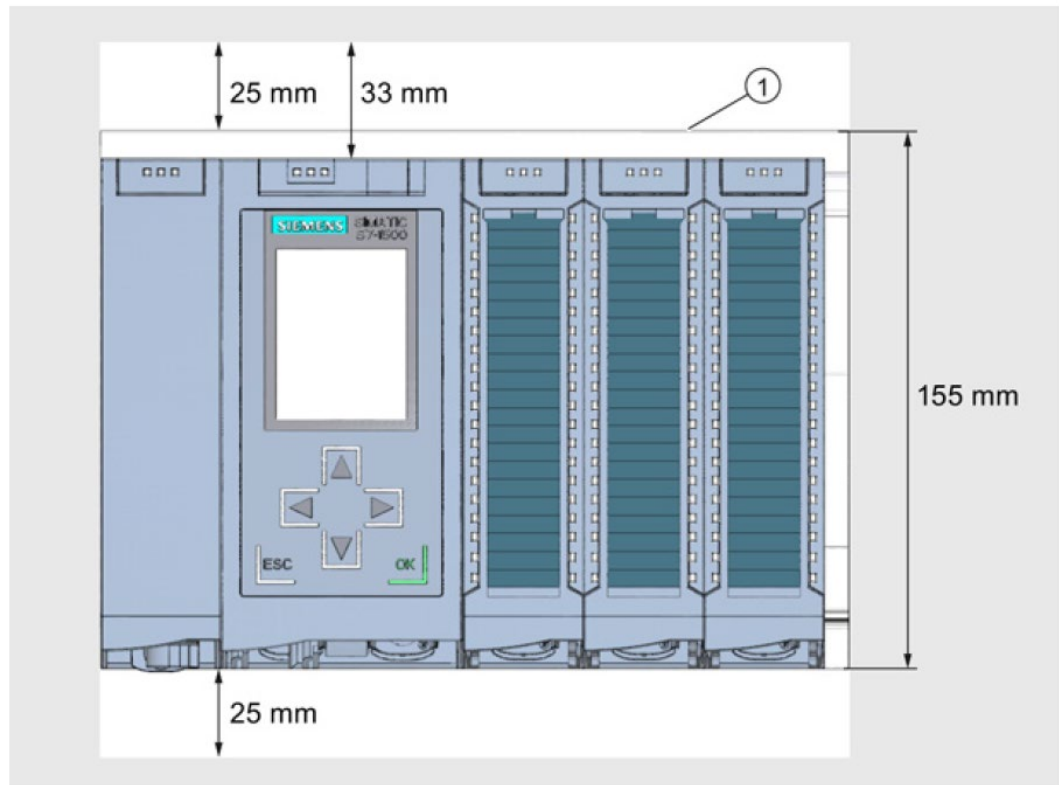
En PLC, som indeholder en masse elektronik, skal under de forhold, den normalt monteres i, fungere tilfredsstillende. At der er opnået EMC betyder, at PLC'en er konstrueret så støjimmunt, at den kan tåle at befinde sig i den støj, der er normal for dens placering.

PLC'ens egen støjudsendelse bør være af en sådan beskaffenhed, at dens omgivelser kan leve med det.

Der henvises i øvrigt til afsnittet om EMC.

Installation

Der bør altid være god plads i et styreskab til PLC'en med hjælpeudstyr som klemrækker, relæer, strømforsyning mv. PLC'en bør altid sidde længst muligt borte fra transformere og relæudstyret.



① Upper edge of the mounting rail

Diagrammet viser de afstande, som Siemens foreskriver, der skal overholdes til andet udstyr i forbindelse med montage af en S7-1500 PLC.

Transformeren udstråler magnetfelter, som kan virke forstyrrende på PLC'en, relæudstyret giver HF-indstråling fra gnister i kontaktor, da luft virker som leder over korte afstande, så derfor: Placér, hvis muligt, PLC'en i et andet tavleudsnit.

Temperaturforholdene skal være i orden. Elektronik arbejder ikke tilfredsstillende, når temperaturen enten er for høj eller for lav.

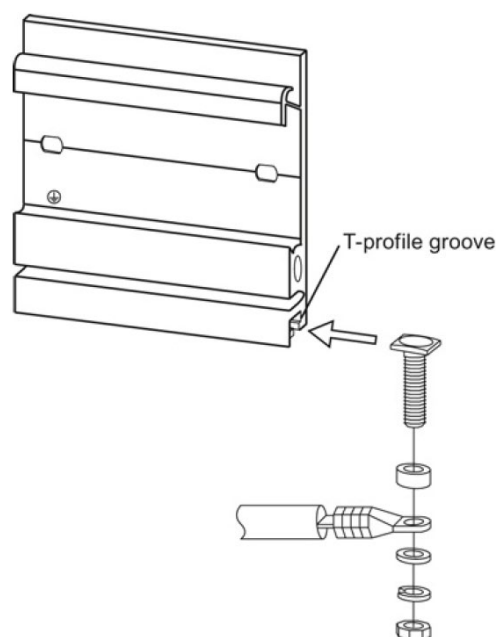
Hold afstand mellem ind- og udgange. Signalkabler fra følere og effektkabler fra motorer må ikke sammenføres f.eks. i ledningskanaler (transienter fra udgang flyttes til indgange ved indstråling).

Forsyningskabler, der fører store strømme, bør altid holdes langt væk fra signalkabler, selvom signalkablerne er skærmede. Montering af signalkabler tæt op ad maskindele, metalstativer, chassisplader eller andre jordede dele nedsætter faren for indstråling i eller udstråling fra et kabel.

Signalkabler fra digitale følere til PLC'en kan normalt udføres uden brug af skærmede kabler, men er der tale om nærføring til kraftige støjfelte, parallelføring over større afstande og i nærheden af netforsyningskabler samt kabler, hvor store startstrømme forekommer, anbefales det at skærme kablerne.

Signalkabler fra analoge transmittere og andet analogt udstyr udføres normalt som skærmet kabel. Det er af største vigtighed, at skærmen monteres og afsluttes korrekt. Skærmen forbindes til jordpotentiale i begge ender som hovedregel. Hvis det giver problemer med strøm i skærmen, forbindes skærmen kun til jordpotentiale i den ene ende.

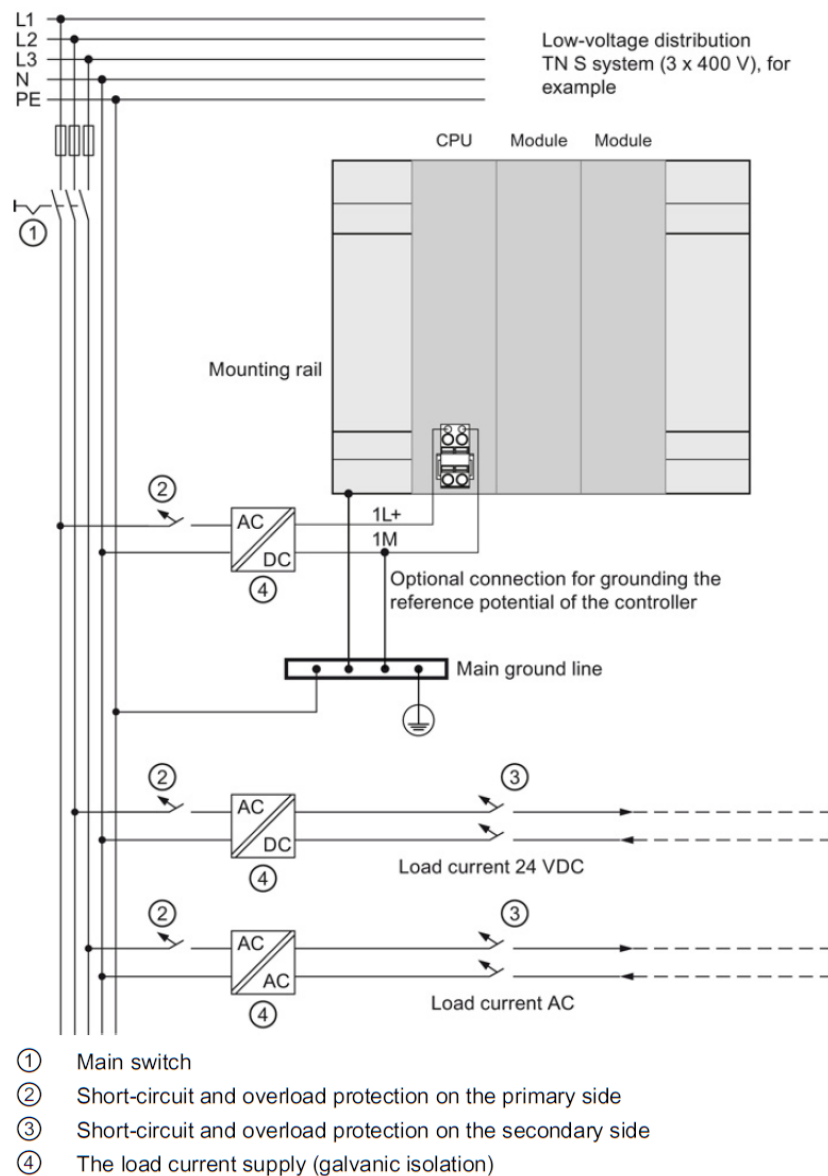
Siemens anbefaler, at man forbinder skærmen til skinnen, hvorpå PLC'en er monteret på følgende måde:



1. Afisolér jordlederen med en minimumsdiameter på 10 mm² og monter en ringterminalen til en M6 bolte på lederen med en krympetangen.
2. Skub den medfølgende bolt ind i T-profilens rille.
3. Indsæt afstandsstykket, ringterminalen med jordforbindelse, fladskive og låseskive på bolten (i nævnte rækkefølge). Skru møtrikken på og spænd denne. (moment 4 Nm).
4. Tilslut den modsatte ende af jordkablet til den centrale jordingspunkt / beskyttelseslederskinne (PE).

Forsyning

Elektrisk forsyning



Diagrammet viser en typisk forsyning til en maskine med en PLC. Da det drejer sig om en maskine, er forsyningen udformet efter DS/EN60 204-1.

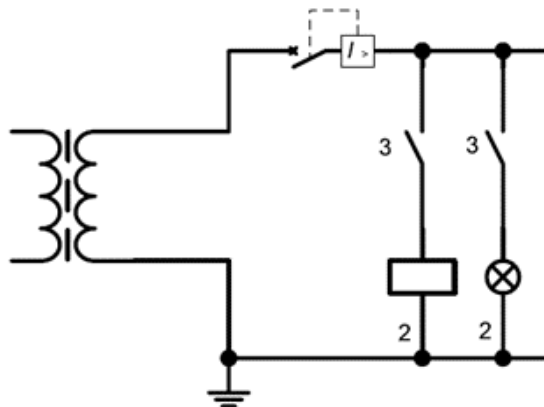
Maskinen forsynes via hovedsikringer, som normalt er placeret i den faste installation. Forsyningen er her vist som de tre faser, L1, L2, L3, nulleader N og beskyttelsesleder PE (Protective Earth). PE skal tilsluttes maskinens hovedjordklemme, som har forbindelse til hele maskinstellet.

Forsyningen tilsluttes maskinens forsyningsadskiller ①

PLC's strømforsyning tilsluttes via egen automatsikring ②, da hovedsikringerne typisk er for store til en forsvarlig kortslutningsbeskyttelse. HUSK PE-tilslutning.

Der skal anvendes transformer eller stabiliserede-spændingsforsyninger^④ til forsyning af styrekredse, så både relæ- og magnetventilspoler sikres galvanisk adskillelse gennem disse. Transformer eller stabiliserede-spændingsforsyninger har egen kortslutnings- og overbelastningsbeskyttelse på primærsiden ^②, da sekundærsiden kan forsynes med flere kortslutnings- og overbelastningsbeskyttelser^③

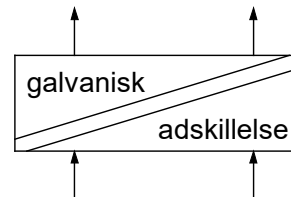
DS/EN 60204-1, Maskinsikkerhed – Elektrisk materiel på maskiner § 9.4.3 foreskriver, at der ikke må ske utilsigtet funktion som følge af jordfejl. For at kunne opfylde denne bestemmelse kan den ene side af styrekredsen forbindes til beskyttelseskredsen (PE). Kontakter og spoler skal forbindes ifølge §9.4.3.1.2 Spolens ene klemme skal forbindes direkte til den side, som er forbundet til beskyttelseskredsen (2), mens kontakter (3) skal være forbundet mellem spolens anden klemme og den side, som ikke er forbundet med beskyttelseskredsen.



Galvanisk adskillelse

Med S7-1500 er der galvanisk adskillelse mellem:

- Den primære side af systemets strømforsyning (PS) og alle andre kredsløbskomponenter
- (PROFIBUS/PROFINET) kommunikationsgrænseflader for CPU'en og alle andre kredsløbskomponenter
- Belastningskredsløbene / proceselektronikken og alle andre kredsløbskomponenter i S7-1500



Højfrekvente interferensstrømme og kapacitivtkoblet strømme ledes gennem integrerede RC-led eller kondensatorer.

Den følgende figur viser en forenklet fremstilling af de ovenfor anførte krav.

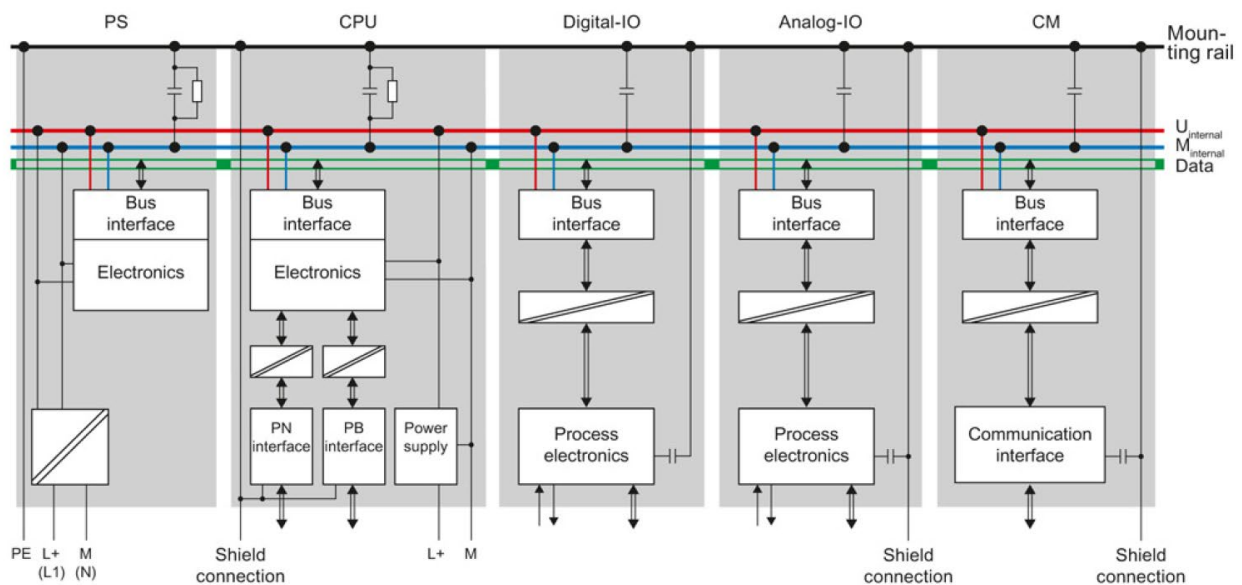


Figure 6-2 Potential relationships for S7-1500 with CPU 1516-3 PN/DP

Mindre styringer

Til mindre PLC styringer kan samme 24 VDC-forsyning bruges til hele maskinens DC-kreds. Dog skal man være opmærksom på, at EMC-beskyttelsen nedsættes.

Regler for kabling

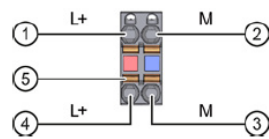
Nedenfor er vist de regler, som Siemens angiver i forbindelse med tilslutning af en S7-1500 PLC.

| Wiring rules for... | | CPU | System power and load current supply |
|---|---|---------------------------------------|---------------------------------------|
| Connectible conductor cross-sections for solid wires | | - | - |
| Connectible conductor cross-sections for stranded wires | Without end sleeve | 0.25 to 2.5 mm ² | 1.5 mm ² |
| | With end sleeve | 0.25 to 1.5 mm ² | 1.5 mm ² |
| Number of wires per connection | | 1 | 1 |
| Length of stripped wires | | 10 to 11 mm | 7 to 8 mm |
| End sleeves according to DIN 46228 | without plastic sleeve | Design A, 10 mm long | Design A, 7 mm long |
| | with plastic sleeve 0.25 to 1.5 mm ² | Design E, 10 mm long | Design A, 7 mm long |
| Sheath diameter | | - | 8.5 mm |
| Tool | | 3 to 3.5 mm screwdriver, conic design | 3 to 3.5 mm screwdriver, conic design |
| Connection system | | Push-in terminal | Screw terminal |
| Tightening torque | | - | from 0.5 Nm to 0.6 Nm |

| Wiring rules for... | | 40-pin front connector (screw terminal, for 35 mm module) | 40-pin front connector (push-in terminal, for 35 mm module) | 40-pin front connector (push-in terminal, for 25 mm module) |
|---|---|--|--|---|
| Connectible conductor cross-sections for solid wires | | up to 0.25 mm ² | up to 0.25 mm ² | up to 0.25 mm ² |
| Connectible conductor cross-sections for stranded wires | Without end sleeve | 0.25 to 1.5 mm ² | 0.25 to 1.5 mm ² | 0.25 to 1.5 mm ² (max. 40 x 0.75 mm ²) |
| | With end sleeve | 0.25 to 1.5 mm ² | 0.25 to 1.5 mm ² | 0.25 to 1.5 mm ² (max. 32 x 0.75 mm ² ; 8 x 1.5 mm ²) |
| Number of wires per connection | | 1 or combination of 2 wires up to 1.5 mm ² (total) in the same end sleeve | 1 or combination of 2 wires up to 1.5 mm ² (total) in the same end sleeve | 1 or combination of 2 wires up to 1.5 mm ² (total) in the same end sleeve |
| Length of stripped wires | | 10 to 11 mm | 8 to 11 mm (corresponding to length of end sleeve**: 8 mm, 10 mm) | 8 to 11 mm (corresponding to length of end sleeve**: 8 mm, 10 mm) |
| End sleeves according to DIN 46228 | without plastic sleeve | Design A, 10 mm and 12 mm long | Design A, 10 mm long | Design A, 10 mm long |
| | with plastic sleeve 0.25 to 1.5 mm ² | Design E, 10 mm and 12 mm long | Design E, 8 mm and 10 mm long | Design E, 8 mm and 10 mm long |
| Sheath diameter | | - | - | - |
| Tool | | 3 to 3.5 mm screwdriver, conic design | 3 to 3.5 mm screwdriver, conic design | 3 to 3.5 mm screwdriver, conic design |
| Connection system | | Screw terminal | Push-in terminal | Push-in terminal |
| Tightening torque (screw terminal) | | from 0.4 Nm to 0.7 Nm | - | - |
| Max. actuation force for complete opening of the push-in terminal | | - | 40 N | 40 N |

Forsyning af CPU

Ledninger til CPU'ens strømforsyning må, når der anvendes terminalrør, maksimalt være 1,5 mm² (maks. 10A).



- ① + 24 V DC of the supply voltage
- ② Mass of the supply voltage
- ③ Mass of the supply voltage for looping (current limited to 10 A)
- ④ + 24 V DC of the supply voltage for looping (current limited to 10 A)
- ⑤ Spring opener (one spring opener per terminal)

Beskyttelseskreds

Gennemgående forbindelser i beskyttelseskredsen skal besigtiges for at sikre overensstemmelse med pkt. 8 i DS/EN60 204-1, og det kontrolleres, at der ikke er løse forbindelser i beskyttelseslederens tilslutningspunkter. Ud over det skal der foretages en prøve efter pkt. 18 i DS/EN60 204-1.

Ledninger uden for kapslingen

En 1-leder skal være mindst 1 mm² og flerleder 0,75 mm². DS/EN60 204-1 tillader, at forskellige former for spænding og spændingsstørrelse overføres i samme kabel, bare alle ledere opfylder kravet til den højeste spænding.

Af hensyn til driftssikkerhed (på grund af støj) og personsikkerhed (plus håndværksmæssige traditioner) er kablerne normalt opdelt i effektkredse, signalkredse og styrekredse. Styrekredse kan være opdelt yderligere i aftastere og belastninger (f.eks. magnetventiler).

Hvis det ikke er over længere strækninger, sker der ikke noget ved at oplægge kablerne sideløbende. Hvis det drejer sig om lange strækninger, bør effektkabler oplægges særskilt, specielt, hvis de er tilsluttet SSR- (Solid State Relæ) styret belastninger eller frekvensomformere.

Overførsel af analoge signaler kræver normalt altid et skærmet kabel.

EMC (Engelsk)

12.2 Protection against electromagnetic interference

12.2.1 Basic Points for System Installations Conforming with EMC Requirements

Definition: EMC

EMC (electromagnetic compatibility) describes the capability of electrical equipment to operate free of errors in a given electromagnetic environment, without being subject to external influence and without influencing external devices in any way.

Introduction

Although your S7-300 and its components are developed for an industrial environment and high electromagnetic compatibility, you should draw up an EMC installation plan before you install the controller under consideration of all possible interference sources.

Possible effects of interference

Electromagnetic noise can influence a PLC in various ways:

- Electromagnetic fields having a direct influence on the system
- Interference via bus signals (PROFIBUS DP etc.)
- Interference coupling via the system wiring
- Interference influencing the system via the power supply and/or protective ground

The figure below shows the likely paths of electromagnetic interference.

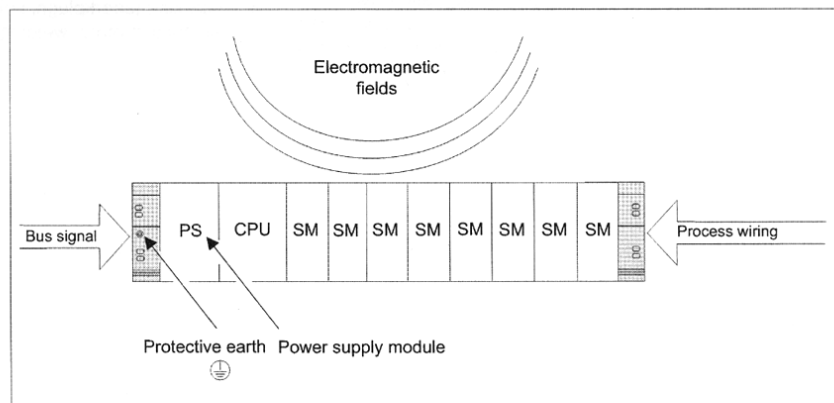


Figure 12-1 Possible paths of electromagnetic interference

Coupling mechanisms

Depending on the emitting media (line or isolated) and the distance between the interference source and the device, four different coupling mechanisms can influence the PLC.

Table 12-5 Coupling mechanisms

| Coupling mechanisms | Cause | Typical interference sources |
|--------------------------|---|--|
| Electrical coupling | Galvanic or mechanical coupling always occurs when two circuits use one common cable. | <ul style="list-style-type: none"> • Clocked devices (influence on the network due to converters and third-party power supply modules) • Starting motors • Potential differences on component enclosures with common power supply • Static discharge |
| Capacitive coupling | Capacitive or electrical coupling occurs between conductors connected to different potentials. The coupling effect is proportional to voltage change over time. | <ul style="list-style-type: none"> • Interference coupling due to parallel routing of signal cables • Static discharge of the operator • Contactors |
| Inductive coupling | Inductive or magnetic coupling occurs between two current circuit loops. Current flow in magnetic fields induces interference voltages. The coupling effect is proportional to current change over time. | <ul style="list-style-type: none"> • Transformers, motors, arc welding devices • Power supply cables routed in parallelism • Switched cable current • High-frequency signal cable • Coils without suppression circuit |
| Radio frequency coupling | Radio frequency coupling occurs when an electromagnetic wave reaches a conductor system. This wave coupling induces currents and voltages. | <ul style="list-style-type: none"> • Neighboring transmitters (e.g. radio phones) • Sparking (sparkplugs, collectors of electrical motors, welding devices) |

12.2.2 Five Basic Rules for Ensuring EMC

If you comply with these five basic rules ...

you can ensure EMC in many cases!

Rule 1: Large-area grounding

When you install the automation equipment, make sure that surfaces of inactive metal parts are well bonded to chassis ground (see the following sections).

- Bond all passive metal parts to chassis ground, ensuring large area and low-impedance contact.
- When using screw connections on varnished or anodized metal parts, support contact with special contact washers or remove the protective insulating finish on the points of contact.
- Wherever possible, avoid the use of aluminum parts for ground bonding. Aluminum oxidizes very easily and is therefore less suitable for ground bonding.
- Create a central connection between chassis ground and the equipotential grounded/protective conductor system.

Rule 2: Proper cable routing

Ensure proper cable routing when you wire your system (see the section below on *Indoor/outdoor cable routing*).

- Sort your wiring system into groups (high-voltage/power supply/signal/data cables).
- Always route high-voltage, signal or data cables through separated ducts or in separate bundles.
- Install the signal and data cables as close as possible to grounded surfaces (e.g. supporting beams, metal rails, steel cabinet walls).

Rule 3: Mounting the cable shielding

Take care that all cable shielding is properly fastened (refer to the section on *Shielding of cables*).

- Always use shielded data cable. Always connect both ends of the shielding to ground on a large area.
- Analog cables must always be shielded. For the transmission of low-amplitude signals it might prove to be more efficient to have only one side of the shielding connected to ground.
- Directly behind the cable entry in the cabinet or enclosure, terminate the shielding on a large area of the shielding/protective ground bar and fasten it with the help of a cable clamp. Then, route the cable to the module; however, do not connect the shielding once again to ground in this place.
- Connections between the shielding/protective ground busbar and the cabinet/enclosure must be of a low impedance.
- Always install shielded data cables in metal/metallized connector housings.

Rule 4: Special EMC measures

Some special applications might require special EMC measures (refer to the section on *How to protect digital output modules against inductive surge voltage*).

- Connect anti-surge elements to all inductive devices not controlled by S7-300 modules.
- For cabinet or cubicle lighting in the immediate range of your controller, use incandescent lamps or interference suppressed fluorescent lamps .

Rule 5: Homogeneous reference potential

Create a homogeneous reference potential and ground electrical equipment whenever possible (refer to the section on *Equipotential bonding*).

- Route your equipotential conductors over a wide area if potential differences exist or are expected between your system components.
- Make sure you carefully direct your grounding measures. Grounding measures protect the controller and its functions.

Form a star circuit to connect the equipment in your system and the cabinets containing central/expansion units to the grounding/protective conductor system. This prevents the formation of ground loops.

12.2.3 EMC Compatible Installation of PLC

Introduction

Quite often it is the case that interference suppression measures are not taken until corruption of user signals is detected after the controller is actually in operation.

Frequently, the causes of such interference are found in inadequate reference potentials as a result of faulty installation. This section shows you how to avoid such errors.

Inactive metal parts

Inactive parts are referred to as electrically conductive elements, separated from active elements by a basic insulating and only subject to electrical potential if an error occurs.

Installation and ground bonding of inactive metal parts

Bond all inactive metal parts to a large-surface ground when you install the S7-300. Proper ground bonding ensures a homogeneous reference potential for the controller and reduces the effect of interference coupling.

The ground connection establishes an electrically conductive interconnection of all inactive parts. The sum of all interconnected inactive parts is referred to as chassis ground.

This chassis ground must never develop a hazardous potential even if a fault occurs. Therefore, chassis ground must be connected to the protective conductor using cables with an adequate conductor cross-section. To avoid ground loops, physically separate chassis ground elements (cabinets, parts of the building construction or machine) must be bonded to the protective conductor system in a star circuit.

Observe the following for ground connection:

- In the same way as with active elements, exercise meticulous care to interconnect inactive metal elements.
- Always make sure that you have a low-impedance interconnection between metal elements (e.g. large and highly conductive contact surface).
- The protective insulating finish on varnished or anodized metal elements must be pierced or removed. Use special contact washers or completely remove the finish on the point of contact.
- Protect your connecting elements against corrosion (e.g. with a suitable grease)
- Interconnect moving chassis ground elements (e.g. cabinet doors) with flexible ground straps. Always use short ground straps with a large surface (the surface is decisive for the diversion of high-frequency currents).

12.2.4 Examples of an EMC Compatible Installation

Introduction

Below you can find two examples of an EMC compatible PLC installation.

Example 1: EMC compatible cabinet installation

The figure below shows a cabinet installation with the measures described above (bonding of inactive metal parts to chassis ground and connecting the cable shielding to ground). This sample applies only to grounded operation. Note the points in the figure when you install your system.

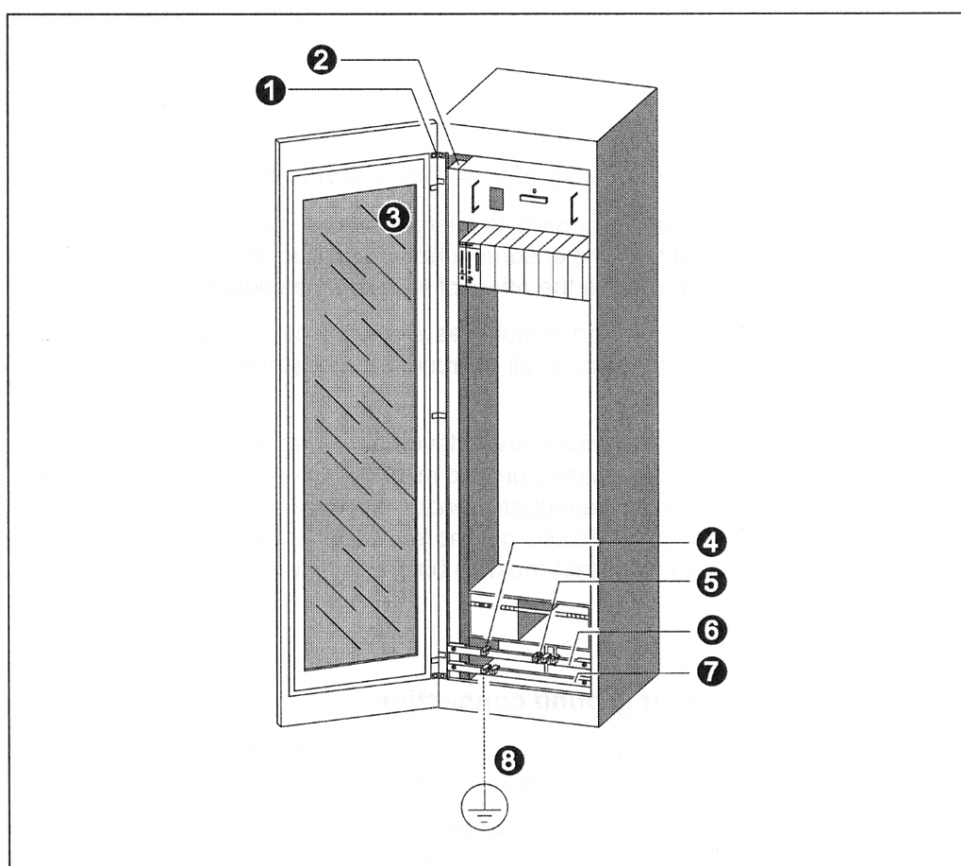


Figure 12-2 Example of an EMC compatible cabinet installation

Key to example 1

The numbers in the following list refer to the numbers in the figure above.

Table 12-6 Key to example 1

| No | Description | Explanation |
|----|--|--|
| 1 | Ground straps | If no large-surface metal-to-metal connections are available, you must either interconnect inactive metal parts (e.g. cabinet doors or mounting plates) or bond them to chassis ground using ground straps. Use short ground straps with a large surface. |
| 2 | Supporting bars | Interconnect the supporting bars on a large area to the cabinet walls (metal-to-metal connection). |
| 3 | Mounting the rail | The mounting bar and rack must be interconnected with large-area metal-to-metal connections. |
| 4 | Signal cables | Connect the shielding of signal cables on a large area of the protective conductor/additional shielding busbar and fasten them with cable clamps. |
| 5 | Cable clamp | The cable clamp must cover a large area of the shielding braid and ensure good contact. |
| 6 | Shielding busbar | Interconnect the shielding busbar on a large surface with the supporting bars (metal-to-metal connection). The cable shielding is terminated on the busbar. |
| 7 | Protective conductor busbar | Interconnect the protective conductor busbar on a large surface with the supporting bars (metal-to-metal connection). Interconnect the protective conductor busbar and the protective ground system, using a separate cable (minimum cross-section 10 mm ²). |
| 8 | Cable to the protective ground system (equipotential ground) | Interconnect the cable on a large area with the protective ground system (equipotential ground). |

Example 2: EMC compatible wall mounting

When operating your S7 in a low-noise environment that conform with permitted ambient conditions (see Appendix *Ambient conditions*), you can also mount your S7 in frames or to the wall.

Interference coupling must be diverted to large metal surfaces. Therefore, always mount standard profile/shielding/protective conductor rails on metal parts of the construction. Steel sheet panels reference potential surfaces have been found especially suitable for wall-mounting.

Provide a shielding busbar for connecting your cable shielding. This shielding busbar can also be used as protective ground bar.

Note the following points on framewall-mounting:

- When mounting on varnished or anodized metal parts, use special contact washers or remove the insulating layers.
- Provide a large-surface and low-impedance metal-to-metal connection for fastening the shielding/protective ground bar.
- Always touch-protect live mains conductors.

The figure below shows an example of EMC compatible wall-mounting of an S7.

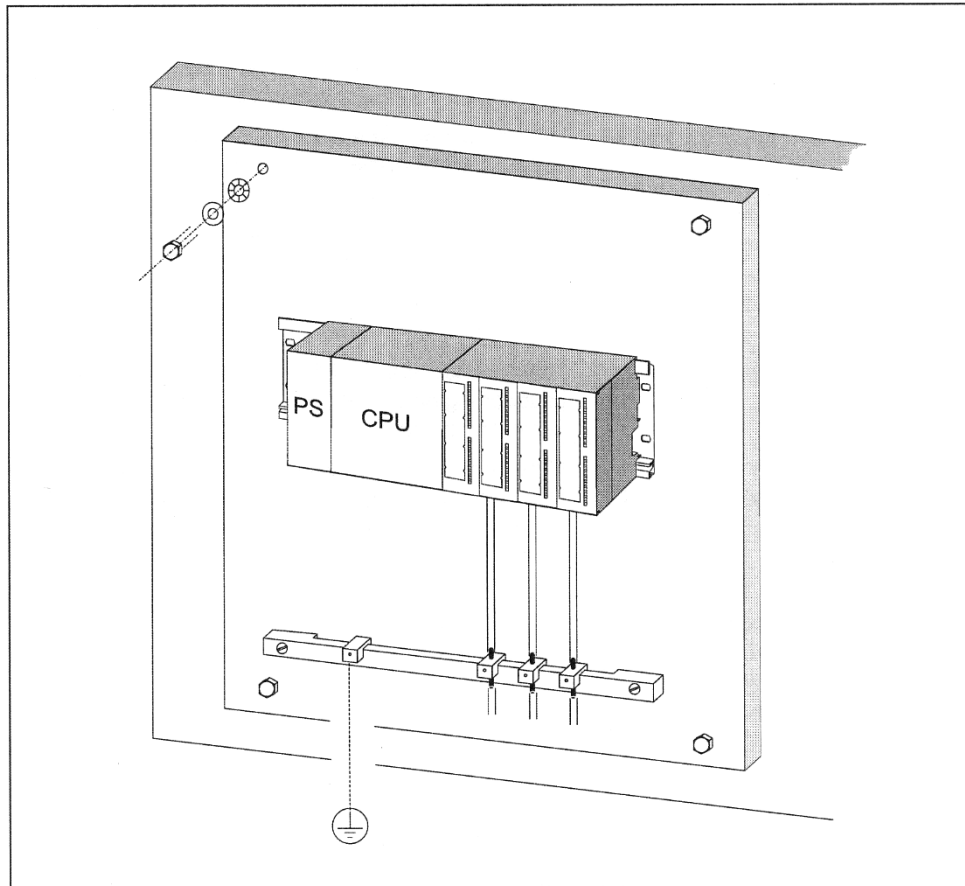


Figure 12-3 Example of EMC compatible wall-mounting

12.2.5 Shielding of Cables

Purpose of the shielding

A cable is shielded to attenuate the effects of magnetic, electrical and electromagnetic interference on the cable.

Operating principle

Interference currents on cable shielding is diverted to ground conductive interconnection between the shielding and the cabinet. To avoid interference as a result of these currents, it is imperative to provide a low-impedance connection to the protective conductor.

Suitable cables

Whenever possible, use cables equipped with a shielding braid. Shielding density should be at least 80%. Avoid cables with film shielding, because the film can be easily damaged by tensile or pressure stress, thus reducing its shielding effect.

Handling of the shielding

Note the following points on handling the shielding:

- Always use metal clamps to mount shielding braid. The clamps must contact a large area of the shielding and provide appropriate contact force.
- Directly behind the cabinet's cable entry, terminate the shielding on a shielding bus. Then, route the cable to the module; however, do not connect the shielding once again to ground in this place.
- In installations outside of cabinets (e.g. for wall-mounting) you can also terminate the shielding on a cable duct.

The figure below shows some options for mounting shielded cables, using cable clamps.

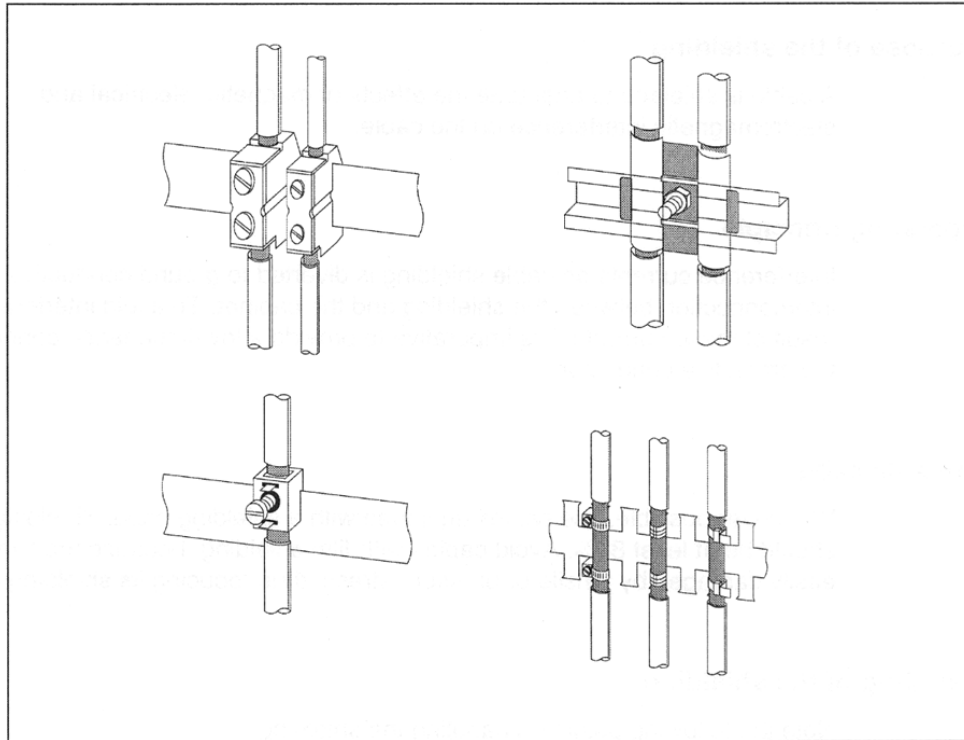


Figure 12-4 Mounting cable shielding

12.2.6 Equipotential bonding

Potential differences

Potential differences can occur between separate system elements. This can result in high equipotential currents, e.g. if the cable shielding is terminated at both ends and grounded to different system components.

The cause of potential difference can be differences in the power supplies.



Warning

Cable shielding is not suitable for equipotential bonding. Always use the prescribed cables (e.g. with a cross-section of 16 mm²). When installing MPI/DP networks, provide a sufficient conductor cross-section. Otherwise, interface hardware might get damaged or even be destroyed.

Equipotential bonding conductor

To reduce potential differences and ensure proper functioning of your electronic equipment, you must install equipotential bonding conductors.

Note the following points on the use of equipotential bonding conductors:

- The lower the impedance of an equipotential bonding conductor, the more effective is equipotential bonding.
- When shielded signal cables interconnect two system components and the shielding is connected on both ends to ground/protective conductors, the impedance of the additional equipotential bonding conductor must not exceed 10% of the shielding impedance.
- Determine the cross-section of your equipotential bonding conductor on the basis of the maximum equalizing current that will flow through it. The equipotential bonding conductor cross-section that has proven best in practice is 16 mm².
- Always use equipotential bonding conductors made of copper or galvanized steel. Always connect the cables on a large surface to the equipotential busbar/protective conductor and protect it against corrosion.
- Route your equipotential bonding conductor to minimize the area between the equipotential bonding conductor and signal lines as far as possible (see the figure below).

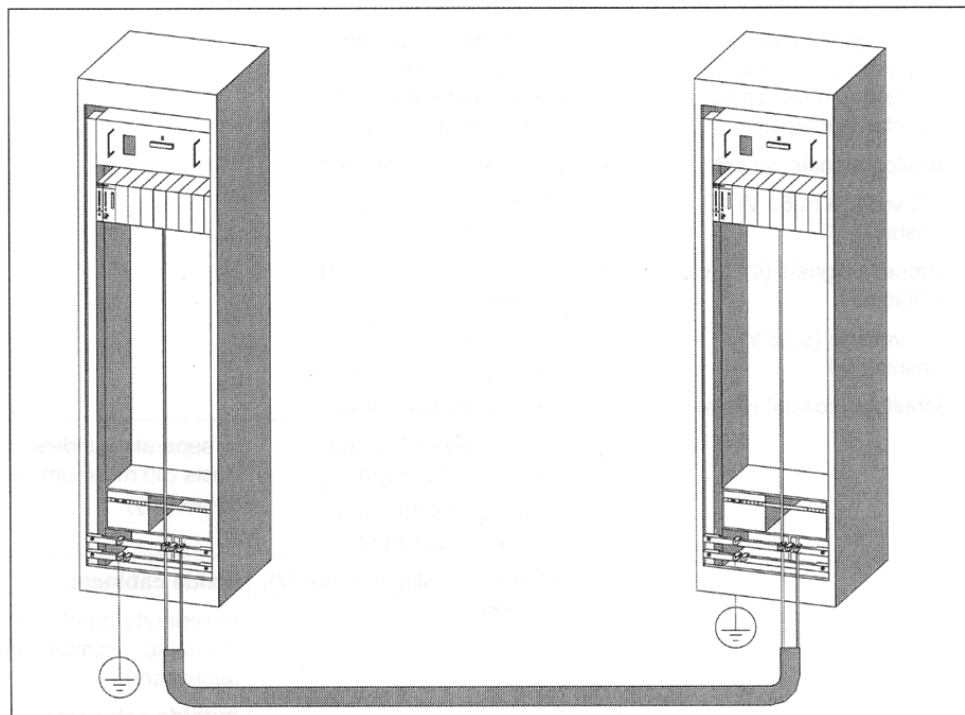


Figure 12-5 Equipotential bonding

12.2.7 Cable Routing inside Buildings

Introduction

Inside buildings (inside and outside cabinets), clearances must be maintained between groups of different cables to achieve the necessary electromagnetic compatibility (EMC). The table contains information on the general rules governing clearances to enable you to choose the right cables.

How to read the table

To find out how to run two cables of different types, proceed as follows:

1. Look up the type of the first cable in column 1 (Cables for ...).
2. Look up the type of the second cable in the corresponding field in column 2 (and cables for ...).
3. Note the applicable directives in column 3 (Run ...).

Table 12-7 Routing cables inside buildings

| Cables for ... | and cables for ... | Run ... |
|---|--|--|
| <ul style="list-style-type: none"> • Bus signals, shielded (for example, PROFIBUS, PROFINET) • Data signals, shielded (programming devices, operator panels, printers, counter inputs, etc.) • Analog signals, shielded • DC voltage (≤ 60 V), unshielded • Process signals (≤ 25 V), shielded • AC voltage (≤ 5 V), unshielded • Monitors (coaxial cable) | <ul style="list-style-type: none"> • Bus signals, shielded (for example, PROFIBUS, PROFINET) • Data signals, shielded (programming devices, operator panels, printers, counter inputs, etc.) • Analog signals, shielded • DC voltage (≤ 60 V), unshielded • Process signals (≤ 25 V), shielded • AC voltage (≤ 25 V), unshielded • Monitors (coaxial cable) | In common bundles or cable ducts |
| | <ul style="list-style-type: none"> • DC voltage (> 60 V and ≤ 400 V), unshielded • AC voltage (> 25 V and ≤ 400 V), unshielded | In separate bundles or cable ducts (no minimum clearance necessary) |
| | <ul style="list-style-type: none"> • DC and AC voltage (> 400 V), unshielded | Inside cabinets: In separate bundles or cable ducts (no minimum clearance necessary) Outside cabinets: On separate cable racks with a clearance of at least 10 cm |

| Cables for ... | and cables for ... | Run ... |
|--|---|--|
| <ul style="list-style-type: none"> • DC voltage ($> 60\text{ V}$ and $\leq 400\text{ V}$), unshielded • AC voltage ($> 25\text{ V}$ and $\leq 400\text{ V}$), unshielded | <ul style="list-style-type: none"> • Bus signals, shielded (for example, PROFIBUS, PROFINET) • Data signals, shielded (programming devices, operator panels, printers, counter inputs, etc.) • Analog signals, shielded • DC voltage ($\leq 60\text{ V}$), unshielded • Process signals ($\leq 25\text{ V}$), shielded • AC voltage ($\leq 25\text{ V}$), unshielded • Monitors (coaxial cable) | In separate bundles or cable ducts (no minimum clearance necessary) |
| | <ul style="list-style-type: none"> • DC voltage ($> 60\text{ V}$ and $\leq 400\text{ V}$), unshielded • AC voltage ($> 25\text{ V}$ and $\leq 400\text{ V}$), unshielded | In common bundles or cable ducts |
| | <ul style="list-style-type: none"> • DC and AC voltage ($> 400\text{ V}$), unshielded | Inside cabinets: In separate bundles or cable ducts (no minimum clearance necessary) Outside cabinets: On separate cable racks with a clearance of at least 10 cm |
| DC and AC voltage ($> 400\text{ V}$), unshielded | <ul style="list-style-type: none"> • Bus signals, shielded (e.g. PROFIBUS, PROFINET) • Data signals, shielded (programming devices, operator panels, printers, counter inputs, etc.) • Analog signals, shielded • DC voltage ($\leq 60\text{ V}$), unshielded • Process signals ($\leq 25\text{ V}$), shielded • AC voltage ($\leq 25\text{ V}$), unshielded • Monitors (coaxial cable) | Inside cabinets: In separate bundles or cable ducts (no minimum clearance necessary) Outside cabinets: On separate cable racks with a clearance of at least 10 cm |
| | <ul style="list-style-type: none"> • DC and AC voltage ($> 400\text{ V}$), unshielded | In common bundles or cable ducts |

12.2.8 Outdoor cable routing

Rules for EMC compatible cable routing

The same EMC compatibility rules apply both for indoor and outdoor routing of cables. The following also applies:

- Running cables on metal cable trays.
- Electrical connection of the joints of cable trays/ducts.
- Ground the cable carriers.
- If necessary, provide adequate equipotential bonding between connected devices.
- Take the necessary (internal and external) lightning protection and grounding measures in as far as they are applicable to your particular application.

Rules for lightning protection outside buildings

Run your cables either:

- in metal conduits grounded at both ends, or
- in concrete cable ducts with continuous end-to-end armoring.

Overvoltage protection equipment

An individual appraisal of the entire plant is necessary before any lightning protection measures are taken.

Additional information on lightning protection ...

can be found in the Lightning and surge voltage protection section.

Idriftsætning af PLC-styrede anlæg

Generelt

Idriftsætning af en maskine eller et procesanlæg kan være en meget kompliceret og tidskrævende fase, hvor teori og praksis skal afprøves. Det er på dette tidspunkt, det skal bevises, om idé og virkelighed passer sammen. Det er tidspunktet, hvor alle de komponenter, der indgår i maskinen/procesanlægget, skal vise, at de fungerer sammen. Det er også tidspunktet, hvor de forskellige styringer som pneumatik, hydraulik, elektronik, PLC-hard- og software samt kommunikationen mellem enhederne skal vise, at de kan fungere enkeltvis og som en helhed, der styrer/regulerer maskinen/processen.

Der er i denne fase følgende fejlmuligheder:

- Forrådningsfejl
- Konstruktionsfejl
- Manglende justeringer
- Komponentfejl
- Softwarefejl
- Dokumentationsfejl

Den følgende idriftsætningsvejledning vil omhandle de metoder og muligheder, der ligger i en systematisk idriftsætning af en PLC.

Vejledningen tager udgangspunkt i "DS_EN 60204-1 Kapitel 18 verifikation", hvor yderligere oplysninger kan læses, specielt mht. punkterne 18.2 til 18.5. Afsnittet angiver de almindelige bestemmelser for det elektriske materiel på maskiner.

Vejledningen er meget generel, så der skal altid tages udgangspunkt i det aktuelle anlæg. Afhængig af anlæggets opbygning kan der være flere eller færre komponenter, men rækkefølgen for idriftsætningen vil være den samme.

Pkt.1 Visuel kontrol / Synsprøve

- Forsyningsklemmer
- Jordtilslutninger
- Moduler korrekt monteret
- Relæer m.m.
- Frakobling af alle sikringer m.m.

Pkt. 2 DS/EN 60204-1 Kapitel 18 Verifikation

”DS_EN 60204-1 Kapitel 18 verifikation” giver en udførlig vejledning til udførelse af de nedenfor beskrevne punkter. Derfor er de kun meget kort nævnet her.

- Verificering af at det elektriske materiel er i overensstemmelse med den tekniske dokumentation.
- 18.2.2 Beskyttelseskredsens kontinuitet kontrolleres ved at sende en strøm mellem min. 0,2 A og ca. 10 A ved maksimalt 24 VAC eller DC (SELV) igennem kredsen. Herefter kan kredsens modstand beregnes og sammenlignes med den modstand, som må forventes ud fra kredsens længde og tværsnit.
- 18.2.3-4 Ved fejlbeskyttelse i form af automatisk afbrydelse af strømmen skal metoderne/kravene i afsnit 18.3 og 18,4 følges og opfyldes. Dette er ikke beskrevet i denne bog.
- 18.3 Måling af isolationsmodstanden. Ved 500 VDC mellem effektledere og beskyttelseskredsen: Min. 1MΩ.
- 18.4 Spændingsprøver med to gange mærkeværdien eller 1000 V ved 50 Hz. Komponenter, der ikke kan modstå prøvespændingen, skal frakobles.
- 18.5 Beskyttelse mod restspændinger, der skal aflades til max 60 V inden 5 sek.
- 18.6 Funktionsprøver af det elektriske materiel, specielt sikkerheds- og beskyttelsesfunktioner skal prøves.
- 18.7 Genprøvning, hvis en del af maskinen ændres eller modificeres.

En god metode til overholdelse af punkt 18.6 er ikke beskrevet i DS/EN 60204-1, så de næste punkter i dette afsnit er en vejledning i, hvordan dette kan gøres.

Pkt. 3 Test og indkobling af forsyninger til styreformål

- Før forsyningen indkobles, skal man sikre sig, at alt andet er frakoblet af hensyn til utilsigtet igangsætning
- Afbryd alle handleorganer vha. sikringer, motorværn mm.
- Indkobl spændingsforsyningen: Mål og kontroller polaritet, størrelse, ripple

Pkt. 4 Sikkerhedskreds

- Da sikkerhedskredse, f.eks. nødstop, fungerer uafhængigt af en PLC, kontrolleres disse kredse så tidligt som muligt i idriftsætningsfasen
- Nødstop, lågekontakter, hardware-endestop mm. kontrolleres

Pkt. 5 Fortrådning af indgange

- Kontroller vha. status-LED eller programmerings-PC alle tilsluttede indgange
- Foretag samtidig evt. justeringer af føler

Pkt. 6 Test og indkobling af udgangsforsyninger, effektforsyninger

- Kontrol af forsyningerne
- Indstil motorværn, minimalt tryk og hastighed på pneumatik og hydraulik.

Pkt. 7 Fortrådning og test af udgange

- Start med en kontrol af udgangsforsyningerne
- Ved hjælp af programmeringsenheden aktiveres de enkelte udgange. Anvend force-instruktioner eller fremstil et lille testprogram
- Først aktiveres de udgange, som sikrer energitilførslen, f.eks. hydraulikpumper, obs. mht. omløbsretning, oliestand mm.
- Kontroller motorers omløbsretninger/pneumatik/hydraulik/bevægelser mm.
- Når de enkelte udgange er aktiveret, kan mængde, tryk osv. indstilles
- Herefter afprøves de enkelte handleorganer mht. fortrådning, hastighed, kraft og slaglængde
- Da udgangene tvangsstyres, skal der udvises stor forsigtighed
- Aktiver en udgang ad gangen og vær klar ved nødstopet
- Hastigheder indstilles

Pkt. 8 Software downloades og afprøves

- Download PLC-program
- Modulafprøv softwaren og foretag evt. justeringer af kvitteringer, handlinger
- Start med at afprøve manuel drift

Pkt. 9 Optimering

- Optimering af anlægget mht. driftssikkerhed, betjening og kapacitet

Reparation og fejlfinding

Generelt

I forbindelse med vedligeholdelse af produktionsmaskiner udgør reparation og fejlfinding den mest krævende del af vedligeholdelsesarbejdet.

Dette skyldes primært, at enhver reparation er uforudsigelig både med hensyn til fejlomfang, fejlårsag, samt de forskellige måder en fejl kan optræde på.

I de ofte meget komplekse maskiner og anlæg foregår der et hurtigt og præcist samspil mellem:

- Mekanik
- Elektriske og elektroniske komponenter
- Computersystemer, der indeholder forskellige hard- og softwareelementer
- Regulatorer
- Avancerede motorer
- Hydraulik
- Pneumatik
- Bearbejdningssværktøjer, værktøjsmagasiner
- Emnegenkendelse, emnehåndtering
- Datakommunikation mellem enhederne

Derfor bør en person, der arbejder med reparation og vedligeholdelse af maskiner, være i besiddelse af følgende egenskaber:

- Et godt kendskab til førnævnte teknologier
- Være i stand til at bevare overblikket
- Være systematisk og konsekvent
- Være kritisk
- Være observerende og kunne tænke logisk

Derudover kræves der gode sprogkunderskaber og samarbejdsevner samt en ansvarlig og pålidelig optræden.

Systematisk, logisk fejlfinding

Systematisk, logisk fejlfinding er en kunst, som kræver megen erfaring. Men ved at lære nogle simple grundregler, får man et godt fundament.

Forudsætningen for at udføre en systematisk fejlfinding er:

- En god teoretisk baggrund
- Systematik
- Logisk konsekvens
- God måleteknik
- Diagram- og symbolkendskab
- En vis portion fantasi

En systematisk fejlfinding kan groft deles op i følgende tre faser:

- Observering
- Vurdering
- Handling

For at sikre et veludført job skal følgende forudsætninger være opfyldt:

- En udførlig fejlmelding
- Ajourført dokumentation
- Relevante måleinstrumenter
- Aktuell logbog

Fejlmelding

Eftersom operatøren har oplevet, hvorledes maskinen er stoppet, er disse iagttagelser et vigtigt grundlag for den videre fejlsøgning.

Operatøren kan udføre visuel kontrol, enkle test ved hjælp af sin erfaring og de til rådighedsstående midler, f.eks. betjeningsvejledning eller fejlmeldinger fra styringen.

For at vedligeholdelsespersonalet kan udføre fejlsøgningen effektivt, er det vigtigt med en nøjagtig fejlbeskrivelse udført af betjeningspersonalet.

En udførlig fejlmelding bør indeholde følgende hv-ord:

Hvem?

Navn
Afdeling
Funktion

Hvornår?

Dato
Tidspunkt

Hvor?

Fejlomfang
Område
Underdel af anlæg
Enkelt maskine

Hvad?

Hvilken funktion(er) svigter
Både i automatik og manuel?

Hvordan?

Optræder fejlen
Bestemte emner
Alle hastigheder
Bestemte ydre påvirkninger

Måleinstrumenter

Før hver eneste måling skal man gøre sig følgende punkter klart:

- Hvordan virker det foreliggende kredsløb?
- Hvilke instrumenter skal anvendes?
- Er der risiko for overbelastning og ødelæggelse af instrumentet?
- Hvorledes skal instrumentet tilsluttes for ikke at påvirke måleresultatet?
- Hvilke måleresultater forventes?
- Hvor nøjagtigt skal måleresultatet aflæses?

Til normal fejlfinding er et moderne universalinstrument som regel tilstrækkelig.

Medmindre det drejer sig om strømmålinger på transmittere eller evt. mindre motorstrømme, anvendes faktisk udelukkende spændingsmålinger.

Ohm-målinger bruges næsten kun til dobbelttjek af fejlårsagen, når komponenten er demonteret og dermed uden spænding.

Fejlomfang

Man indleder fejlfindingsopgaver ved at fastslå fejlomfanget, dvs. vurdere hvilke områder der er fejlramte.

Del styringen op i sammenhængende logiske grupper, hovedfunktioner/bifunktioner.

Kontroller hovedstrøm, styrestrøm inden der måles inde i styringen for derved at kontrollere instrument, forsyning samt forståelse af dokumentationen.

Er hele anlægget ude af drift, er alle motorerne i et afsnit af anlægget stoppet eller er det kun en enkelt?

Fejlmeldingen drejer sig oftest om en handling, der udebliver eller ikke stopper, altså startes der på udgangssiden af styringen.

Manuel/Automatik

Hvis der er mulighed for at koble anlægget i manuel drift, kontrolleres om den fejlbehæftede handling kan aktiveres/stoppes.

Er dette ikke tilfældet, fortsættes fejlfindingen i denne driftsform, og fejlen vil som regel ligge i selve udgangsdelen/forsyningen eller handleorganet.

Alt andet lige vil en fejlfindingssituation i manuel drift altid være væsentlig lettere at overskue sammenlignet med automatisk drift. Hvis anlægget fungerer i manuel drift, men svigter i automatisk drift, så vil fejlen som regel være et kvitteringssignal på indgangssiden til styringen.

Systematisk fejlfinding

Den normale procedure for en systematisk fejlfinding er:

Observer og analyser

- Observer fejlens optræden
- Analyser fejlårsager
- Anvend indbyggede diagnoseværktøjer

Overvej en strategi

- Overvej nøje en strategi til at fastslå, hvorledes observationer/analyserne kan bekræftes eller forkastes

Mål og vurder resultatet

- Anvend dokumentationen
- Foretag den ønskede måling
- Vurder kritisk måleresultatet

Fastslå fejlårsag

- Fejlfindingen er udført, når fejlårsagen med sikkerhed kan bestemmes
- Hvis muligt skal der foretages et dobbelttjek af den defekte komponent/modul
- Hvis muligt så bestem fejlårsagen for at imødegå en fremtidig gentagelse

Typiske fejlårsager er:

- Slitage
- Defekt komponent/modul
- Konstruktionsfejl
- Betjeningsfejl

Fejlfinding på sekventielle styringer

Hvis styringen af en maskine er udført sekventielt, vil der være mulighed for en systematisk/hurtig fejlfindingsprocedure.

Hvis maskinen stopper i et trin, er fejlen enten, at handlingen i det pågældende trin ikke er blevet udført, eller at kvitteringssignaler for handlingen ikke returneres til styringen.

En anden mulighed er, at et sekvenstrin springes over. Dvs. at handlingen i et trin ikke udføres, men styringen hopper over et trin. Denne fejl optræder, når kvitteringssignalet fra den ønskede handling er kortsluttet.

1. Bestem aktuelle trinnummer

Ved anvendelse af enten diagnosefunktioner, display, operatørpanel, PLC-testfunktioner eller vha. af maskinens dokumentation, f.eks. funktionsdiagram eller sekvensdiagram, bestemmes maskinens aktuelle trin.

2. Kontrol af handlinger i det aktuelle trin

På baggrund af dokumentationen skal handleorganerne i det aktuelle trin kontrolleres. Er handlingen sket? Hvis ja, gå da til punkt 3. Hvis nej, se da nedenfor:

- Kontrol af signalstatus på udgangskortet
- Kontrol af aktivering af kontaktor, ventil, tryk m.m.
- Kontrol af manglende bevægelighed pga. klemning, beskadigelse, støv, smøring, korrosion og lignende

3. Kontrol af kvitteringssignaler for det næste trin

Kvitteringssignalerne for handlingerne i det aktuelle trin kontrolleres på indgangskortet.

4. Kontrol af ind- og udgangsmoduler

Hvis ikke fejlen er lokaliseret under pkt. 1-3, er det mest sandsynligt, at fejlen ligger i et ind- eller udgangsmodul.

Ved hjælp af operatørpanelet eller en programmeringsenhed kontrolleres i statusmode, om de berørte indgangssignaler bliver overført til PLC'en. Hvis ikke skiftes det pågældende indgangsmodul.

På tilsvarende måde kontrolleres, at de interne udgangssignaler bliver overført til udgangsmodulet. Evt. tvangsstyres den aktuelle udgang, for at fastslå om udgangsmodulet er defekt.

Fejlårsager

Efter at fejlen er lokaliseret, skal årsagen til fejlen analyseres for at undgå, at samme type fejl optræder senere.

De mest typiske fejlårsager er:

Klima- og miljøårsager:

- Fugtighed
- Kondensvand
- Temperatur
- Støv
- Korrosion
- Olie
- Dampe
- Vibrationer

Fysiske/mekaniske årsager:

- Overbelastning
- Ældning, slaphed
- Slitage
- Støbefejl
- For stort tryk
- Beskadigelse

Elektriske årsager:

- Overbelastning
- Ældning
- Forkert tilslutning
- For høj strøm/spænding
- Nedsat isolationsevne
- Korrosion
- Erosion

Andre årsager:

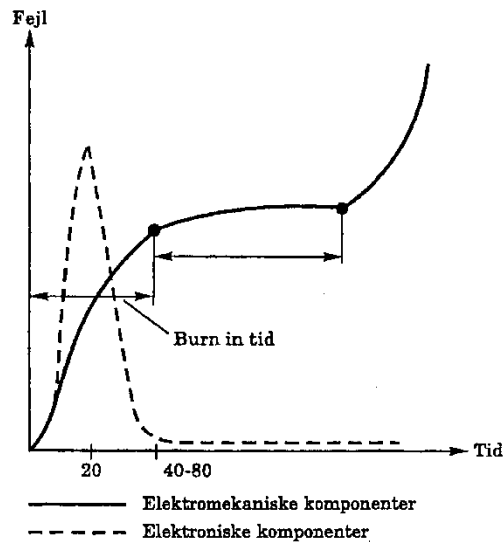
- Forkert dimensionering/indstilling/justering
- Dårlig kvalitet
- Forkert betjening
- Vold
- Påkørsel
- Hændeligt uheld

Ud fra konkluderede fejlårsager bør man finde frem til, på hvilken måde den netop fundne fejl kan undgås i fremtiden, og fejlen bør beskrives i en driftsjournal, logbog eller lignende. Derved sikres, at der ved gentagne fejl på samme komponent med tilsvarende årsag kan iværksættes foranstaltninger, som reducerer fremtidige fejlretninger.

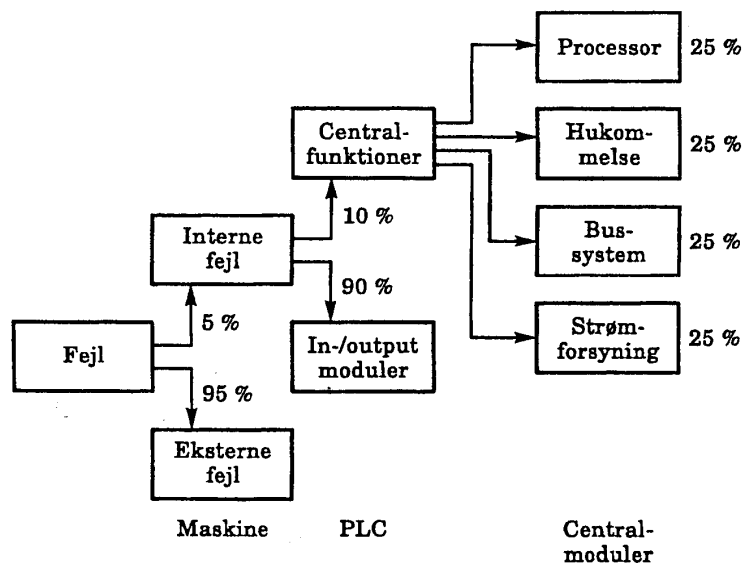
Fejlstatistik

Med til vurderingen hører også begrebet fejlhyppighed, dvs. hvor stor er sandsynligheden for, at lige denne komponenttype er fejlbehæftet?

Nedenstående kurve viser fejlhyppigheden for elektromekaniske og elektroniske komponenter som en funktion af tiden.



På baggrund af ovenstående kan den nedenstående grafiske fremstilling af sandsynligheden for en PLC-fejl på et PLC-styret anlæg ikke overraske.



Fejlfindingens 10 bud

1. Optræd altid velovervejet og konsekvent
2. Udfør kun strengt logiske handlinger
3. Husk altid rækkefølgen: Observering, vurdering og derefter handling
4. Udnyt altid de indbyggede faciliteter maksimalt
5. Gå fra grove analyser mod fine analyser
6. Hvis begyndt, så fortsæt i samme retning, indtil analysen helt sikkert kan bekræftes/forkastes
7. Udfør aldrig to procedurer samtidigt
8. Vær meget omhyggelig for at undgå følgeskader
9. Vær meget kritisk mht. vurdering af måleresultater
10. Overhold altid de gældende sikkerhedsbestemmelser

Sekvenseksempel på Allen-Bradley

Sekvensteknik

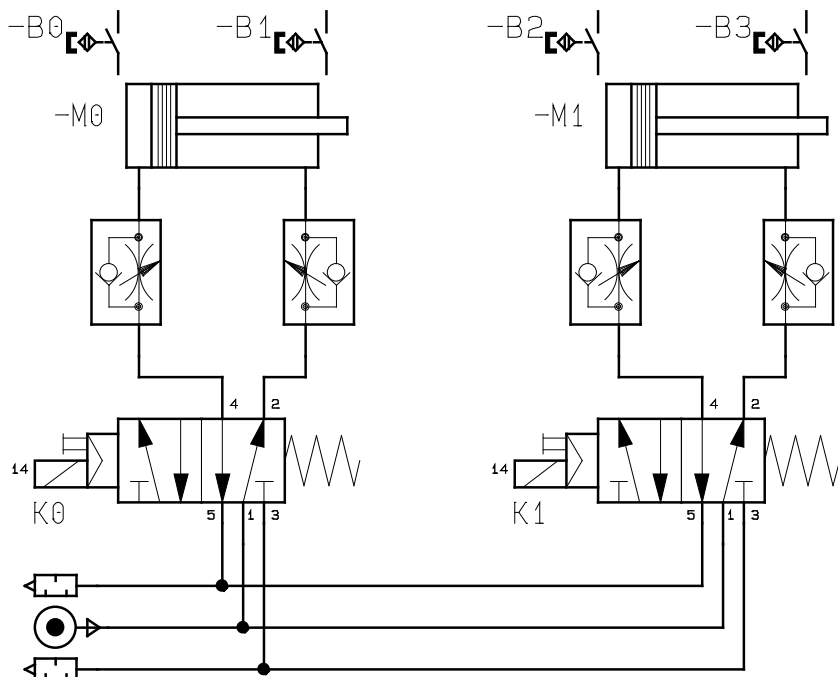
I automatiske anlæg, hvor de forskellige handlinger sker i en fast sekvens (rækkefølge), kan styringen udarbejdes systematisk. Dette har den fordel, at styringen udarbejdes efter en helt bestemt fremgangsmåde og derfor er mere overskuelig og lettere at lave systematisk fejlfinding på.

Sekvenskæden bygges normalt op af de interne hjælperelæer i PLC'en. I nogle PLC'er findes der specielle instruktioner (step) til denne opbygning af sekvenskæder.

Når man programmerer en sekvenskæde i en PLC, er fremgangsmåden, at et trin resetter det foregående og forbereder det næste trin.

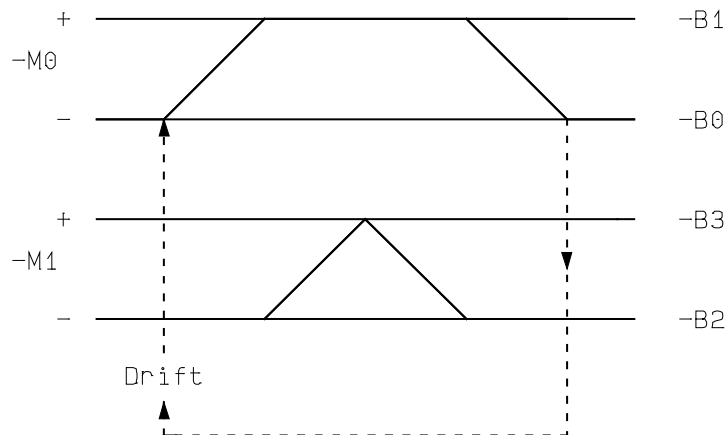
Det følgende er et eksempel på opbygning af en sekvenskæde.

Styringseksempel



PLC-styringen skal styre 2 cylindre, der anvendes som en del af et større maskinanlæg.

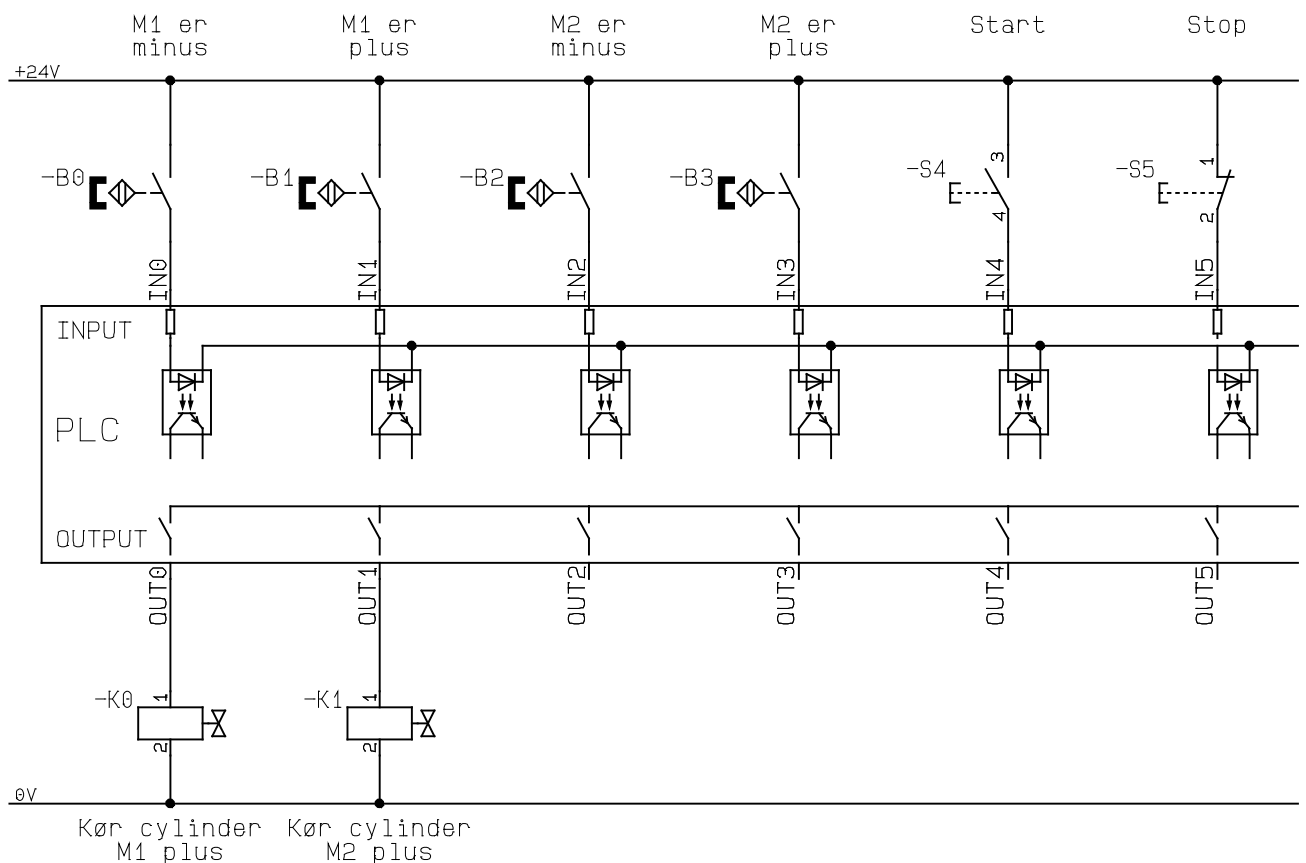
Funktionsdiagram



Diagrammet viser sekvensen for de to cylindre.

Sekvensen fortsætter med at køre, indtil drift ikke er aktiv mere, herefter stopper sekvensen i udgangsposition.

PLC-diagram



På PLC'en anvendes følgende tilslutningsdiagram.

Controller Tag Tabel

| Controller Tags - PLC_bog(controller) | | | | |
|---------------------------------------|------------------|------------------|-----------|--|
| Scope: PLC_bog | | Show: All Tags | | Enter Name |
| Name | Alias For | Base Tag | Data Type | Description |
| B0_M1_er_minus | Local:1:I.Data.0 | Local:1:I.Data.0 | BOOL | Reed foeler på cylinder NO |
| B1_M1_er_plus | Local:1:I.Data.1 | Local:1:I.Data.1 | BOOL | Reed foeler på cylinder NO |
| B2_M2_er_minus | Local:1:I.Data.2 | Local:1:I.Data.2 | BOOL | Reed foeler på cylinder NO |
| B3_M2_er_plus | Local:1:I.Data.3 | Local:1:I.Data.3 | BOOL | Reed foeler på cylinder NO |
| Drift | | | BOOL | |
| K0_Koer_Cylinder_M1_plus | Local:4:O.Data.0 | Local:4:O.Data.0 | BOOL | Tænd spole på monostabil ventil |
| K1_Koer_Cylinder_M2_plus | Local:4:O.Data.1 | Local:4:O.Data.1 | BOOL | Tænd spole på monostabil ventil |
| S4_Start | Local:1:I.Data.4 | Local:1:I.Data.4 | BOOL | Impulstryk knap NO |
| S5_Stop | Local:1:I.Data.5 | Local:1:I.Data.5 | BOOL | Impulstryk knap NC |
| Udgangs_Position | | | BOOL | |
| Trin | | | INT | |
| Trin.0 | | | BOOL | Initialtrin |
| Trin.1 | | | BOOL | Kør cylinder M1 plus |
| Trin.2 | | | BOOL | Kør cylinder M2 plus og hold cylinder M1 plus |
| Trin.3 | | | BOOL | Kør cylinder M2 minus og hold cylinder M1 plus |
| Trin.4 | | | BOOL | Kør cylinder M2 minus |

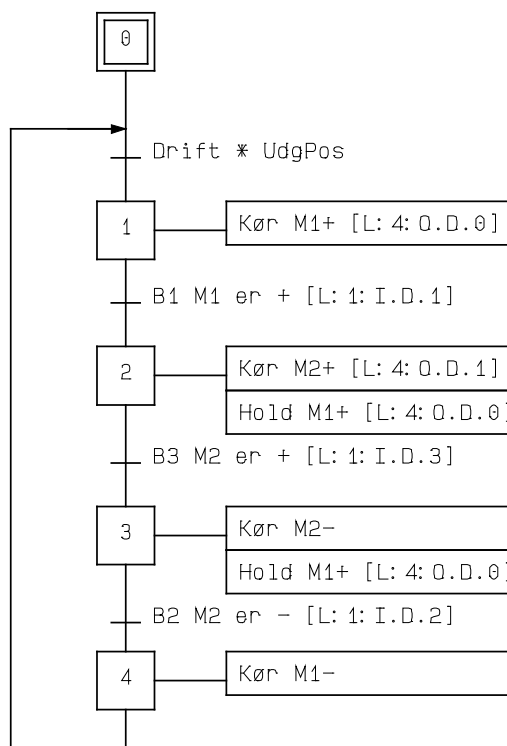
I PLC-programmet anvendes følgende tag.

EN 60848

Generelt

For at dokumentere en sekventiel styring anvendes et sekvensdiagram i henhold til EN 60848. Denne standard er også godkendt af Dansk Standard.

Sekvensdiagram



Beskrivelse af sekvens

Sekvensen er delt op i 5 trin, hvor trin 0 er et Initialiseringstrin, hvor sekvensen er i udgangsstilling og afventer startsignal fra S4.

Betingelserne for at aktivere trin 1 er, at trin 0 samt drift og udgangsposition er "on". Drift er aktiv, når der trykkes start og indtil, der trykkes stop. Udgangsposition er de to følere B0 og B2, som skal være aktive, når cylinderne er i deres hvileposition.

Når trin 1 bliver aktiv, sendes cylinder M1 i plus.

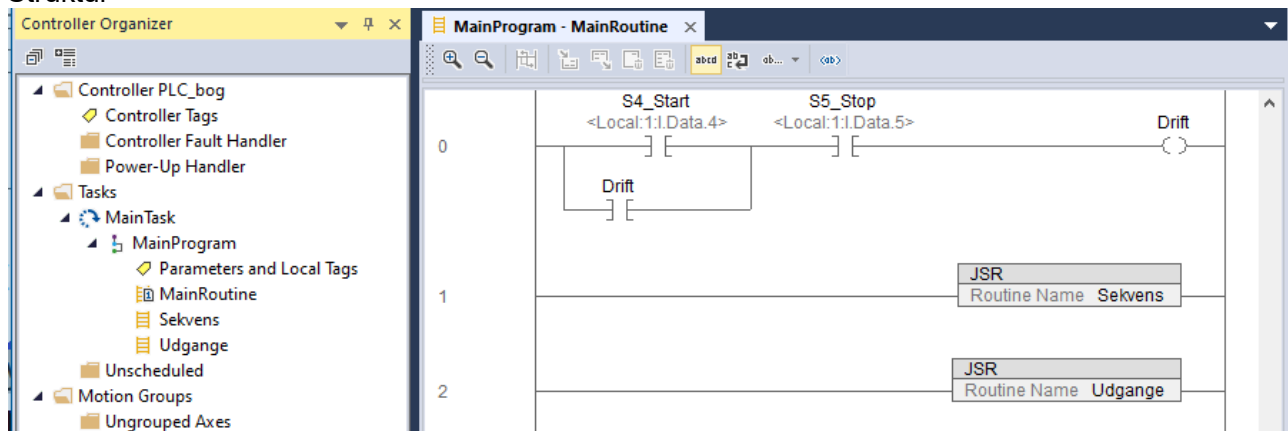
Når cylinder M1 er i plus, aktiveres føler B1, hvorefter trin 2 bliver aktiv. I trin 2 sendes cylinder M2 i plus.

Når cylinder M2 er i plus, aktiveres føler B3, hvorefter trin 3 bliver aktiv. I trin 3 sendes cylinder M2 i minus.

Når cylinder M2 er i minus, aktiveres føler B2, hvorefter trin 4 bliver aktiv. I trin 4 sendes cylinder M1 i minus.

Når cylinder M1 er i minus, aktiveres føler B0, hvorefter trin 1 bliver aktiv, og sekvensen starter forfra igen.

PLC-program Struktur



I programeksemplet er der anvendt følgende programfiler for at skabe overblik:

MainRoutine:

Denne blok læses automatisk af PLC'en (cyklisk programafvikling), og her er instruktioner om at læse resten af programmet, samt fælles startbetingelser.

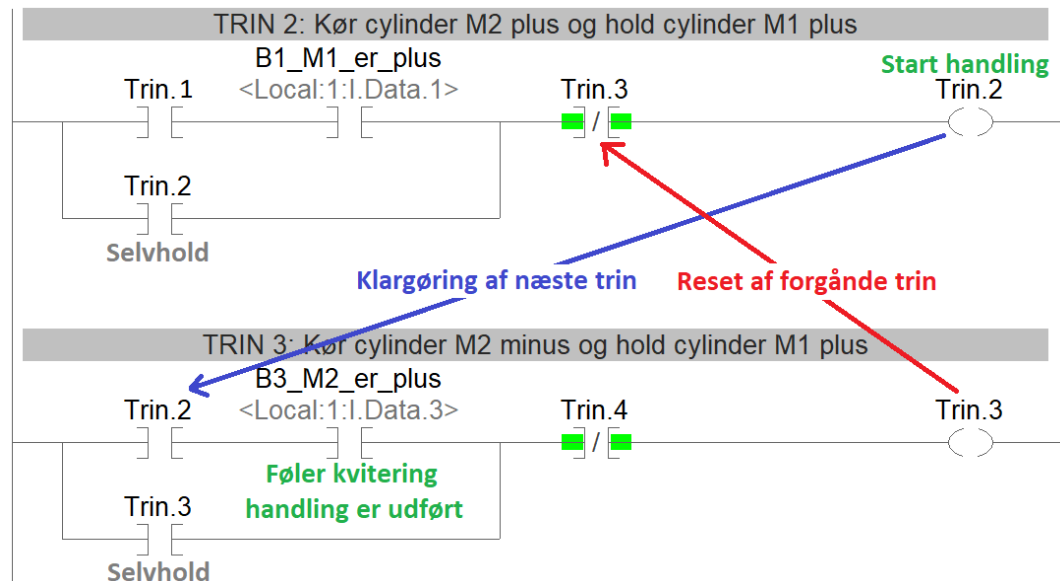
Sekvens:

Programfile for sekvens.

Udgange:

Programfile for aktivering af udgange.

Trin-opbygning

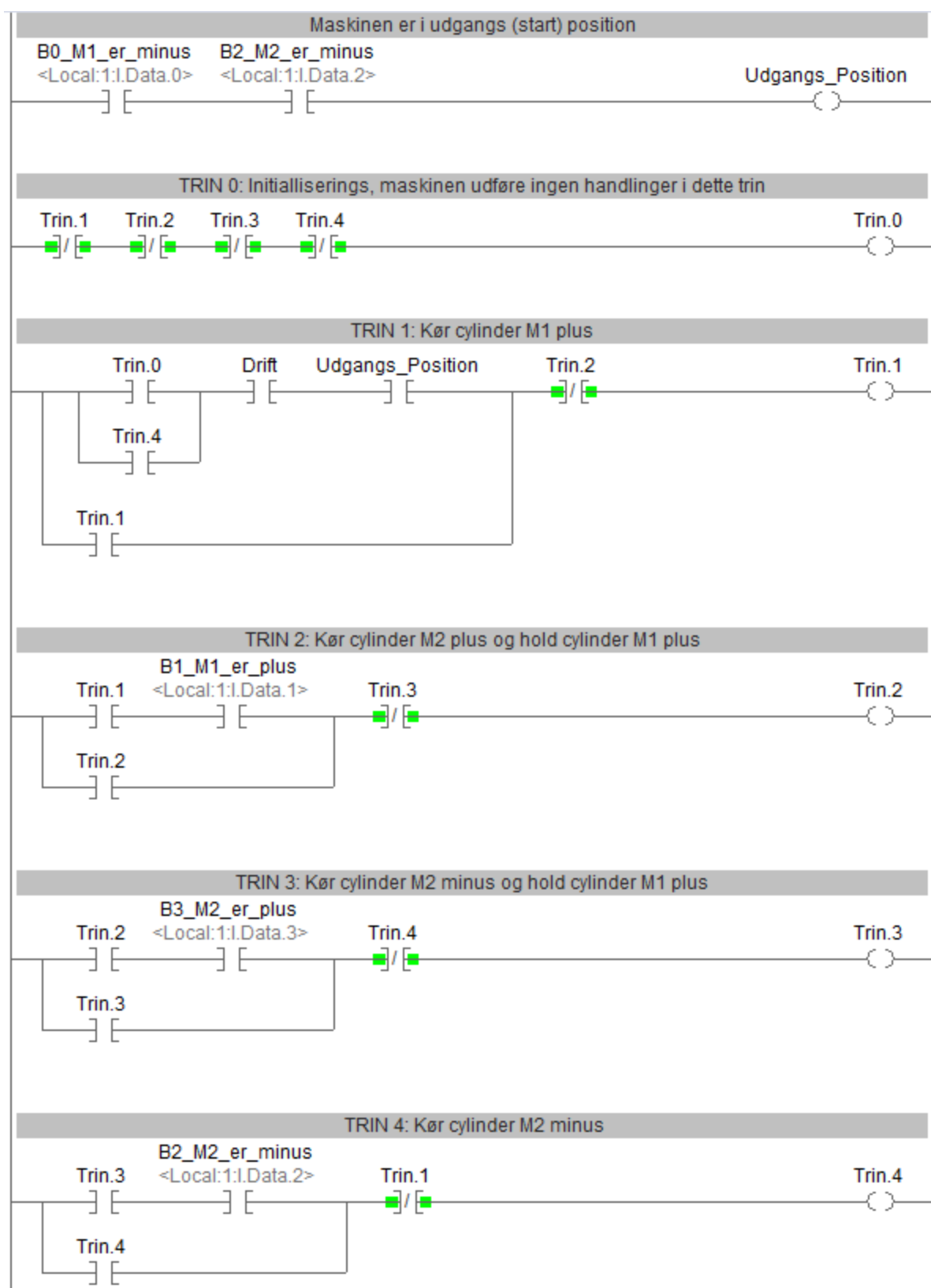


En sekvenskæde programmeres ud fra sekvensdiagrammet og består af et antal trin. Disse sekvenstrin skal programmeres med selvhold.

Princippet for opbygning af en sekvensstyring er:

- TRIN 2: Udfører en handling
(I dette tilfælde er det at tænde en udgang, som får en cylinder (M2) til at køre plus. Denne funktion er programmeret i programfilen *udgange*)
- TRIN 2: Forbereder det næste trin, Trin 3.
- TRIN 3: Går "on", når der kvitteres for, at handlingen er udført.
(Dette gøres ved, at cylinderen aktiverer en føler, her er det -B3)
- TRIN 3: Resette det foregående trin, Trin 2
- Der kan i en enkelt sekvenskæde aldrig være mere end et trin "ON" ad gangen
- Dette system kan fortsættes i det uendelige

Sekvensprogram



Udgangsposition

Det er altid en god ide at kontrollere, at maskinen er i udgangsstilling, inden den startes op. Derfor er der lavet et tag, der hedder `Udgangs_Position`, som kun er aktivt, når alle cylindre er i deres hjemmestilling. I dette tilfælde minus.

Initialiseringstrin

Initialiseringstrinnet er et starttrin, og det er fra dette "hvile" trin, at sekvensen startes. Når sekvensen er færdig, vil den stå i trin 4.

Det vil sige, at initialiseringstrinnet kun er aktivt ved nystart af PLC'en. Efter start vil sekvensen stå i trin 4.

Initialiseringstrinnet bruges bl.a. til at spærre for uønsket genstart, når der køres fortsat sekvens, og sekvensen er opbygget sådan, at maskinen kommer i udgangsposition undervejs.

Initialiseringstrinnet kan laves på mange måder, og i afsnittet Sekvens Siemens med Flip-Flop er der vist et andet eksempel.

Fordelen ved at lave initialiseringstrinnet på den viste måde er, at trinnet også kan bruges ved fejlfinding til at danne sig et hurtigt overblik over, hvor i sekvensen maskinen er gået i stå, da lige netop dette trin vil stå aktivt.

Trin1

Trin 1 adskiller sig lidt fra de øvrige trin, da det både skal kunne startes, når sekvensen ved førstegangstopstart står i Trin 0, eller når sekvensen har kørt og står i Trin 4.

Trin 1 er også det trin, der resetter Trin 4, da det jo er næste trin i forhold til Trin 4.

Fortsat sekvens

Hvis man ønsker, at maskinen skal fortsætte med at køre sin sekvens i en uendelighed, efter der er trykket start, og indtil der trykkes stop, er der brug for et drift-tag, der kan huske, at start har været påvirket, og som først går off, når der trykkes stop.

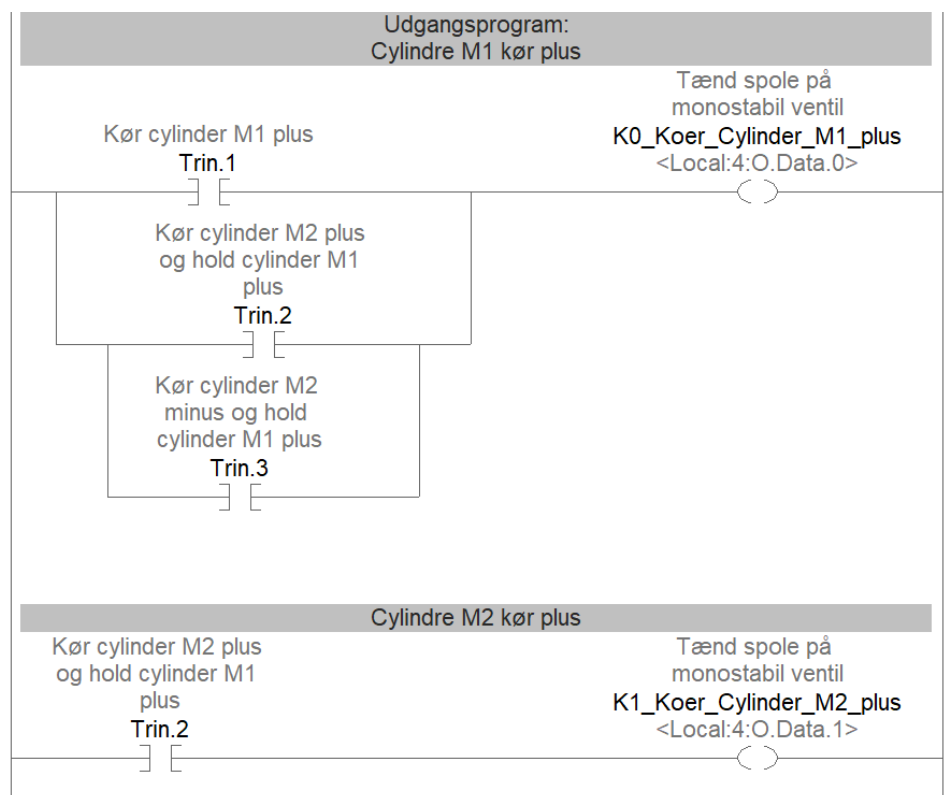
Ønskes der kun et gennemløb af sekvensen, skal drift-tagget skiftes ud med startknappen (S4).

Udgangsprogram

Ved at programmere udgangsprogrammet sidst i PLC-programmet opnås et godt overblik over hvilke trin, der styrer den enkelte udgang.

Endvidere bliver det forholdsvis simpelt at udvide med flere sekvenser.

Et senere ønske om at tilføje en manual betjening til maskinen klares ved at programmere de manuelle flag parallelt over de enkelte trin-flag i udgangsprogrammet.

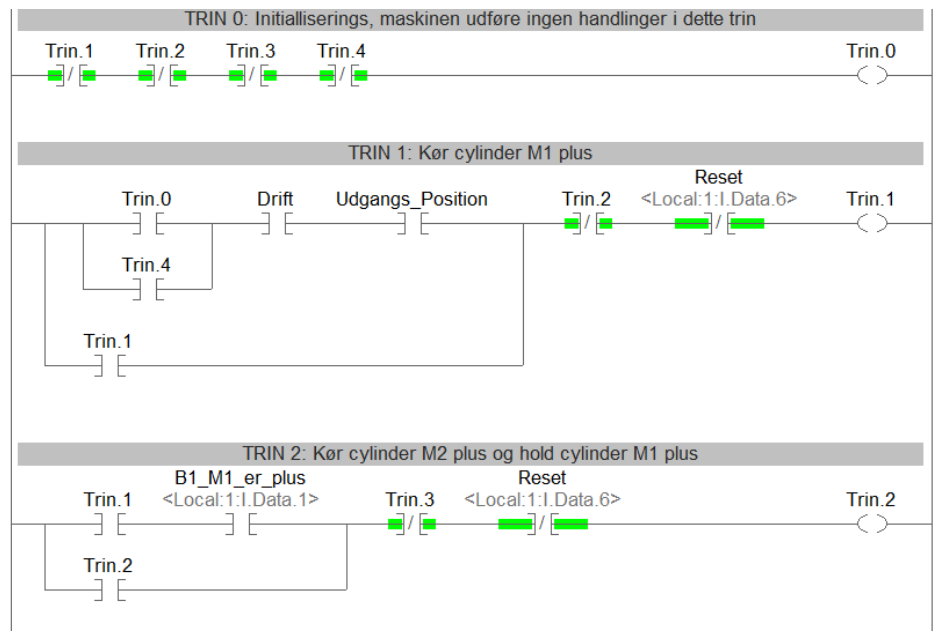


Der anvendes pneumatiske ventiler med fjeder-retur, også kaldet monostabile ventiler, og derfor holdes signalet på K0 i trin 1, 2 og 3, mens K1 kun får signal i trin 2.

Reset

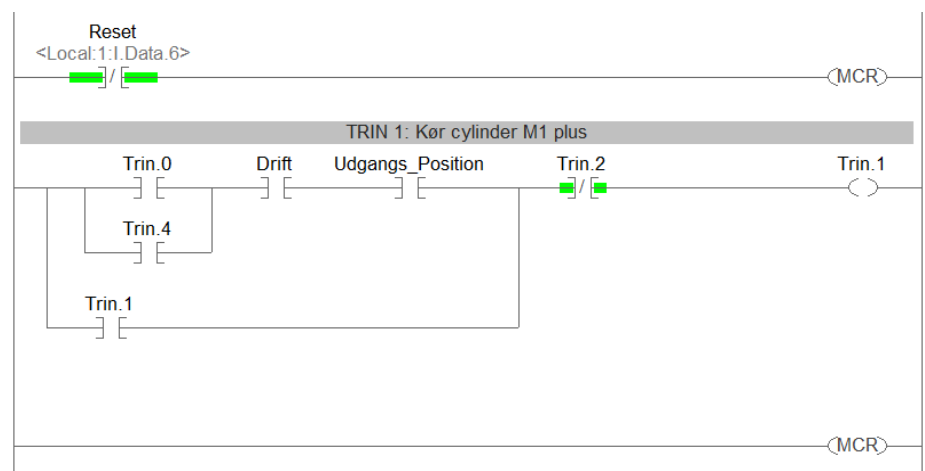
Reset benyttes, hvor det ønskes at resette (nulstille) sekvensen, f.eks. efter at nødstop har været aktiv.

Når reset aktiveres, skal det fungere således, at uanset hvilket trin sekvenskæden befinder sig i, så nulstilles sekvensen fra trin 1 til sidste trin. For overskuelighedens skyld er der kun medtaget trin 0, trin 1 og trin 2 i dette eksempel:



Reset kan også laves ved hjælp af instruktionen Master Control Reset (MCR).

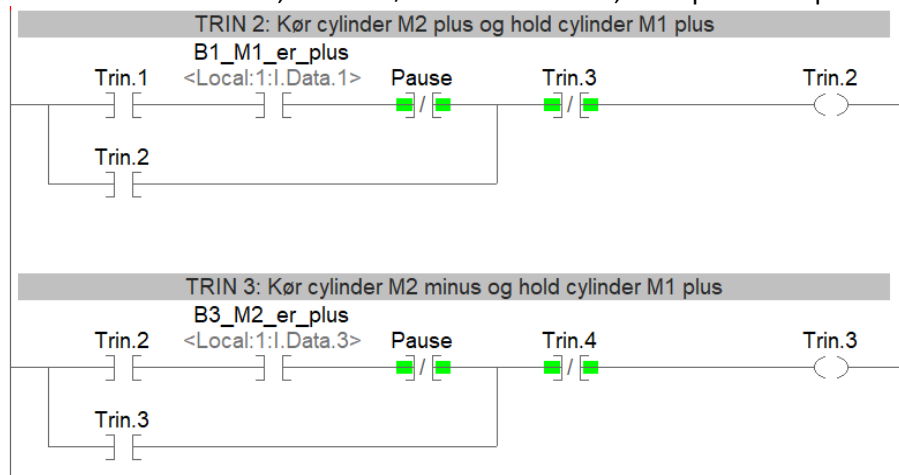
Når Reset-tagget går off, vil MCR slukke alle udgange imellem de to MCR-instruktioner. I dette eksempel er det kun trin 1, der slukkes. Denne metode er lettere at programmere, men kan være svære at overskue ved fejlfinding, specielt i store programmer.



Pause

Ønskes det, at sekvensen skal stoppe, efter en igangsat handling er udført og ikke fortsætte til næste trin, altså en pausefunktion, kan det gøres på følgende måde:

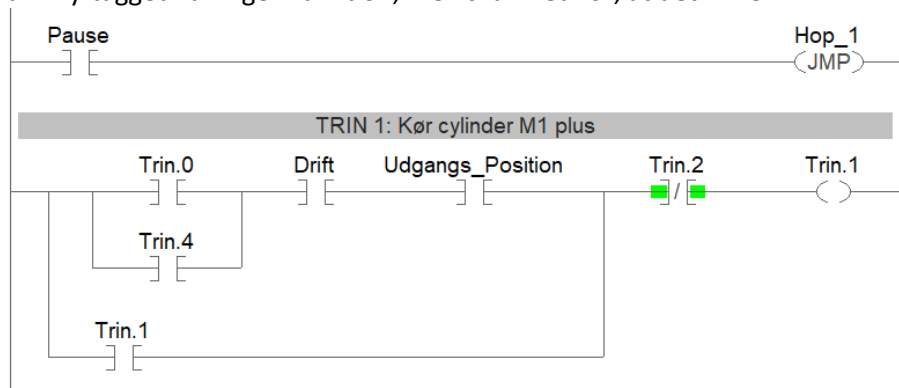
Hvis trin 2 er aktiv, når pause aktiveres, vil dette trin forblive aktivt. Trin 3 vil ikke kunne blive aktiv, selvom føler B3 bliver aktiv, fordi pause vil spærre.



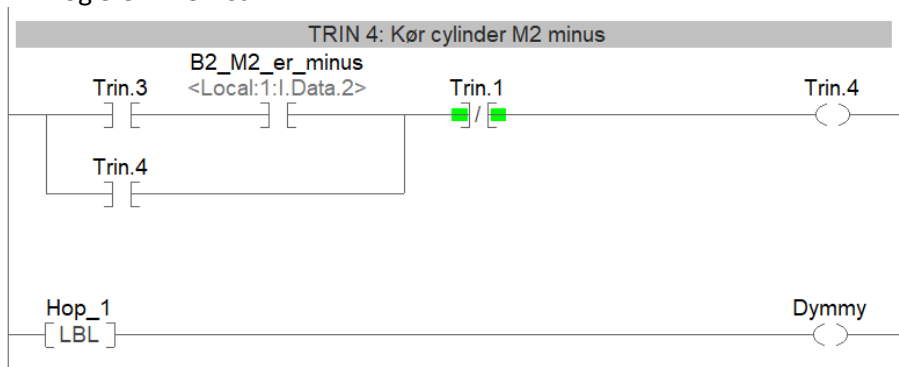
Pause kan også laves med instruktionerne Jump to Label (JMP) og Label (LBL).

Når JMP aktiveres stopper PLC'en med at læse og opdaterer programmet mellem JMP og LBL.

Dummy-tagget har ingen funktion, men skal med for, at det virker.



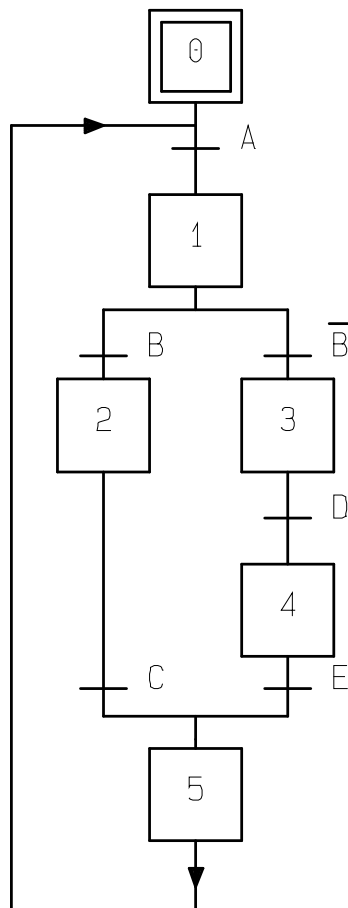
Trin 2 og 3 er ikke vist



Sekvens Eller-delning på Allen-Bradley

Hvis det ønskes at dele sekvenskæden, således at der er mulighed for at vælge imellem at køre to forskellige forløb, altså enten forløb 1 **eller** forløb 2, kan det gøres således:

Sekvensdiagram



Betingelser for deling:

Når kæden står i trin 1, kan den dele sig og køre enten trin 2, hvis B er on eller trin 3, hvis B er off.

Bemærk, at det, ved at bruge B on/off som startbetingelse for de to kæder, er sikret, at kun en kæde kan starte ad gangen. B kan jo ikke både være on og off samtidigt.

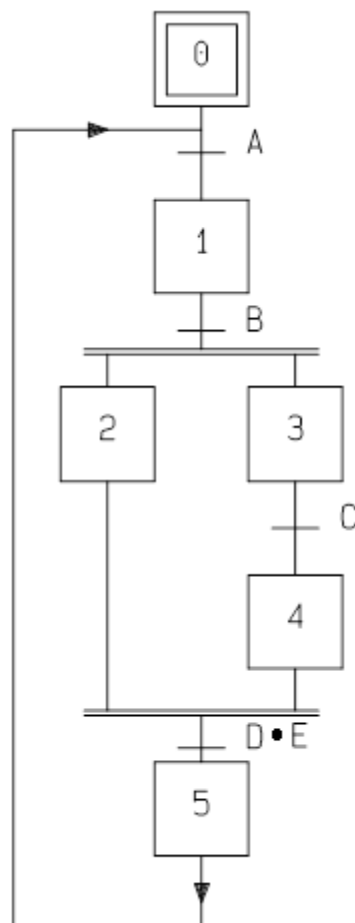
Som det fremgår af sekvensdiagrammet, skal trin 1 klargøre både trin 2 og trin 3. Trin 1 skal resettes af enten trin 2 **eller** trin 3.

I trin 5 samles kæden igen, og trin 5 skal derfor klargøres af enten trin 2 og C **eller** trin 4 og E. Trin 5 skal resette både trin 2 og trin 4.

Sekvens Og-delning på Allen-Bradley

Hvis det ønskes at dele sekvenskæden, således at der er mulighed for at køre to forskellige forløb samtidigt, altså så både forløb 1 **og** forløb 2 kan køre på en gang, gøres det således:

Sekvensdiagram

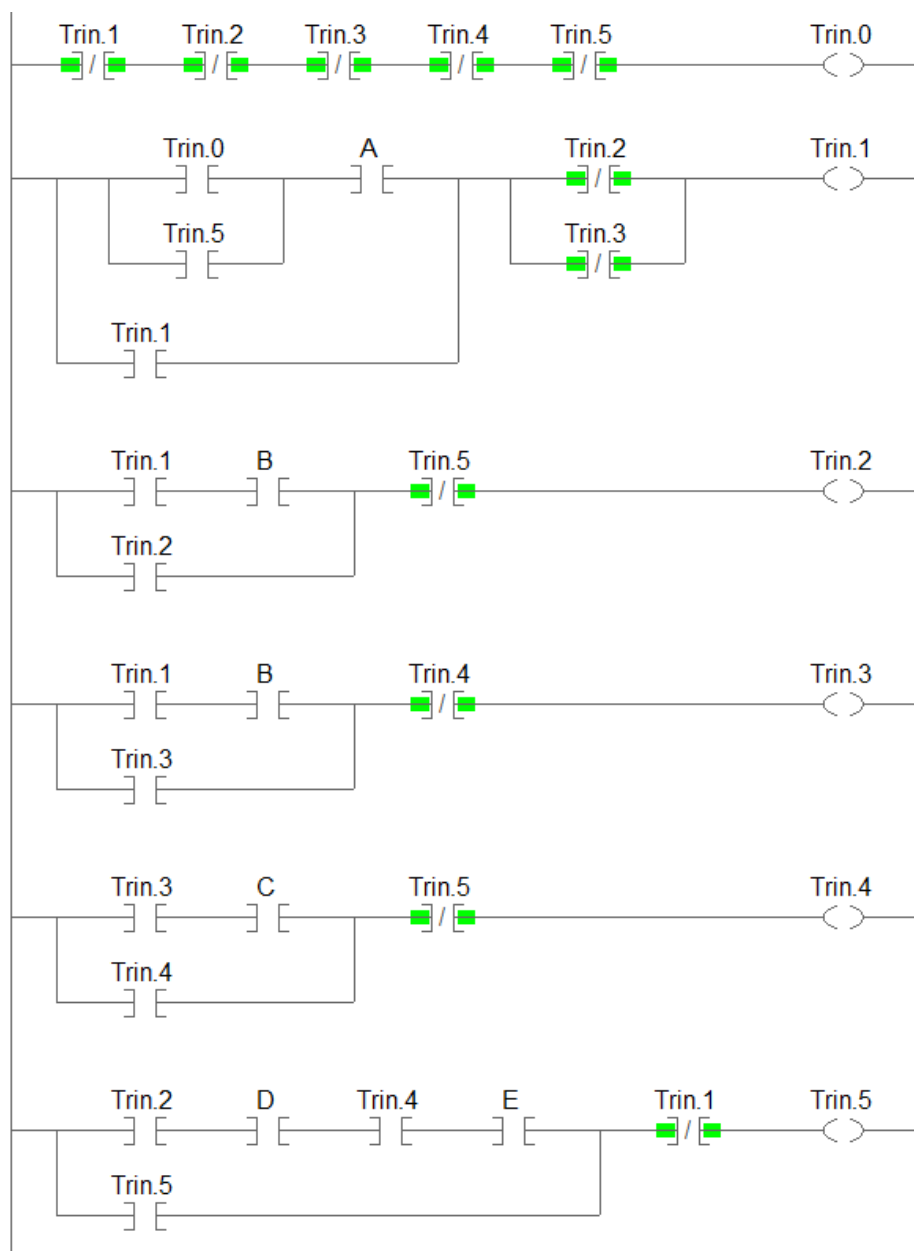


Betingelser for deling:

Når kæden står i trin 1, kan den dele sig og køre både trin 2 **og** trin 3, hvis B er on.

Bemærk, at der ved at bruge B on som startbetingelse for de to kæder er sikret, at begge kæder starter samtidig.

PLC-program



Deling af kæden:

Som det fremgår af sekvensdiagrammet, skal trin 1 klargøre både trin 2 og trin 3. Trin 1 skal først resettes, når trin 2 **og** trin 3 begge er on.

Samling af kæden:

I trin 5 samles kæden igen. Trin 5 skal derfor klargøres af både trin 2 og trin 4 samt betingelse D og E.
Trin 5 skal resette både trin 2 og trin 4.

Sekvens Siemens med flip-flop

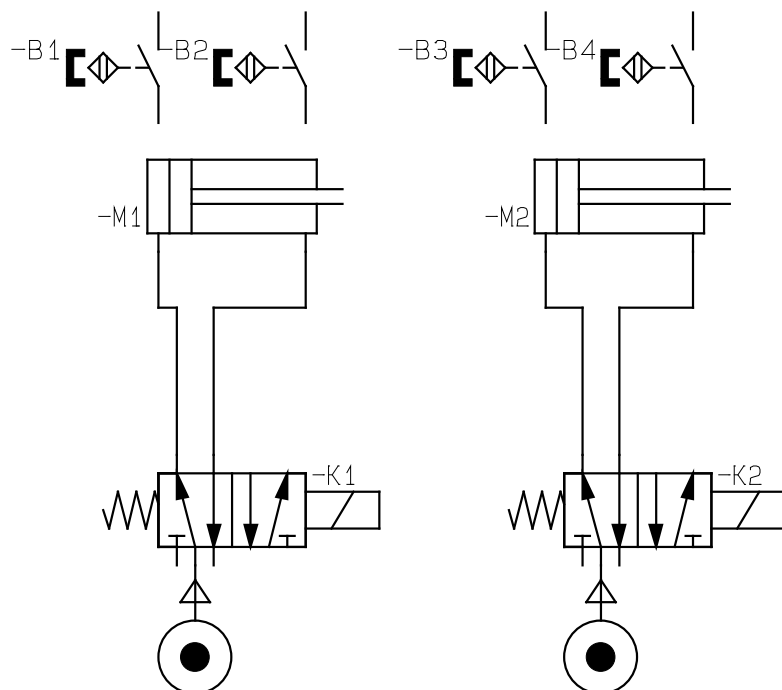
Sekvensteknik

I automatiske anlæg, hvor de forskellige handlinger sker i en fast sekvens (rækkefølge), kan styringen udarbejdes systematisk. Dette har den fordel, at styringen udarbejdes efter en helt bestemt fremgangsmåde og derfor er mere overskuelig og lettere at lave systematisk fejlfinding på.

Sekvenskæden bygges normalt op af de interne hjælperelæer i PLC'en. I nogle PLC'er findes der specielle instruktioner (step) til denne opbygning af sekvenskæder. Når man programmerer en sekvenskæde i en PLC, er fremgangsmåden, at et trin resetter det foregående og forbereder det næste trin.

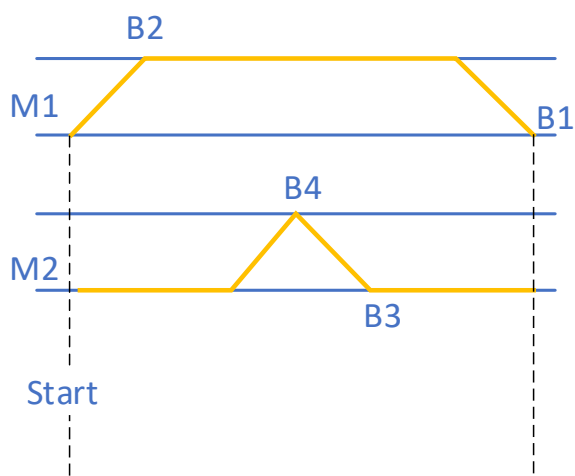
Det efterfølgende er et eksempel på opbygning af en sekvenskæde med flip-flop.

Styringseksempel



PLC-styringen skal styre 2 cylindre, der anvendes som en del af et større maskinanlæg.

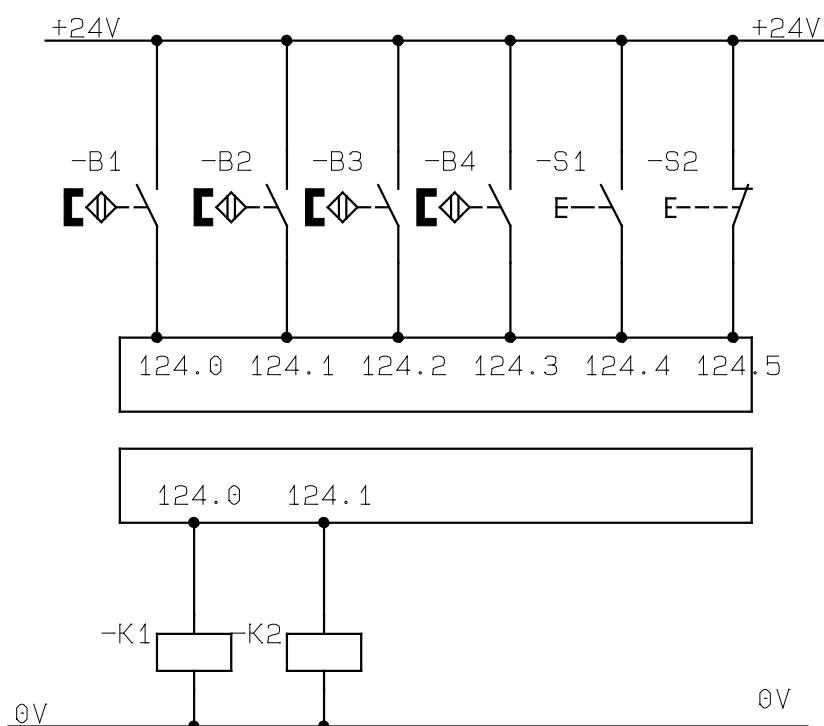
Funktionsdiagram



Diagrammet viser sekvensen for de to cylindre.

Sekvensen fortsætter med at køre, indtil start ikke er aktiv mere, herefter stopper sekvensen i udgangsposition.









PLC-diagram



På PLC'en anvendes følgende tilslutningsdiagram.

PLC-tags

I PLC-programmet anvendes følgende PLC-tags:

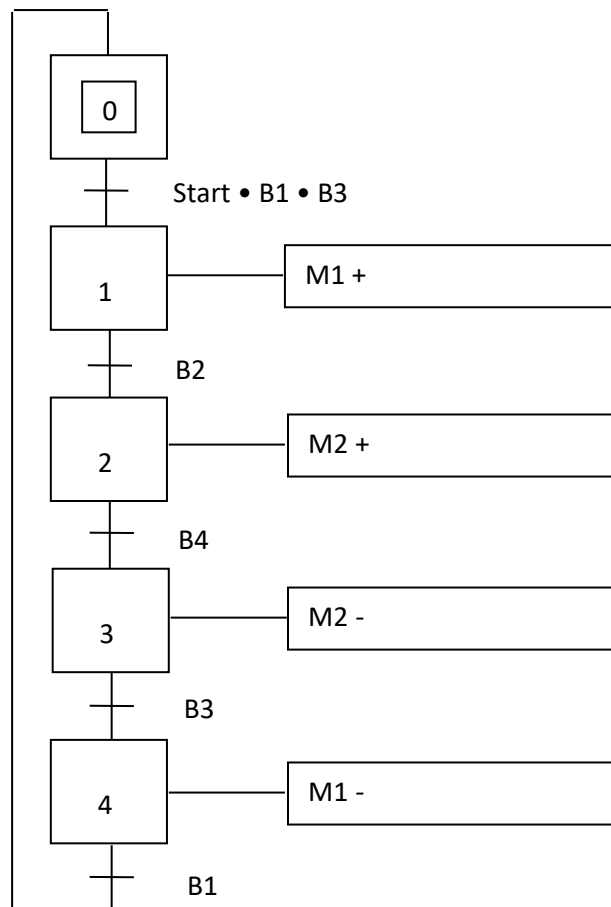
| | Name | Tag table | Data type | Address |
|---|-----------|-------------------|-----------|---------|
|  | B1 | Default tag table | Bool | %I124.0 |
|  | B2 | Default tag table | Bool | %I124.1 |
|  | B3 | Default tag table | Bool | %I124.2 |
|  | B4 | Default tag table | Bool | %I124.3 |
|  | S1 | Default tag table | Bool | %I124.4 |
|  | S2 | Default tag table | Bool | %I124.5 |
|  | K1 | Default tag table | Bool | %Q124.0 |
|  | K2 | Default tag table | Bool | %Q124.1 |
| | <Add new> | | | |

EN 60848

Generelt

For at dokumentere en sekventiel styring anvendes et sekvensdiagram i henhold til EN 60848. Denne standard er også godkendt af Dansk Standard.

Styringseksempel



Beskrivelse af sekvens

Sekvensen er delt op i 5 trin, hvor trin 0 er et initialiseringstrin, hvor sekvensen er i udgangsstilling og afventer startsignal fra Start.

Betingelserne for at aktivere trin 1 er, at trin 0 samt Start, B1 og B3 er aktive.

Når trin 1 bliver aktiv, sendes M1 i plus.

Når cylinder M1 er i plus, aktiveres B2, hvorefter trin 2 bliver aktiv. I trin 2 sendes M2 i plus.

Når cylinder M2 er i plus, aktiveres B4, hvorefter trin 3 bliver aktiv. I trin 3 sendes M2 i minus.

Når cylinder M2 er i minus, aktiveres B3, hvorefter trin 4 bliver aktiv. I trin 4 sendes M1 i minus.

Når cylinder M1 er i minus, aktiveres B1, hvorefter trin 0 bliver aktiv, og sekvensen er klar til opstart igen.

Henvisning

Der findes en standard for PLC-sproget til sekvensprogrammering. Den standard benævnes IEC 61131-3, og du kan læse mere om standarden på www.plcopen.org

PLC-program

Struktur

I programeksemplet er der anvendt følgende struktur:

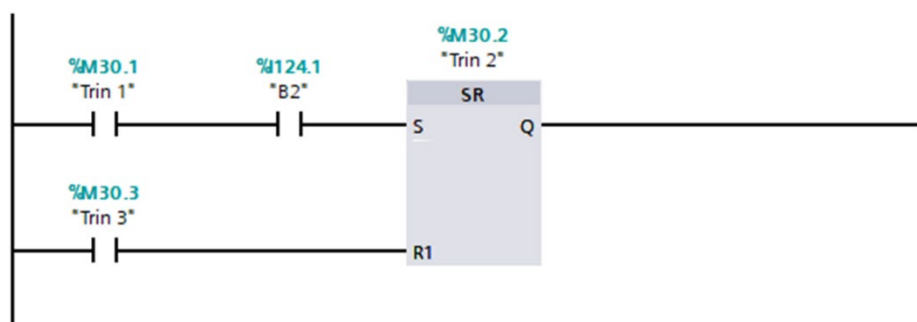
| Call structure of PLC_1 | | |
|-------------------------|---|-----------------|
| Call structure | ! | Address |
| ▼ Main | | OB1 |
| Block_1 | | FC1 @Main ► NW1 |
| Block_2 | | FC2 @Main ► NW2 |

OB 1 = Cyklisk programbearbejdning

FC 1 = Program for sekvens

FC 2 = Program for udgangsmatrix

Flip-flop



Et sekvenstrin programmeres ofte med flip-flop. På set-indgangen "S" tilføjes de betingelser, der skal være til stede for, at trinnet aktiveres. Typisk er betingelserne det foregående trin samt relevante signaler fra indgange.

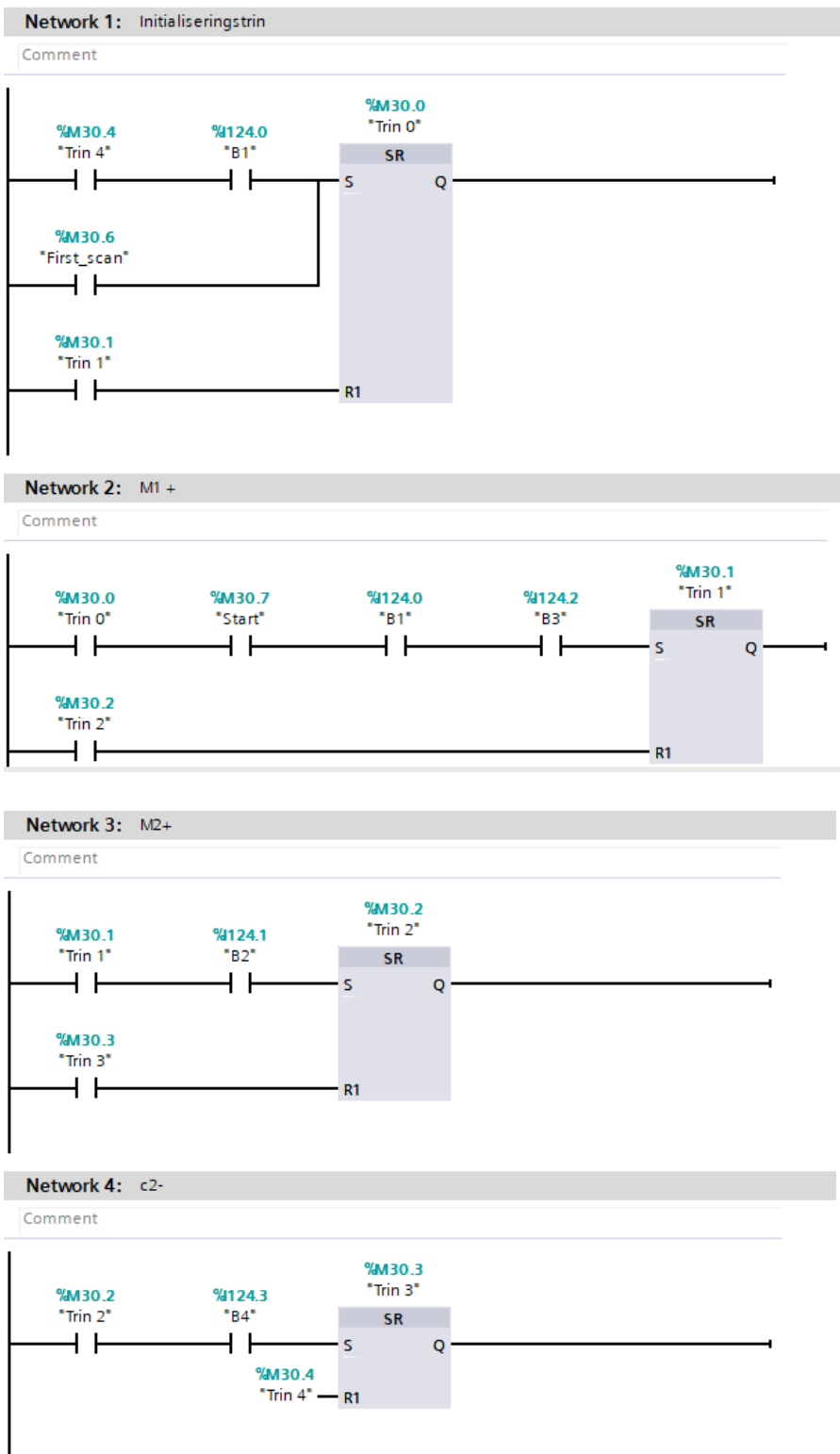
På reset-indgangen "R" tilføjes betingelser for reset, typisk det efterfølgende trin.

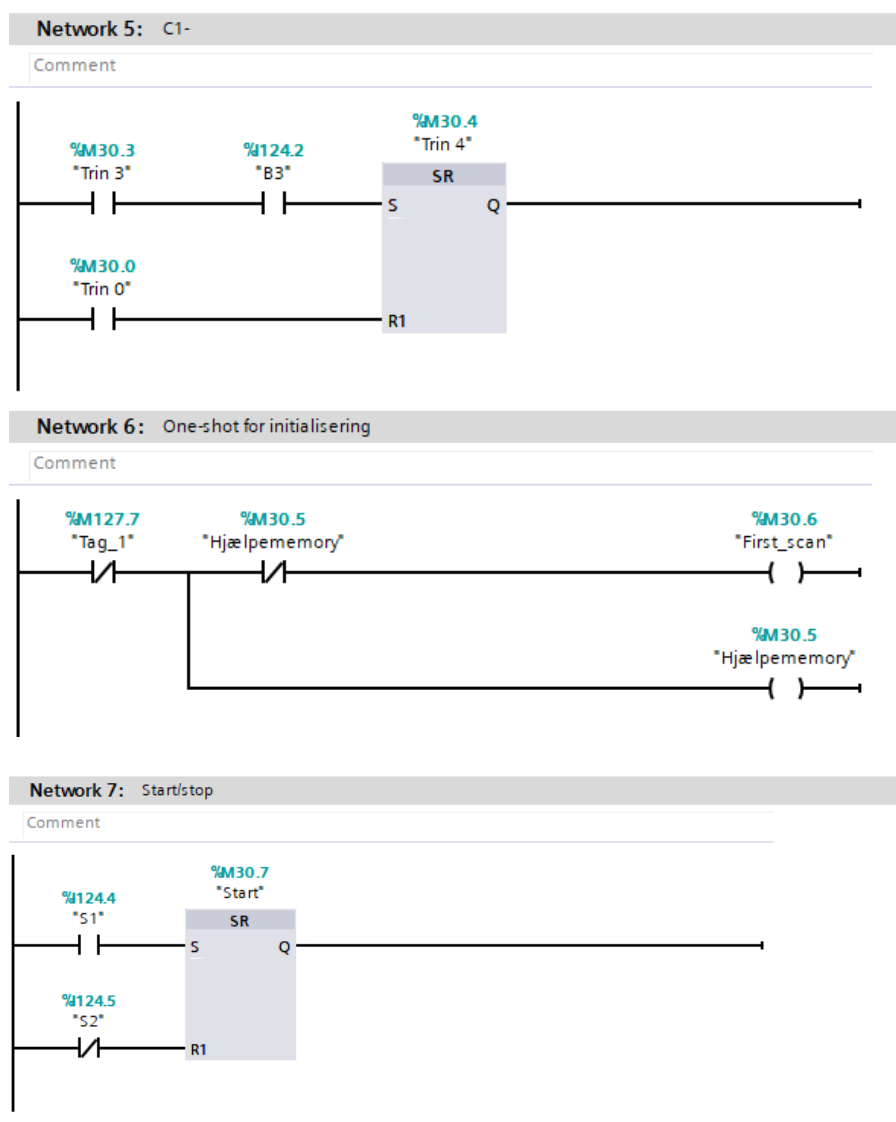
Princippet for opbygning af en sekvensstyring er:

- Udfører en handling
- Forbereder det næste trin
- Resette det foregående trin

I det efterfølgende eksempel er der anvendt 5 trin, hvilket medfører, at der anvendes 5 flip-flop til trinstyringen.

Eksempel sekvens FC1





Initialiseringstrin

Initialiseringstrinnet er et starttrin, og det er fra dette "hvile"-trin, at sekvensen startes, og når sekvensen er færdig, returneres til dette trin.

For at kunne aktivere initialiseringstrinnet ved nystart af PLC'en er det nødvendigt med en startpuls i det første scan. Hvis denne første scanpuls ikke er indbygget i PLC'en, skal den programmeres.

Network 1 = Initialiseringstrin

Network 6 = One-shot 1. scan

Network 2-5 = Trinstyring for sekvens

M127.7 er et "Dummy"-mærke, dvs. en bit, der ikke ændrer status. I dette tilfælde vil "Dummy"-bittet være konstant "1".

Årsagen til "Dummy"-bittet skal være der er, at der skal være en kontakt foran en udgang, der har flere udgange.

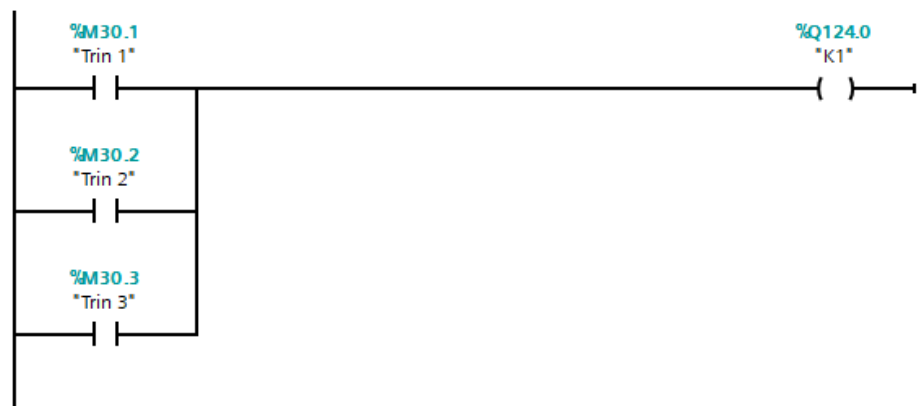
I network 6 aktiveres "first_scan" i et enkelt scan, når PLC'en startes op. "first_scan" setter initialiseringstrinnet i network 1.

Udgangsmatrix FC 2

Ved at programmere udgangsmatrixen sidst i PLC-programmet opnås et godt overblik over hvilke trin, der styrer den enkelte udgang. Endvidere bliver det forholdsvis simpelt at udvide med flere sekvenser, og et senere ønske om at tilføje en manual betjening til maskinen klares ved at programmere de manuelle flag parallelt over de enkelte trinflag i udgangsmatrixen.

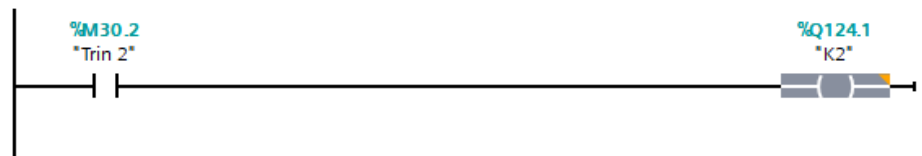
Network 1: Udgang K1 for cylinder M1

Comment



Network 2: Udgang K2 for cylinder M2

Comment

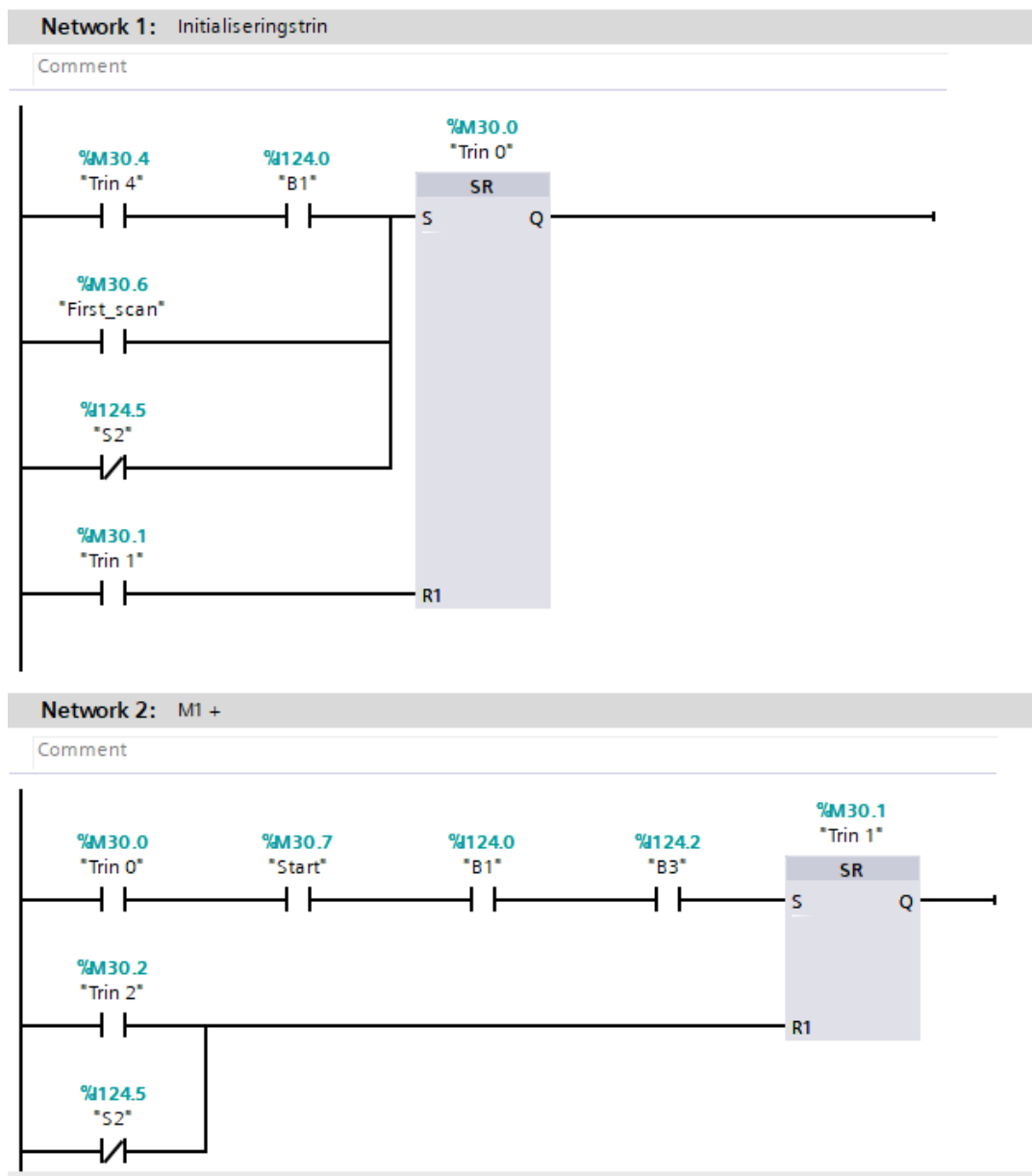


Der anvendes pneumatiske ventiler med fjeder-retur, derfor holdes signalet på K1 i trin 1, 2 og 3, mens K2 kun får signal i trin 2.

Reset

Når man aktiverer S2 "Sekvens reset", skal hele sekvensen nulstilles. Når reset aktiveres, skal det fungere således, at uanset hvilket trin sekvenskæden befinder sig i, så nulstilles sekvensen fra trin 1 til trin 4, og initialiseringstrinnet aktiveres.

For overskuelighedens skyld er der kun medtaget trin 0 og trin 1 i dette eksempel.



Sekvens med Move og Compare

Der findes flere forskellige metoder til programmering af sekvenser i en PLC. I dette afsnit beskrives programmet for den samme sekvens som i det foregående afsnit, men nu anvendes der ordinstruktioner i stedet for flip-flop.

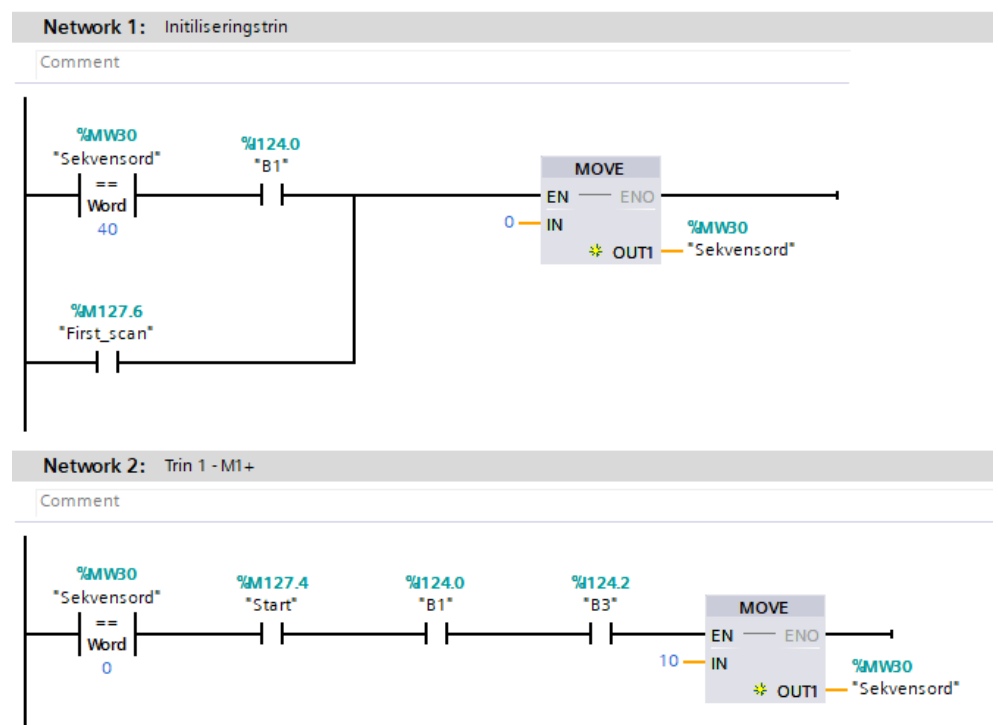
Det betyder, at man bruger instruktionen "Move" til at flytte en talværdi ind på en ordadresse. Sammenligning af hvilket trin programmet er kommet til, foretages med instruktionen "Compare".

I dette eksempel er der valgt følgende talværdier:

Trin 0 = 0
 Trin 1 = 10
 Trin 2 = 20
 Trin 3 = 30
 Trin 4 = 40

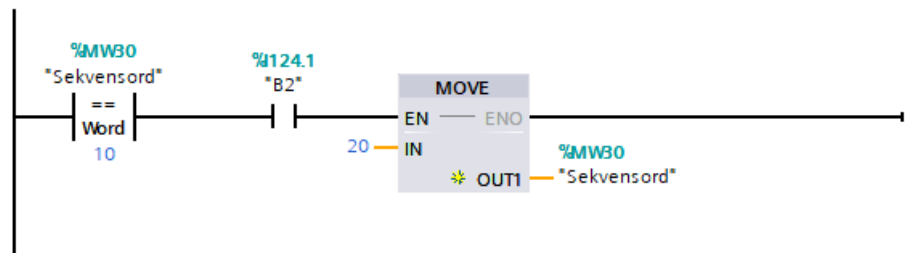
Det har den fordel at man let kan tilføje et ekstra trin, ved at kalde det trin 1.1 = 11

Programeksempel

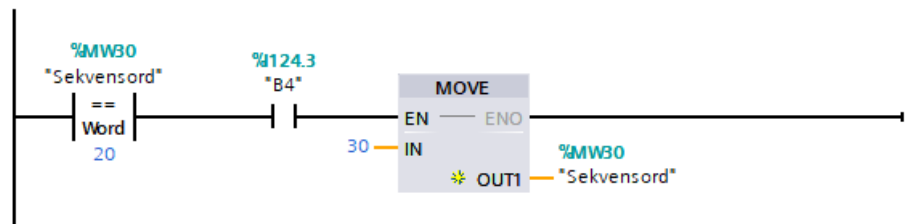


Network 3: Trin 2 - M2+

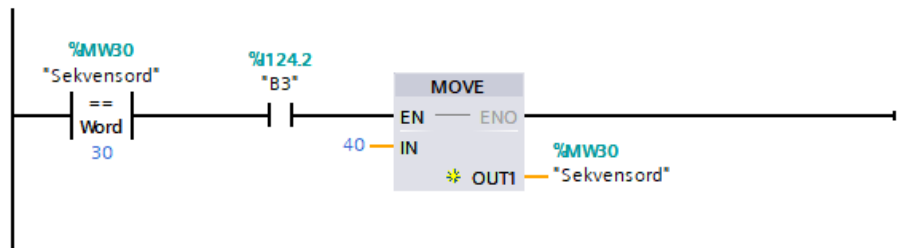
Comment

**Network 4: Trin 3 - M2-**

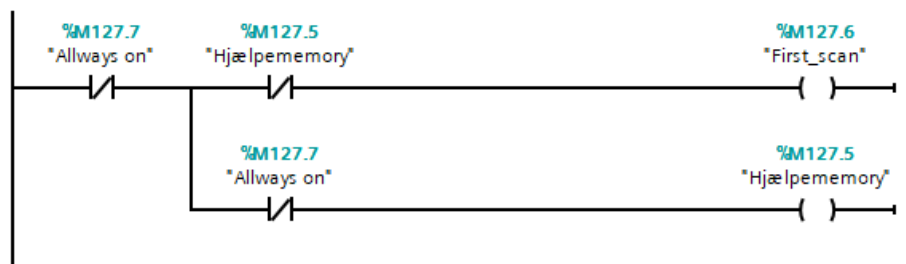
Comment

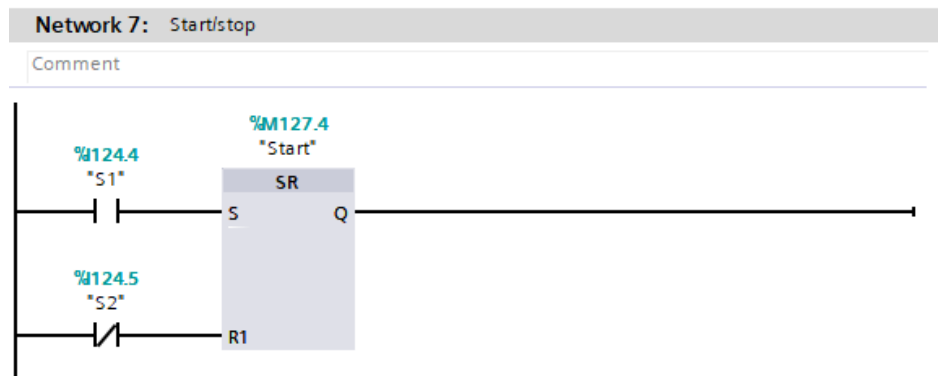
**Network 5: Trin 4 - M1-**

Comment

**Network 6: One-shot for initialisering**

Comment





Initialisering og start/stop er det samme som i foregående eksempel.

Når PLC'en sættes i run, bliver "First_scan" aktiv i et scan. Dette betyder, at kredsløbet i netværk 1 bliver aktiv, og der lægges tallet 0 ned i "Sekvensord". Dette betyder, at sekvensen er i trin 0 og afventer startkommando.

Når startbetingelser er aktive, og start aktiveres, bliver kredsløbet i netværk 2 aktiv, og der lægges tallet 10 ned i "Sekvensord".

Netværk 3 bliver aktiv, når trin 1 er aktiv og føler B4. Det medfører, at tallet 20 lægges ned i "Sekvensord"

Sekvensen fortsætter på samme måde med de efterfølgende trin.

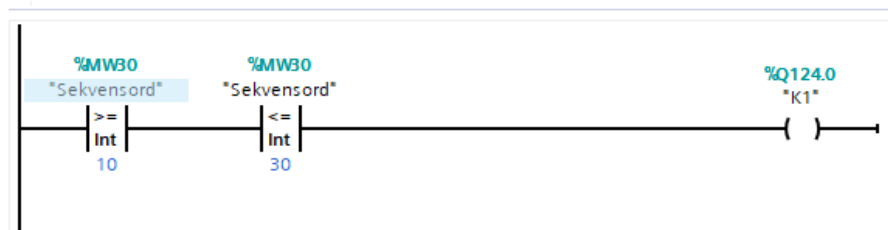
Udgangsmatrix

Udgangsmatrix programmeres med sammenligningsfunktioner.

Når et trin er aktiv, lægges talværdien ned i "Sekvensord":

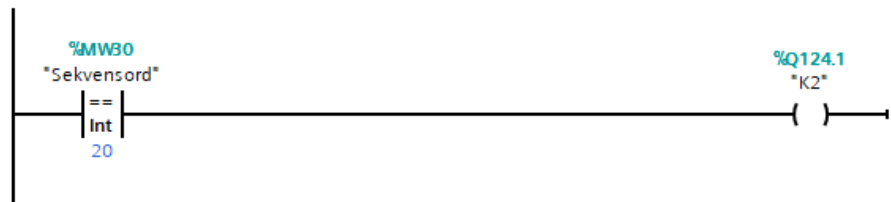
Trin 0 = 0
 Trin 1 = 10
 Trin 2 = 20
 Trin 3 = 30
 Trin 4 = 40

Cylinder M1 skal være aktiv i plusstilling, når trin 1, trin 2 og trin 3 er aktiv. Det betyder, når sekvensord er lig med 10, 20 eller 30, så skal M2 være i plus. K1 bliver aktiv, når "Sekvensord" er ≥ 10 og ≤ 30 .



Network 9: Udgangsmatrix M2

Comment



Cylinder M2 skal være aktiv i plusstilling, når trin 2 er aktiv. Det betyder, at når sekvensord er lig med 20, så skal M2 være i plus.

K2 bliver aktiv, når "Sekvensord" er = 20.

Sekvens-standarder

Generelt

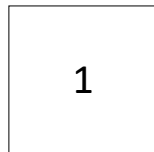
For at sikre en standardiseret dokumentation er der udarbejdet standarder for beskrivelse af en sekvens.

Den mest anvendte er IEC/EN 60848, der også benævnes Grafcet. Grafcet er et fransk navn, som betyder **G**raphe **F**onctionnel de **C**ommande **E**tape **T**ransition. Denne standard beskriver en sekvens set udefra. Det betyder, at der ikke tages hensyn til, om det er et relæ-, hydraulik-, pneumatik- eller en PLC-styring.

Standarden for PLC-programmering hedder IEC 61131-3. I denne standard er der angivet, hvordan sekvenser programmeres. Sekvensstandarden kaldes også **Sequential Function Charts (SFC)**. Siemens Graph 7-sproget overholder den SFC-standard.

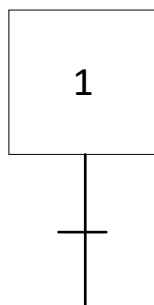
IEC/EN 60848

Standarden er udviklet til at beskrive et sekventielt system, hvilket betyder, at de grundlæggende elementer er:



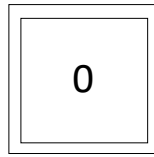
- Step (trin)

Tallet betyder, at det er trin 1.



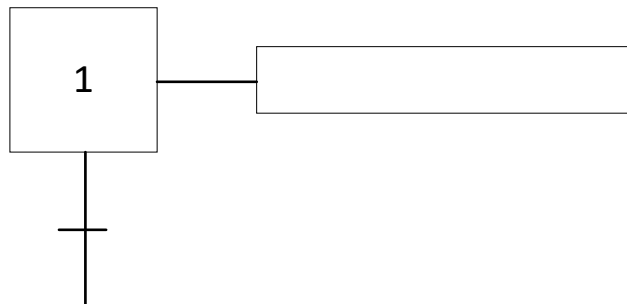
- Transition (kviktering)

Her er trin 1 vist med en efterfølgende kviktering. På tværstregen angives signalet for kviktering mellem de enkelte trin.



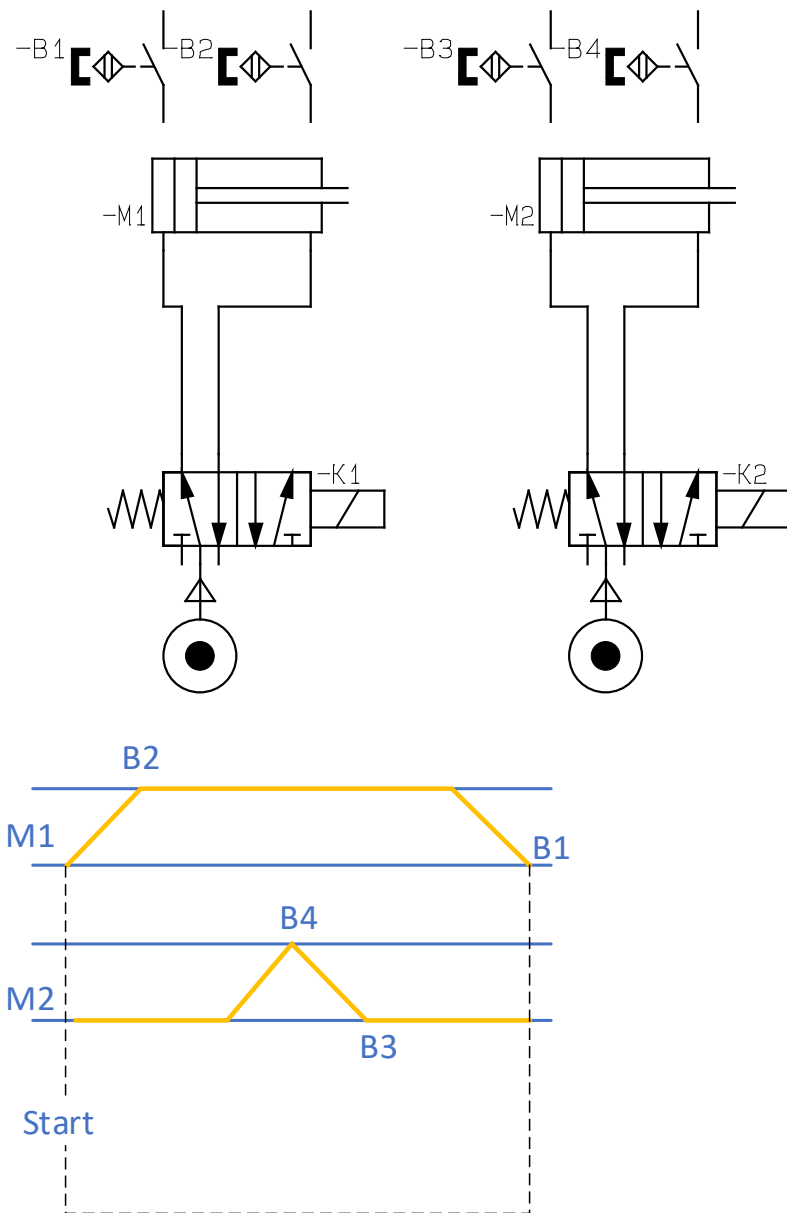
Her er trinnet vist med en dobbeltfirkant. Det betyder, at det er initialiseringstrinnet, som er det trin, hvor sekvensen starter.

I dette eksempel har den fået benævnelsen trin 0.



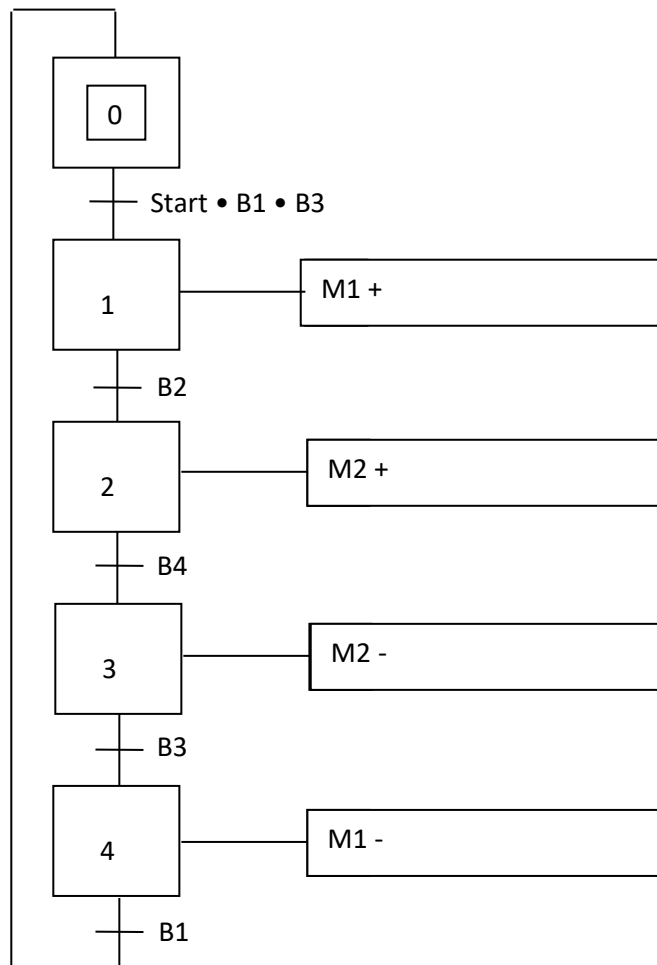
Kassen til højre for trinnet er en action, hvilket betyder, at det er kommando til, hvad der skal ske i trinnet.

Sekvenseksempel



Vi har en sekvens med to cylindre, der kører i henhold til følgende diagram.

Sekvensdiagram



Sekvensen står i dens udgangsstilling, som er initialiseringstrin 0. Her står den og venter på startsignal.

Transition for at få trin 1 aktiv er følgende:

Trin 0, Start, B1 og B3 skal være aktive for, at trin 1 bliver aktiv. Når trin 1 er aktiv, gives signal til cylinder M1, hvorefter den kører frem mod plus.

Betingelser skrives med boolsk algebra:

- betyder og (AND)
- + betyder eller (OR)

Husk, at de normale regneregler for parenteser også gælder her.

Hvis et signal ikke skal være der, sættes en lille streg over betingelsen $\overline{B1}$ en såkaldt negering.

Transition for at få trin 2 aktiv er følgende:

Trin 1 og B2 skal være aktive for, at trin 2 bliver aktiv. Når trin 2 er aktiv, gives signal til cylinder M2, hvorefter den kører frem mod plus.

Transition for at få trin 3 aktiv er følgende:

Trin 2 og B4 skal være aktive for, at trin 3 bliver aktiv. Når trin 3 er aktiv, fjernes signal til cylinder M2, hvorefter den kører frem mod minus.

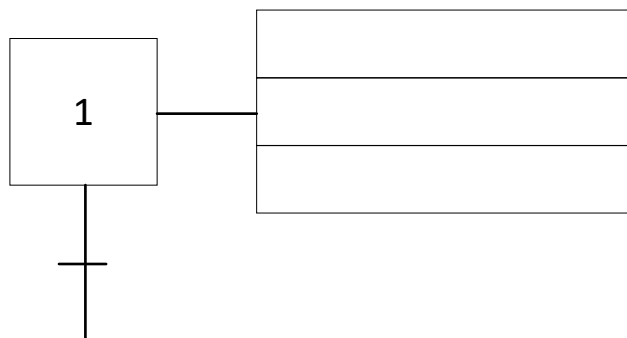
Transition for at få trin 4 aktiv er følgende:

Trin 3 og B3 skal være aktive for, at trin 4 bliver aktiv. Når trin 4 er aktiv, fjernes signal til cylinder M1, hvorefter den kører frem mod minus.

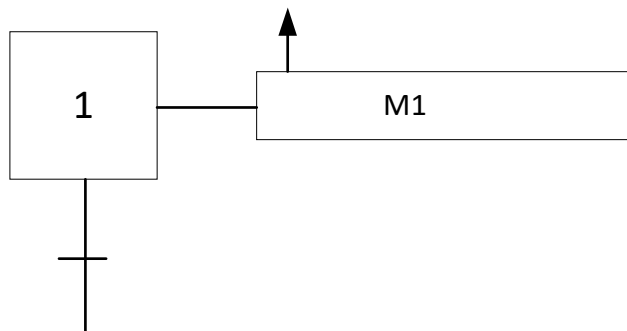
Transition for at få trin 0 aktiv er følgende:

Trin 4 og B1 skal være aktive for at trin 0 bliver aktiv. Når trin 0 igen er aktiv, kører sekvensen igen, hvis ellers startbetingelserne er opfyldt.

Symboler og deling af sekvens

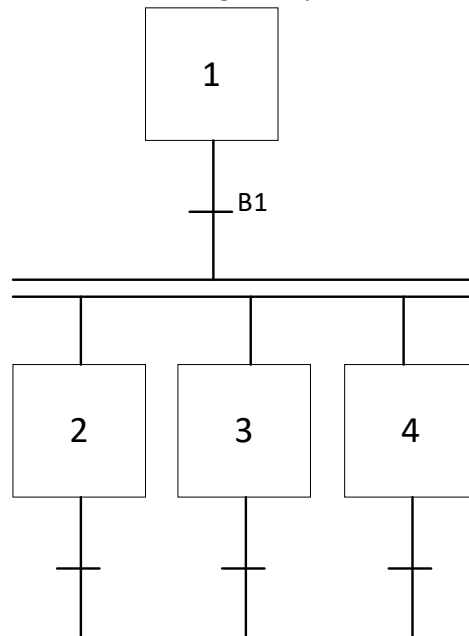


Hvis der i et trin skal udføres flere kommandoer, så tegnes der nogle ekstra bokse.

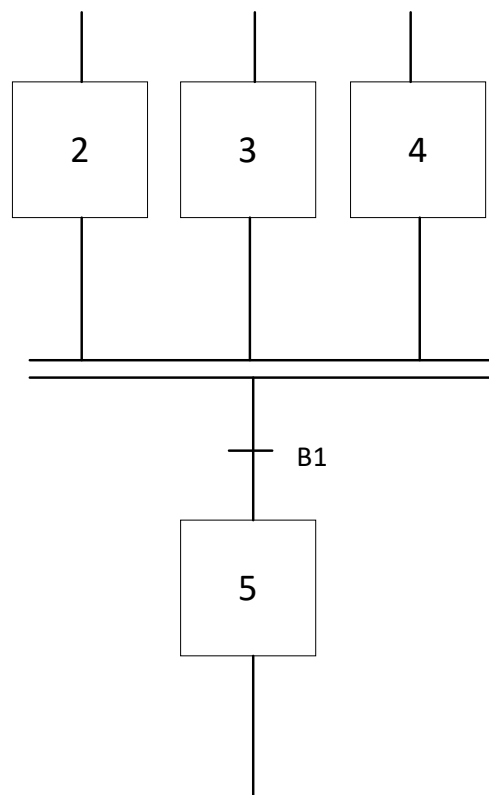


Hvis der er hukommelse på en kommando, så angives det med en pil.

De to dobbeltstreger betyder **OG**-funktion.

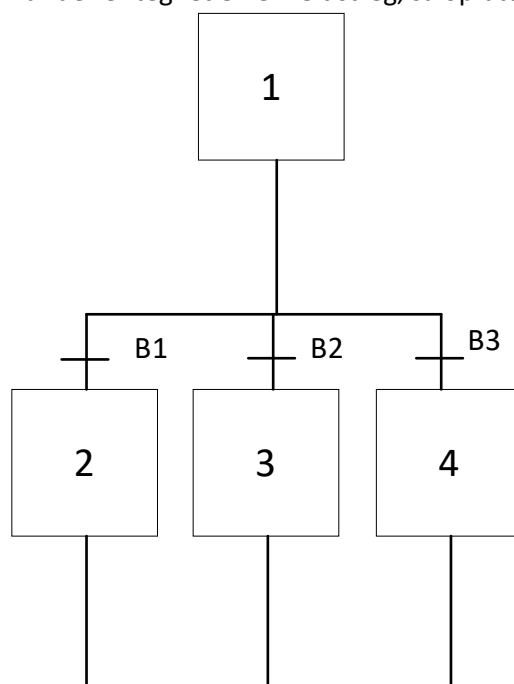


Når trin 1 er aktiv, og B1 aktiveres, fortsættes samtidigt parallelt i trin 2 **og** 3 **og** 4.

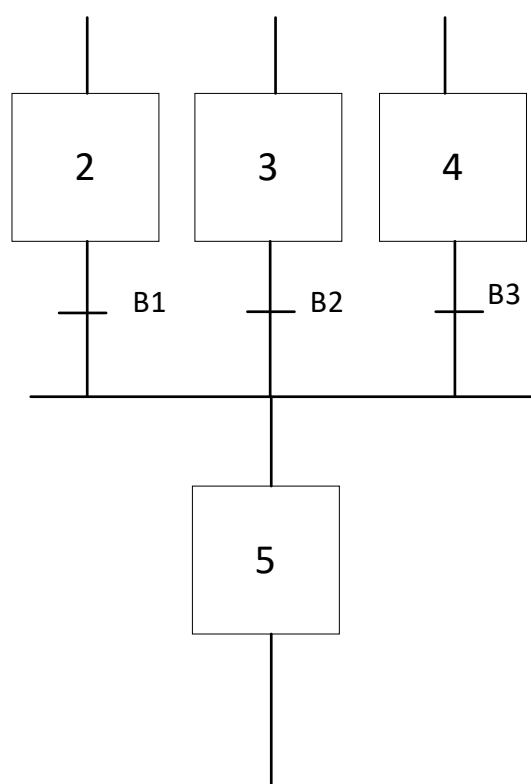


I dette eksempel skal både trin 2 **og** trin 3 **og** trin 4 **og** B1 være aktiv, før sekvensen går videre til trin 5.

Når der er tegnet en enkelt streg, så opfattes det som en **Eller**-forbindelse



Der fortsættes i trin 2 **eller** 3 **eller** 4, når trin 1 er aktiv samt det tilhørende signal; B1 for trin 2, B2 for trin 3 eller B3 for trin 4 er aktivt.







For at komme videre i trin 5 kræves det, at trin2 og B1 **eller** trin 3 og B2 **eller** trin 4 og B3 er aktive.

Symboler IEC/EN 60848

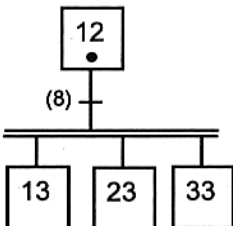
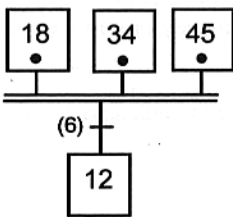
Generelt

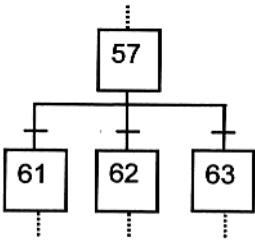
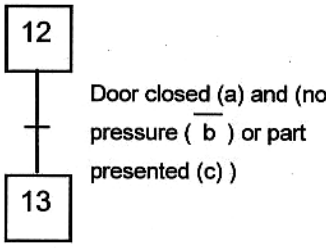
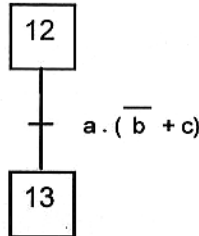
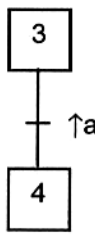
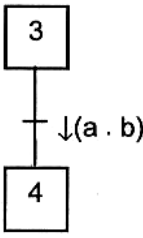
Det følgende viser nogle af de symboler, som anvendes for at beskrive en sekvens.

Trin

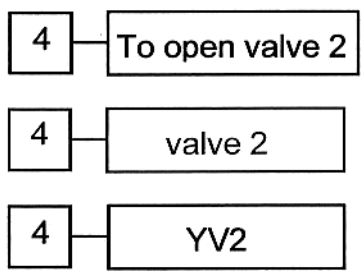
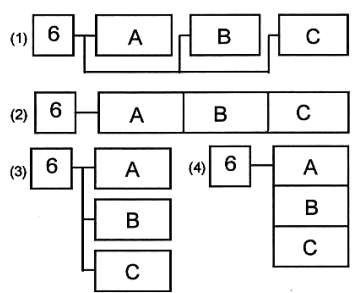
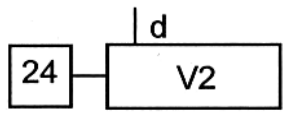

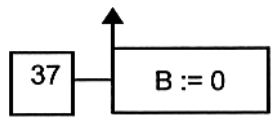
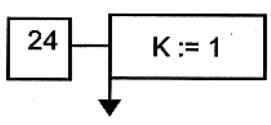
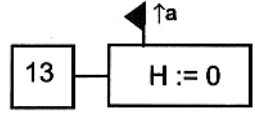
| Symbol | Beskrivelse |
|---|---|
|  | Et trin tegnes enten som et aktiv eller et ikke aktivt trin. Hvert trin benævnes med et nummer. Det viste trin er tegnet ikke aktiv. Hvis man har behov for at tegne et aktivt trin, tilføjes en sort prik. |
|  | Symbolet viser et initialiseringstrin. |
|  | Symbolet viser en undersekvens, som kan køre parallelt med de andre sekvenser. |
|  | Macrostep, underrutine som skal programmeres med et start- og et sluttrin |

Kvitteringer

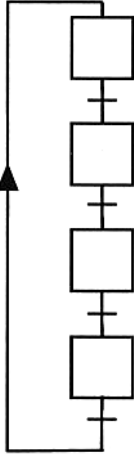
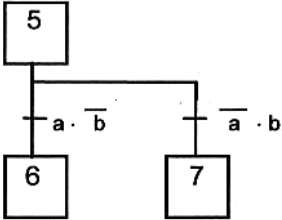
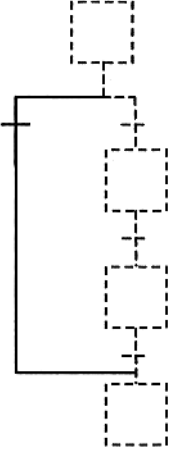
| Symbol | Beskrivelse |
|---|---|
|  | Trin 12 er tegnet aktiv (angives med en sort prik), der fortsættes i trin 13, 23 og 33, når trin 12 er aktiv, og kvittering nr. 8 er til stede. |
|  | Trin 18, 34 og 45 er tegnet aktiv. Der fortsættes i trin 12, når kvittering nr. 6 og trin 18 og trin 34 og trin 45 er aktiv |

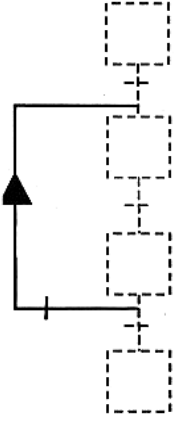
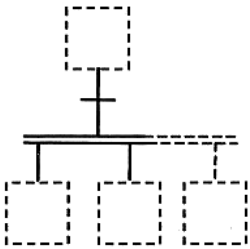
| | |
|---|---|
|  | <p>Der fortsættes i de efterfølgende trin, når trin 57 er aktiv samt en af de viste kvitteringer.</p> |
|   | <p>I dette eksempel er kvitteringer beskrevet med tekst.</p> <p>I dette eksempel er kvitteringer vist med boolsk algebra</p> |
|  | <p>Kvittering a er kun aktiv, når der er et skift fra logisk 0 til logisk 1. (kanttrigget)</p> |
|  | <p>Kvitteringen er aktiv, når det logiske resultat af a og b skifter fra logisk 1 til logisk 0.</p> |

Udgange, actions

| Symbol | Beskrivelse |
|---|---|
|  | I dette eksempel er der vist, hvordan de enkelte udgangssignaler kan angives. |
|  | I dette eksempel er der vist forskellige muligheder for at tegne actions ind i de enkelte trin. |
|  | I dette eksempel er der tegnet en betingelse på udgangssignalet. For at V2 bliver aktiv, kræves det, at trin 24 og signal d er aktive. |
|  | <p>Tildeling af værdi til en variabel. I de viste actions gemmes værdien i en "stored" funktion.</p> <p>Værdien A settes til værdien 1.</p> <p>Der er tegnet :=, der fortæller, at funktionen er "stored".</p> <p>Værdien B settes til værdien 0.</p> |
|  | Udgangsværdien B bliver lig med logisk 0, når trin 37 skifter fra logisk 0 til logisk 1. |
|  | Udgangsværdien K bliver lig med logisk 1, når trin 24 skifter fra logisk 1 til logisk 0. |
|  | Udgangsværdien H bliver lig med logisk 0, når trin 13 er aktiv, og kvittering a skifter fra logisk 0 til logisk 1. |

Sekvensopbygning

| Symbol | Beskrivelse |
|---|--|
|  | <p>Skemaet viser en normal sekvens, som genstarter, når den er løbet igennem.</p> |
|  | <p>Valg af sekvens, signalerne a og b bestemmer, hvilket sekvensgennemløb der foretages. Eller-delning</p> |
|  | <p>I dette eksempel er der vist, hvordan man hopper trin over.</p> |

| | |
|---|--|
|  | <p>I dette eksempel er der vist, hvordan man hopper trin tilbage i sekvenskæden.</p> |
|  | <p>I dette eksempel er det vist, hvordan man aktiverer parallelle sekvenser. Og-delinger</p> |

IEC 61131-3

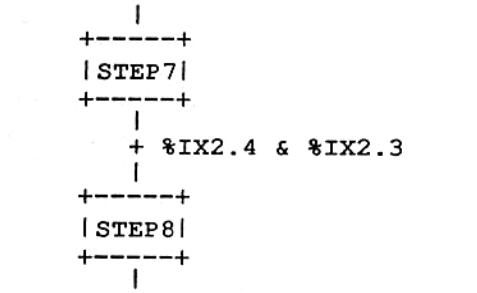
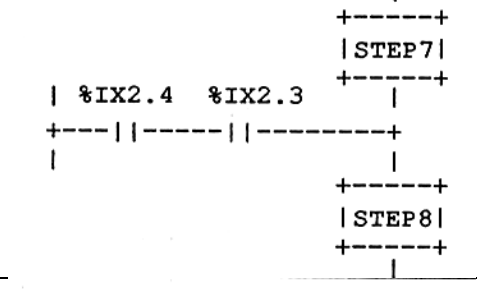
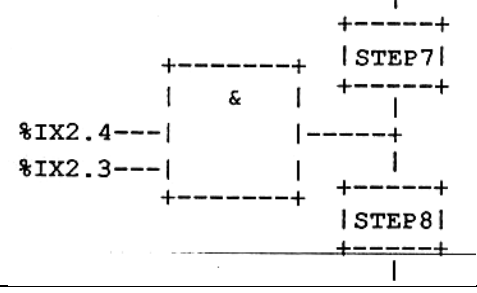
Generelt

Standarden for PLC-programmeringssprog hedder IEC 61131-3. Som en del af denne standard er det angivet, hvorledes en sekvens skal/bør programmeres. Sekvensstandarden i IEC 61131-3 benævnes SFC, **Sequential Function Chart**.

Når man tegner en sekvens i henhold IEC 61131-3, er det set inde fra PLC'en, hvilket betyder, at man skal tage hensyn til, om kontakter og signaler er NC/NO. Hvis man tegner i henhold til IEC 60848, så ses sekvensen fra maskinsiden, hvor man beskriver, hvilke følere og kontakter der er aktiveret/ikke aktiveret.

I dette afsnit vises nogle enkelte elementer fra standarden.

Step og kvittering

| Symbol | Beskrivelse |
|--|--|
|  | Eksemplet viser 2 trin, hvor kvitteringen skrives med boolsk algebra. |
|  | Eksemplet viser 2 trin, hvor kvitteringen skrives med et Ladder Diagram |
|  | Eksemplet viser 2 trin, hvor kvitteringen skrives med et logikdiagram, Function Block Diagram. |

Step med action

| Symbol | Beskrivelse |
|--|--|
| <pre> +-----+-----+-----+ --- "a" "b" "c" --- +-----+-----+-----+ "d" +-----+-----+-----+ </pre> | <p>Diagrammet viser en action på et step.</p> <p>a. Kommandotype iht. tabel</p> <p>b. Kommandonavn</p> <p>c. Boolsk-indikator, f.eks. kvittering.</p> <p>d. Anvendes, hvis kommando skrives med kode</p> |
| <pre> +-----+ +-----+-----+-----+ S8 --- L ACTION_1 DN1 +-----+ t#10s + DN1 </pre> | <p>Eksemplet viser en udgang, der benævnes action, som er aktiv i 10 s.</p> |

Tabel for kommandotype

| No. | Qualifier | Explanation |
|-----|-----------|-----------------------------|
| 1 | None | Non-stored (null qualifier) |
| 2 | N | Non-stored |
| 3 | R | overriding Reset |
| 4 | S | Set (Stored) |
| 5 | L | time Limited |
| 6 | D | time Delayed |
| 7 | P | Pulse |
| 8 | SD | Stored and time Delayed |
| 9 | DS | Delayed and Stored |
| 10 | SL | Stored and time Limited |

Sekvens Siemens Graph 7

Generelt

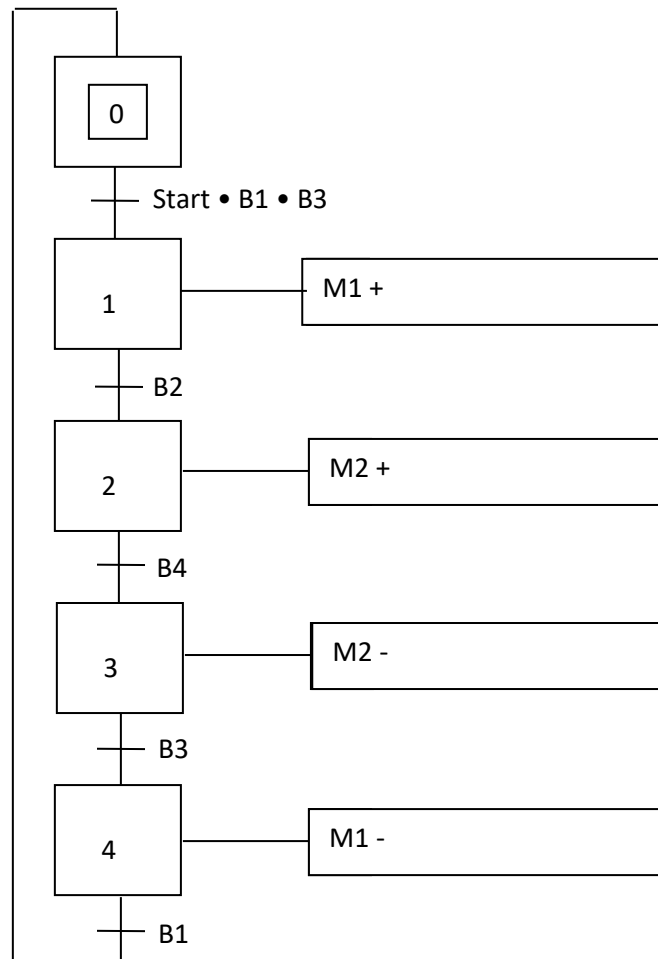
Siemens har udviklet software, hvor man programmerer direkte i et grafisk sekvenssprog, SFC (**S**equential **F**unction **C**hart).

Det medfører, at sekvensen tegnes på computeren, hvorefter softwaren selv genererer den programkode, der er nødvendig.

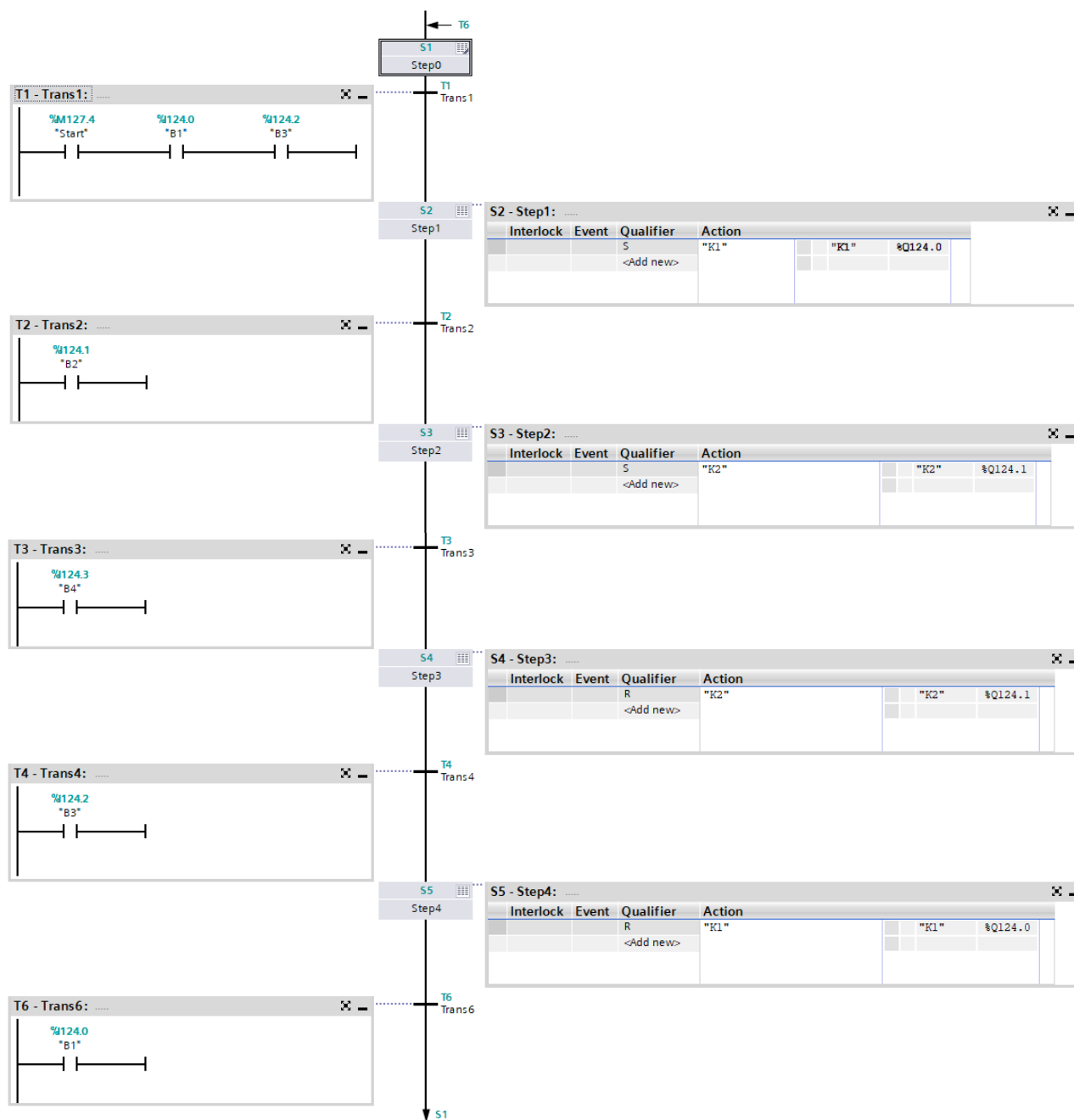
En Graph 7-blok skal altid programmeres i en FB.

Det efterfølgende er et eksempel på en sekvens programmeret i Graph 7 – det samme eksempel som tidligere anvendt i dette afsnit.

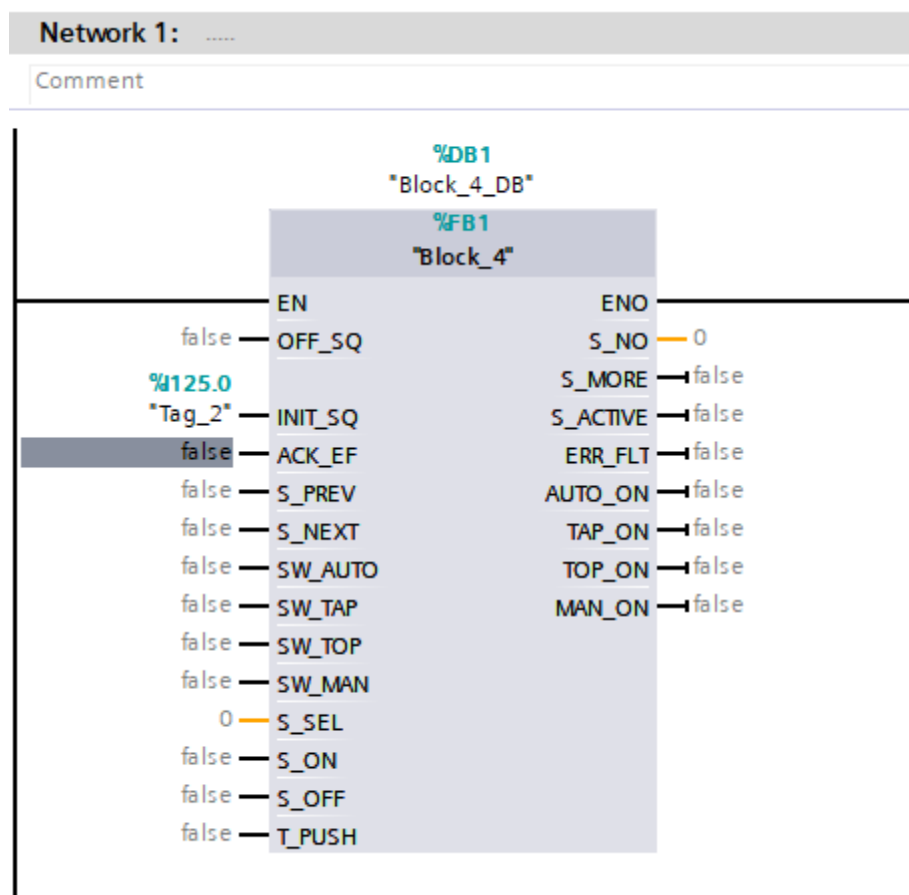
Sekvensdiagram



Programeksempel



Kald af blok



For at sekvensen kan eksekveres, så skal FB 1 kaldes fra f.eks. OB 1.

I dette eksempel er der kun brugt INIT_SQ – I125.0.

Når I 125.0 aktiveres, resettes sekvensen, og den stiller sig i initialiseringstrinnet og er klar til start.

Talsystemer

Titalssystemet

Opbygning

Når man arbejder med programmerbare styringer og lignende elektroniske systemer, vil man støde på forskellige talsystemer.

Det talsystem, som vi er vant til at arbejde med, og som vi bruger til at lægge sammen og trække fra med og til at regne vores løn ud efter, kaldes titalssystemet eller det decimale system.

Det skyldes, at titalssystemet består af præcist ti forskellige tal, og for det andet er hvert ciffers placering i et helt tal baseret på potenser af 10.

| Tusinde | Hundrede | Tiere | Enere |
|---------|----------|-------|-------|
| 2 | 0 | 2 | 4 |

Dette kan ses af årstallet 2024.

Som det ses af eksemplet, er det ikke kun tallets egen værdi, der tæller.

Tallets placering spiller også en rolle.

Tager vi totallet, betyder det ikke to, men angiver antallet af tusinder og antallet af tiere.

Potenser

| Tusinde | Hundrede | Tiere | Enere |
|---------------|--------------|-------------|------------|
| 2 | 0 | 2 | 4 |
| $10^3 = 1000$ | $10^2 = 100$ | $10^1 = 10$ | $10^0 = 1$ |

De enkelte cifres betydning kan også vises med titalspotenser.

Totalssystemet

Opbygning

Alle elektroniske regne- og styresystemer arbejder med totalssystemet. Dette talsystem har kun to forskellige cifre at arbejde med, nemlig 0 og 1.

Det er også 0 og 1, man arbejder med i styringer, og det er 0 og 1, Boolsk algebra kan anvendes på.

Et tal, som er skrevet i totalssystemet, kan anskueliggøres i totalspotenser.

| $2^5 = 32$ | $2^4 = 16$ | $2^3 = 8$ | $2^2 = 4$ | $2^1 = 2$ | $2^0 = 1$ |
|------------|------------|-----------|-----------|-----------|-----------|
| 1 | 1 | 0 | 0 | 1 | 0 |

Indeks

Når man arbejder med forskellige talsystemer, er det klogt at sætte et indeks på tallet for at fortælle, hvilket talsystem man arbejder med.

Omskrivning

Det binære tal er navnet på tal i totalssystemet og kan derfor omskrives til decimaltal.

$$\begin{array}{lll}
 2^5 & = 32 \cdot 1 & = 32 \\
 2^4 & = 16 \cdot 1 & = 16 \\
 2^3 & = 8 \cdot 0 & = 0 \\
 2^2 & = 4 \cdot 0 & = 0 \\
 2^1 & = 2 \cdot 1 & = 2 \\
 2^0 & = 1 \cdot 0_B & = 0
 \end{array}$$

$$50_D = 110010_B$$

Sekstentalssystemet

Opbygning

Nogle programmerbare styresystemer anvender et sekstentalssystem eller hexadecimalt talsystem, som i daglig tale blot benævnes hex.

Et sekstentalssystem har 16 forskellige cifre. Da vi kun har 10 forskellige talsymboler, har det været nødvendigt at tage nogle bogstaver til hjælp.

Den hexadecimal talrække ser således ud:

| Hex | Dec. |
|-----|------|
| 0 | 0 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |
| A | 10 |
| B | 11 |
| C | 12 |
| D | 13 |
| E | 14 |
| F | 15 |

Talsystemer og format

Fra hex til decimal

| $16^3 = 4096$ | $16^2 = 256$ | $16^1 = 16$ | $16^0 = 1$ |
|---------------|--------------|-------------|------------|
| Hex | | | |
| D | 7 | A | 1 |
| Binær | | | |
| 1101 | 0111 | 1010 | 0001 |

Her benyttes sekstentalspotensen til at beskrive et hexadecimaltal.

$$16^3 = 4096 \cdot D = 53248$$

$$16^2 = 256 \cdot 7 = 1792$$

$$16^1 = 16 \cdot A = 160$$

$$16^0 = 1 \cdot 1 = 1$$

$$55201_D = D7A1_H = 1101\ 0111\ 1010\ 0001_B$$

Konverteringstabel

Tabel til omregning mellem de tre nævnte talsystemer.

| Binær | Hex | Decimal |
|--------|-----|---------|
| 0000 | 0 | 0 |
| 0001 | 1 | 1 |
| 0010 | 2 | 2 |
| 0011 | 3 | 3 |
| 0100 | 4 | 4 |
| 0101 | 5 | 5 |
| 0110 | 6 | 6 |
| 0111 | 7 | 7 |
| 1000 | 8 | 8 |
| 1001 | 9 | 9 |
| 1010 | A | 10 |
| 1011 | B | 11 |
| 1100 | C | 12 |
| 1101 | D | 13 |
| 1110 | E | 14 |
| 1111 | F | 15 |
| 1 0000 | 10 | 16 |
| 1 0001 | 11 | 17 |
| 1 0010 | 12 | 18 |
| 1 0011 | 13 | 19 |
| 1 0100 | 14 | 20 |

Bit, Byte, Word og Dword

For at kunne skrive de forskellige talformater kræves der det korrekte antal bit i adressen.

Det efterfølgende er en beskrivelse af bit, byte, word og double word.

Bit

PLC'en kan kun bearbejde de to signaltilstande »0« og »1«, da den arbejder digitalt.

Bit = 0

| |
|---|
| 0 |
|---|

 Bit = 1

| |
|---|
| 1 |
|---|

En celle, der indeholder signaltilstanden »0« eller »1«, benævnes et bit. Ordet stammer fra **BI**nary digi**T** eller oversat til dansk et »binært ciffer«.

Byte

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

MSB

LSB

Ved at gruppere 8 bit dannes en byte.

MSB Most Significant Bit (Mest betydende bit).

LSB Least Significant Bit (Mindst betydende bit).

Word

| Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|--------|--------|--------|--------|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |

MSB

LSB

Ved at sammensætte 2 byte dannes et WORD eller på dansk et ord, som beslaglægger 16 bit.

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--------------|
| Bit 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | Bit 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | | |

BCD



| Binær | Decimal | BCD |
|--------------|----------------|------------|
| 0000 | 0 | 0000 |
| 0001 | 1 | 0001 |
| 0010 | 2 | 0010 |
| 0011 | 3 | 0011 |
| 0100 | 4 | 0100 |
| 0101 | 5 | 0101 |
| 0110 | 6 | 0110 |
| 0111 | 7 | 0111 |
| 1000 | 8 | 1000 |
| 1001 | 9 | 1001 |
| 1010 | 10 | 0001 0000 |
| 1011 | 11 | 0001 0001 |
| 1100 | 12 | 0001 0010 |
| 1101 | 13 | 0001 0011 |
| 1110 | 14 | 0001 0100 |
| 1111 | 15 | 0001 0101 |

| | | | | |
|----------------|------|------|------|------|
| Decimal | 2 | 0 | 1 | 4 |
| BCD | 0010 | 0000 | 0001 | 0100 |

Positive/negative heltal

Hvis indholdet kan betragtes som et positivt decimaltal, kan der fremstilles positive hele tal fra 0 til 2^{16} . 0 65536

Ved matematiske beregninger, indlæsning af analoge signaler og lignende er der behov for at få positive og negative heltal omsat til digitale bitmønstre.

| | | | | | | | | | | | | | | | |
|----|----------|----------|----------|----------|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| S | 2^{14} | 2^{13} | 2^{12} | 2^{11} | 2^{10} | 2^9 | 2^8 | 2^7 | 2^6 | 2^5 | 2^4 | 2^3 | 2^2 | 2^1 | 2^0 |

S = 0 = positivt tal

S = 1 = negativt tal

MSB anvendes som fortegnsbite, hvor MSB = 0 medfører en positiv talværdi, og med MSB = 1 er talværdien negativ.

Positive tal fremstilles som normalt, og da MSB skal være nul, medfører dette, at den maksimale værdi bliver 2^{15} , altså +32767.

2-komplement

De negative tal fremstilles i 2-komplement.

Instuktionssættet i CPU'en understøtter, at negative tal fremstilles i 2-komplement-format for at lette regneoperationer.

2-komplement formatet er det mest anvendte format til fremstilling af hele negative decimaltal i en PLC, men det er desværre ikke helt ukompliceret.

Et tals komplement er differencen mellem tallet og et referencetal. I det binære talsystem anvendes 1-komplement og 2-komplement.

For at bevare overblikket ser vi efterfølgende på et eksempel med 4 bit, idet fremgangsmåden jo ikke ændres med 8 eller 16 bit.

1-komplementet til et binært tal fås ved at invertere hver enkelt bit.

| | S | 4 | 2 | 1 |
|--------------|---|---|---|---|
| Binær | 0 | 1 | 1 | 0 |
| 1 komplement | 1 | 0 | 0 | 1 |

Hvis vi adderer 1 til 1-komplementet, fås 2-komplementet til tallet.

| | S | 4 | 2 | 1 |
|--------------|---|---|---|---|
| Binær | 0 | 1 | 1 | 0 |
| 1 komplement | 1 | 0 | 0 | 1 |
| +1 | | | | 1 |
| 2 komplement | 1 | 0 | 1 | 0 |

En huskeregel for at fremstille et binært tal i 2-komplement-format er at notere den positive binære værdi, og startende fra LSB mod MSB skrives alle bit til og med det første »1« tal uforandret, resten inverteres.

Som det fremgår i eksemplet, har vi nu et negativt tal, fordi fortegnet i MSB = »1«. Ved udlæsning af den negative talværdi til et decimaltal udlæses »0« med den normale vægtning af bitpositionen, og derefter skal der yderligere trækkes 1 fra udlæsningen.

Eksempel med 2-komplement værdien: $1\ 0\ 1\ 0 = -(4 + 1) - 1 = -6$

Datatyper iht. 61131-3

En datatype er en klassifikation, som definerer de mulige værdier, der kan anvendes ved variable.

Datatyper er defineret i standarden DS/EN 61131-3, som er programmeringssproget for en PLC.

| No. | Description | Keyword | Default Initial value | N (bits) ^a |
|---|--|-----------------------|-------------------------|--------------------------|
| 1 | Boolean | BOOL | 0, FALSE | 1 ^h |
| 2 | Short integer | SINT | 0 | 8 ^c |
| 3 | Integer | INT | 0 | 16 ^c |
| 4 | Double integer | DINT | 0 | 32 ^c |
| 5 | Long integer | LINT | 0 | 64 ^c |
| 6 | Unsigned short integer | USINT | 0 | 8 ^d |
| 7 | Unsigned integer | UINT | 0 | 16 ^d |
| 8 | Unsigned double integer | UDINT | 0 | 32 ^d |
| 9 | Unsigned long integer | ULINT | 0 | 64 ^d |
| 10 | Real numbers | REAL | 0.0 | 32 ^e |
| 11 | Long reals | LREAL | 0.0 | 64 ^f |
| 12a | Duration | TIME | T#0s | .. ^b |
| 12b | Duration | LTIME | LTIME#0s | 64 ^{m, q} |
| 13a | Date (only) | DATE | NOTE | .. ^b |
| 13b | Long Date (only) | LDATE | LDATE#1970-01-01 | 64 ⁿ |
| 14a | Time of day (only) | TIME_OF_DAY or TOD | TOD#00:00:00 | .. ^b |
| 14b | Time of day (only) | LTIME_OF_DAY or LTOD | LTOD#00:00:00 | 64 ^{o, q} |
| 15a | Date and time of Day | DATE_AND_TIME or DT | NOTE | .. ^b |
| 15b | Date and time of Day | LDATE_AND_TIME or LDT | LDT#1970-01-01-00:00:00 | 64 ^{p, q} |
| 16a | Variable-length single-byte character string | STRING | ' ' (empty) | 8 ^{i, g, k, l} |
| 16b | Variable-length double-byte character string | WSTRING | " " (empty) | 16 ^{i, g, k, l} |
| 17a | Single-byte character | CHAR | '\$00' | 8 ^{g, l} |
| 17b | Double-byte character | WCHAR | "\$0000" | 16 ^{g, l} |
| 18 | Bit string of length 8 | BYTE | 16#00 | 8 ^{j, g} |
| 19 | Bit string of length 16 | WORD | 16#0000 | 16 ^{j, g} |
| 20 | Bit string of length 32 | DWORD | 16#0000_0000 | 32 ^{j, g} |
| 21 | Bit string of length 64 | LWORD | 16#0000_0000_0000_0000 | 64 ^{j, g} |
| NOTE Implementer specific because of special starting date different than 0001-01-01. | | | | |

Se evt. også https://en.wikipedia.org/wiki/IEC_61131-3

BOOL

Binær værdi på 1 bit, kan antage værdien 1 - 0, eller true - false

SINT, INT, DINT, LINT

INT betyder integer, eller oversat til dansk heltal. Et heltal er et tal uden komma. Et heltal kan både være positivt eller negativt.

Forskellen på de fire taltyper er antal af bit, det indeholder.

SINT fylder 8 bit

INT fylder 16 bit

DINT fylder 32 bit

LDINT fylder 64 bit

Når der er flere bit i en datatype, kan der skrives større tal.

USINT, UINT, UDINT, ULINT

UINT betyder unsigned integer, eller oversat til dansk positivt heltal.

Forskellen på unsigned og signet heltal er, at denne type kun kan skrive positive heltal. Ellers er det samme opbygning som integer.

USINT fylder 8 bit

UINT fylder 16 bit

UDINT fylder 32 bit

ULINT fylder 64 bit

Når der er flere bit i en datatype, kan der skrives større positive tal.

REAL, LREAL

REAL tal betyder, at det er et kommatal. Et real tal kaldes også for floating point, som lige netop betyder kommatal på engelsk.

REAL fylder 32 bit

LREAL fylder 64 bit

TIME, LTIME

Disse to datatyper bruges til tidsangivelse, f.eks. i forbindelse med timere

DATE, LDATE

Datatype for dato

TOD, LTOD

Datatype for time of day, klokkeslæt med timer-minutter og sekunder

DT, LDT

Datatype for dato og klokkeslæt

STRING, WSTRING

Datatype for en tekststreng

CHAR, WCHAR

Datatype for en karakter

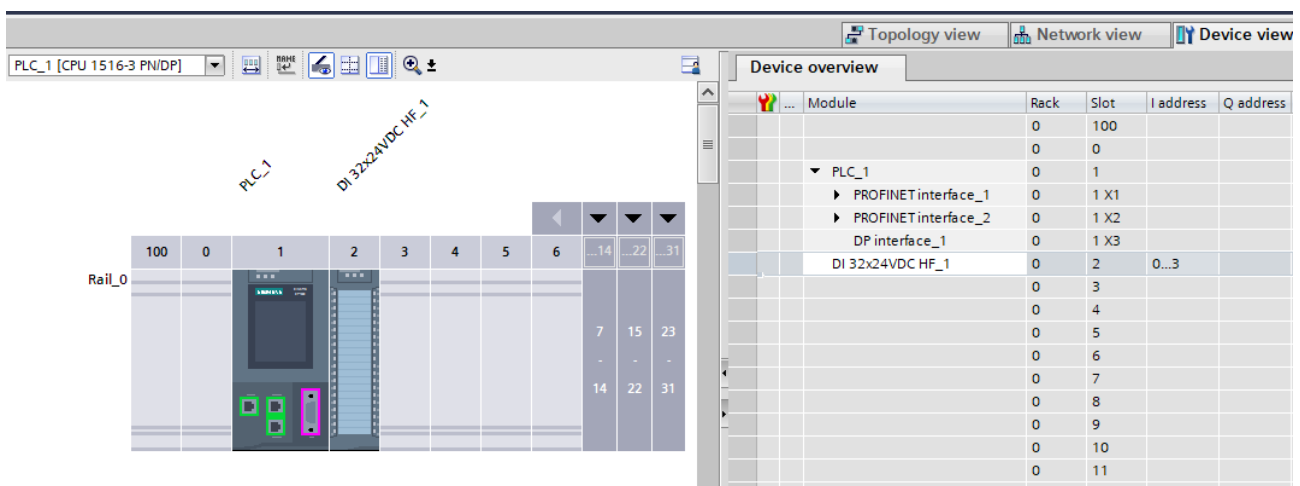
BYTE, WORD, DWORD, LWORD

Datatyper for hex-værdier. En hex-værdi går fra 0 til F
Byte fylder 8 bit
Word fylder 16 bit
Dword fylder 32 bit
Lword fylder 64 bit

Siemens adressering, talformat og word-instruktioner

Adressering

Hardware



Billedet viser en hardwareopsætning, hvor der er brugt en Siemens S7-1500 CPU. På modulplads 2 er der monteret et inputkort med 32 digitale indgange. I adressefeltet er adresse angivet som "0...3". Den værdi, der vises, er byteadressen på indgangen.

Det er denne type adresseringer og talformat, der behandles i dette afsnit.

En indgang kan adresseres på bit-, byte-, word- eller dword-niveau.

Der skal 8 bit til en byte, 16 bit til et word, og der skal 32 bit til et doubleword (dword).

Adresserne på indgangskort på position 2 bliver følgende:

| Bit | Byte | Word | D-Word |
|-------|------|------|--------|
| I 0.0 | IB 0 | IW 0 | ID 0 |
| I 0.1 | | | |
| I 0.2 | | | |
| I 0.3 | | | |
| I 0.4 | | | |
| I 0.5 | | | |
| I 0.6 | | | |
| I 0.7 | | | |
| I 1.0 | IB 1 | | |
| I 1.1 | | | |
| I 1.2 | | | |
| I 1.3 | | | |
| I 1.4 | | | |
| I 1.5 | | | |
| I 1.6 | | | |
| I 1.7 | | | |
| I 2.0 | IB 2 | IW 2 | |
| I 2.1 | | | |
| I 2.2 | | | |
| I 2.3 | | | |
| I 2.4 | | | |
| I 2.5 | | | |
| I 2.6 | | | |
| I 2.7 | | | |
| I 3.0 | IB 3 | | |
| I 3.1 | | | |
| I 3.2 | | | |
| I 3.3 | | | |
| I 3.4 | | | |
| I 3.5 | | | |
| I 3.6 | | | |
| I 3.7 | | | |

Hvis indgangen anvendes som en bit-værdi, så adresseres fra I 0.0, I 0.1, I 0.3 osv.

Hvis man adresserer via byte, skrives IB0 for de første 8 bit, IB 1 for de næste 8bit osv.

Ved word hedder det så IW0, IW2 og endelige på dword, så er adressen ID0.

Husk at være opmærksom på, at der er overlap. F.eks. dækker IW0 IB 0 og 1, IW 1 dækker IB 1 og 2 og IW 2 dækker IB 2 og 3.

Datatyper

I en Siemens PLC anvendes der en række forskellige datatyper, der anvendes, når man skal definere parametre.

5.4.1 Bool, Byte, Word, and DWord data types

Table 5-29 Bit and bit sequence data types

| Data type | Bit size | Number type | Number range | Constant examples | Address examples |
|-----------|----------|------------------|----------------------------------|-------------------|---|
| Bool | 1 | Boolean | FALSE or TRUE | TRUE | I1.0 Q0.1 M50.7 DB1.DBX2.3 Tag_name |
| | | Binary | 2#0 or 2#1 | 2#0 | |
| | | Unsigned integer | 0 or 1 | 1 | |
| | | Octal | 8#0 or 8#1 | 8#1 | |
| | | Hexadecimal | 16#0 or 16#1 | 16#1 | |
| Byte | 8 | Binary | 2#0 to 2#1111_1111 | 2#1000_1001 | IB2 MB10 DB1.DBB4 Tag_name |
| | | Unsigned integer | 0 to 255 | 15 | |
| | | Signed integer | -128 to 127 | -63 | |
| | | Octal | 8#0 to 8#377 | 8#17 | |
| | | Hexadecimal | B#16#0 to B#16#FF, 16#0 to 16#FF | B#16#F, 16#F | |

| Data type | Bit size | Number type | Number range | Constant examples | Address examples |
|-----------|----------|-------------------|--|---------------------------------|------------------------------|
| Word | 16 | Binary | 2#0 to 2#1111_1111_1111_1111 | 2#1101_0010_1001_0110 | MW10 DB1.DBW2 Tag_name |
| | | Unsigned integer | 0 to 65535 | 61680 | |
| | | Signed integer | -32768 to 32767 | 72 | |
| | | Octal | 8#0 to 8#177_777 | 8#170_362 | |
| | | Hexadecimal | W#16#0 to W#16#FFFF, 16#0 to 16#FFFF | W#16#F1C0, 16#A67B | |
| DWord | 32 | Binary | 2#0 to 2#1111_1111_1111_1111_1111_1111 | 2#1101_0100_1111_1110_1000_1100 | MD10 DB1.DBD8 Tag_name |
| | | Unsigned integer* | 0 to 4_294_967_295 | 15_793_935 | |
| | | Signed integer* | -2_147_483_648 to 2_147_483_647 | -400000 | |
| | | Octal | 8#0 to 8#37_777_777_777 | 8#74_177_417 | |
| | | Hexadecimal | DW#16#0000_0000 to DW#16#FFFF_FFFF, 16#0000_0000 to 16#FFFF_FFFF | DW#16#20_F30A, 16#B_01F6 | |

* The underscore "_" is a thousands separator to enhance readability for numbers greater than eight digits.

Integer data types

Table 5-30 Integer data types (U = unsigned, S = short, D= double)

| Data type | Bit size | Number Range | Constant examples | Address examples |
|-----------|----------|---------------------------------|-------------------|----------------------------|
| USInt | 8 | 0 to 255 | 78, 2#01001110 | MB0, DB1.DBB4, Tag_name |
| SInt | 8 | -128 to 127 | +50, 16#50 | |
| UInt | 16 | 0 to 65,535 | 65295, 0 | MW2, DB1.DBW2, Tag_name |
| Int | 16 | -32,768 to 32,767 | 30000, +30000 | |
| UDInt | 32 | 0 to 4,294,967,295 | 4042322160 | MD6, DB1.DBD8, Tag_name |
| DInt | 32 | -2,147,483,648 to 2,147,483,647 | -2131754992 | |

Floating-point real data types

Real (or floating-point) numbers are represented as 32-bit single-precision numbers (Real), or 64-bit double-precision numbers (LReal) as described in the ANSI/IEEE 754-1985 standard. Single-precision floating-point numbers are accurate up to 6 significant digits and double-precision floating point numbers are accurate up to 15 significant digits. You can specify a maximum of 6 significant digits (Real) or 15 (LReal) when entering a floating-point constant to maintain precision.

Table 5-31 Floating-point real data types (L=Long)

| Data type | Bit size | Number range | Constant Examples | Address examples |
|-----------|----------|--|-----------------------------|---|
| Real | 32 | -3.402823e+38 to -1.175 495e-38, ±0, +1.175 495e-38 to +3.402823e +38 | 123.456, -3.4, 1.0e-5 | MD100, DB1.DBD8, Tag_name |
| LReal | 64 | -1.7976931348623158e+308 to -2.2250738585072014e-308, ±0, +2.2250738585072014e-308 to +1.7976931348623158e+308 | 12345.123456789e40, 1.2E+40 | DB_name.var_name Rules: <ul style="list-style-type: none"> No direct addressing support Can be assigned in an OB, FB, or FC block interface table |

Calculations that involve a long series of values including very large and very small numbers can produce inaccurate results. This can occur if the numbers differ by 10 to the power of x, where $x > 6$ (Real), or 15 (LReal). For example (Real): $100\,000\,000 + 1 = 100\,000\,000$.

Time and Date data types

Table 5-32 Time and date data types

| Data type | Size | Range | Constant Entry Examples |
|-----------|---------|--|--|
| Time | 32 bits | T#-24d_20h_31m_23s_648ms to T#24d_20h_31m_23s_647ms Stored as: -2,147,483,648 ms to +2,147,483,647 ms | T#5m_30s T#1d_2h_15m_30s_45ms TIME#10d20h30m20s630ms 500h10000ms 10d20h30m20s630ms |
| Date | 16 bits | D#1990-1-1 to D#2168-12-31 | D#2009-12-31 DATE#2009-12-31 2009-12-31 |

| Data type | Size | Range | Constant Entry Examples |
|--------------------------------|-------------|---|---|
| Time_of_Day | 32 bits | TOD#0:0:0.0 to TOD#23:59:59.999 | TOD#10:20:30.400 TIME_OF_DAY#10:20:30.400 23:10:1 |
| DTL (Date and Time Long) | 12 bytes | Min.: DTL#1970-01-01-00:00:00.0 Max.: DTL#2262-04-11:23:47:16.854 775 807 | DTL#2008-12-16-20:30:20.25 0 |

BCD Formater

I en Siemens PLC ser BCD formatet ud sådan:

| Format | Size (bits) | Numeric Range | Constant Entry Examples |
|--------|-------------|---------------------|-------------------------|
| BCD16 | 16 | -999 to 999 | 123, -123 |
| BCD32 | 32 | -9999999 to 9999999 | 1234567, -1234567 |

Variable formater

I en Siemens PLC anvendes denne form for adressering på følgende variabler:

| Memory area | Description | Force | Retentive |
|--|--|-------|-------------------|
| I Process image input | Copied from physical inputs at the beginning of the scan cycle | No | No |
| I_:P ¹ (Physical input) | Immediate read of the physical input points on the CPU, SB, and SM | Yes | No |
| Q Process image output | Copied to physical outputs at the beginning of the scan cycle | No | No |
| Q_:P ¹ (Physical output) | Immediate write to the physical output points on the CPU, SB, and SM | Yes | No |
| M Bit memory | Control and data memory | No | Yes (optional) |
| L Temp memory | Temporary data for a block local to that block | No | No |
| DB Data block | Data memory and also parameter memory for FBs | No | Yes (optional) |

Ord-instruktioner

I det følgende vises nogle grundlæggende ordinstruktioner, som kan anvendes til en hel række forskellige formål.

For at PLC'en kan behandle talværdier korrekt, er det et krav, at der regnes med de samme taltyper (data types). Man kan ikke gange et kommatal med et heltal, uden at en af de to tal bliver konverteret.

I dette afsnit behandles følgende grundlæggende ord-instruktioner:

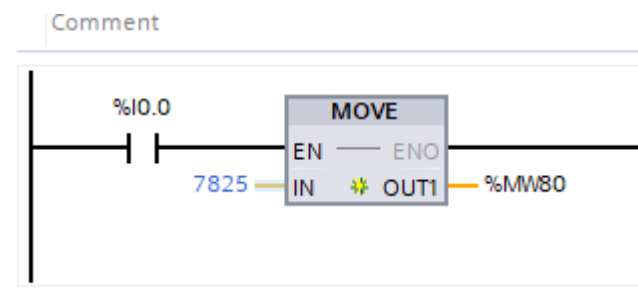
- Move (bruges til at flytte data)
- Talkonverteringer
- Matematiske funktioner

For at gøre forklaringerne nemmere, er der valgt kun at bruge memory-område for data.

Move-instruktion

Eksempel 1:

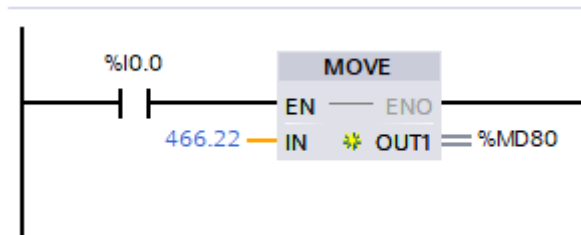
Der skal flyttes et heltal (INT) 7825 ind på adressen MW 80. Funktionen skal udføres, når I 0.0 er aktiv.



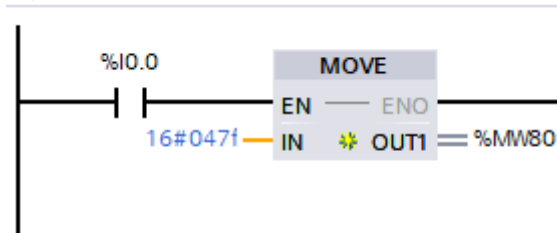
Move-instruktionen kan flytte alle typer af data fra "IN" til "OUT".

"EN" betyder enable, og er den indgang, der aktiverer move-instruktionen.

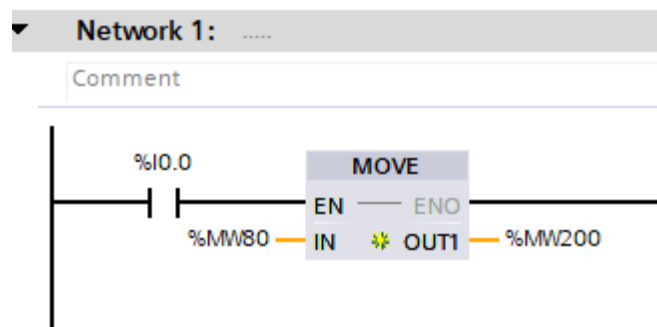
Når I 0.0 er aktive flyttes INT-tallet 7825 over på adresse MW80. I dette eksempel er man nødt til at bruge en wordadresse, som modtager, fordi et INT-tal fylder et word.



I dette eksempel er INT-tallet erstattet med et kommatal, som også kaldes et REAL-tal. Et REAL-tal fylder altid et dword, derfor bruges en dword-adresse som modtager, i dette eksempel er det MD 80.



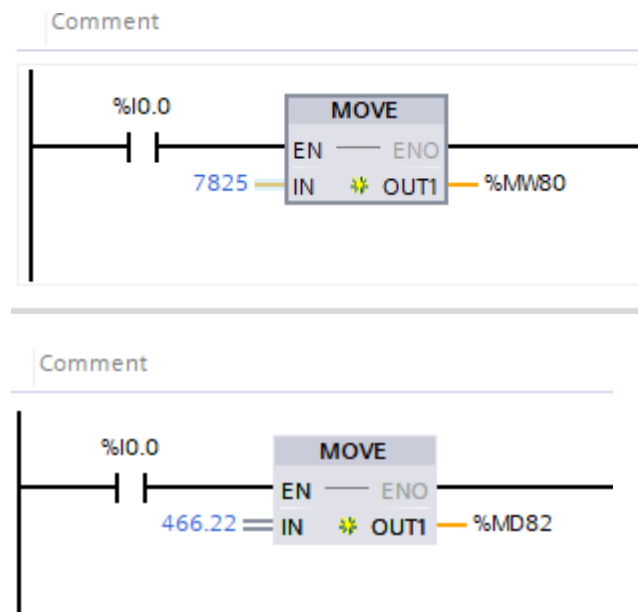
I dette eksempel flyttes en Hex-værdi, som fylder et word til MW 80.



Der kan også flyttes data mellem to forskellige adresse. I dette eksempel flyttes dataværdien fra MW 80 til MW 200.

Talkonverteringer

Eksempel 2:

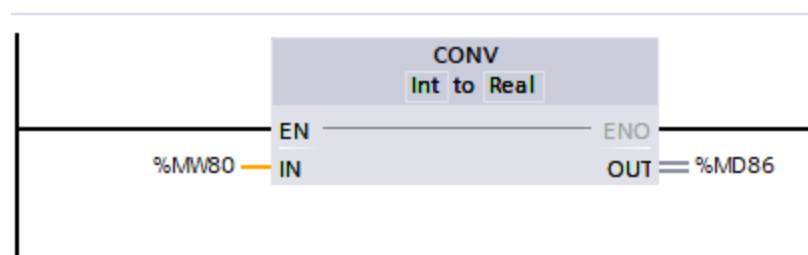


I dette eksempel har vi to talværdier, et INT-tal der er gemt på MW80, og et REAL tal som er gemt på MD82.

Disse to talværdier skal ganges med hinanden.

For at det kan lade sig gøre, skal det ene tal konverteres, så det er samme taltype.

I dette eksempel vil vi konvertere INT-tallet til REAL.



Instruktionen, som skal bruges, hedder CONV. I dette eksempel konverteres fra INT til REAL. INT-værdien på MW 80 ændres til en REAL-værdi og gemmes på MD86.

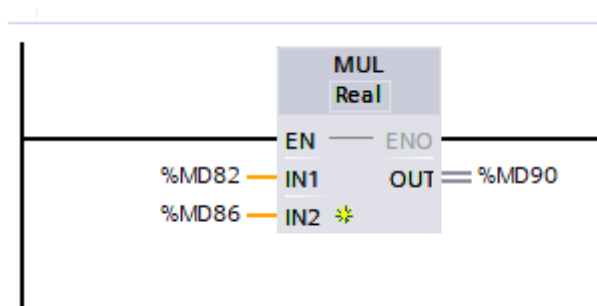
Matematiske funktioner

I PLC'en findes der en række matematiske funktioner, som bruges til beregninger.

Eksempel 3:

Vi fortsætter med det foregående eksempel, hvor vi har to REAL-tal, der skal ganges sammen.

De to REAL-tal er gemt på MD82 og MD86.



Instruktionen, som skal bruges, hedder MUL (multipliser). I eksemplet ganges værdierne fra MD82 og MD86 sammen, og resultatet placeres på MD90.

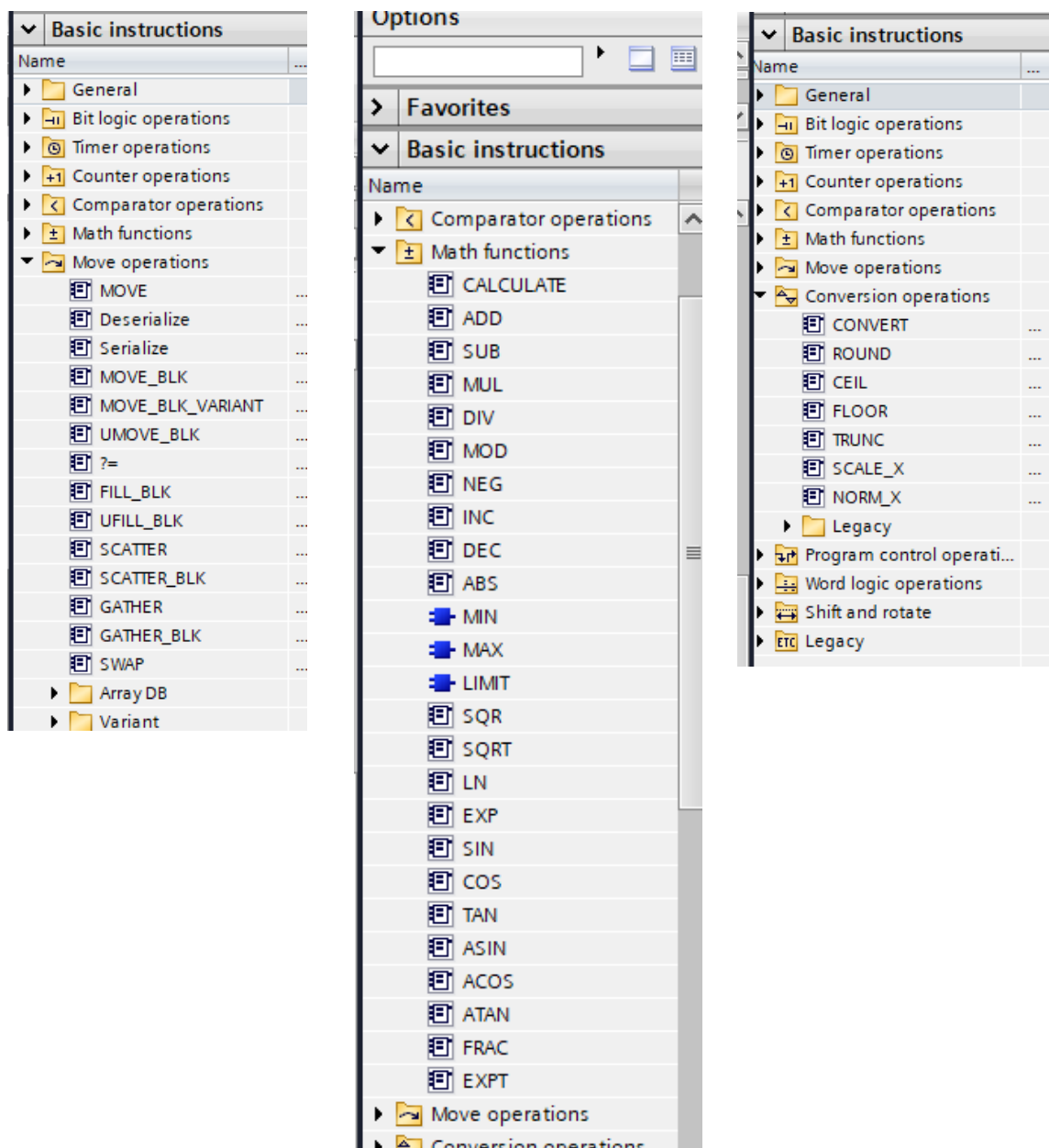
Datahukommelse

I dette eksempel er der brugt Memory-adresser for at flytte rundt på data. Det er valgt for at simplificere det.

I stedet for Memory-adresser vil man normalt anvende datablokke og lokale data – mere om det senere i kompendiet.

Instruktioner

De enkelte instruktioner kan findes i TIA-programmet under "Basic instructions".

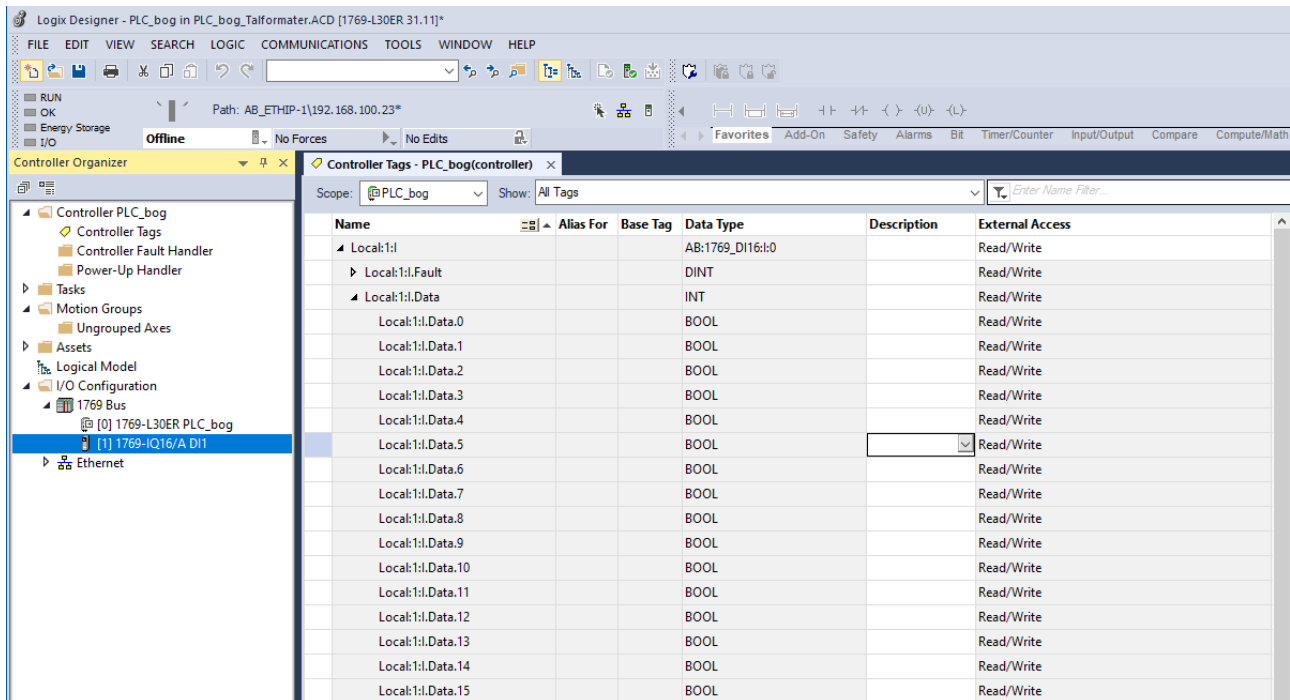


Der findes en lang række instruktioner. Hvis du vil læse mere om de enkelte instruktioner, skal du gå ind i din TIA-software og klik på funktionen i listen, så den er valgt. Derefter hent hjælp med knappen F1.

Allen-Bradley adressering, talformat og word-instruktioner

Adressering

Hardware



Billedet viser en hardwareopsætning, hvor der er brugt en Compact Logix L30ER CPU. I slot [1] er der monteret et inputkort 1769-IQ16/A med 16 digitale indgange.

I "Controller Tags" er kortes adresse angivet som "Local:1:I.Data.0" til "Local:1:I.Data.0.15". Dette er de 16 indganges bit-adresser. "Local:1:I.Data" er word-adressen for hele indgangskortet.

Det er denne type adresseringer og talformat, der behandles i dette afsnit.

En indgang kan adresseres på bit, byte og word niveau.

Der skal 8 bit til en byte, 16 bit til et word.

Adressering

Når man vil adresser forskellige data i PLC'en, er et kendskab til bit, byte og word en fordel.

Hvis vi i PLC'ens "Controller Tags" ser på et digitalt inputkort med 16 indgange, ser det således ud:

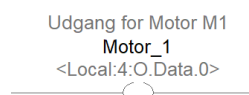
| Name | Alias For | Base Tag | Data Type | De | External Access | Constant | Style |
|-------------------|-----------|----------|------------------|------------|-----------------|--------------------------|---------|
| Local:4:O | | | AB:1769_DO16:0:0 | Read/Write | | <input type="checkbox"/> | |
| Local:4:O.Data | | | INT | Read/Write | | | Binary |
| Local:4:O.Data.0 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.1 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.2 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.3 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.4 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.5 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.6 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.7 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.8 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.9 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.10 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.11 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.12 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.13 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.14 | | | BOOL | Read/Write | | | Decimal |
| Local:4:O.Data.15 | | | BOOL | Read/Write | | | Decimal |

Bit adresser

Herunder er der lavet et alias tag for den første indgang på kortet. Tagget hedder "Start_knap", og det er alias for bit-adressen "Local:1:I.Data.0", hvor .0 angiver, at det er en bit-adresse.

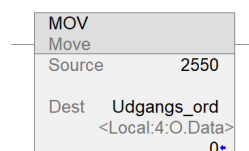
| Name | Alias For | Base Tag | Data Type | Description |
|---------|------------------|------------------|-----------|---------------------|
| Motor_1 | Local:4:O.Data.0 | Local:4:O.Data.0 | BOOL | Udgang for Motor M1 |

Når man programmerer, kan man bruge en Alias-adresse i stedet for den fysiske adresse. Det kan gøre det lettere at programmere, da man så skal huske på funktioner og ikke lange adresser. Når bit-adressen bruges i programmet, ser det således ud:



Word-adresser

Herunder bruges en ordinstruktion, som flytter værdien 2550 til tagget "Udgangs_ord", som er alias for et udgangskort med 16 udgange. Da det er en talværdi, skal der bruges et helt word. Adressen på kortet er "Local:4:O.Data". Bemærk, at her bruges .0 ikke, da det jo angav en bit-værdi.



Datatyper

I en Allen-Bradley PLC anvendes der en række forskellige datatyper, når man skal definere parametre.

PLC'en understøtter følgende datatyper defineret i IEC 61131-3. De foruddefinerede datatyper er:

Logisk niveau:

| Data type | Størrelse | Område | Beskrivelse |
|-------------|--|--|---|
| BOOL | 1-bit | 0 = cleared/off 1 = set/on | boolean |
| PLC visning | <div> <div>Name</div> <div>Tag_navn_0</div> </div> | <div> <div>Value</div> <div>0</div> </div> | <div> <div>Force Mask</div> <div></div> </div> <div> <div>Style</div> <div>Decimal</div> </div> <div> <div>Data Typ</div> <div>BOOL</div> </div> <div> <div>Description</div> <div></div> </div> <div> <div>Constant</div> <div><input type="checkbox"/></div> </div> |

Heltal med fortegn:

| | | | |
|-------------|--|---|---|
| SINT | 1-byte Short (kort) Integer | -128 to 127 | Short (kort) Integer. Heltal med fortegn. 8 bit, fra 0 to bit 7, hvor bit 7 er fortegn-bit. <div> <div>Bit 7 6 0</div> <div>1 111 1111</div> <div>Fortegn Betydende tal</div> </div> |
| PLC visning | <div> <div>Name</div> <div>Tag_navn_1</div> </div> | <div> <div>Value</div> <div>127</div> </div> | <div> <div>Force Mask</div> <div></div> </div> <div> <div>Style</div> <div>Decimal</div> </div> <div> <div>Data Typ</div> <div>SINT</div> </div> <div> <div>Description</div> <div></div> </div> <div> <div>Constant</div> <div><input type="checkbox"/></div> </div> |
| INT | 2-byte Integer | -32,768 to 32,767 | Word/ord. Heltal med fortegn. 16 bit, fra 0 to bit 15, hvor bit 15 er fortegn-bit. <div> <div>Bit 15 14 8 7 0</div> <div>1 111 1111 1111 1111</div> <div>Fortegn Betydende tal</div> </div> |
| PLC visning | <div> <div>Name</div> <div>Tag_navn_2</div> </div> | <div> <div>Value</div> <div>32767</div> </div> | <div> <div>Force Mask</div> <div></div> </div> <div> <div>Style</div> <div>Decimal</div> </div> <div> <div>Data Typ</div> <div>INT</div> </div> <div> <div>Description</div> <div></div> </div> <div> <div>Constant</div> <div><input type="checkbox"/></div> </div> |
| DINT | 4-byte Dobbelt Integer | -2,147,483,648 to 2,147,483,647 | Dobbelt word/ord. Heltal med fortegn. 32 bit, fra 0 to bit 31, hvor bit 31 er fortegn-bit. <div> <div>Bit 31 30 24 23 16 15 8 7 0</div> <div>1 111 1111 1111 1111 1111 1111</div> <div>Fortegn Betydende tal</div> </div> |
| PLC visning | <div> <div>Name</div> <div>Tag_navn_3</div> </div> | <div> <div>Value</div> <div>2147483647</div> </div> | <div> <div>Force Mask</div> <div></div> </div> <div> <div>Style</div> <div>Decimal</div> </div> <div> <div>Data Typ</div> <div>DINT</div> </div> <div> <div>Description</div> <div></div> </div> <div> <div>Constant</div> <div><input type="checkbox"/></div> </div> |

Komatal med fortegn:

| Data type | Størrelse | Område | Beskrivelse | | | | | | | | | | | | | | |
|-------------|--|---|--|----------|-------------|--------------------------|-------|----------|-------------|----------|------------|-----------|--|-------|------|--|--------------------------|
| REAL | 4-byte floating-point number | -3.402823E ³⁸ to -1.1754944E ⁻³⁸ (negative values) and 0 and 1.1754944E ⁻³⁸ to 3.402823E ³⁸ (positive values) | Dobbelt word/ord Hele tal med fortegn og flydende kommaplacering. Flydende koma vil sige, at dets placering bestemmes af eksponenten. <div><div><div><div>1.2345 =</div><div><div>12345</div><div>betydende tal</div></div><div>×</div><div><div><div>10⁻⁴</div><div>base</div></div><div>eksponent</div></div></div><div><div><div>Bit 313023220</div><div><div><div>1</div><div>1111 1111</div><div>11 1111 1111 1111 1111</div></div><div>FortegnEksponentBetydende tal</div></div></div></div></div></div> | | | | | | | | | | | | | | |
| PLC visning | <table><tr><th>Name</th><th>Value</th><th>Force Mask</th><th>Style</th><th>Data Typ</th><th>Description</th><th>Constant</th></tr><tr><td>Tag_navn_4</td><td>1234.1234</td><td></td><td>Float</td><td>REAL</td><td></td><td><input type="checkbox"/></td></tr></table> | | | Name | Value | Force Mask | Style | Data Typ | Description | Constant | Tag_navn_4 | 1234.1234 | | Float | REAL | | <input type="checkbox"/> |
| Name | Value | Force Mask | Style | Data Typ | Description | Constant | | | | | | | | | | | |
| Tag_navn_4 | 1234.1234 | | Float | REAL | | <input type="checkbox"/> | | | | | | | | | | | |

Dataformater til tekst

For at gemme bogstaver og tal som ASCII-tegn i tags bruges standard-datatypen STRING, som gemmer op til 82 tegn.

Hver STRING indeholder følgende:

| Name | Data Type | Beskrivelse | Notes | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------|---|-----------------------|--|-------------|-----------|-------------|--------------------------|------|-------|------------|-------|-----------|-------------|----------|----------|--|---------|-------|--------|--|--------------------------|--------------|--|---|---------|------|--|--|---------------|--|-------|-------------|----------|--|--|------------------|--|-----|-------|------|--|--|------------------|--|-----|-------|------|--|--|------------------|--|-----|-------|------|--|--|------------------|--|-----|-------|------|--|--|------------------|--|-----|-------|------|--|--|------------------|--|--------|-------|------|--|--|
| LEN | DINT | antal tegn i strengen | LEN opdaterer automatisk til det nye antal tegn, når du bruger: <ul style="list-style-type: none">Strengbrowseren for at indtaste tegn.Instruktioner, der læser, konverterer eller manipulerer en streng. LEN viser længden af den aktuelle streng. DATA-medlemmet kan indeholde yderligere, gamle tegn, som ikke er inkluderet i LEN-antallet. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DATA | SINT-array | ASCII-tegn i strengen | For at få adgang til tegnene i strengen skal du adressere navnet på tagget. For at få adgang til tegnene i string_1-tagget skal du f.eks. indtaste string_1. Hvert element i DATA-arrayet indeholder et tegn. Opret nye strengtyper, der gemmer færre eller flere tegn. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| PLC-visning | <table><tr><th>Name</th><th>Value</th><th>Force Mask</th><th>Style</th><th>Data Type</th><th>Description</th><th>Constant</th></tr><tr><td>String_1</td><td></td><td>'Tekst'</td><td>{...}</td><td>STRING</td><td></td><td><input type="checkbox"/></td></tr><tr><td>String_1.LEN</td><td></td><td>5</td><td>Decimal</td><td>DINT</td><td></td><td></td></tr><tr><td>String_1.DATA</td><td></td><td>{...}</td><td>{...} ASCII</td><td>SINT[82]</td><td></td><td></td></tr><tr><td>String_1.DATA[0]</td><td></td><td>'T'</td><td>ASCII</td><td>SINT</td><td></td><td></td></tr><tr><td>String_1.DATA[1]</td><td></td><td>'e'</td><td>ASCII</td><td>SINT</td><td></td><td></td></tr><tr><td>String_1.DATA[2]</td><td></td><td>'k'</td><td>ASCII</td><td>SINT</td><td></td><td></td></tr><tr><td>String_1.DATA[3]</td><td></td><td>'s'</td><td>ASCII</td><td>SINT</td><td></td><td></td></tr><tr><td>String_1.DATA[4]</td><td></td><td>'t'</td><td>ASCII</td><td>SINT</td><td></td><td></td></tr><tr><td>String_1.DATA[5]</td><td></td><td>'\$00'</td><td>ASCII</td><td>SINT</td><td></td><td></td></tr></table> | | | | | | | Name | Value | Force Mask | Style | Data Type | Description | Constant | String_1 | | 'Tekst' | {...} | STRING | | <input type="checkbox"/> | String_1.LEN | | 5 | Decimal | DINT | | | String_1.DATA | | {...} | {...} ASCII | SINT[82] | | | String_1.DATA[0] | | 'T' | ASCII | SINT | | | String_1.DATA[1] | | 'e' | ASCII | SINT | | | String_1.DATA[2] | | 'k' | ASCII | SINT | | | String_1.DATA[3] | | 's' | ASCII | SINT | | | String_1.DATA[4] | | 't' | ASCII | SINT | | | String_1.DATA[5] | | '\$00' | ASCII | SINT | | |
| | Name | Value | Force Mask | Style | Data Type | Description | Constant | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | String_1 | | 'Tekst' | {...} | STRING | | <input type="checkbox"/> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | String_1.LEN | | 5 | Decimal | DINT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | String_1.DATA | | {...} | {...} ASCII | SINT[82] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | String_1.DATA[0] | | 'T' | ASCII | SINT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | String_1.DATA[1] | | 'e' | ASCII | SINT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | String_1.DATA[2] | | 'k' | ASCII | SINT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | String_1.DATA[3] | | 's' | ASCII | SINT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | String_1.DATA[4] | | 't' | ASCII | SINT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| String_1.DATA[5] | | '\$00' | ASCII | SINT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Ord-instruktioner

I det efterfølgende vises nogle grundlæggende ordinstruktioner, som kan anvendes til en helt række forskellige formål.

For at PLC'en kan behandle talværdier korrekt, så er det et krav at der regnes med de samme taltyper. Man kan ikke gange et kommatal med et heltal, uden at en af de to tal bliver konverteret.

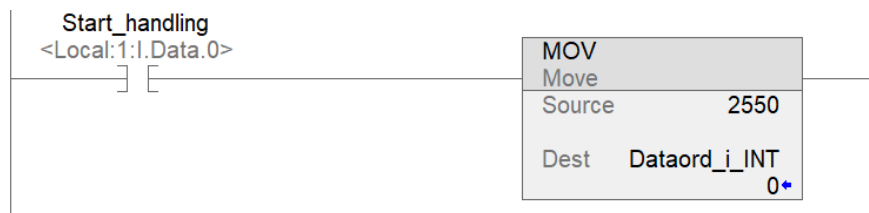
I dette afsnit behandles følgende grundlæggende ord-instruktioner:

- Move (bruges til at flytte data)
- Talkonverteringer
- Matematiske funktioner

For at gøre forklaringerne nemmere er der valgt primært at bruge base tags i de næste eksempler.

Move-instruktion

Der skal flyttes et heltal (INT) 2550 ind i tagget "Dataord_I_INT". Funktionen skal udføres, når indgang "Local:1:I.Data.0" er aktiv.

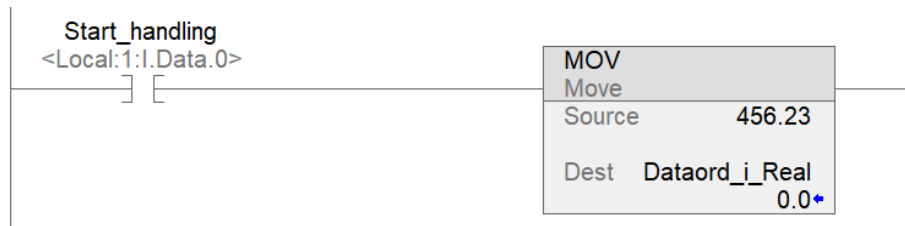


Move-instruktionen kan flytte alle typer af data fra "Source" til "Dest".

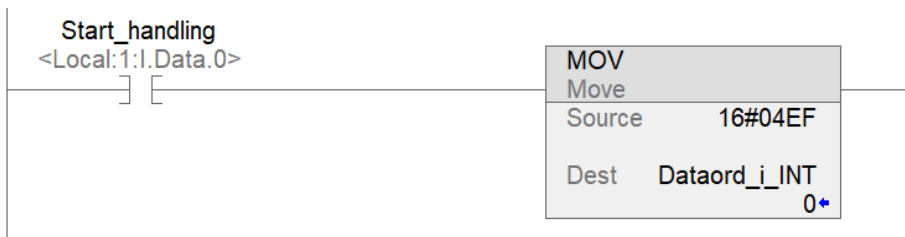
Når der er signal ind på blokken, udføres move-instruktionen.

Når indgangen "Local:1:I.data.0" er aktiv, flyttes INT-tallet 2550 over på tagadressen "Dataord_I_INT".

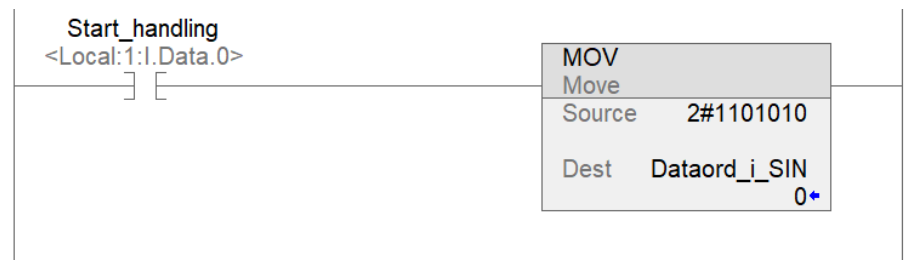
Den binærkode for 2550_{dec} er 1001 1111 0110_{bin}. Altså bruger decimaltallet 12 bit. Derfor skal der bruges et tag med data type "INT". Denne datatype fylder et word altså 16 bit.



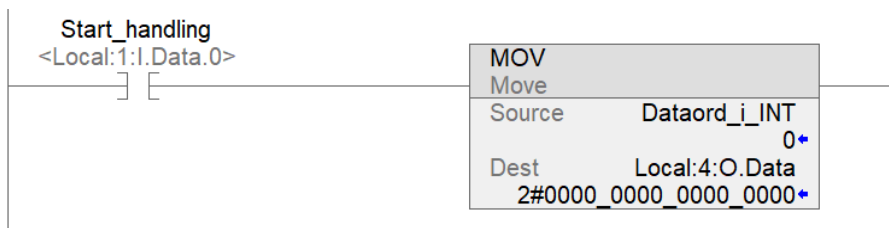
I dette eksempel er INT-tallet erstattet med et kommatal, som også kaldes et REAL-tal. Et REAL-tal fylder altid 32 bit, derfor bruges et tag med datatype "REAL".



I dette eksempel flyttes en Hex-værdi, som fylder et word til tagadresse "Dataord_i_INT", som har datatype INT og fylder 16 bit.

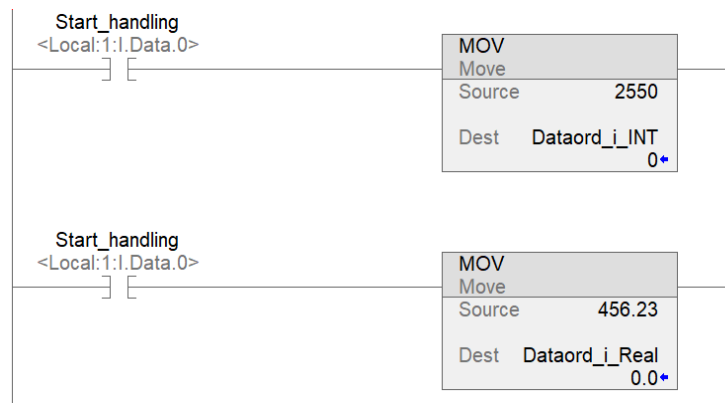


I dette eksempel flyttes en binærværdi, der fylder 7 bit-værdi til tagadresse "Dataord_i_SINT", som har datatype SINT og fylder 8 bit.



Der kan også flyttes data mellem to forskellige adresse. I dette eksempel flyttes dataværdien fra tagget "Dataord_i_INT" til det digitale udgangskort "Local:4:O.Data".

Talkonverteringer



I dette eksempel har vi to talværdier, et helt tal uden komma 2550, der er gemt på "Dataord_I_INT", og et komma tal 456.23, som er gemt i "Dataord_I_Real".

Disse to talværdier, med meget forskellig opbygning af den bagvedliggende binærkode, skal ganges med hinanden.

Det er ikke noget problem fordi, PLC'en automatisk vil konvertere det hele tal i INT til et REAL-tal. Derefter vil de blive ganget sammen. Om resultatet er i INT eller REAL, afhænger af den datatype, man har givet det tag, som resultatet gemmes i. OBS. CPU'en bruger ekstra resurser på disse konverteringer, så ved store programmer er det en fordel, hvis datatyperne passer sammen.

For yderligere oplysninger om autokonvertering se hjælp i PLC-programmeringssoftwaren Logix Designer – søg på "Data Conversions".

Det, at PLC'en automatisk konverterer, betyder, at det kun er i ganske særlige situationer, der skal bruges konverteringsinstruktioner. Disse er vist i tabellen nedenfor.

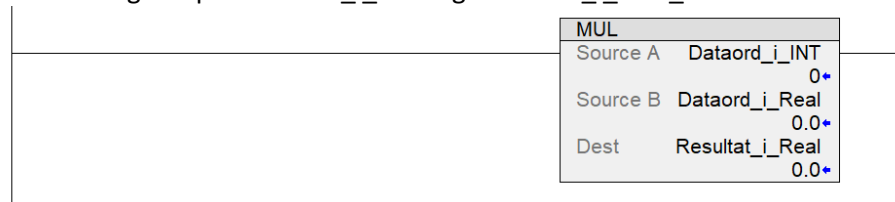
| Hvis du vil | Brug instruktion |
|---|------------------|
| Konverter radianer til grader. | DEG |
| Konverter grader til radianer. | RAD |
| Konverter en heltalsværdi INT til en BCD-værdi. | TOD |
| Konverter en BCD-værdi til en INT heltalsværdi. | FRD |
| Fjern brøkdelen af en værdi. | TRN |

Matematiske funktioner

I PLC'en findes der en række matematiske funktioner, som bruges til beregninger.

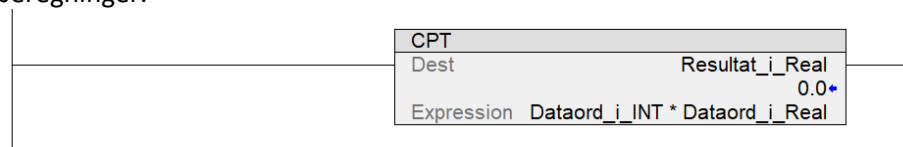
Vi fortsætter med det foregående eksempel, hvor vi har et INT-tal og et REAL-tal, der skal ganges sammen.

De to tal er gemt på "Dataord_i_INT" og "Dataord_i_Real".



Instruktionen, som skal bruges, hedder MUL (multiplikation). I eksemplet ganges værdierne fra de to tags sammen, og resultatet placeres på tagget "Resultat_i_Real".

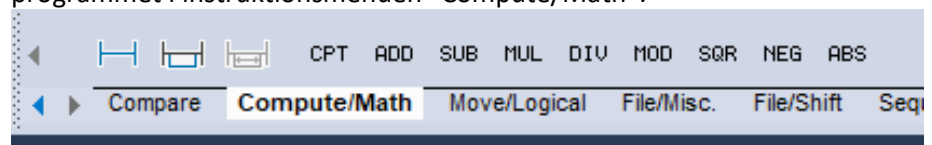
En anden måde at udføre beregningen ovenfor er at bruge instruktionen "CPT Compute". Denne instruktion kan være en fordel ved større beregninger.



OBS. Dette er en instruktion, der kun findes i Allen-Bradley PLC'er, da det ikke er en IEC-standardinstruktion.

Instruktioner

De enkelte matematik-instruktioner kan findes i Logix Designer-programmet i instruktionsmenuen "Compute/Math":



Forklaring til mest brugte instruktioner:

| | | |
|-----|---|----------------------|
| ADD | + | plus/addition |
| SUB | - | minus/substraktion |
| MUL | * | gange/multiplikation |
| DIV | / | division |
| SQR | √ | kvadratrod |

Der findes en lang række instruktioner. Hvis du vil læse mere om de enkelte instruktioner, så gå ind i dit Logix Designer-program og læs i hjælp.

Siemens bloktyper, OB, FC, FB, og DB

Generelt

Når man arbejder med PLC-programmering på en Siemens PLC, er det nødvendigt at kende blokbegrebet, bloktyperne og deres anvendelse.

Der er hensigtsmæssig at opdele sit PLC-program i mindre blokke.

Herved forbedres overskueligheden og struktureringen. Ligeledes er det hensigtsmæssig at bestemme, hvilke blokke der skal kaldes op og i hvilken rækkefølge.

Bloktyper

I en Siemens PLC findes der følgende bloktyper:

OB: Organisationsblokke

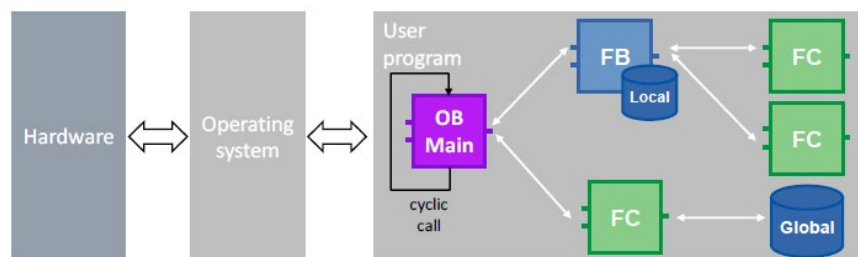
FB: Funktionsblokke

FC: Funktioner

DB: Datablokke

Struktur

OB, FB og FC indeholder PLC-program, mens en DB indeholder data.



PLC'ens operativsystem kalder de forskellige organisationsblokke (OB) op. Hver OB har sin egen funktion, mere om det senere.

Den vigtigste OB er OB1.

OB1 er cyklisk programbearbejdning, hovedprogrammet der bearbejdes cyklisk (i hvert scan).

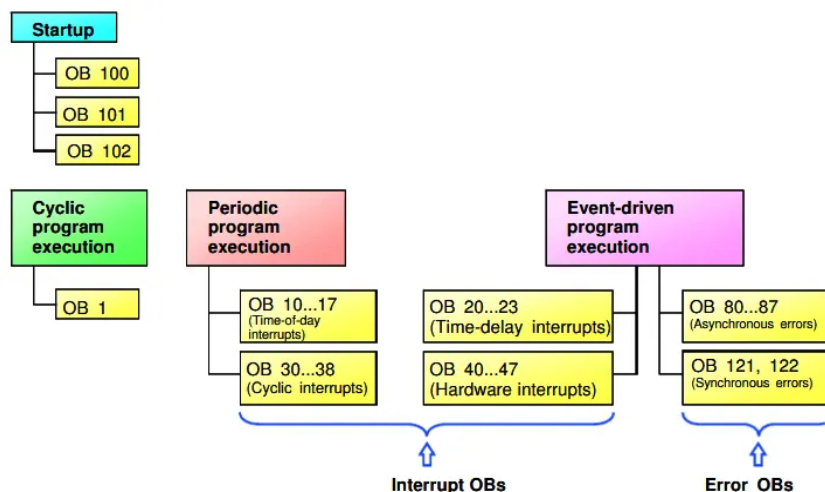
Det betyder, at når PLC'en sættes i run, så bearbejdes det PLC-program, som er gemt i OB1. OB1 er den blok på ovenstående billede, der er benævnt OB main.

Fra OB1 kaldes de funktioner (FC) og funktionsblokke (FB), der indeholder PLC-programmet.

Fra alle programelementer er det muligt at hente data fra en datablok (DB). Datablokke kan være lokale eller globale.

Organisationsblokke, OB

OB'er kaldes op af PLC'ens operativsystem, når en given hændelse indtræder.



Billedet viser et eksempel på forskellige type af OB'er.

Startup:

Det er OB'er, som gennemløbes en gang ved opstart.

Cyclic program execution:

Cyklisk program, som bearbejdes en gang for hvert scan.

Periodic program execution:

Er programblokke der læses med fast scantid, eller på faste tidspunkter.

Interrupt:

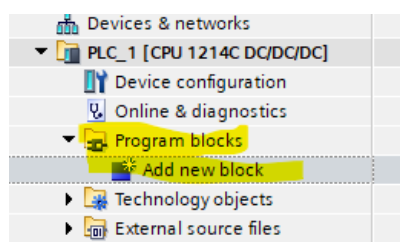
Det er OB'er, som kaldes ved bestemte hændelser, f.eks. hardwarefejl.

Error OB:

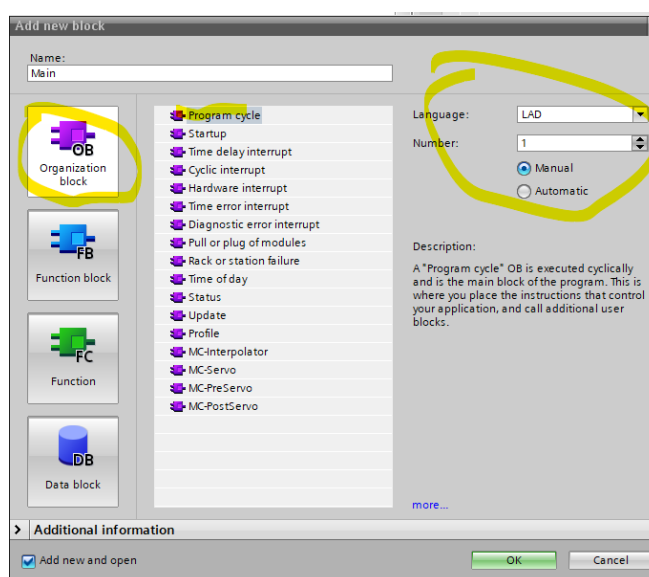
Det er OB'er, som kaldes, når der opstår en fejl.

Det er ikke alle Siemens PLC'er, der har alle typer af OB'er. Det skal man læse i PLC'ens data.

Indsæt OB i programmet



For at indsætte en OB, klikkes på "Add new block".



I dette eksempel indsættes en OB1.

Husk at vælge manuel nummerering, ellers tager programmet det næste frie nummer.

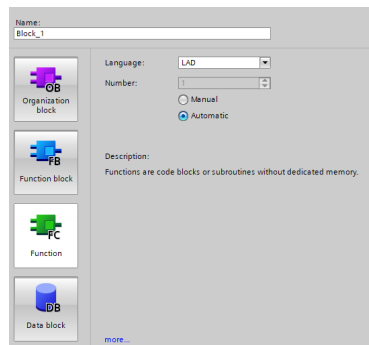
Der er kun medtaget 3 forskellige OB i dette eksempel. For en mere detaljeret beskrivelse skal man se i manualen for CPU'en.

OB 1: Cyklisk programbearbejdning, hovedprogrammet der bearbejdes cyklisk (i hvert scan).

OB 200: Hardwareinterrupt, kaldes når et modul melder hardware interrupt.

OB 100: Startprogram, initialisering. Kaldes op under start, hvorved systemet kan initialiseres.

Funktioner, FC

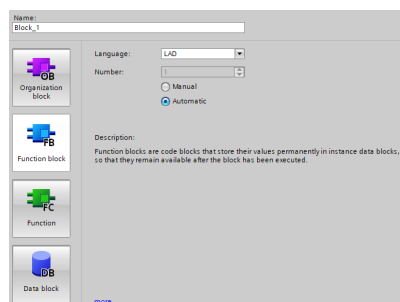


En FC er en programblok, som bruges til cyklisk programbearbejdning. En FC har ikke tilknyttet en fast datablok, derfor gemmes blokparametre ikke ved gennemløb.

En FC kan kaldes med parametring, og den har mulighed for at returnere resultat.

En FC bliver f.eks. brugt til almindelige digitale programmer

Funktionsblok (FB)

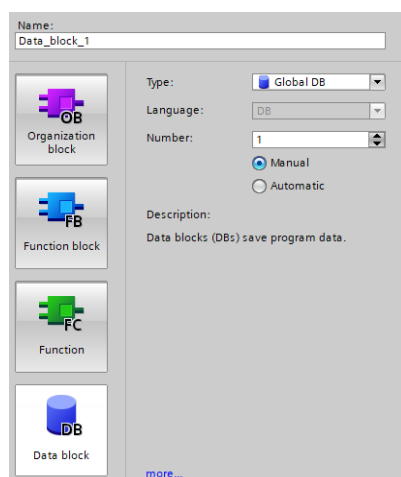


En FB ligner på mange måder en FC. Den store forskel er dog, at der altid tilknyttes en datablok til en FB. Det betyder, at blokparametre bliver gemt for hvert gennemløb.

Den faste datablok, der tilknyttes kaldes en Instance DB.

En FB bliver f.eks. brugt til en PID-regulator.

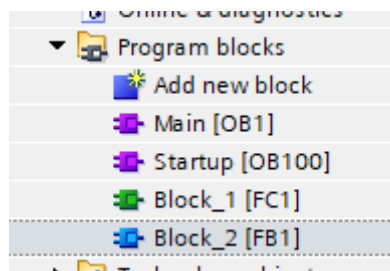
Datablokke, DB



Det viste eksempel er en Global DB, hvilket betyder, at den kaldes fra alle programblokke.

En datablok indeholder forskellige typer af data. Det kan f.eks. være opsamlede målinger, setpunkter, alarmpunkter osv.

Eksempel



I dette eksempel er der fire blokke:

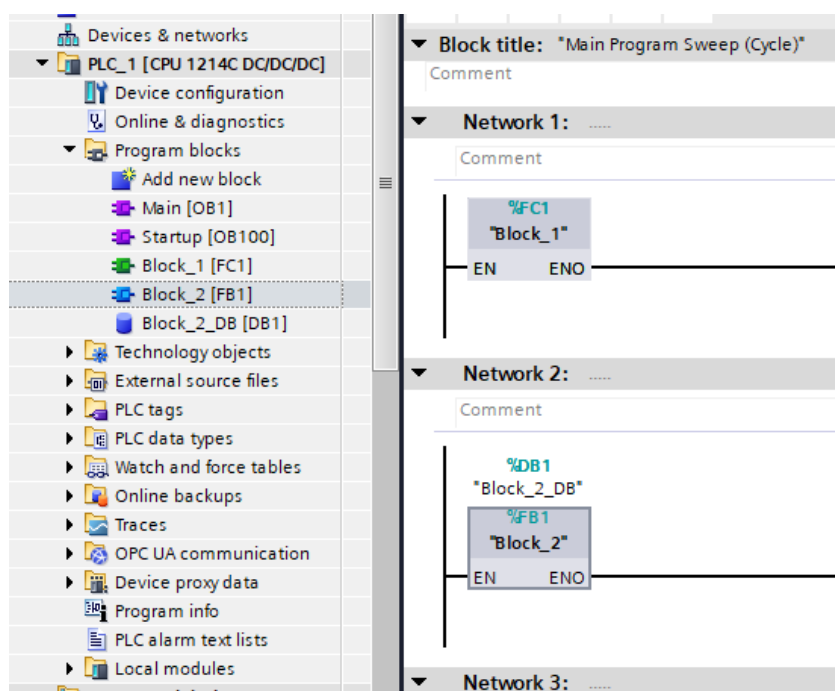
OB 1: Cyklisk programbearbejdning

OB 100: Startup OB, som gennemløbes en gang ved opstart

FC 1: Function

FB 1: Functionblock

Programstruktur

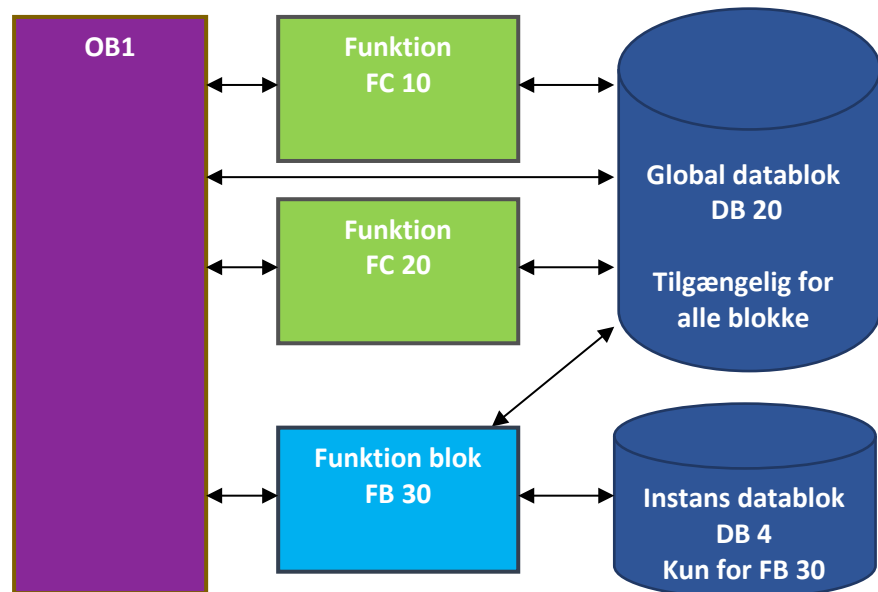


De to programblokke trækkes over i OB 1 med drag and drop.

Når man trækker FB over, kommer programmet automatisk og spørger på valg af datablok (DB). I dette eksempel er der brugt en DB1. DB1 oprettes automatisk og tilføjes bloklisten.

I bogen findes der et kapitel, som beskriver oprettelse og anvendelse af FC/FB samt et kapitel, der skriver mere om datablokke (DB).

Siemens Datablokke



Beskrivelse

Datablokke (DB'er) er blokke i CPU-lageret, i hvilke programmet lagrer data. Når en blok (FC, FB eller OB) kaldes op, belægger den lagerplads i det lokale dataområde.

Foruden dette lokale dataområde kan blokken åbne et lagerområde i form af en DB. I modsætning til dataene i det lokale område bliver dataene i DB'en ikke slettet, når DB'en lukkes, eller bearbejdningen af blokken er afsluttet.

Der er to forskellige typer af datablokke, som anvendes til forskellige formål, afhængig af deres forbindelse til programblokken.

Fælles datablokke (Globale DB'er)

Globale (fælles) datablokke kan anvendes af alle logiske blokke i programmet. Alle FB'er, FC'er og OB'er kan læse data fra de globale (fælles) DB'er og skrive data ind i dem. Før programblokkene kan få adgang til dataene, skal datablokken åbnes. Data i DB'erne forbliver lagrede, når datablokken er lukkes.

Instansdatablokke

En instansdatablok er tilordnet til en funktionsblok. Herved kan de data, der er lagret i datablokken, udelukkende læses og skrives af den tilordnede funktionsblok.

Instansdatablokken forsyner funktionsblokken med lagerplads, og de data, der er lagret i datablokken, bliver ikke slettet, når blokken lukkes (i modsætning til lokale data i en funktion eller en funktionsblok, som bliver slettet, når blokken lukkes). En funktionsblok kan have flere tilordnede instansdatablokke, hvis FB'en kaldes op flere gange.

Datatyper i Siemens PLC

I en Siemens PLC findes der en række forskellige datatyper, det er nærmere beskrevet i kapitlet omkring "Siemens adressering, talformater og Word-instruktioner".

De efterfølgende tabeller er nogle eksempler på datatyper.

Elementære datatyper

| Data type | Length (in bits) | Constants | Variables |
|-----------|------------------|------------------|-----------|
| BOOL | 1 | 1 or 0 | I 1.0 |
| BYTE | 8 | B#16#A9 | MB70 |
| WORD | 16 | W#16#12AF | MW72 |
| DWORD | 32 | DW#16#ADAC1EF5 | QD40 |
| LWORD | 64 | LW#16#5F52DE8B | |
| CHAR | 8 | 'w' | DBB4 |
| INT | 16 | 123 | #Value |
| DINT | 32 | L#65539 | MD80 |
| REAL | 32 | 1.2 or 34.5E-12 | DBD60 |
| SINT | 8 | +/-50 | MB24 |
| USINT | 8 | 50 | MB24 |
| UDINT | 32 | 4875678 | DBD64 |
| LREAL | 64 | LREAL#1.0e-5 | |
| LINT | 64 | LINT#+1543258759 | |

Elementære datatyper bestemmer den nødvendige hukommelsesstørrelse, f.eks. fylder datatypen "WORD" 16 bit i hukommelseslageret (arbejdslageret).

Datatyper for timer, dato og time of day

| Data type | Length (in bits) | Example |
|-----------------------------|------------------|-----------------------------------|
| Timers | | |
| SSTIME | 16 | SST#5s |
| TIME | 32 | T#2h46m30s630ms |
| LTIME | 64 | LT#11350d20h25m14s830ms652us315ns |
| Date and Time-of-day | | |
| DATE | 32 | D#1984-01-01 |
| TIME_OF_DAY (TOD) | 32 | TOD#18:15:18:999 |
| L_TIME_OF_DAY (LTOD) | 64 | LTOD#10:20:30.400_365_215 |
| DATE_AND_TIME (DT) | 64 | DT#1984-01-21-13-45-56-567-3 |
| DATE_AND_LTIME (LDT) | 64 | LDT#2008-01-01-18:00:30:25 |
| DTL | 96 | DTL#1984-01-01-18:00:30:250 |

Komplekse datatyper

| Data type | Length (in bits) | Example |
|--|-------------------------------|---|
| STRING (character string with max. 254 characters) | 8 * (number of characters +2) | 'This is a string' 'SIEMENS' |
| ARRAY (Group of components of the same data type) | user-defined | Measured values: ARRAY[1..20] of INT |
| STRUCT (Structure, Group of components of different data types) | user-defined | Motor: STRUCT Speed : INT Current : REAL END_STRUCT |
| UDT (User Defined Data Type = "Template" consisting of elementary or complex data types) | user-defined | UDT as block UDT as array element STRUCT Speed : INT Current : REAL END_STRUCT Drive: ARRAY[1..4] of UDT1 |

Komplekse datatyper kan kun anvendes i forbindelse med variable deklareret i globale datablokke.

Elementære datatyper

Datatyphen bestemmer den nødvendige hukommelsesstørrelse, f.eks. fylder datatypen "WORD" 16 bit i hukommelseslageret(arbejdslageret).

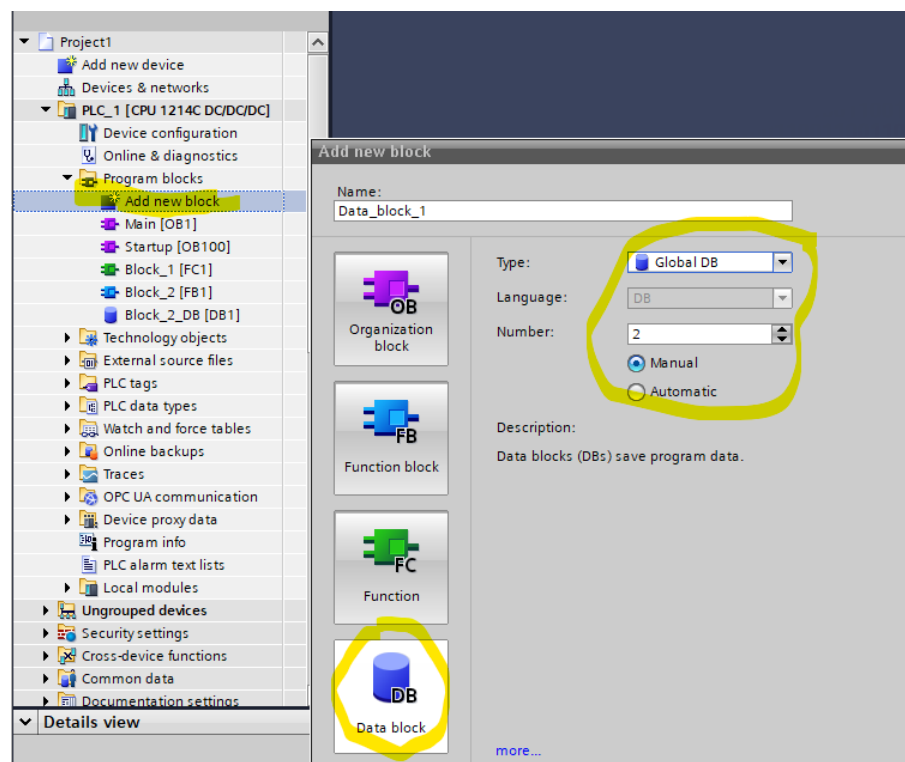
Brugerdefinerede datatyper

En brugerdefineret datatype kan anvendes som en datablok eller som en datatype i en variabel deklARATIONstabel.

UDT fremstilles med datablokeeditoren (LAD, FBD, STL Editor).

En UDT-struktur kan indeholde forskellige kombinationer af elementære og/eller komplekse datatyper. Se mere i kapitlet "Siemens User Defined Type (UDT)".

Oprettelse af Global DB



For at oprette en ny DB vælges "Add new block" – Vælg DB og brug type "Global DB". Der er valgt manuel nummerering, hvilket betyder, at programmøren selv vælger nummer.

Editering af datablok

Det efterfølgende eksempel viser en DB, som er oprettet med nogle variable af forskellige datatyper.

Data_block_1

| | Name | Data type | Start value | Retain | Accessible f... | Writa... | Visible in ... | Setpoint | Comment |
|----|----------------------|------------------------|-------------|--------|-------------------------------------|-------------------------------------|-------------------------------------|----------|--------------------------|
| 1 | Static | | | | | | | | |
| 2 | hjælpe_merke | Bool | false | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Settes true eller false |
| 3 | Klasse_1 | Int | 4000 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Læg 400 som startværdi |
| 4 | Retur_value | Real | 0.0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Returværdi |
| 5 | Felt_timer_Value | Array[0..10] of IEC... | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 6 | Felt_timer_Value[0] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 7 | Felt_timer_Value[1] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 8 | Felt_timer_Value[2] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 9 | Felt_timer_Value[3] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 10 | Felt_timer_Value[4] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 11 | Felt_timer_Value[5] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 12 | Felt_timer_Value[6] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 13 | Felt_timer_Value[7] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 14 | Felt_timer_Value[8] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 15 | Felt_timer_Value[9] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 16 | Felt_timer_Value[10] | IEC_TIMER | | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | | Felt med 10 timerværdier |
| 17 | <Add new> | | | | | | | | |

Navn på variabel

Datatype

Startværdi, som variabel antager ved opstart

Ved at klikke "retain" så huskes værdien ved stop

I disse tre kolonner kan der blokeres for, at HMI-panelet kan tilgå DB'ens data

Beskrivelse

Databloktabellen indeholder indlæsninger til adresse, variabel, datatype, begyndelsesværdi og kommentar til variablerne. Hver linje indeholder en variabeldeklaration. Til "Felt_timer Vallue" som er et Array, er det nødvendigt med flere linjer.

Navn

Dette er det alfanumeriske, symbolske navn for dataelementet.

Type

Dette er datatypen for den enkelte variabel. I dette eksempel har vi Bool (binær), INT (heltal), Real (kommatal) og en IEC timerformat.

Begyndelsesværdi

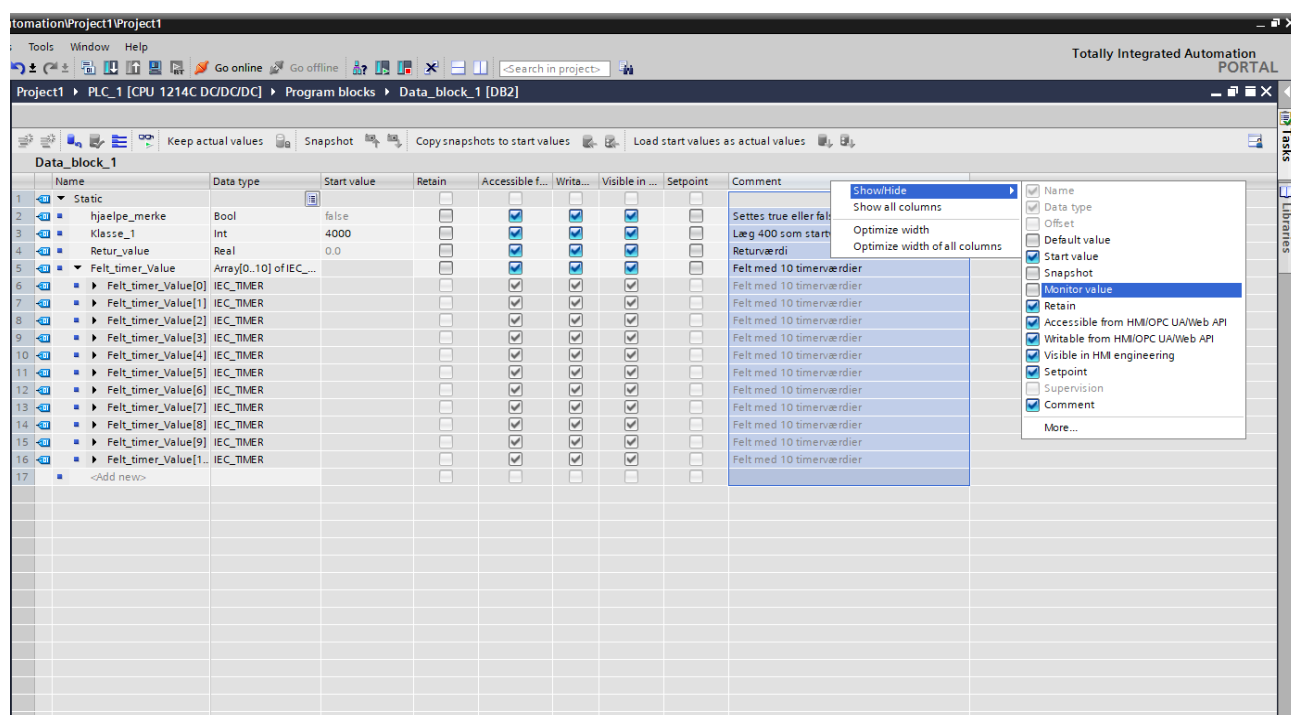
I feltet fastlægges begyndelsesværdien for variabelen, når PLC'en settes i run.

Retain

Det betyder, at der er hukommelse på variablen, hvilket medfører, at hvis PLC kobles i Stop, så huskes værdien.

HMI

De tre kolonner med HMI betyder, at programmøren har mulighed for at blokere for, at et HMI-system kan tilgå DB-variabler.



Hvis du højreklikker på DB-tabellen, har du mulighed for at til- eller frakoble flere variabler.

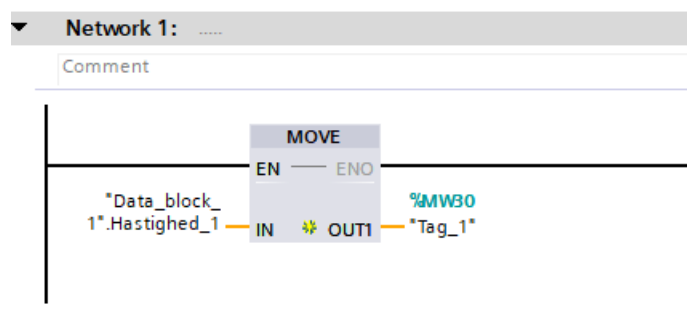
Eksempel på en struktur for en global DB

| Data_block_1 | | | | |
|--------------|-------------|-----------|-------------|-----------------------------|
| | Name | Data type | Start value | Comment |
| Static | | | | |
| | Hastighed_1 | Int | 500 | Korer med 500 omdrejninger |
| | Hastighed_2 | Int | 1000 | Korer med 1000 omdrejninger |
| | Max_Stroem | Real | 0.0 | Værdi for motorstrøm |
| | Motor_temp | Real | 0.0 | Motor temperatur |

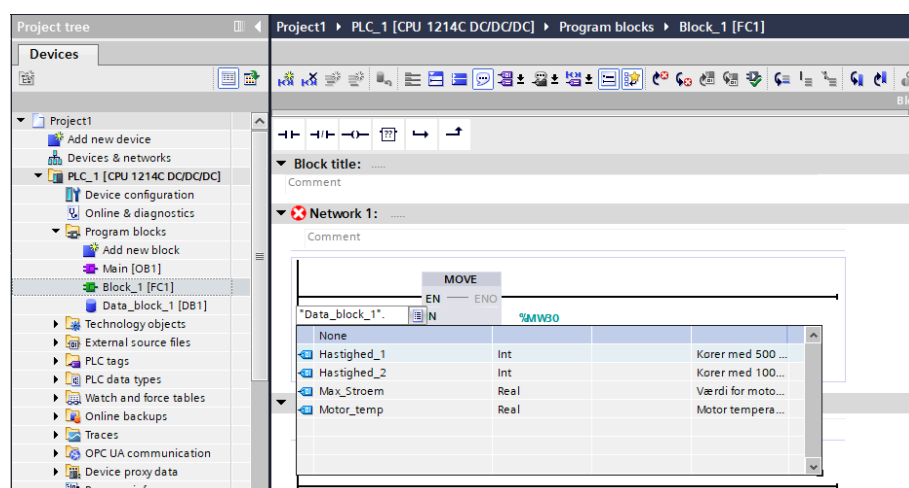
Ovenstående billede viser et eksempel på en datablok med fire variabler.

Man kan i dette eksempel ikke se de enkelte absolutte adresser. Man kun se de symbolske adresser.

Move



I eksemplet hentes "Hastighed_1" fra Datablok 1 og flyttes (move) til MW30



Når adressen for datablok indtastes, skrives først navnet på datablokken, derefter kommer der en liste frem, hvor variabelen kan vælges.

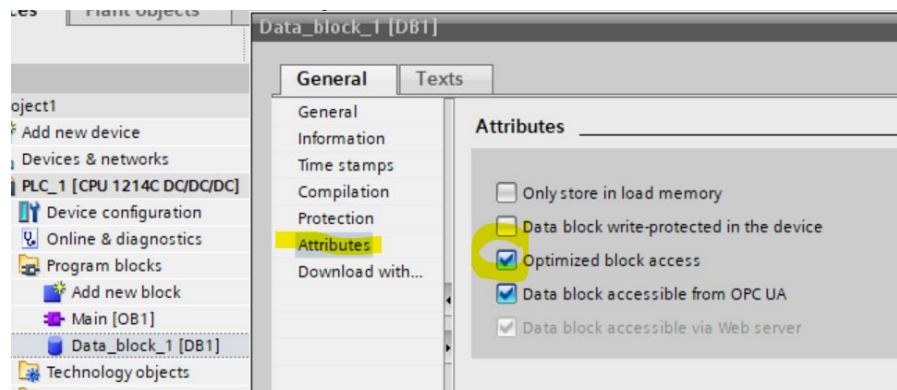
Symbolsk/absolut adressering i Datablok

Symbolsk adressering

| Data_block_1 | | | | |
|--------------|-------------|-----------|-------------|-----------------------------|
| | Name | Data type | Start value | Comment |
| <BI> | Static | | | |
| <BI> | Hastighed_1 | Int | 500 | Korer med 500 omdrejninger |
| <BI> | Hastighed_2 | Int | 1000 | Korer med 1000 omdrejninger |
| <BI> | Max_Stroem | Real | 0.0 | Værdi for motorstrøm |
| <BI> | Motor_temp | Real | 0.0 | Motor temperatur |
| | <Add new> | | | |

I det viste eksempel er der anvendt symbolsk adressering. Softwaren holder selv styr på adressering og placering i DB.

Absolut adressering



For at se de absolutte adresser højreklikkes på DB1, vælg "Attributes" og fjern hak i "Optimized block access". Husk, når du vælger denne indstilling, så kan du kun tilgå adresserne med absolut adressering.

| Data_block_1 | | | | | |
|--------------|------------------|-----------|--------|-------------|-----------------------------|
| | Name | Data type | Offset | Start value | Comment |
| 1 | <BI> Static | | | | |
| 2 | <BI> Hastighed_1 | Int | 0.0 | 500 | Korer med 500 omdrejninger |
| 3 | <BI> Hastighed_2 | Int | 2.0 | 1000 | Korer med 1000 omdrejninger |
| 4 | <BI> Max_Stroem | Real | 4.0 | 0.0 | Værdi for motorstrøm |
| 5 | <BI> Motor_temp | Real | 8.0 | 0.0 | Motor temperatur |

Der er nu dukket en ekstra kolonne op med adresser.

Hastighed_1 hentes på DB1.DBW0

Hastighed_2 hentes på DB1.DBW2

Max_Stroem hentes på DB1.DBD4

Mator_temp hentes på DB1.DBD8

Bit-værdi adresseres DBX ??

Byte-værdi adresseres på DBB?

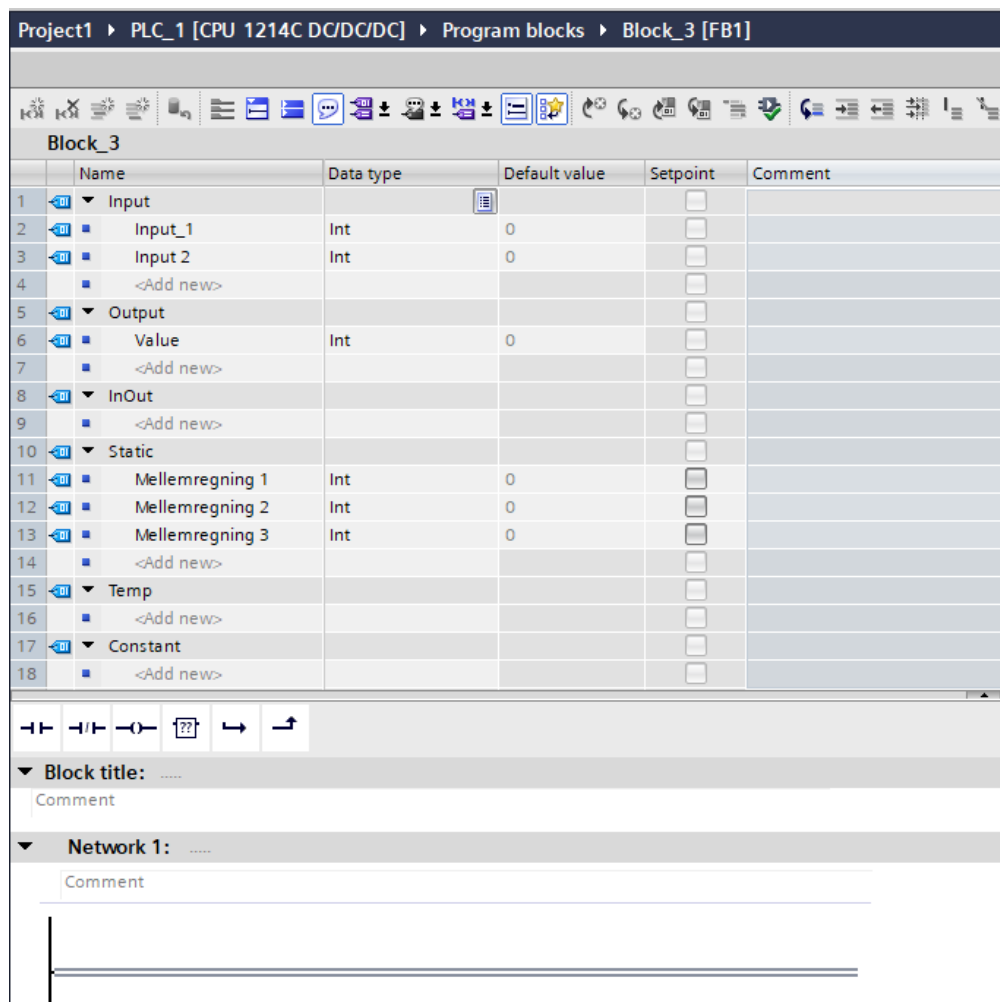
Word-værdi adresseres på DBW?

Dobbeltord adresseres på DBD?

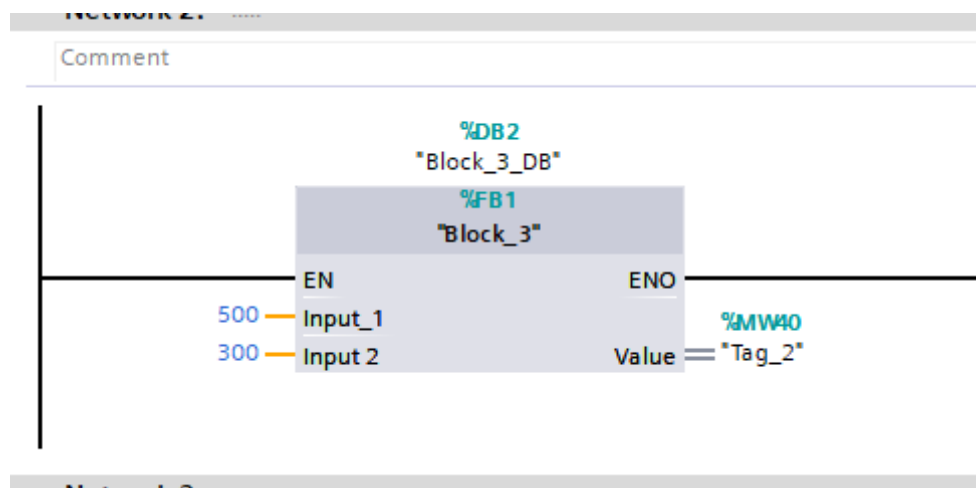
Instance datablok

De datablokke, vi har gennemgået indtil nu, er globale DB, som kan tilgås fra alle programblokke.

En Instance datablok er en DB, som er tilknyttet en FB. FB/FC-blokke bliver behandlet i et særskilt kapitel, så her gennemgås kun selve datablokken.



Billedet viser en FB1, hvor der anvendes følgende deklarationstabel – der ikke vist noget program i blokken.



Når blokken FB1 kaldes fra OB1, tilføjes en instance datablok, som i dette eksempel er valgt til DB2.

Project1 ► PLC_1 [CPU 1214C DC/DC/DC] ► Program blocks ► Block_3_DB [DB2]

Keep actual values Snapshot Copy snapshots to start values

Block_3_DB

| | Name | Data type | Start value | Retain | Accessible f... | Writa... | Visible in |
|----|-----------------|-----------|-------------|--------------------------|-------------------------------------|-------------------------------------|-------------------------------------|
| 1 | Input | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 2 | Input_1 | Int | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| 3 | Input_2 | Int | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| 4 | Output | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 5 | Value | Int | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| 6 | InOut | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 7 | Static | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 8 | Mellemregning 1 | Int | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| 9 | Mellemregning 2 | Int | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| 10 | Mellemregning 3 | Int | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |

Billedet viser instance datablokken DB2. Alle de deklarerede værdier fra FB 1, bliver oprettet som variable i datablokken.

Der er ingen forskel på opbygningen af en instance DB og en global DB.

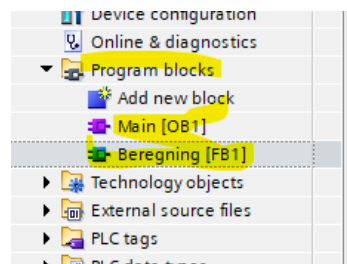
Forskellige typer af Instance DB

Der findes i en Siemens PLC tre forskellige typer af Instance datablokke:

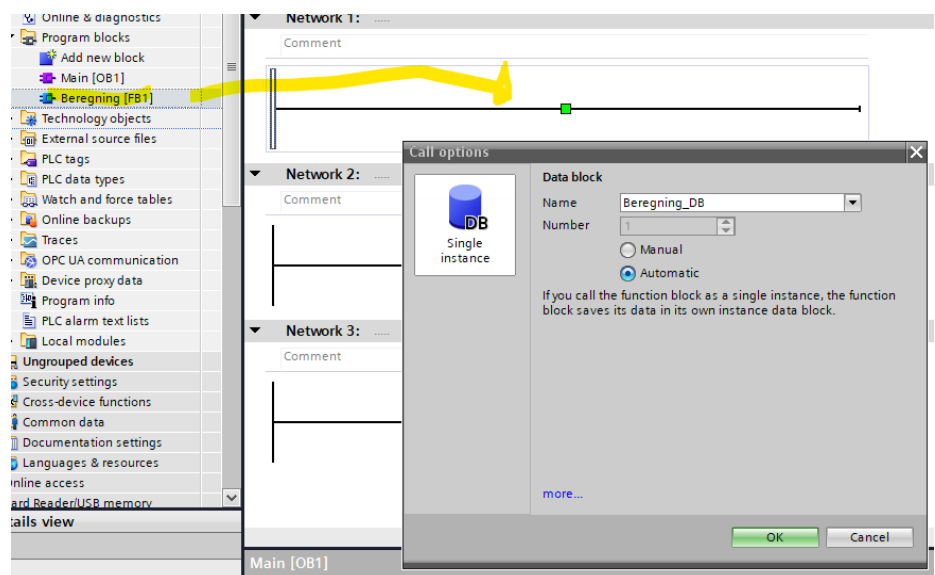
- Single Instance DB
- Parameter Instance block
- Multi instance block

I de efterfølgende afsnit gennemgås de tre typer, samt eksempler på deres anvendelse.

Single Instance DB

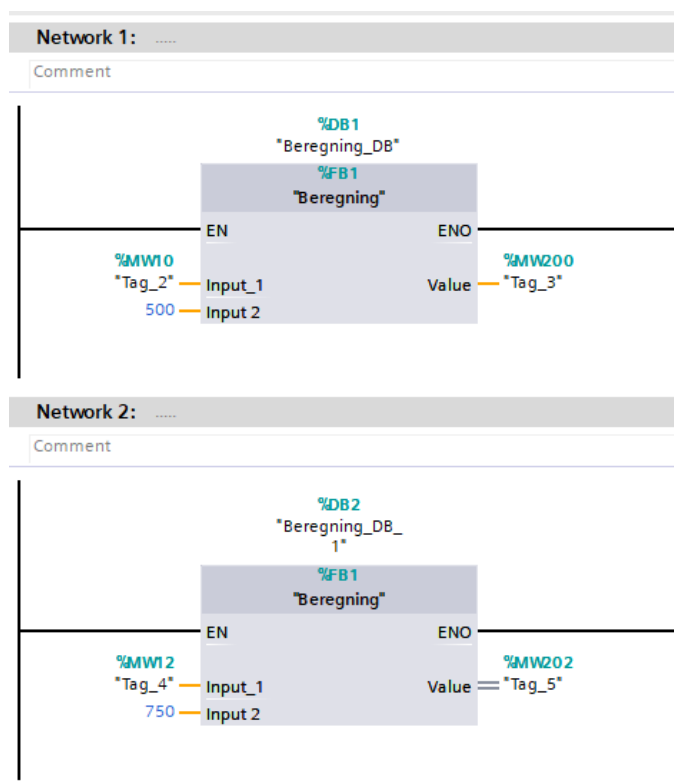


I dette eksempel har vi en OB1 og en FB1. FB indeholder nogle beregninger og skal kaldes to gange med forskellige parametre.



FB1 trækkes over i netværk 1 i OB1.

Software foreslår selv, at det skal være en "Single Instance DB". I dette eksempel vælges DB1.

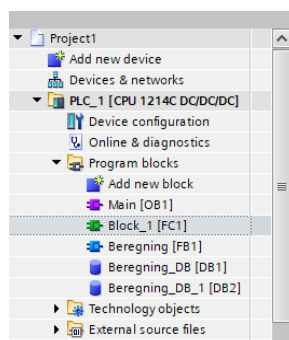


Tegningen viser, at FB1 kaldes både i netværk 1 og netværk 2 med forskellige parametre. I netværk 1 tilknyttes DB1, mens DB2 tilknyttes i netværk 2.

Hver gang man kalder en FB, skal den have sit eget DB-område, ellers fungerer den ikke korrekt.

I dette eksempel definerer vi en ny DB for hvert kald af FB, dette kaldes for "Single Instance DB".

Parameter Instance block

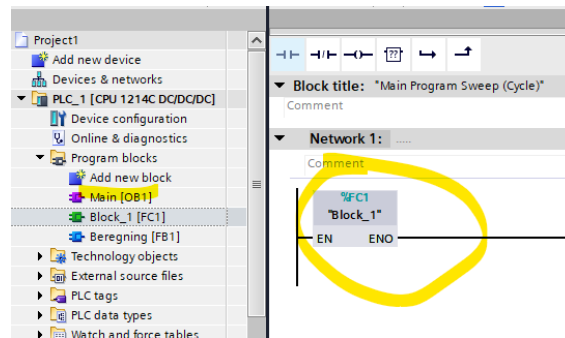


I dette eksempel har vi en OB1, der kalder en FC1, der igen kalder FB1 to gange som i foregående eksempel. Det betyder, at FB1 er indlejret i FC1.

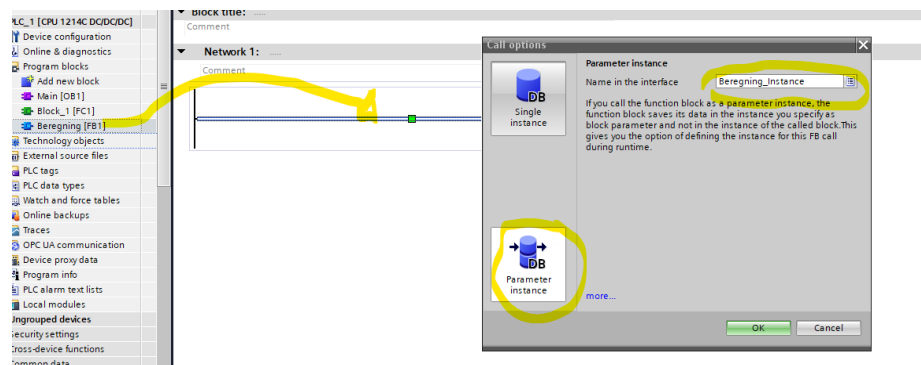
Hvis vi anvendte "Single Instance blokke" ved kald af de to FB1 i FC 1, så ville vi kun kunne anvende FC 1-kaldet den ene gang.

Derfor har vi i en Siemens PLC mulighed for at anvende "Parameter Instance blok".

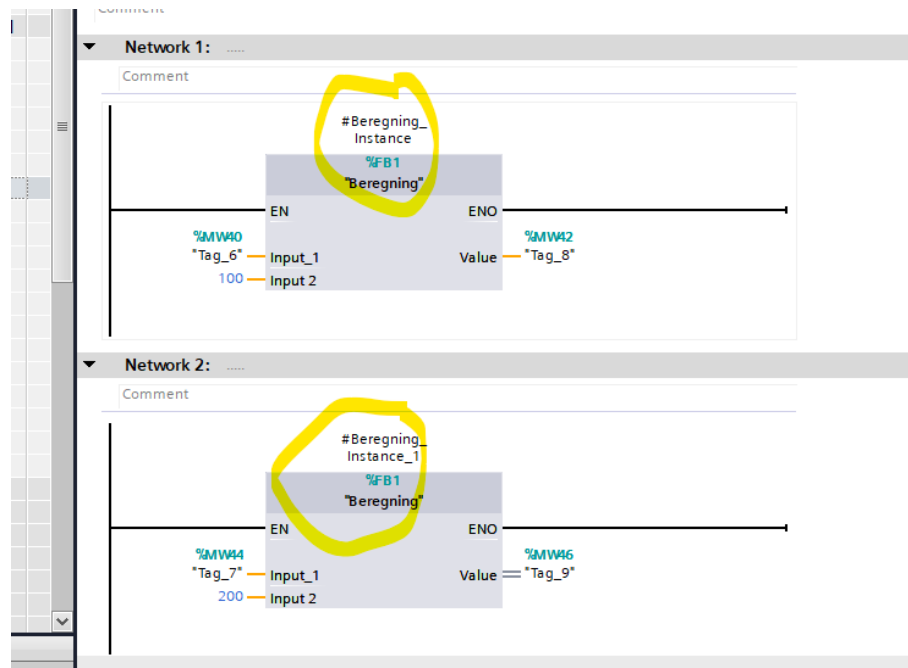
En "Parameter Instance blok" betyder egentlig, at DB bliver variabel og skal defineres ved kald af blokken.



I OB1 kaldes FC1

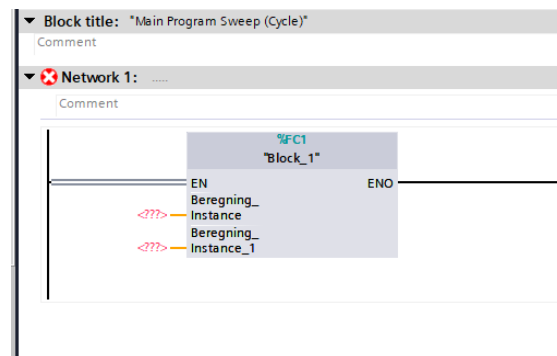


FB1 trækkes over i FC1. Der er 3 niveauer af programkald, derfor får man mulighed for at vælge "Parameter Instance Blok". Navnet på blokken følger programnavnet, men kan selvfølgelig ændres.



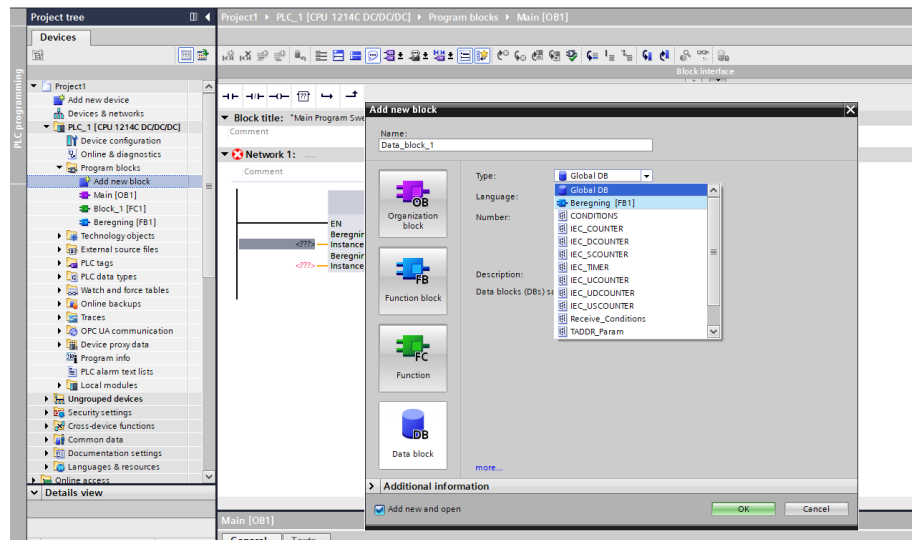
Dette er så den færdige FC1, hvor de to FB så kaldes i henholdsvis netværk 1 og 2.

Som det ses på dette skærmbillede, er de to DB nu blevet variable og skal parametres ved kald.



Når blokken kaldes fra OB1, skal de to DB defineres.

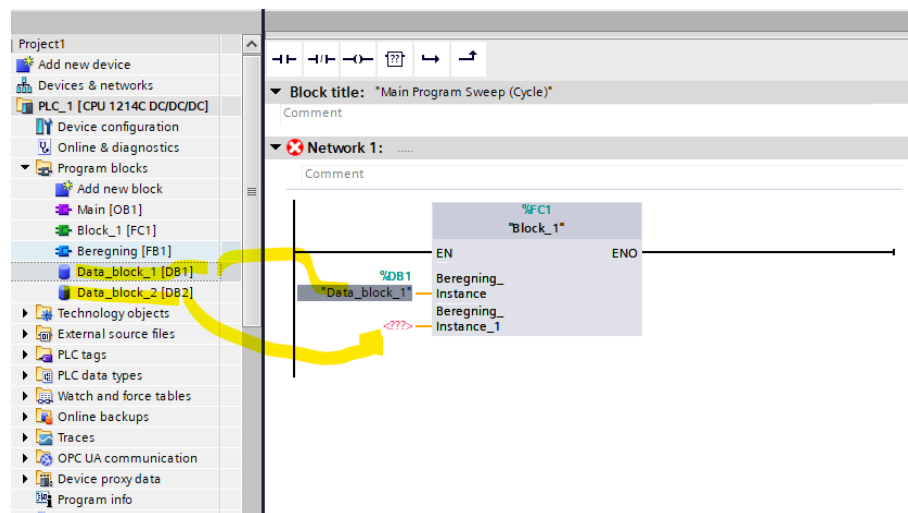
Oprettelse af DB for "Parameter Instance Blok"



Vælg "Add new block" – Vælg Data Block – under type vælges den DB, som passer til FB'en. I dette eksempel hedder den beregning (FB1).

Det er vigtigt, at man vælger den korrekte type, ellers oprettes DB med de forkerte parametre.

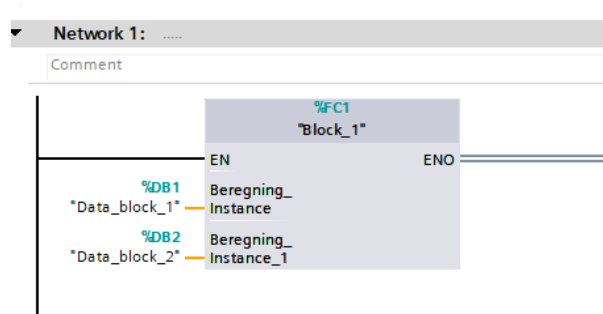
I dette eksempel har vi brug for to datablokke, derfor oprettes den to gange med hvert sit nummer. I dette eksempel er nummeret valgt til DB 1 og 2.



De to datablokke trækkes over på FC 1-blokken.

I dette eksempel er der bare navngivet med default værdier, de kan ændres efter behov.

Det færdige programkald af FC1 fra OB1 kommer så til at sådan ud:



DB1 bliver så instance datablok til første gang, FB1 kaldes i FC1, DB2 bliver instance datablok for anden gang, FB1 kaldes i FC1.

Det er vigtigt, at der vælges forskellige DB, ellers fungerer det ikke korrekt.

I dette eksempel er der kun fokuseret på variable DB'er. Ofte vil der også være behov for at in- og out-variabler bliver parametriserbare.

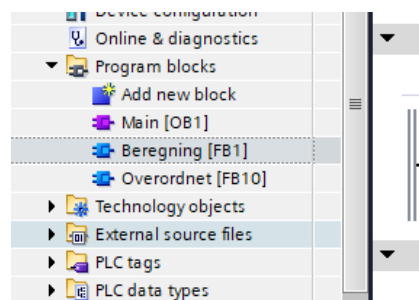
Multi instance block

Den sidste type af instance DB i en Siemens PLC er en "Multi Instance block".

Forskellen for denne og den foregående er, at den kun kan anvendes, når der defineres FB for programkald.

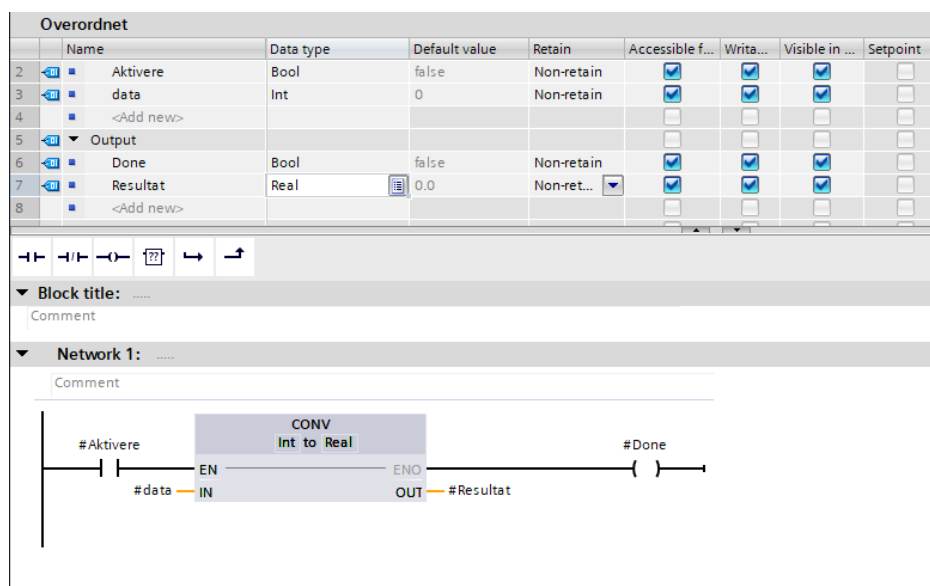
På mange måder er "Multi Instance block" nemmere at håndtere for programmøren end "Parameter Instance Blok".

Eksempel



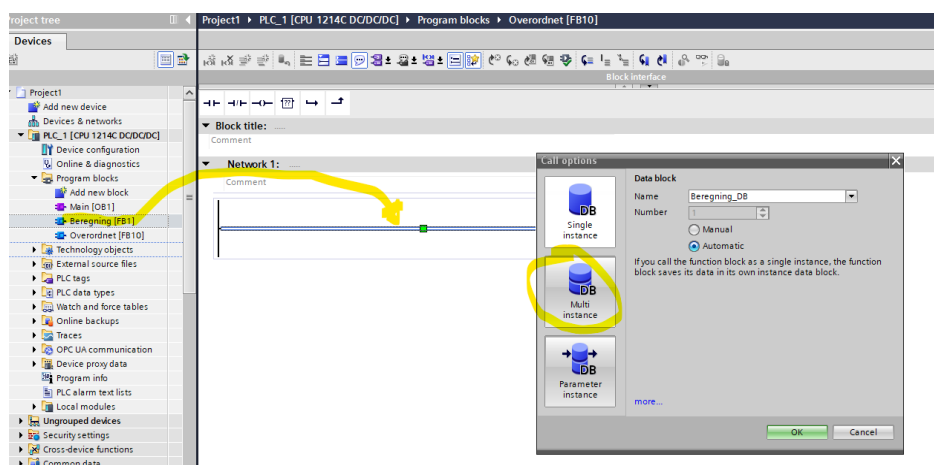
I dette eksempel har vi en OB 1, som kalder en FB10, som kalder en FB1 to gange.

FB 1

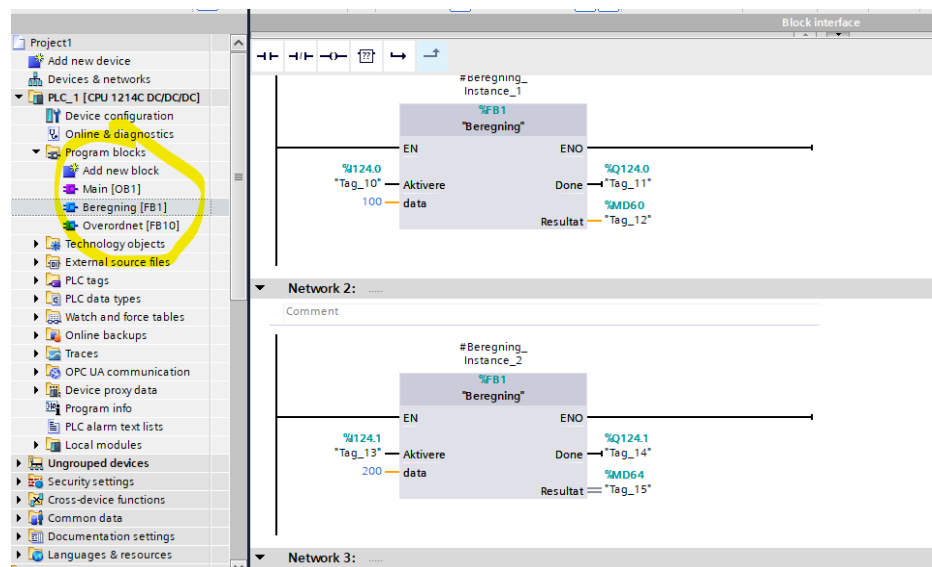


FB1 indlæser en talværdi, hvis den digitale indgang er aktiv. Dette tal konverteres til REAL og leveres på output.

Programmet i sig selv giver ikke den store mening og er kun tænkt som et simpelt eksempel.

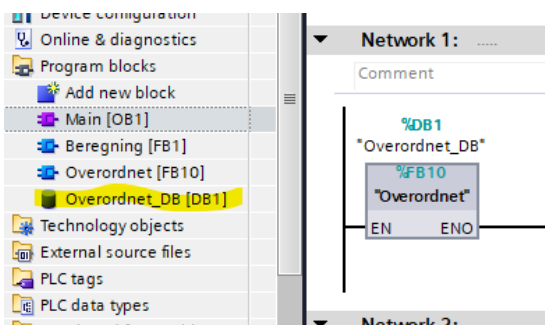


Når FB1 trækkes over i FB10, får man nu mulighed for at vælge "Multi Instance Block".



Her ses den færdige FB10-blok, hvor FB1 kaldes to gange.

Hvis man kigger i bloklisten, ses det, at der endnu ikke defineret en datablok til kaldet. Det sker først, når den overordnede blok FB10 kaldes fra OB1.



Når FB 10 kaldes op fra OB1, defineres der en datablok. I dette eksempel bliver det DB1, som er en single datablok.

Hvis man anvender "Multi Instance Block", så betyder det, at den samme DB bruges til flere forskellige kald, men dataområde i DB er unikke og tilknyttet det enkelte kald.

Denne løsning vil reducere antallet af datablokke i PLC'en, hvis man anvender mange FB-kald.

Datablok 1 (DB1)

| | Name | Data type | Start value | Retain | Accessible f... | Writa... | Visible in ... | Setpoint | Comment |
|----|----------------------|-------------|-------------|--------|-----------------|----------|----------------|----------|---------|
| 1 | Input | | | | | | | | |
| 2 | Output | | | | | | | | |
| 3 | InOut | | | | | | | | |
| 4 | Static | | | | | | | | |
| 5 | Beregning_Instance | *Beregning* | | | | | | | |
| 6 | Beregning_Instance_1 | *Beregning* | | | | | | | |
| 7 | Input | | | | | | | | |
| 8 | Aktivere | Bool | false | | | | | | |
| 9 | data | Int | 0 | | | | | | |
| 10 | Output | | | | | | | | |
| 11 | Done | Bool | false | | | | | | |
| 12 | Resultat | Real | 0.0 | | | | | | |
| 13 | InOut | | | | | | | | |
| 14 | Static | | | | | | | | |
| 15 | Beregning_Instance_2 | *Beregning* | | | | | | | |
| 16 | Input | | | | | | | | |
| 17 | Aktivere | Bool | false | | | | | | |
| 18 | data | Int | 0 | | | | | | |
| 19 | Output | | | | | | | | |
| 20 | Done | Bool | false | | | | | | |
| 21 | Resultat | Real | 0.0 | | | | | | |
| 22 | InOut | | | | | | | | |
| 23 | Static | | | | | | | | |

Hvis vi kigger ind i DB1, så kan vi se, at der er oprettet et unikt dataområde for de to kald af FB1, som sker i FB10.

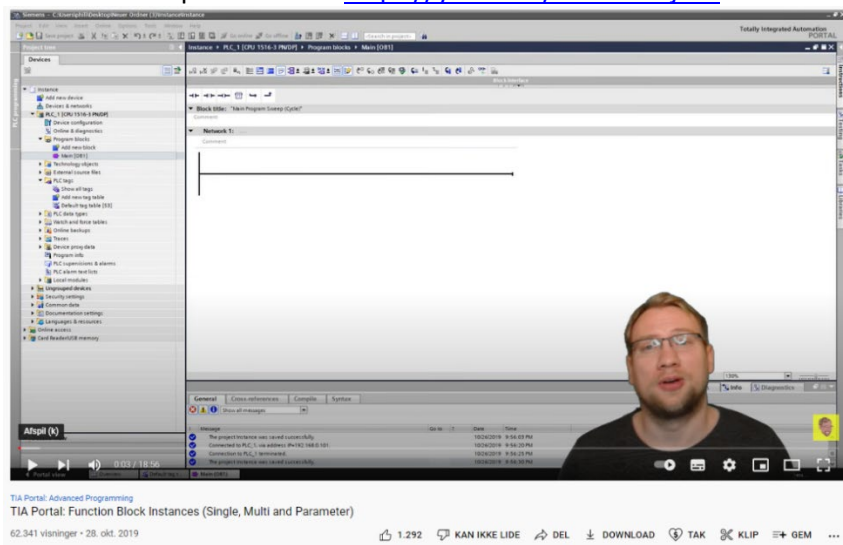
I dette eksempel er der også kun fokuseret på variable DB'er. Ofte vil der også være behov for at flere in- og out-variabler bliver parameterbare.

YouTube

Der findes en gut på YouTube, som laver mange instruktionsfilm til Siemens PLC.

Han har også lavet en omkring TIA Portal: Function Block Instances (Single, Multi and Parameter)

Du finder ham på dette link: <https://youtu.be/an3sUW6j0vY>



Siemens User Defined Type (UDT)

Generelt

I en Siemens PLC er det muligt at definere sine egne datatyper udover de standardtyper, som findes i forvejen.

De typer, man definerer selv, hedder User Defined Type (UDT), eller på dansk "brugerdefineret datatype".

En brugerdefineret datatype anvendes, hvor vi har et bestemt datasæt, som skal bruges mange gange i PLC-programmet.

De brugerdefinerede datatyper gemmes som en UDT-blok på harddisken. UDT-blokken anvendes som skabeloner, så en mængde indtastningsarbejde i datablokke undgås.

En brugerdefineret datatype indeholder elementære datatyper, komplekse datatyper og/eller andre brugerdefinerede datatyper.

Programeksempel

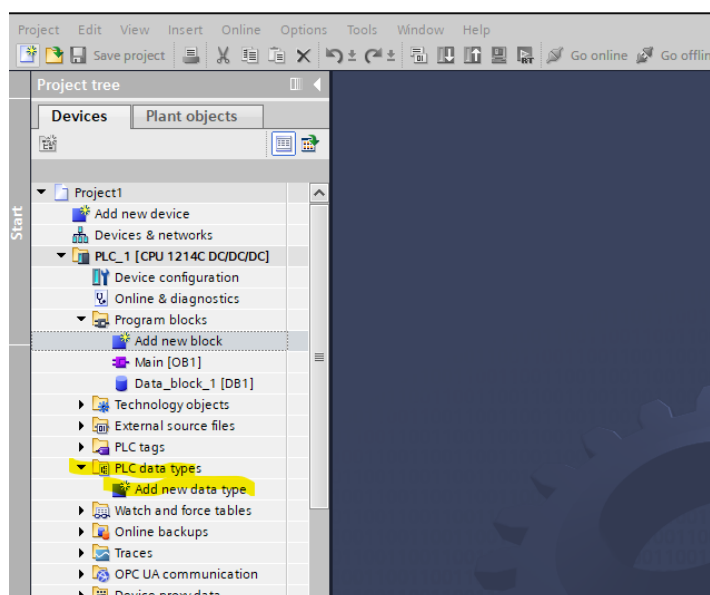
I det følgende eksempel er der brugt en UDT til at definere datastrukturen i en recept.

Recepten indeholder følgende elementer:

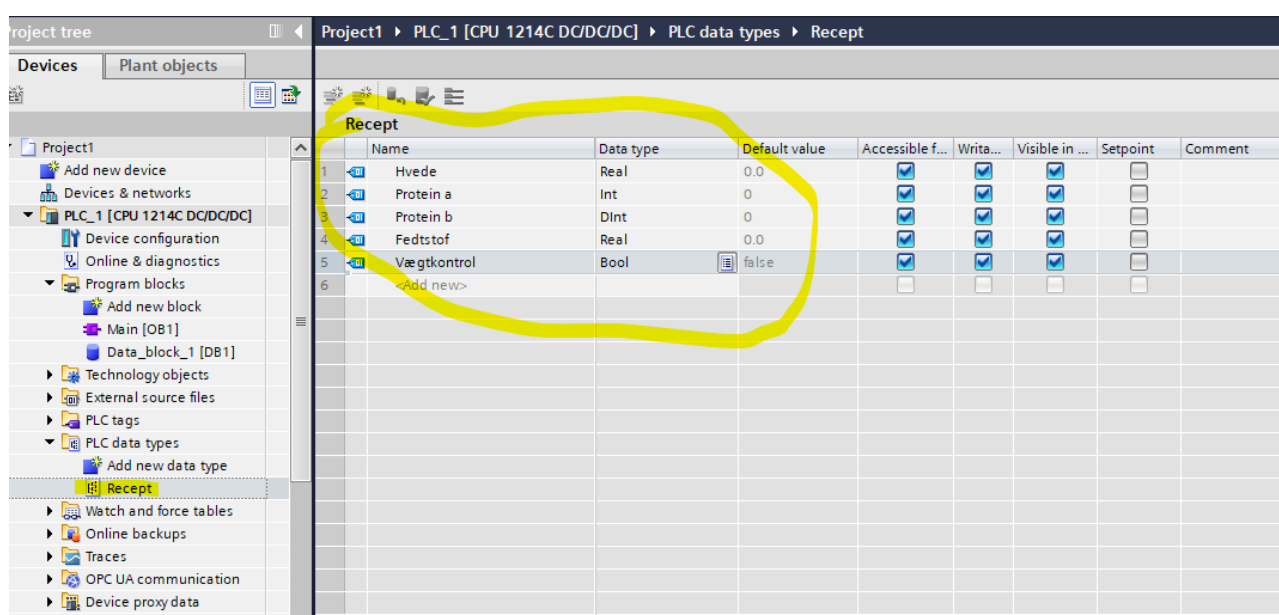
Hvede: Antal gram (Real tal)
Protein a: Antal liter (INT tal)
Protein b: Antal milliliter (DINT)
Fedtstof: Antal gram (Real tal)
Vægtkontrol: bool 1=ja 0=nej

Denne recept skal bruges til tre forskellige recepter, som benævnes: Recept 1, Recept 2 og Recept 3.

Oprette datatype



Klik på "Add new data type" i mappen "PLC data types".



De fem variabler defineres med deres respektive datatype, samtidigt er datatypen omdøbt til "Recept".

Oprette datablok med Array

I dette eksempel skulle vi bruge tre recepter, hvilket betyder, at UDT-recept skal gentages tre gange i et array.

Dette array oprettes i en almindelig global DB, som i dette eksempel er valgt til DB1.

| Data_block_1 | | | | | | | | | |
|--------------|-------------|-------------------------|-------------|--------------------------|-------------------------------------|-------------------------------------|-------------------------------------|--------------------------|---------|
| | Name | Data type | Start value | Retain | Accessible f... | Writa... | Visible in ... | Setpoint | Comment |
| 1 | ▼ Static | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | |
| 2 | ▼ Recept | Array[0..2] of "Recept" | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 3 | ▼ Recept[0] | "Recept" | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 4 | Hvede | Real | 0.0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 5 | Protein a | Int | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 6 | Protein b | DInt | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 7 | Fedtstof | Real | 0.0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 8 | Vægtkontrol | Bool | false | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 9 | ▼ Recept[1] | "Recept" | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 10 | Hvede | Real | 0.0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 11 | Protein a | Int | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 12 | Protein b | DInt | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 13 | Fedtstof | Real | 0.0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 14 | Vægtkontrol | Bool | false | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 15 | ▼ Recept[2] | "Recept" | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 16 | Hvede | Real | 0.0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 17 | Protein a | Int | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 18 | Protein b | DInt | 0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 19 | Fedtstof | Real | 0.0 | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 20 | Vægtkontrol | Bool | false | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| 21 | <Add new> | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | |

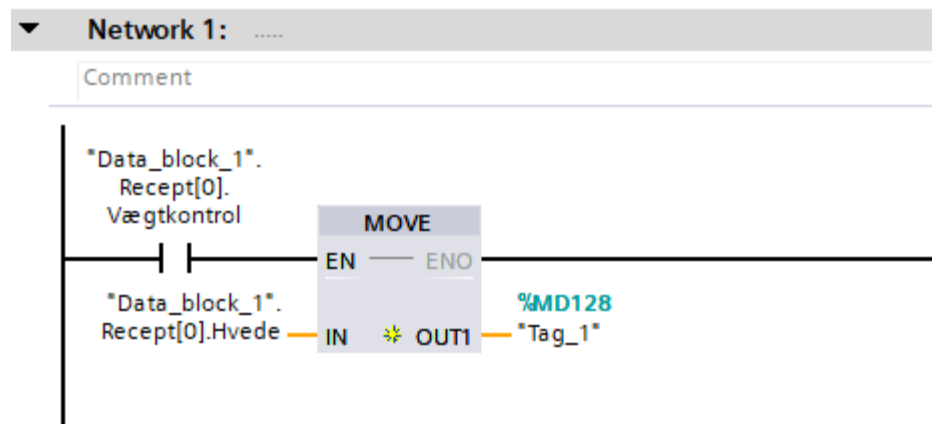
Et array kan oversættes til en tabel, hvor der i dette eksempel er tre rækker.

Hver række indeholder en UDT med 5 rækker, derfor fylder det hele 15 rækker i alt.

Der er forskel på, hvor meget hukommelsesplads den enkelte variabel fylder, fordi taltyperne er forskellige. Der er brugt taltyper, der fylder et bit, et word eller et dobbeltord.

Billedet viser DB1, efter den er pakket ud.

Programeksempel

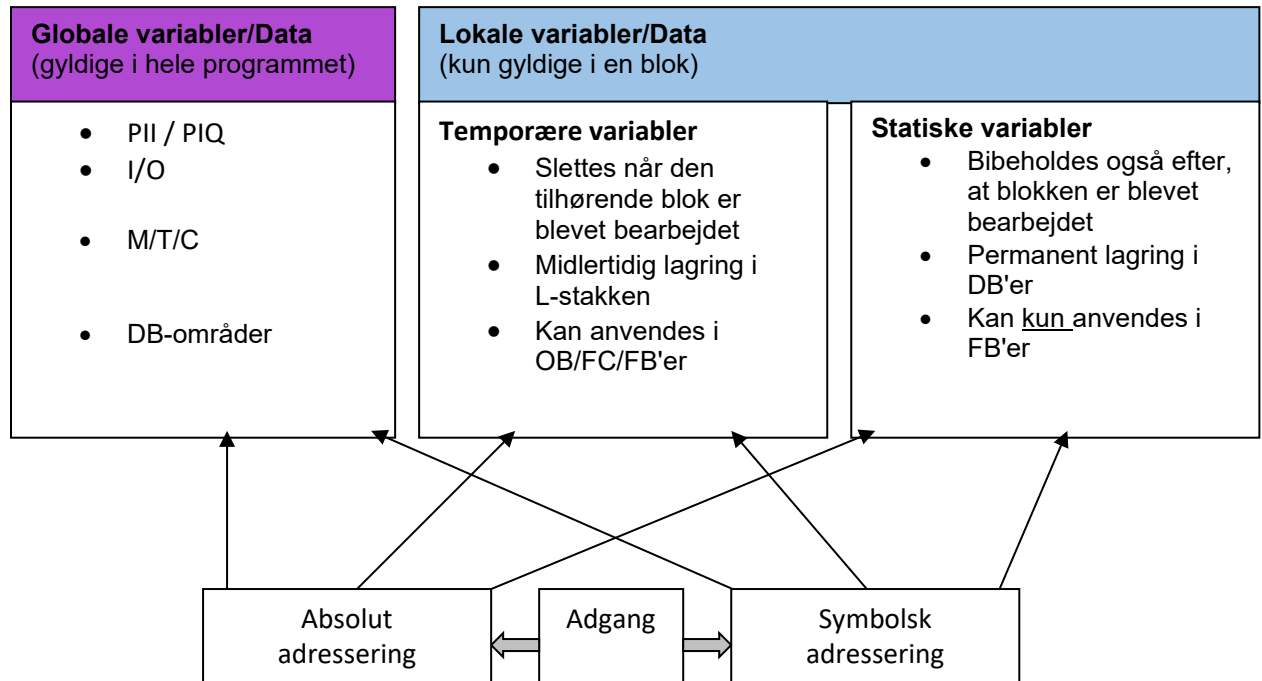


I dette programeksempel flyttes data fra DB 1 – Recept 0 – Hvede til Memory dobbeltord MD 128.

Move-funktionen udføres kun, når den boolske værdi på DB 1 – recept 0 – Vægtkontrol er aktiv.

Siemens funktioner og funktionsblokke

Indledning



Generelt

Indtil nu er alle ind- og udgange i bogen blevet adresseret med deres aktualoperander (globale navne absolut eller symbolsk), og det har ikke været muligt at lave parameteroverførsel til blokke.

Programmering med globale navne absolut eller symbolsk er det hurtigste at anvende, når man ønsker at fremstille en programdel, som kun skal anvendes en enkelt gang f.eks. en specialdel til en maskine.

Til hyppigt gentagne funktioner i et større system ønsker man at fremstille universelle, parameterbare blokke (FC'er, FB'er).

Disse blokke får defineret ind- og udgangsparametre, som får tilordnet aktualoperander (f.eks. de fysiske ind- og udgange), når blokken bliver kaldt op.

Hvilke aktualoperander (f.eks. de fysiske ind- og udgange) en parameterbar blok skal anvende, bestemmes ved aktualoperand (parameter) tildelingen, når blokken kaldes op.

Blokkens "indre" forbliver uændret.

Lokal variabel

Indtil nu er der kun anvendt globale variabler (bitmærker og datablokke) til f.eks. lagring af produktionsdata. I dette kapitel skal vi behandle datalagring i lokale variabler nærmere.

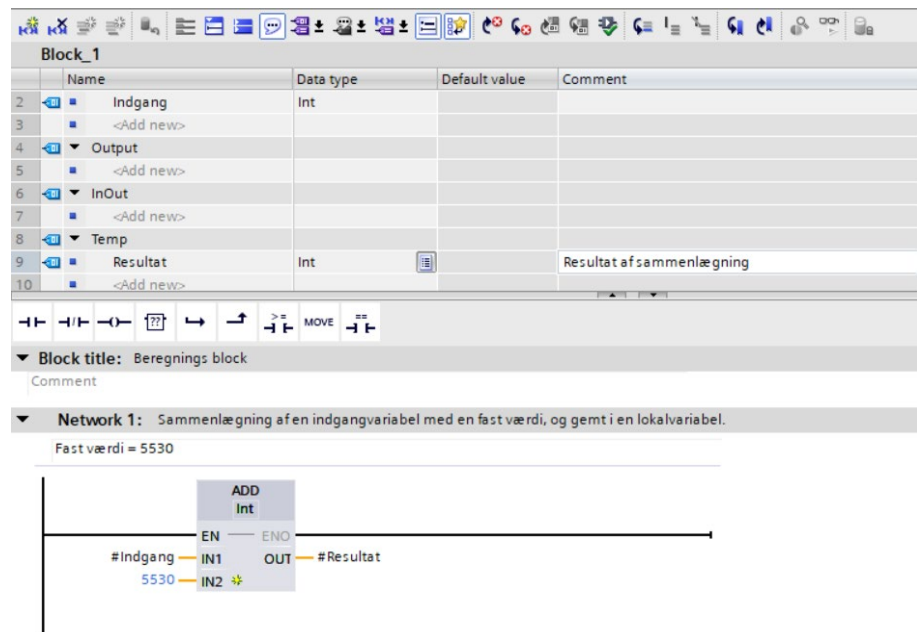
Temporære variabler

Temporære variabler er variabler, som kun har gyldighed, når en blok bliver bearbejdet. Temporære variabler kan anvendes i alle programblokke (OB, FC, FB).

Statiske variabler

Hvis data skal lagres, også efter at blokken er bearbejdet, skal dataene lagres i statiske variabler.
Statiske variabler kan kun bruges i funktionsblokke.

Temporære variabler



Generelt

Temporære variabler kan anvendes i alle programblokke (OB, FC, FB). De anvendes til at gemme informationer i, mens blokken bearbejdes (f.eks. mellemresultater i beregninger). Dataene er tabt, når blokken forlades. Temporære variabler lagres i L-stakken (den lokale datastak), som er et separat hukommelsesområde i CPU'en.

Deklaration

Temporære variabler defineres i deklarationstabellen i en blok. I linjen "temp" indtastes navnet på variabelen og den tilhørende datatype. Det er ikke muligt at definere en initialværdi her. Når blokken gemmes, bliver de absolutte adresser i L-stakkens hukommelsesområde opdateret og kan ses i spalten "Address".

Adgang

I netværk 1 ses et eksempel på en symbolsk adgang til en temporær variabel.

Resultatet af additionen gemmes i den temporære variabel **"Resultat"**.

Det er også muligt at adressere de temporære variabler absolut.

Man bør imidlertid tilstræbe at bruge symbolsk adressering, da programmet ellers bliver vanskeligt at læse.

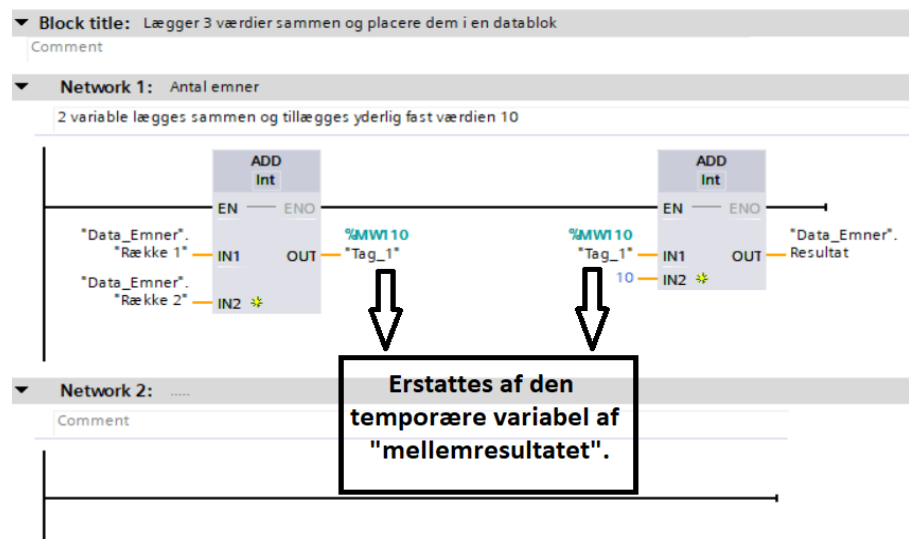
Bemærk

#

Variabelnavne, der begynder med det specielle tegn #, er lokale variabler, som kun er gyldige i den blok, hvor de er blevet deklareret i deklarationstabellen.

Programeditor indsætter automatisk specialkarakteren (#).

Eksempel: Anvendelse af temporære variabler



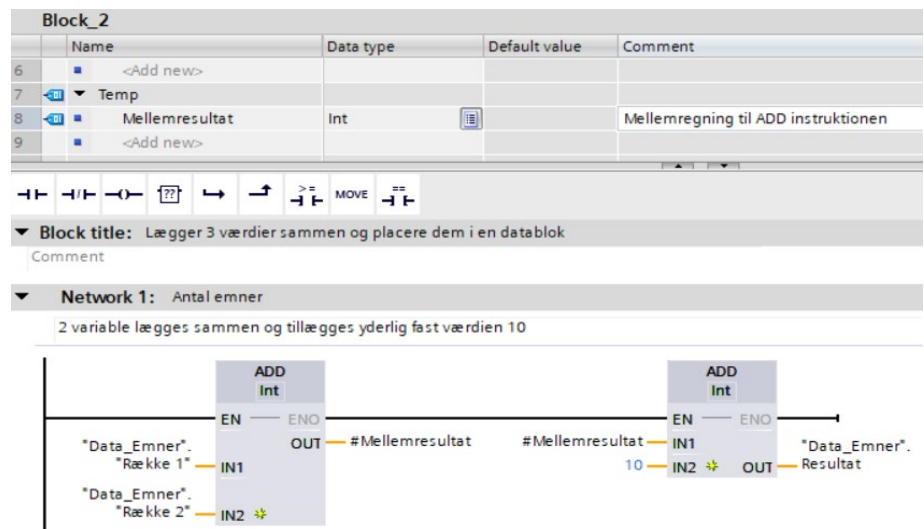
Formål

I stedet for at anvende mærkerord MW110 til lagring af et mellemresultat i FC2 i TIA-portalen, skal vi nu anvende en temporær variabel kaldet **"mellemresultat"**.

Hvis blokken programmeres i LAD eller FBD, vil det være nødvendigt med en variabel (her mærkerområdet) til at forbinde udgangen af additionsinstruktionen til indgangen af den anden additionsinstruktion.

Fremgangsmåde

- Blokken FC2 i TIA-portalen åbnes
- Der defineres en temporær variabel med navnet **"mellemresultat"** og datatypen "Integer" i deklarationstabellen



Resultat

Du kender nu til anvendelsen af temporære variabler.

Eksempel på en meddelelse, der indikerer et problem i processen

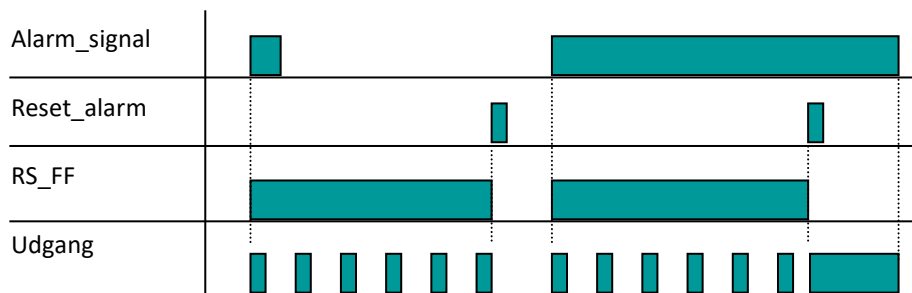
Beskrivelse

En forstyrrelse (Alarm_signal) I 4.1, som opstår fra processen, skal vises på en operatørpult.

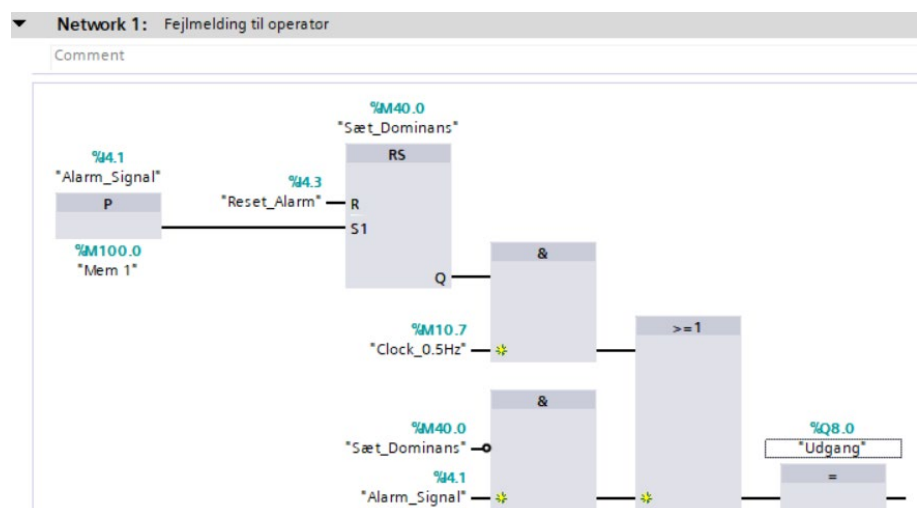
Når fejlen opstår (I 4.1) skal udgangen (Q 8.0) blinke med en frekvens på 1 Hz.

Når fejlen opdages, kvitteres der for meldingen med (Reset_alarm) (I 4.3). Hvis fejlen ikke findes længere, skal udgangen slukke; men er fejlen der stadig efter kvitteringen, skal udgangen skifte til vedvarende lys, indtil Alarm_signal forsvinder.

Funktionsdiagram



Løsningsforeslag



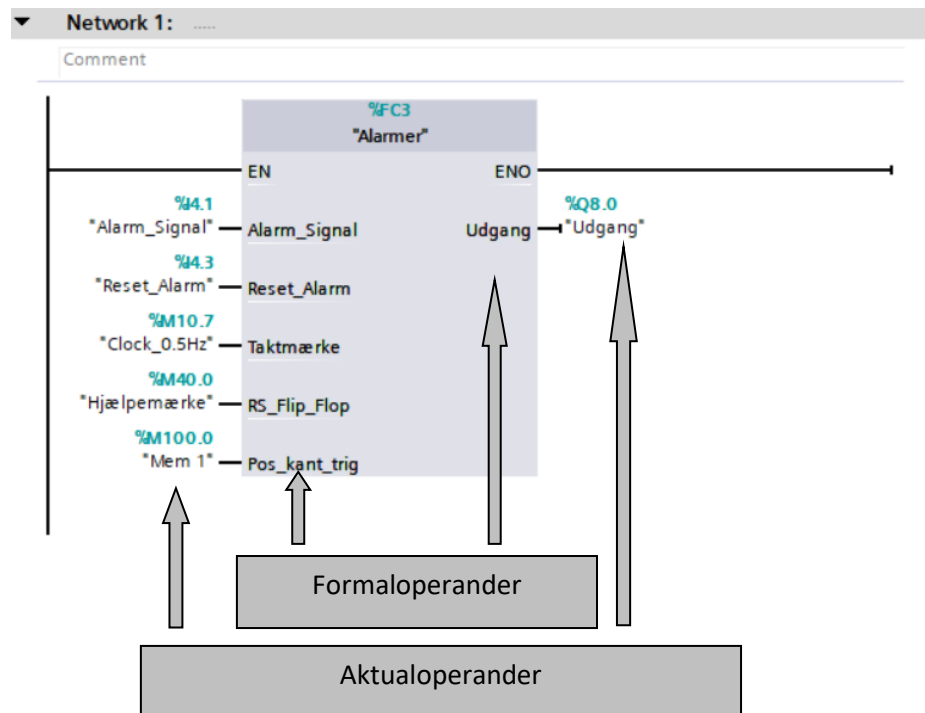
Programmet

Som sikkerhed for at problemer, der kun optræder kortvarigt, bliver konstateret, anvendes en flip-flop med sæt-dominans (M40.0). Forstyrrelsen bliver konstateret på en positiv flanke, da det ellers vil være umuligt at kvittere for en fejl, som fortsat er aktiv.

Hvis hukommelsesmærket (M40.0) er sat, og der endnu ikke er kvitteret for meddelelsen, vil det øverste OG-led bevirke, at LED-dioden blinker. Blinkhastigheden afhænger af blinktaktmærket (M10.7), der defineres i hardwareopsætningen.

Det nederste OG-led sætter LED-dioden på vedvarende lys, hvis der kvitteres for en fejl, som fortsat findes.

Parametrerbare blokke



Introduktion

Parametrerbare blokke anvendes ved hyppigt gentagne programsektioner. De parametrerbare blokke har følgende fordele:

- Programafsnittet (blokken) skal kun fremstilles én gang
- Blokken bliver kun lagret én gang, og den kan kaldes op, så tit det ønskes
- Blokken programmeres med en formaloperand (formaloperander) som indgangs-, udgangs- og ind/ud-parametre, og kun når blokken kaldes op, bliver de "virkelige" adresser (aktualoperander) tildelt.

Eksempel

Når blokken bearbejdes, bliver instruktionen "Alarm_signal" bearbejdet med henblik på at se, hvilken aktualoperand, der er tilknyttet til formalparameteren "Alarm_signal".

FC/ FB

Parametrerbare blokke kan enten være FC'er eller FB'er.

Parametrerbar FC3

Eksemplet øverst på siden skal anvendes 2 gange i et system. Derfor fremstilles FC3 som en parametrerbar blok og bliver herefter kaldt op 2 gange, hvor hvert opkald er med forskellige aktualoperander.

Deklarationsdelen af en blok

Formal-parameter

| Parameter type | Deklaration | Bruges til | Grafisk visning |
|----------------|-------------|----------------|----------------------|
| Indgang | In | Kun læse | Venstre side af blok |
| Udgang | Out | Kun skrive til | Højre side af blok |
| Ind/udgang | In_out | Læse/skrive | Venstre side af blok |

Deklarationstabel for FC3

| | Name | Data type | Default value | Comment |
|---|---------------|-----------|---------------|---------------------------|
| 1 | Input | | | |
| 2 | Alarm_Signal | Bool | | Alarm fra processen |
| 3 | Reset_Alarm | Bool | | Reset af alarmsignal |
| 4 | Taktmærke | Bool | | Taktmærke til blink |
| 5 | Output | | | |
| 6 | Udgang | Bool | | Alarmudgang til processen |
| 7 | InOut | | | |
| 8 | RS_Flip_Flop | Bool | | Reset_Set flip-flop |
| 9 | Pos_kant_trig | Bool | | Positiv kanttrigning |



Formaladresse

Før fremstillingen af en parametriserbar blok kan begynde, skal formaloperanderne defineres i deklarationstabellen.

Parametertyper

I tabellen øverst på siden ses de forskellige parametertyper og deres anvendelse.

Vær opmærksom på, at når der både skal læses og skrives til en formaloperand, skal den defineres som en ind-/ud-parameter.

Eksempel med FC3

I tabellen næstøverst på siden ses deklarationstabellen for meddelelsesvisningen fra de foregående sider. Da hukommelsesparametren (RS_Flip_Flop) både læses (&-leddet), og der skrives til den (sæt/resæt), skal mærket defineres som en ind-/ud-parameter.

Bemærk

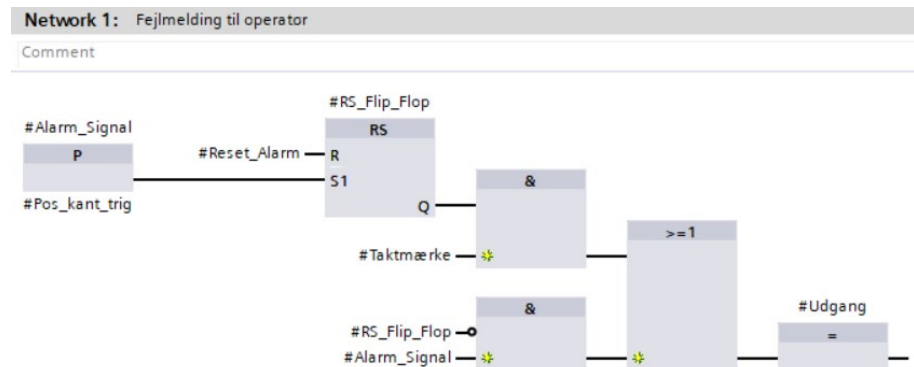
Der er kun én række for hver parametertype i deklarationstabellen. Hvis man skal bruge flere parametre af samme type f.eks. indgangs-parametre, skal man blot trykke "Return" efter den første indgangs-parameterdefinition, og der vil dukke en ny indgangsparameterlinje frem. En anden metode til at få en ekstra deklarationslinje indsat på er at markere en deklarationslinje og vælge menuen:

Insert -> Declaration Row -> Before Selection/After Selection.

Vær opmærksom på

Hvis man indsætter eller sletter en eller flere rækker i deklarations-tabellen, efter at blokopkaldet er programmeret, skal man også opdatere blokopkaldet!

Editering af en parametrerbar blok



Bemærk

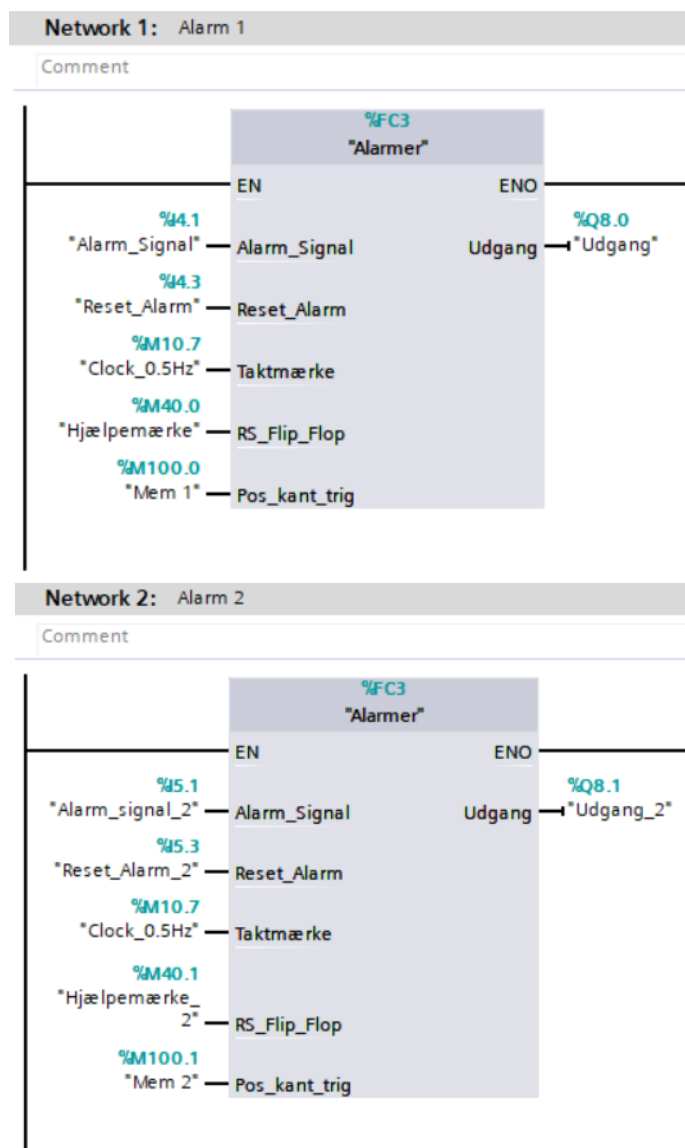
Det gør ingen forskel, om formaloperanden er skrevet med store eller små bogstaver. Tegnet "#" foran formaloperanden indsættes automatisk. Tegnet "#" angiver, at det er en lokal variabel, som er blevet defineret i deklARATIONSTABELLEN i den pågældende blok.

Det er muligt, hvis programmet skrives i LAD eller FBD, at et parameters navn ikke kan vises i en linje, men deles op i to eller flere linjer. Opsætningen af programeditor bestemmer antallet af tegn i en linje. Opsætningen af programeditor sker via menuen.

Symboler

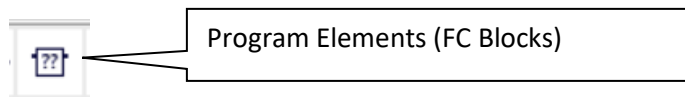
1. Hvis det symbolske navn bruges, når en blok editeres, søger editoren efter det indtastede navn i deklARATIONSTABELLEN. Hvis det symbolske navn findes, accepteres det indtastede navn som en lokal variabel, og systemet indsætter karakteren # foran det symbolske navn.
2. Hvis det symbolske navn ikke kan genkendes som en lokal variabel, søger editor i symboltabellen efter et globalt symbol, der matcher indtastningen. Hvis det indtastede navn findes i symboltabellen, indsætter systemet det symbolske navn i anførselstegn.
3. Hvis det samme symbolske navn går igen i både den globale symboltabel og i deklARATIONSTABELLEN, vil editor altid indsætte den lokale variabel, hvis det pågældende symbolske navn indtastes i programblokken.
Hvis det globale symbolske navn ønskes indsat, skal man manuelt indsætte anførselstegnene, når det symbolske navn indtastes.

Opkald af en parameterbar blok



Opkald

I LAD/FBD kan et opkald vælges fra kataloget "Program Elements"(Empty Box).



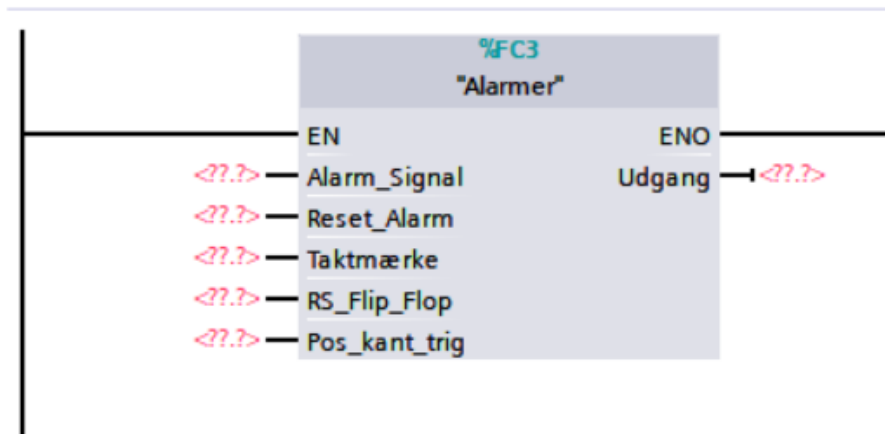
Når den parameterbare blok er indsat, vises spørgsmålstegn (???) ud for blokkens indgang, udgang og ind-/ud-parametre. Spørgsmålstegnene erstattes af de ønskede aktualparametre.

Bemærk

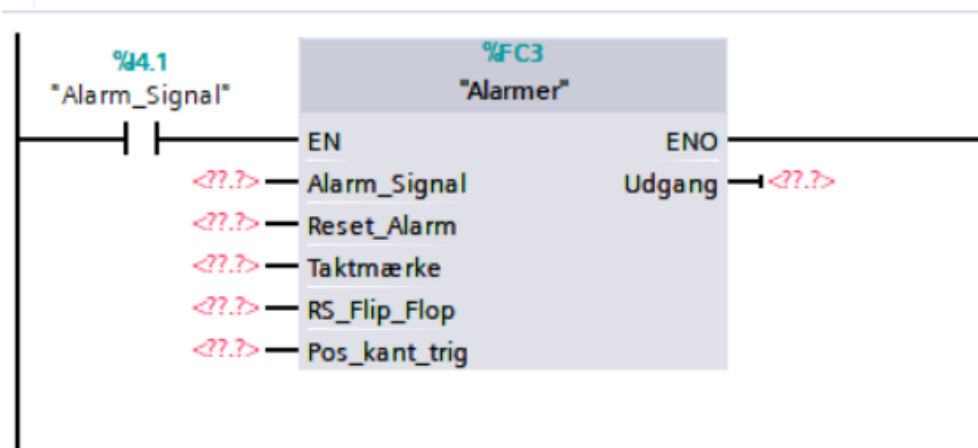
Når en parameterbar FC-blok kaldes op, skal alle blokparametre have tildelt aktualparametre (med undtagelse af EN og ENO).

Anvendelse af EN/ENO-parametrene ved opkald af blokke

Ubetinget opkald



Betinget opkald



Standard – FC'er

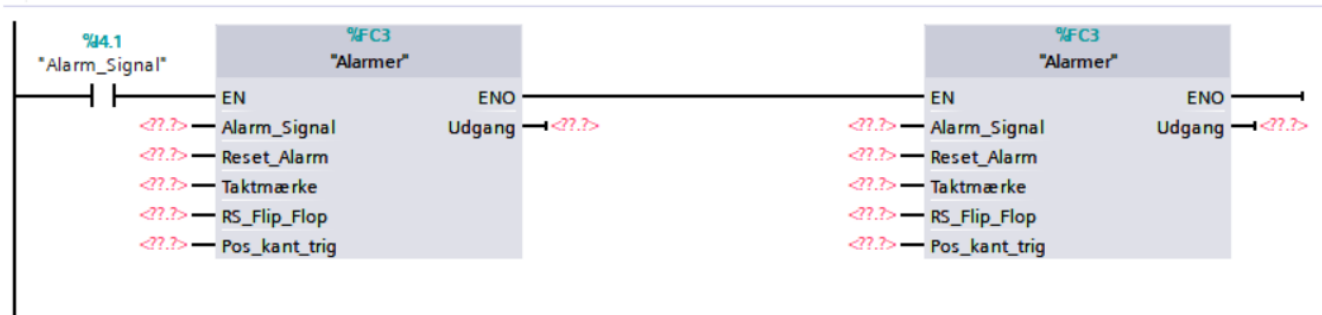
Følgende regel gælder for bearbejdningen af en standard - FC:

1. Hvis $EN = 0$, blokken bearbejdes ikke, og ENO er også $= 0$.
2. Hvis $EN = 1$, blokken bearbejdes, og hvis den blev bearbejdet uden fejl, er ENO også $= 1$.

Hvis der opstår en fejl under bearbejdningen af blokken, vil ENO blive $= 0$.

Bruger – FC'er

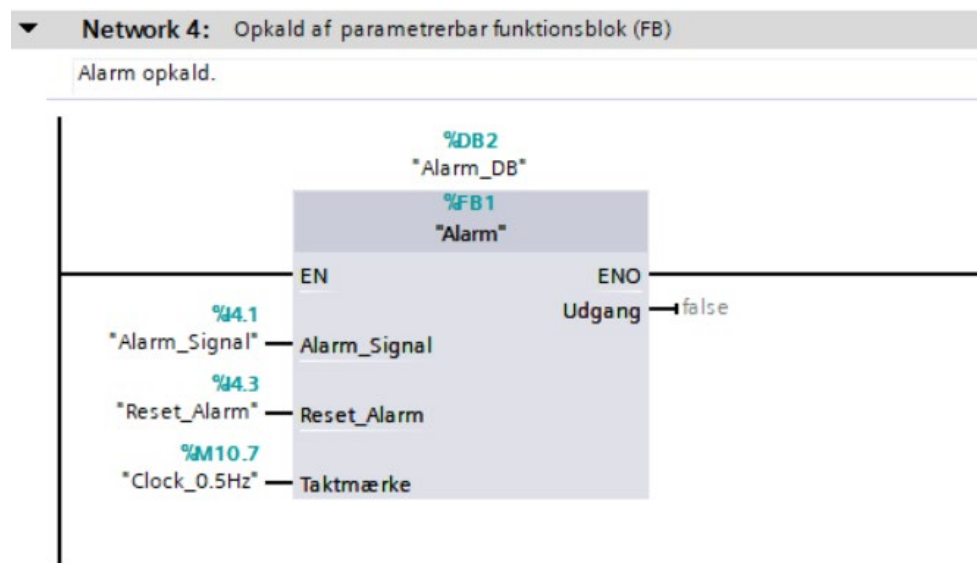
Det har ingen betydning, om en blok er programmeret i LAD, FBD eller STL. Når blokken kaldes op i LAD/FBD, tilføjes parametrene EN og ENO. Igennem EN og ENO er det muligt at videregive RLO (resultatet af den logiske operation).



Sammenkædning

I LAD/FBD kan flere FC'er grupperes sammen efter hinanden i en logisk kæde bundet sammen via EN/ENO.

Funktionsblokke (FB'er)



Deklarationstabel for FB1

| Alarm | | | | | | | | | |
|-------|---------------|-----------|---------------|------------|-------------------------------------|-------------------------------------|-------------------------------------|--------------------------|---------------------------|
| | Name | Data type | Default value | Retain | Accessible f... | Writa... | Visible in ... | Setpoint | Comment |
| 1 | ▼ Input | | | | | | | | |
| 2 | Alarm_Signal | Bool | false | Non-retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Alarm fra processen |
| 3 | Reset_Alarm | Bool | false | Non-retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Reset af alarmsignal |
| 4 | Taktmærke | Bool | false | Non-retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Taktmærke til blink |
| 5 | <Add new> | | | | | | | | |
| 6 | ▼ Output | | | | | | | | |
| 7 | Udgang | Bool | false | Non-retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Alarmudgang til processen |
| 8 | <Add new> | | | | | | | | |
| 9 | ▼ InOut | | | | | | | | |
| 10 | <Add new> | | | | | | | | |
| 11 | ▼ Static | | | | | | | | |
| 12 | RS_Flip_Flop | Bool | false | Retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Reset_Set flip-flop |
| 13 | Pos_kant_trig | Bool | false | Retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Positiv kanttrigning |

Særlige egenskaber ved FB'er

Til forskel fra funktioner (FC'er) har funktionsblokke (FB'er) en remanent hukommelse, da en lokal datablok tilknyttes til funktionsblokken - den såkaldte **Instansdatablok**. Når opkaldet til en FB fremstilles, specificeres nummeret på den tilknyttede instansdatablok, instansdatablokken bliver under programafviklingen åbnet, samtidig med at opkaldet udføres.

En instans-DB kan gemme statiske variabler. De statiske variabler bør kun anvendes i den FB, i hvis deklarationstabel de er deklareret. **Når FB'en forlades, bliver de statiske data bibeholdt.**

Parametre

Når en funktionsblok kaldes op, bliver værdierne af aktualparametrene lagret i instansdatablokken.

Hvis en formalparameter ikke får nogen aktualparametertildeling, vil den sidst gemte værdi i instansdatablokken være gældende for programafviklingen af funktionsblokken.

Der kan specificeres forskellige aktualparametre ved hvert FB-opkald. Når funktionsblokken forlades, gemmes dataene i instansdatablokken.

FB-fordele

Når programmet realiseres i en FC, skal man lede efter frie mærkeadresser eller selv oprette globale datablokadresser. De statiske variabler i en FB derimod bliver oprettet med TIA-portalens software.

Når de statiske variabler anvendes, undgås risikoen for dobbeltbelægning af mærkeområdet.

I stedet for f.eks. formaloperanden hukommelsesmærket "RS_Flip_flop" og POS_KANT_TRIG i FC3 kan de statiske variabler anvendes. Det vil sige, at "RS_Flip_flop" og "POS_KANT_TRIG" defineres som **statiske variabler** i FB'en, hvilket gør opkaldet enklere, da der er to formaloperander mindre.

Alle formalparametre behøver ikke at få tildelt aktualparametre til forskel for FC.

Funktionsblok til en signalvisning

Deklarationstabel for FB1

| Alarm | | | | | | | | | |
|-------|---------------|-----------|---------------|------------|-------------------------------------|-------------------------------------|-------------------------------------|--------------------------|---------------------------|
| | Name | Data type | Default value | Retain | Accessible f... | Writa... | Visible in ... | Setpoint | Comment |
| 1 | ▼ Input | | | | | | | | |
| 2 | Alarm_Signal | Bool | false | Non-retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Alarm fra processen |
| 3 | Reset_Alarm | Bool | false | Non-retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Reset af alarmsignal |
| 4 | Taktmærke | Bool | false | Non-retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Taktmærke til blink |
| 5 | <Add new> | | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | |
| 6 | ▼ Output | | | | | | | | |
| 7 | Udgang | Bool | false | Non-retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Alarmudgang til processen |
| 8 | <Add new> | | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | |
| 9 | ▼ InOut | | | | | | | | |
| 10 | <Add new> | | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | |
| 11 | ▼ Static | | | | | | | | |
| 12 | RS_Flip_Flop | Bool | false | Retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Reset_Set flip-flop |
| 13 | Pos_kant_trig | Bool | false | Retain | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Positiv kantrigning |

Instansdatablok DB2

| Alarm_DB | | | | | | | | | |
|----------|---------------|-----------|-------------|-------------------------------------|-------------------------------------|-------------------------------------|-------------------------------------|--------------------------|---------------------------|
| | Name | Data type | Start value | Retain | Accessible f... | Writa... | Visible in ... | Setpoint | Comment |
| 1 | ▼ Input | | | | | | | | |
| 2 | Alarm_Signal | Bool | false | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Alarm fra processen |
| 3 | Reset_Alarm | Bool | false | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Reset af alarmsignal |
| 4 | Taktmærke | Bool | false | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Taktmærke til blink |
| 5 | ▼ Output | | | | | | | | |
| 6 | Udgang | Bool | false | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Alarmudgang til processen |
| 7 | InOut | | | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | |
| 8 | ▼ Static | | | | | | | | |
| 9 | RS_Flip_Flop | Bool | false | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Reset_Set flip-flop |
| 10 | Pos_kant_trig | Bool | false | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Positiv kantrigning |

Signalvisning

I tidligere sider blev der fremstillet en parametrerbar FC3, der viste en signaltilstand i processen (indikering af et problem).

I stedet for de mærker, som blev brugt i FC3 til at lagre signalet og flanke analysen med, kan de statiske variabler i en FB anvendes. De lagres i instansdatablokken, der refererer til den pågældende FB.

Instans-DB'ens struktur

Når en DB genereres med reference til en FB, fremstiller TIA en datastruktur i en instansdatablok ved at kopiere strukturen fra funktionsblokkens deklarationstabel.

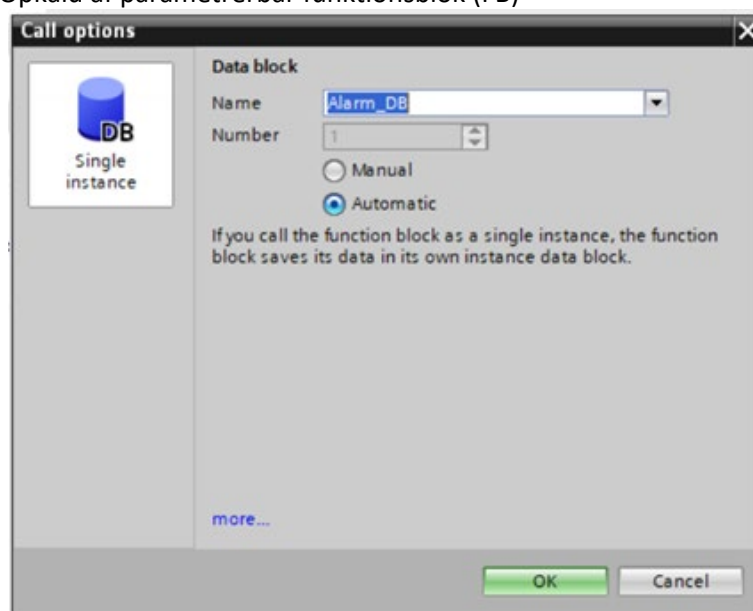
Fremstilling af instansdatablokke

Generering af en instans-DB

Der er to måder, hvorpå en ny instans-DB kan genereres:

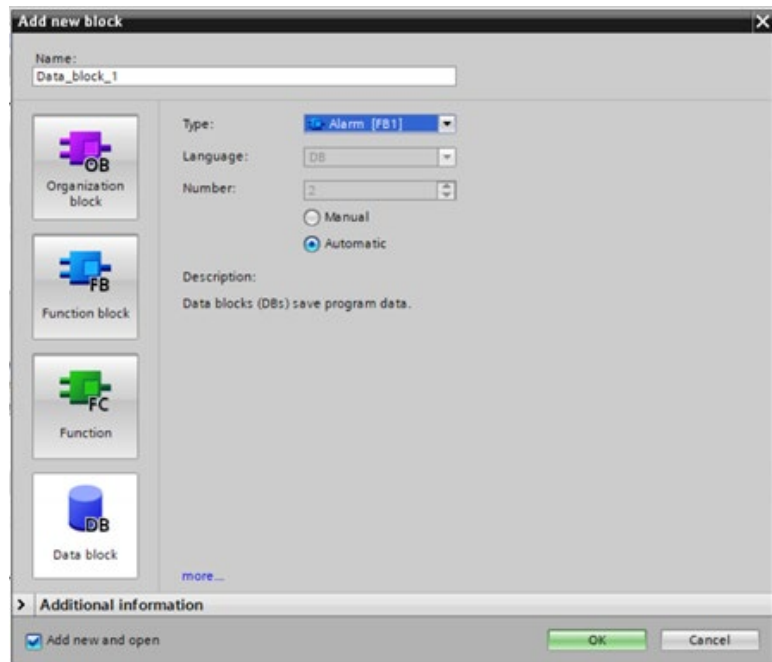
1. I forbindelse med opkaldet af en FB specificeres navnet/nummeret på instans-DB'en. Følgende meddelelse kommer frem: ***"If you call the function block as a single instance, the function block saves its data in its own instance data block."***

Opkald af parameterbar funktionsblok (FB)



2. Ved fremstillingen af en ny DB vælges i dialogboksen muligheden: "Data block referencing a function block" (datablok med reference til en FB).

Fremstilling af ny datablok (DB)



Bemærk

En instans-DB kan kun referere til en FB. Derimod kan en FB godt kan kalde flere forskellige instans-DB'er, én for hvert gang FB bruges.

Hvis en FB ændres ved at tilføje eller slette parametre, skal instans-DB'en genereres igen. Det gøres ved at vælge "Udate block call"

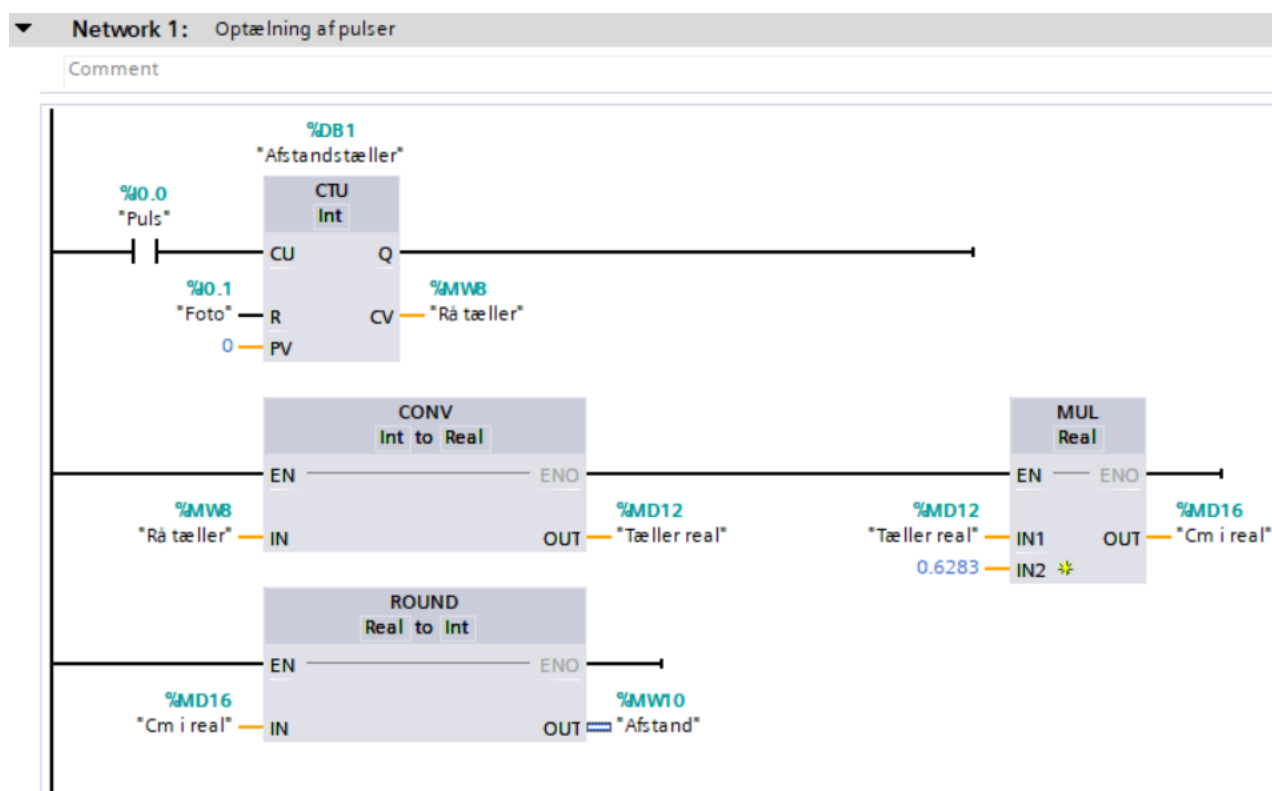
Siemens PLC-simulator

Indledning

PLC-simulatoren S7-PLCSIM bruges til at afprøve programmer, inden de uploades til en rigtig PLC, samt simulering af en PLC i undervisningen. Start med at få installeret SIEMENS S7-PLCSIM-programmet på din PC.

PLC-simulatoren

Inden simulatoren kan bruges, skal der programmeres et PLC-program i TIA-portalen, hvor der er valgt PLC-type, og der bruges ind- og udgange, der passer til denne PLC's konfiguration. Her er FC1 i programmet "Båndomkreds" åbnet.

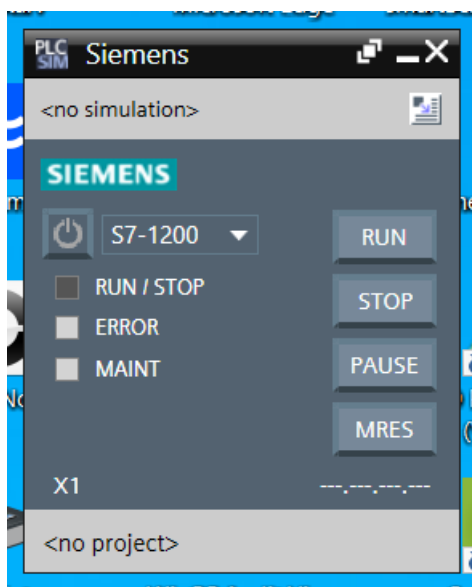


Programeksemplet her illustrerer en tæller på et bånd, der tæller ved hjælp af en pulsgiver på båndrotationen, hvor langt emnet har bevæget sig frem på båndet. Dette eksempel vil være grundlaget for den efterfølgende instruktion i simulatorens brug.

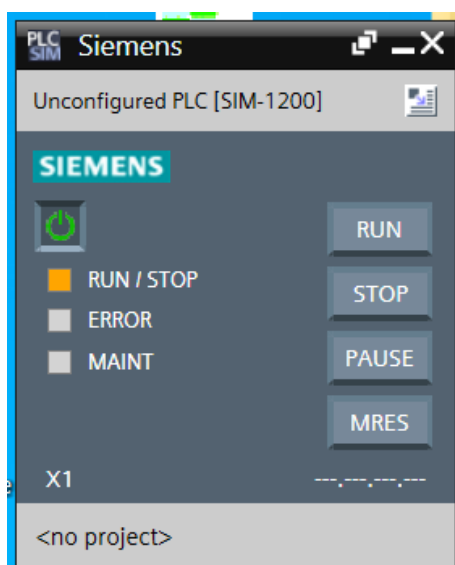
TIA-portalen skal være åben med det færdige program. For at starte simulatoren op skal der trykkes på S7-PLCSIM-ikonet.



Så starter S7-PLCSIM-simulator programmet op, og følgende billede dukker op på skærmen.

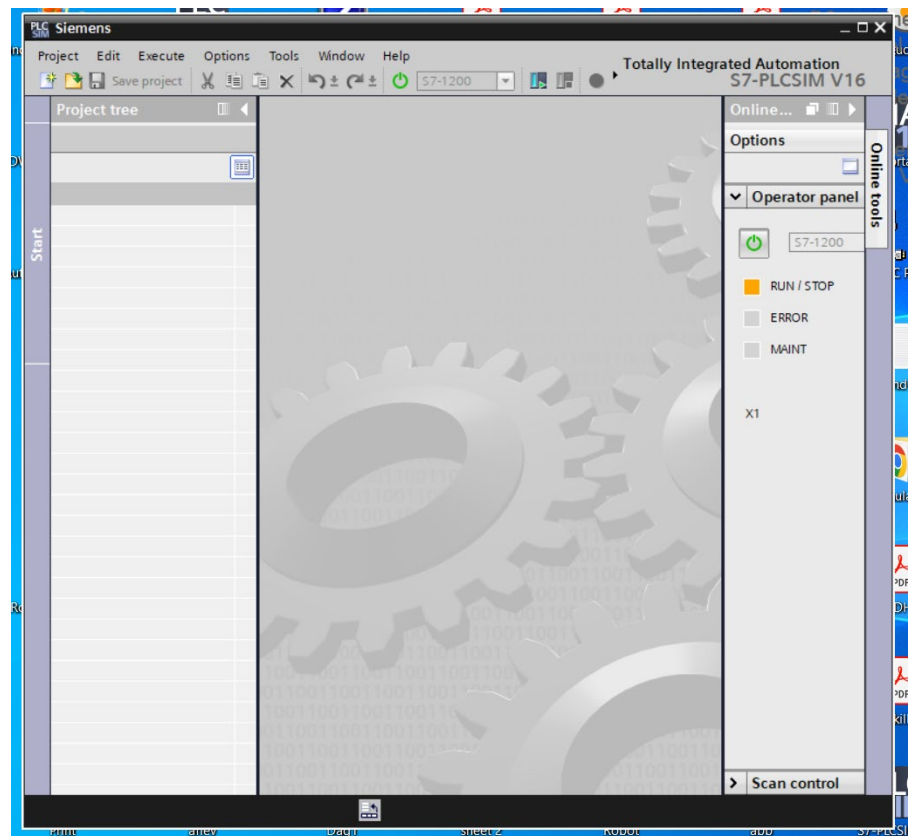


For at aktivere selve simulatoren skal denne tændes på power-knappen ***"Power the PLC on or off"***.

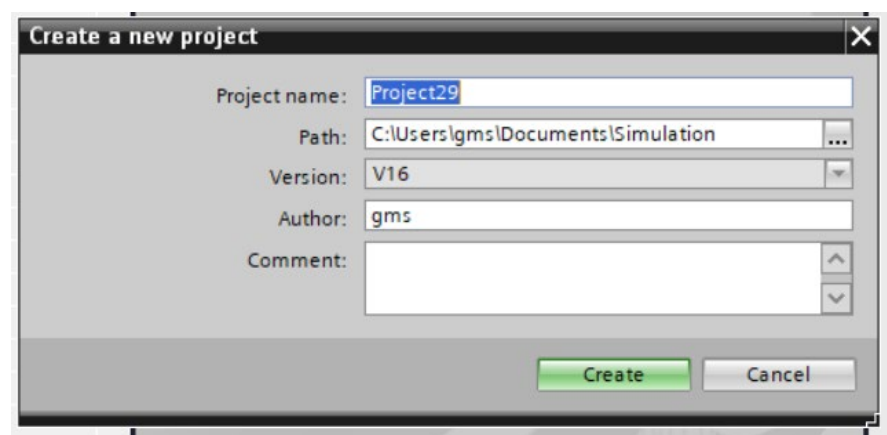


Når power-ikonet skifter til grøn, vil simulatoren være aktiveret med en ukonfigureret PLC, og PLC'en vil være i stop mode.

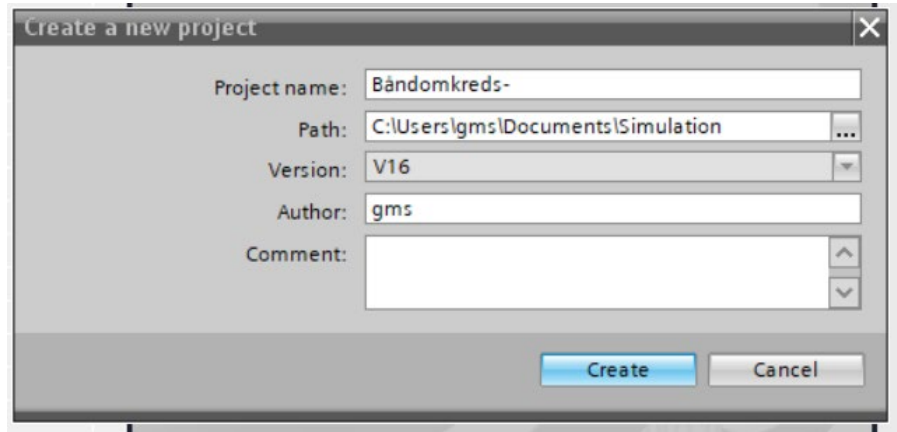
For at komme videre skal der trykkes på projektarket oppe øverst i højre hjørne.



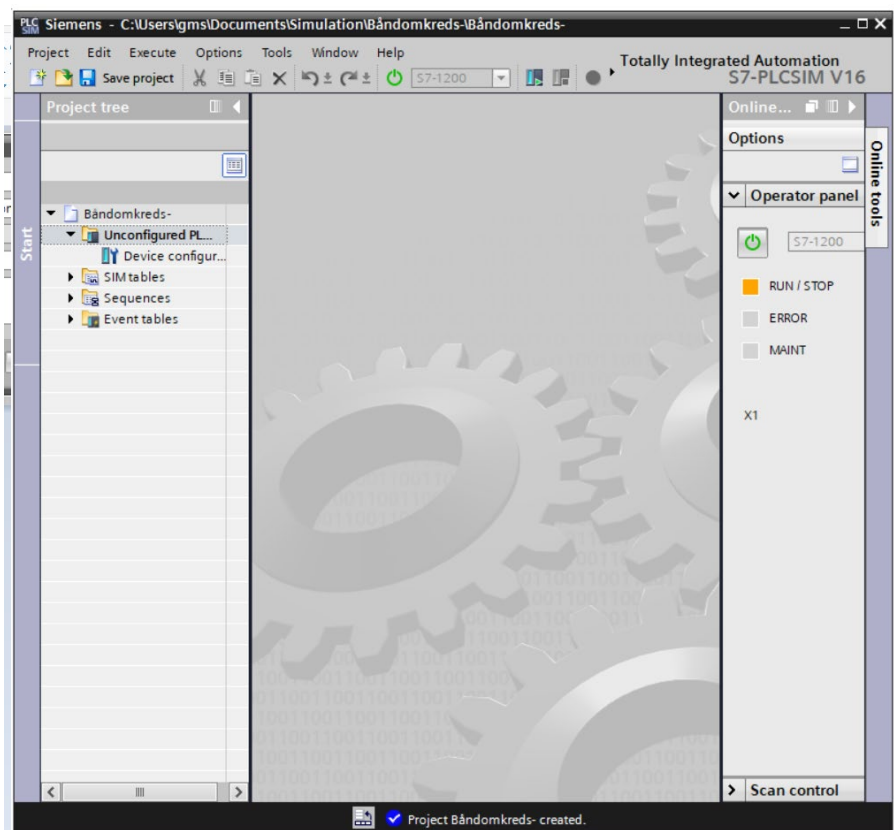
Så åbnes projektstrukturen automatisk i simulatordelen. Her skal der så laves et nyt projekt, der passer til programmet ovre i TIA-portalen. Det er altid en god ide at navngive dette projekt med samme navn som TIA-programmet, da det så altid kan findes igen og bruges sammen med dette program.



Når det er navngivet korrekt, trykkes der på "Create", og simulatoren opretter et nyt tomt dokument med navnet "Båndomkreds", som vi her vil bruge i eksemplet.



Efter et øjeblik vil programstrukturen vise et nyt projekt med navnet "Båndomkreds" med en ukonfigureret PLC og en SIM-tabel.



Nu er simulatoren grundlæggende klar til at blive brugt, så nu skal vi over i TIA-portalen og downloade programmet til simulatoren. Under download til simulatoren vil PLC'en automatisk blive konfigureret tilsvarende PLC'en i TIA-portalen.



Efter tryk på download-ikonet i TIA-portalen kommer menuen frem for download. Her vælges PLCSIM og PNIE_1, og så kan der søges efter en simulator-PLC ved tryk på "Search".

Configured access nodes of "PLC_1"

| Device | Device type | Slot | Interface type | Address | Subnet |
|--------|-------------------|------|----------------|-------------|---------|
| PLC_1 | CPU 1215C DC/D... | 1 X1 | PN/IE | 192.168.0.1 | PN/IE_1 |
| | | | | | |
| | | | | | |

Type of the PG/PC interface:

PG/PC interface:

Connection to interface/subnet:

1st gateway:

Select target device:

| Device | Device type | Interface type | Address | Target device |
|--------|-------------------|----------------|----------------|---------------|
| PLC_1 | CPU 1215C DC/D... | PN/IE | 192.168.0.1 | PLC_1 |
| — | — | PN/IE | Access address | — |

Så fremkommer der en PLC_1, og denne vælges. Herefter trykkes der på load for at downloade programmet til PLC'en. Så vil følgende "Load preview" komme op på skærmen:

Load preview

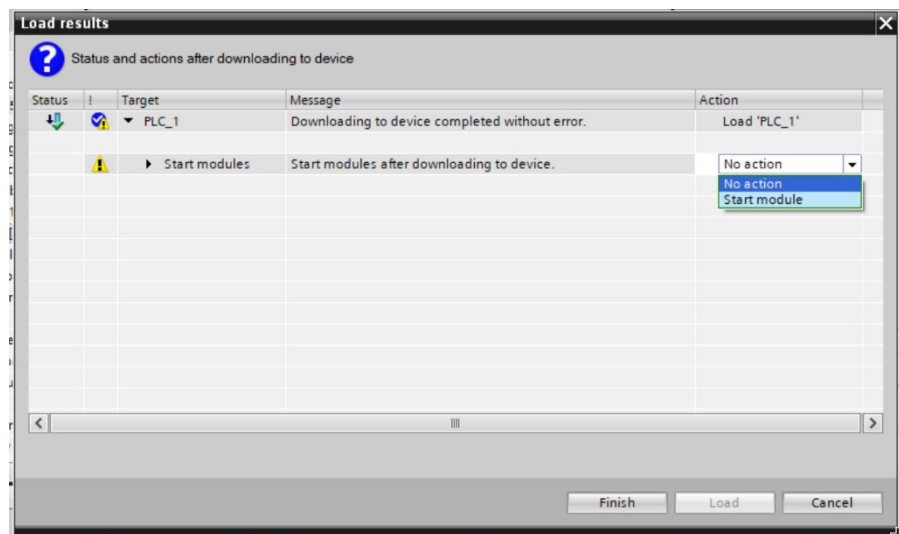
Check before loading

| Status | Target | Message | Action |
|-------------------------------------|-----------------------|--|---------------------|
| <input checked="" type="checkbox"/> | PLC_1 | Ready for loading. | Load 'PLC_1' |
| <input checked="" type="checkbox"/> | Simulated module | The loading will be performed from a simulated PLC. | |
| <input checked="" type="checkbox"/> | Device configurati... | Delete and replace system data in target | Download to device |
| <input checked="" type="checkbox"/> | Software | Download software to device | Consistent download |
| <input checked="" type="checkbox"/> | Text libraries | Download all alarm texts and text list texts to device | Consistent download |

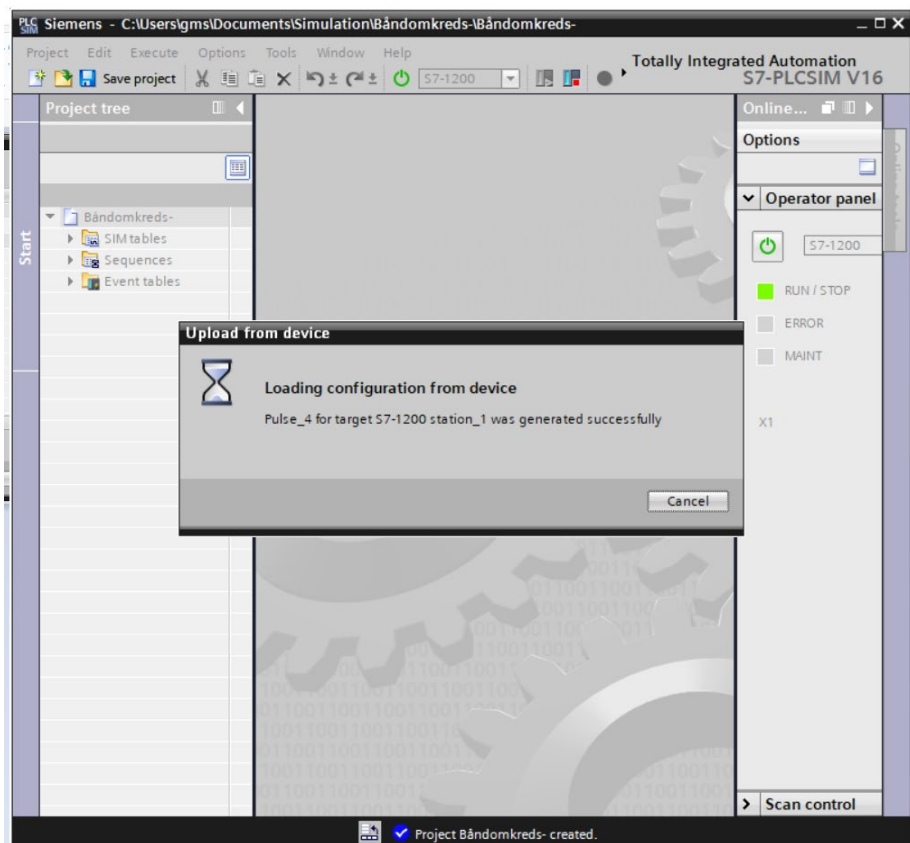
Refresh

Finish Load Cancel

Her ses, at det simulerede PLC-modul også vil blive downloadet sammen med programmet til simulatoren. Så trykkes der på "Load", således dette downloades.

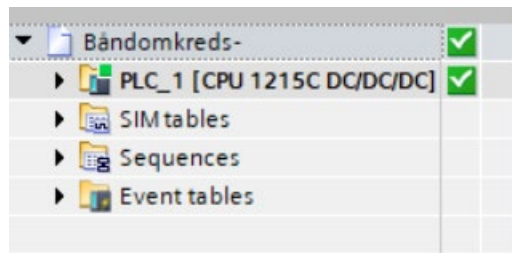


Så skal der vælges "Start module", og der kan trykkes på "Finish". Så vil selve downloaddelen starte ovre i simulatordelen, fuldstændigt som var der en rigtig PLC forbundet til PC'en. Her er det bare en virtuel PLC oprettet i S7-PLCSIM-programmet på PC'en.

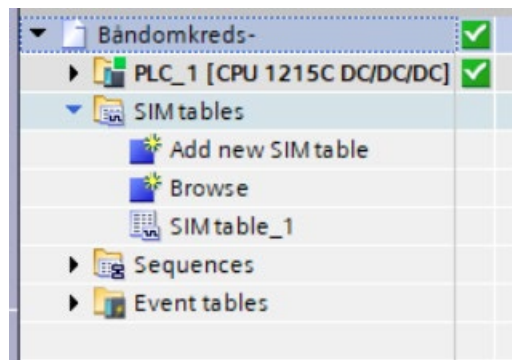


Så vil man kunne se programmet loades ovre i simulatoren, og efter lidt tid vil programmet være overført, og så vil vi næsten være klar til at kunne afvikle programmet på simulatoren. Der er dog lige et par enkelte ting, der skal tilpasses.

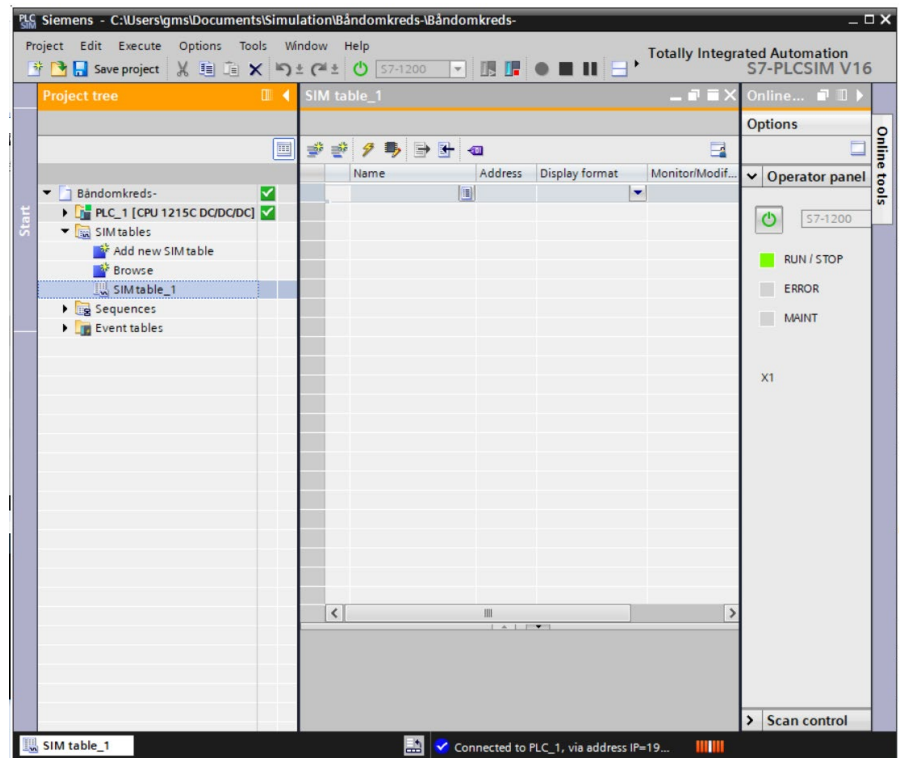
Men allerede her kan man se, at PLC'en er blevet konfigureret til den model, som vi har oprettet i TIA-portalen.



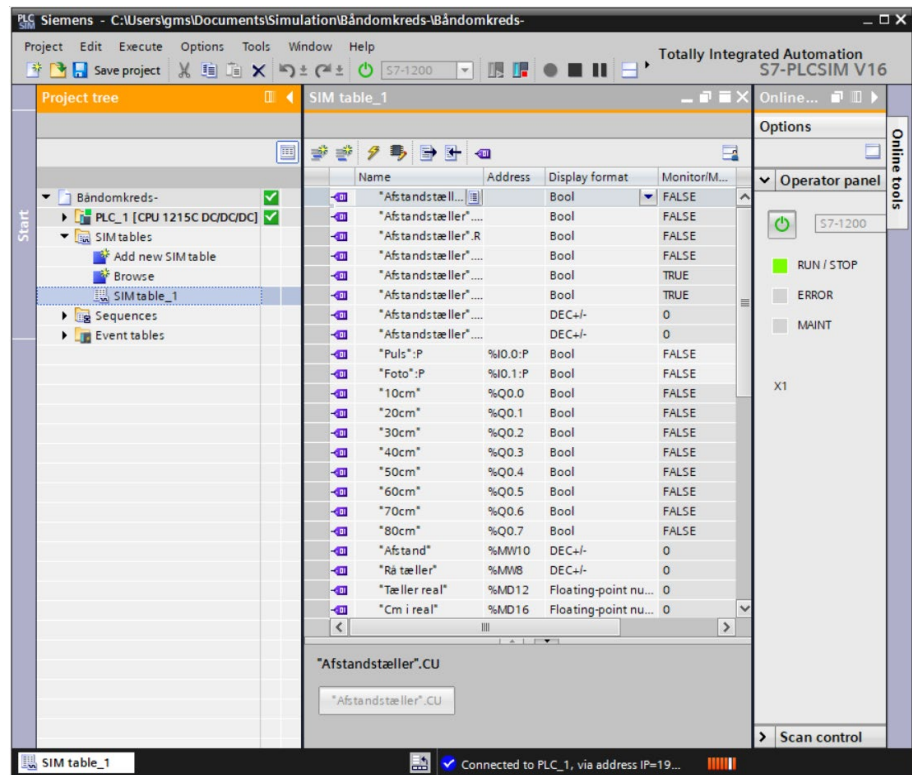
Så åbnes "SIM tables", så vi kan oprette en simuleringstabel, hvor vi kan aktivere indgangene på PLC'en, samt se hvorledes udgange og interne mærker i PLC'en vil forandres i forhold til programafviklingen i simulatoren.



Når "SIM tables" åbnes, vil der fremkomme en "SIM table_1". Denne tabel åbnes, således vi kan få sat værdier ind i tabellen.

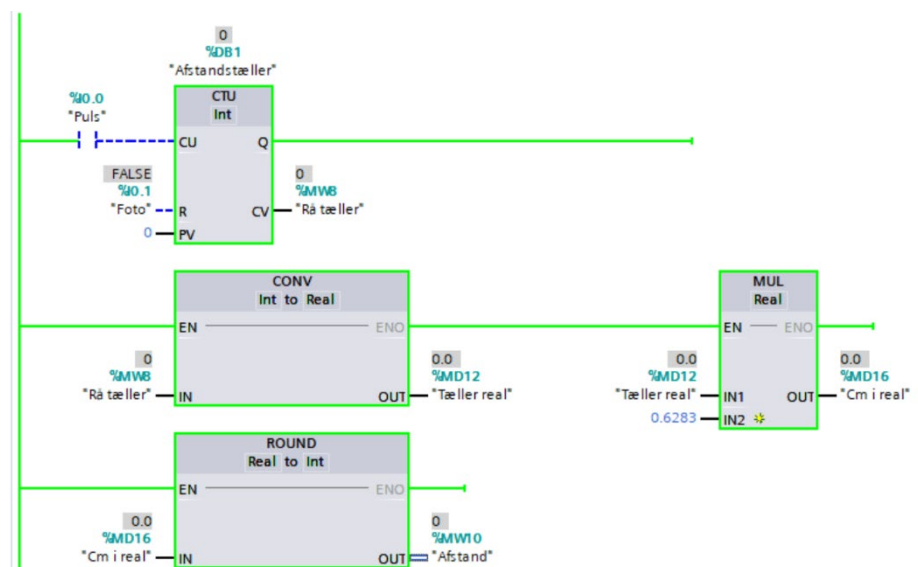


Når "SIM table_1" tabellen er åbnet, er der et lille lilla ikon øverst i tabellen "Load project tags". Denne overfører alle tags fra programmet i TIA-portalens PLC-program, så vi ikke selv behøver skrive disse ind i tabellen.



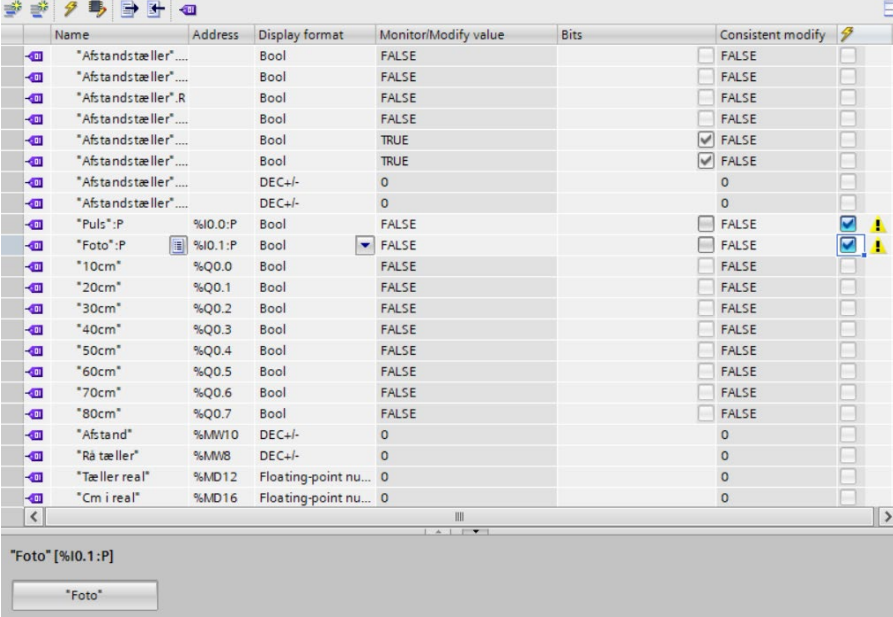
Nu er vi faktisk klar til at simulere programmet fra TIA-portalen, og i højre side af simulatordelen kan vi se, at vores virtuelle PLC er i run, da den grønne LED er tændt.

Så nu springer vi tilbage til PLC-programmet i TIA og sætter "monitoring ON", så vi kan se programmet køre.



Nu har vi forbindelse mellem TIA og simulatoren, og vi er klar til at afprøve programmet.

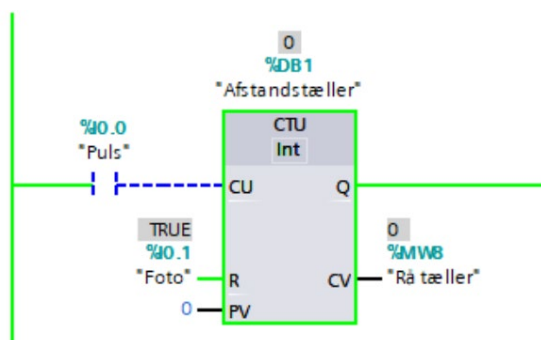
For at aktivere indgange i simulatoren skal disse vinges af i boksene med et flueben under kolonnen med lynet. Så markeres den indgang i rækken, der skal aktiveres. Her har vi valgt "Foto", som er indgang I0.1 i PLC'en. Så fremkommer der en lille knap i bunden af tabellen. Denne knap kan bruges til at aktivere indgangen "Foto".



| | Name | Address | Display format | Monitor/Modify value | Bits | Consistent modify | |
|--------------------------|----------------------|---------|----------------------|----------------------|------|---|-------------------------------------|
| <input type="checkbox"/> | "Afstandstæller".... | | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "Afstandstæller".... | | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "Afstandstæller".R | | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "Afstandstæller".... | | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "Afstandstæller".... | | Bool | TRUE | | <input checked="" type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "Afstandstæller".... | | Bool | TRUE | | <input checked="" type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "Afstandstæller".... | | DEC+/- | 0 | | 0 | |
| <input type="checkbox"/> | "Afstandstæller".... | | DEC+/- | 0 | | 0 | |
| <input type="checkbox"/> | "Puls":P | %I0.0:P | Bool | FALSE | | <input type="checkbox"/> FALSE | <input checked="" type="checkbox"/> |
| <input type="checkbox"/> | "Foto":P | %I0.1:P | Bool | FALSE | | <input type="checkbox"/> FALSE | <input checked="" type="checkbox"/> |
| <input type="checkbox"/> | "10cm" | %Q0.0 | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "20cm" | %Q0.1 | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "30cm" | %Q0.2 | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "40cm" | %Q0.3 | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "50cm" | %Q0.4 | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "60cm" | %Q0.5 | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "70cm" | %Q0.6 | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "80cm" | %Q0.7 | Bool | FALSE | | <input type="checkbox"/> FALSE | |
| <input type="checkbox"/> | "Afstand" | %MW10 | DEC+/- | 0 | | 0 | |
| <input type="checkbox"/> | "Rå tæller" | %MW8 | DEC+/- | 0 | | 0 | |
| <input type="checkbox"/> | "Tæller real" | %MD12 | Floating-point nu... | 0 | | 0 | |
| <input type="checkbox"/> | "Cm i real" | %MD16 | Floating-point nu... | 0 | | 0 | |

"Foto" [%I0.1:P]

Hvis indgangen skal være aktiveret permanent, kan man vinge den af i boksen med et flueben under kolonnen, der hedder "Bits".

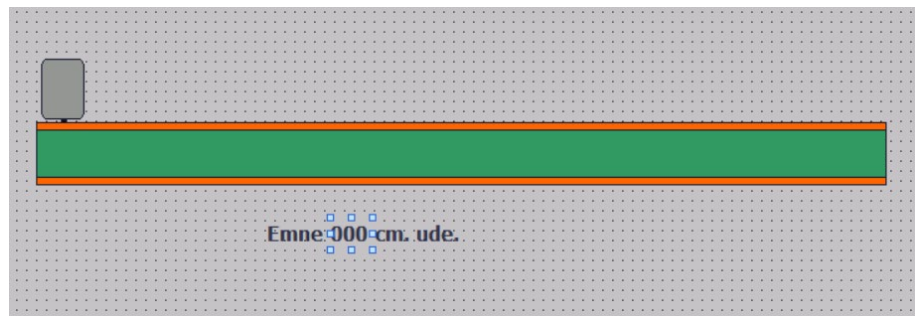


Her har vi aktiveret knappen "Foto" i simulatoren, og som vi kan se, bliver indgangen "Foto" nu aktiv (True) ovre i PLC-programmet TIA.

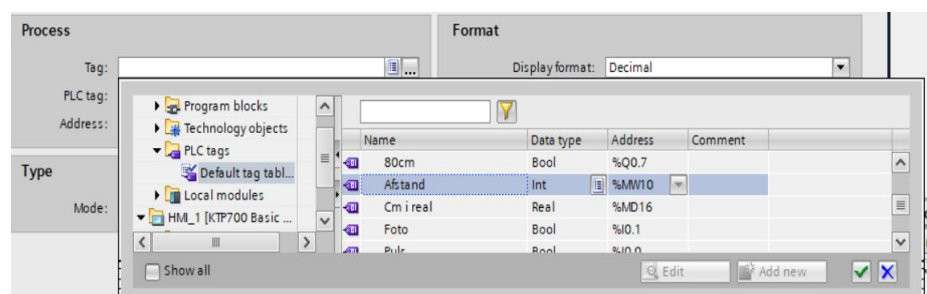
HMI og PLC

Det er også muligt at simulere HMI-skærmen op mod PLC-simulatoren, således der kan afprøves programmer uden en fysisk PLC og HMI-skærm.

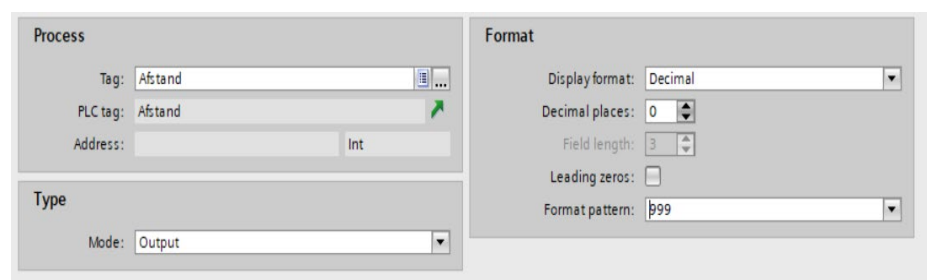
I det foregående afsnit fik vi sat PLC-simulatoren op med programmet "Båndomkreds", så nu vil vi så sætte skærmen op mod dette program og simulere skærmen sammen med PLC-simulatoren.



Vi har her oprettet en skærm, hvor vi kan se, hvor langt ude emnet befinder sig på båndet. Der er en talvisning under båndet i cm, og denne forbinder vi med tagget "Afstand".

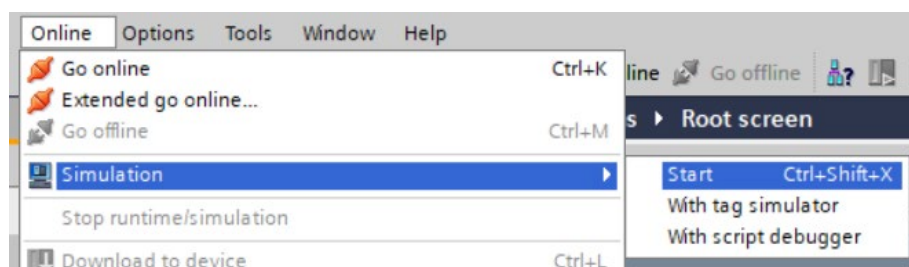


Vi vælger dette tag "Afstand" som output og som et decimaltal med længden 3.

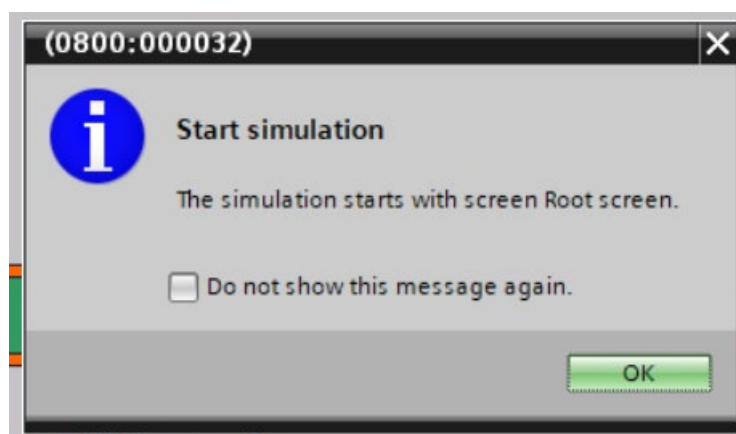


Nu er vi klar til at starte HMI-simulatoren op mod PLC-simulatoren, som vi allerede har startet, og som stadigvæk kører i baggrunden.

Oppe i den øverste bjælke vælges fanen "Online". Når denne åbnes vælges "Simulation", og under denne trykkes der på "Start".



Nu vil HMI-simulatoren starte op, og automatisk forbindes med PLC-simulatoren. Så nu er vi klar til at afprøve programmet med både PLC og HMI-skærm.

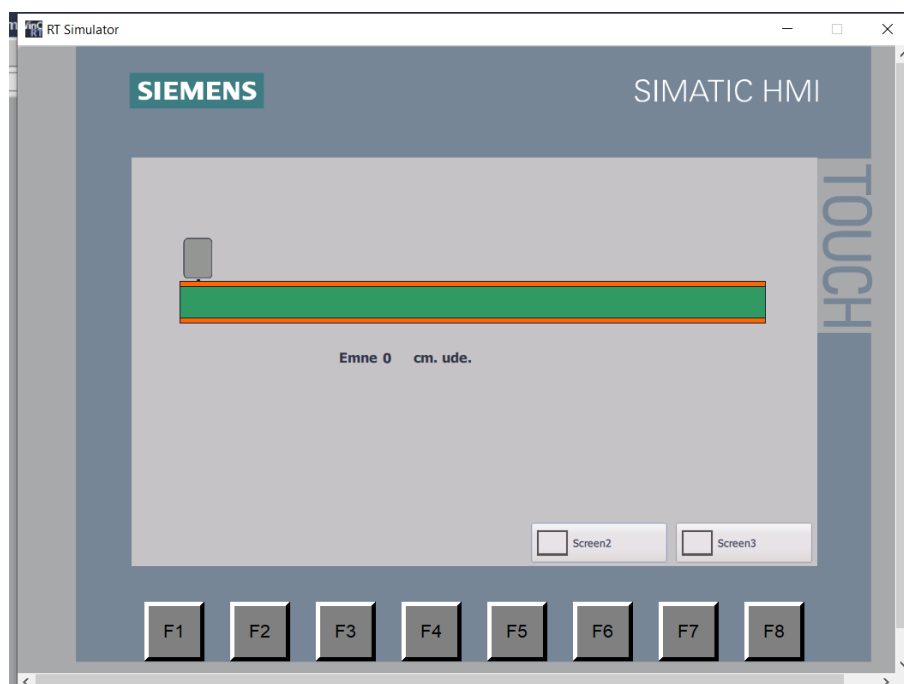


Der vælges "OK" for at starte HMI-simulatoren op.

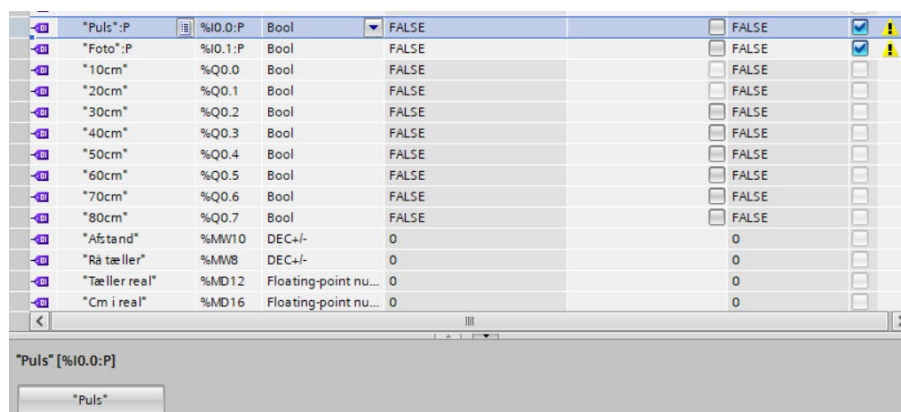


Der vil nu komme et ikon i bunden af skærmen, der viser simulatoren starte op.

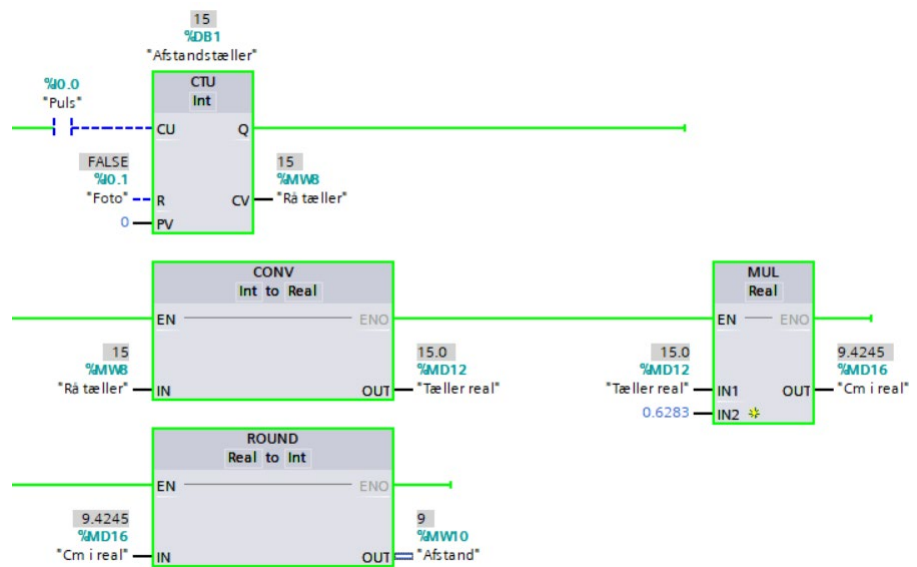
Så trykkes der på dette ikon, og HMI-skærmen dukker op – klar til at blive brugt.



Nu kan vi jo prøve at aktivere tagget "Puls" ovre i PLC-simulatoren, så vi får værdier ind i PLC'en, som så kan vises på HMI-skærmen.

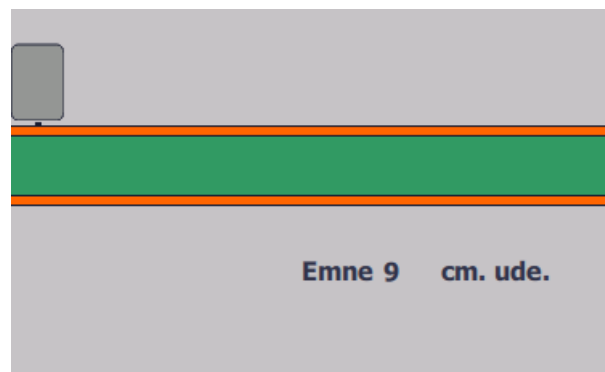


Nu aktiverer vi indgangen "Puls" 15 gange, så kan vi se i programmet, hvor langt ude emnet burde være. Dette skal så være det samme oppe i HMI-skærmens visning.



Dette viser programmet har beregnet, at vores emne nu har bevæget sig 9 cm ud på båndet.

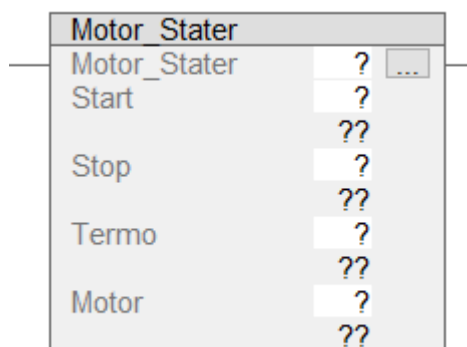
Dette vil også være den værdi, tagget "Afstand" indeholder, og derfor viser vores skærm også, at emnet er 9 cm ude.



Således kan vi altså simulere vores programmer fremover.

Allen-Bradley Add-On-instruktioner

Indledning



Add-On-instruktioner kan med fordel bruges, når man har et maskinanlæg, hvor det samme stykke program skal bruges igen og igen. F.eks. en motorstarter eller en kompliceret beregning eller en standardsekvens.

Add-On-instruktioner giver mulighed for at:

- Definere Add-On-instruktioner, som man kan skrive og genbruge på tværs af flere projekter
- Oprette Add-On-instruktionslogik ved hjælp af disse standardsprog: Ladder Diagram, Function Block Diagram og Structured Text
- Eksportere en Add-On-instruktion til en .L5X-fil, som derefter kan importeres til et andet projekt. Man kan også kopiere og indsætte mellem projekter
- Se revisionshistorik, ændringshistorik og en automatisk genereret hjælpeside for hver instruktion
- Beskytte deres brug og modifikation ved at anvende et tilladelsessæt til hver instruktion eller ved at bruge kildebeskyttelse
- Forebygge ændringer, spore revisioner for Add-On-instruktioner ved at tilføje en instruktionssignatur, som genererer en unik identifikator, og låse instruktionen mod at blive redigeret
- Når en låst instruktion eksporteres til en .L5K- eller .L5X-fil, kodes instruktionen

Programmeringsvejledning

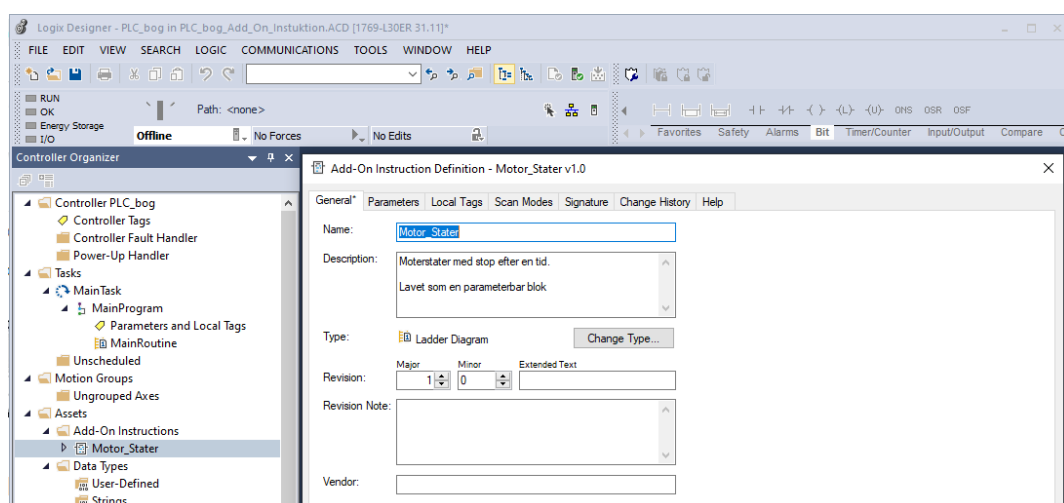
Sådan programmeres en motorstarter med automatisk stop efter tid eller manuelt stop, inden tiden er udløbet.

Opret Add-On-instruktion

Højreklik på "Add-On Instructions" i "Controller Organizer" og vælg "New Add-On Instructions"

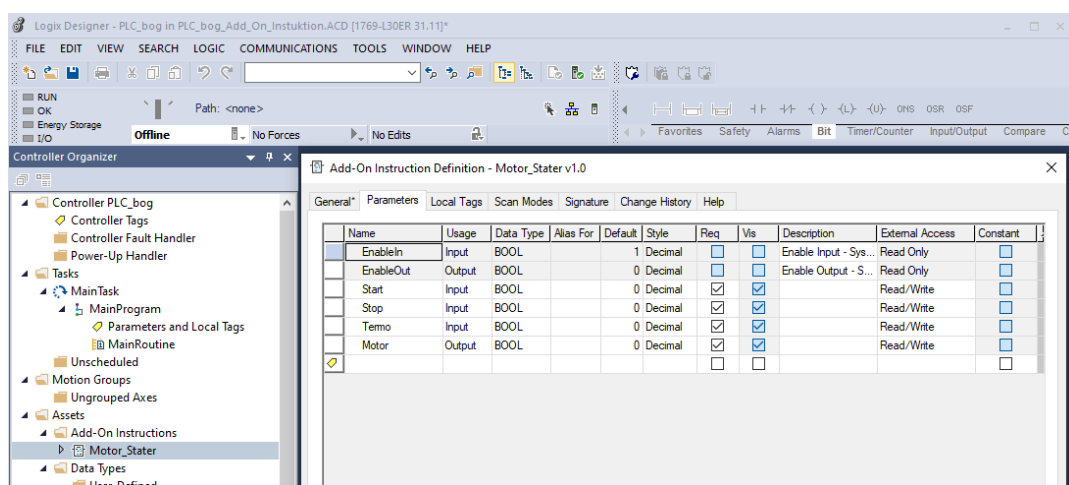
Giv instruktionen et navn, vælg programsprog og sæt flueben i "Open Definition" og klik "OK".

Nu kommer dette vindue op.



Opret ind- og udgange

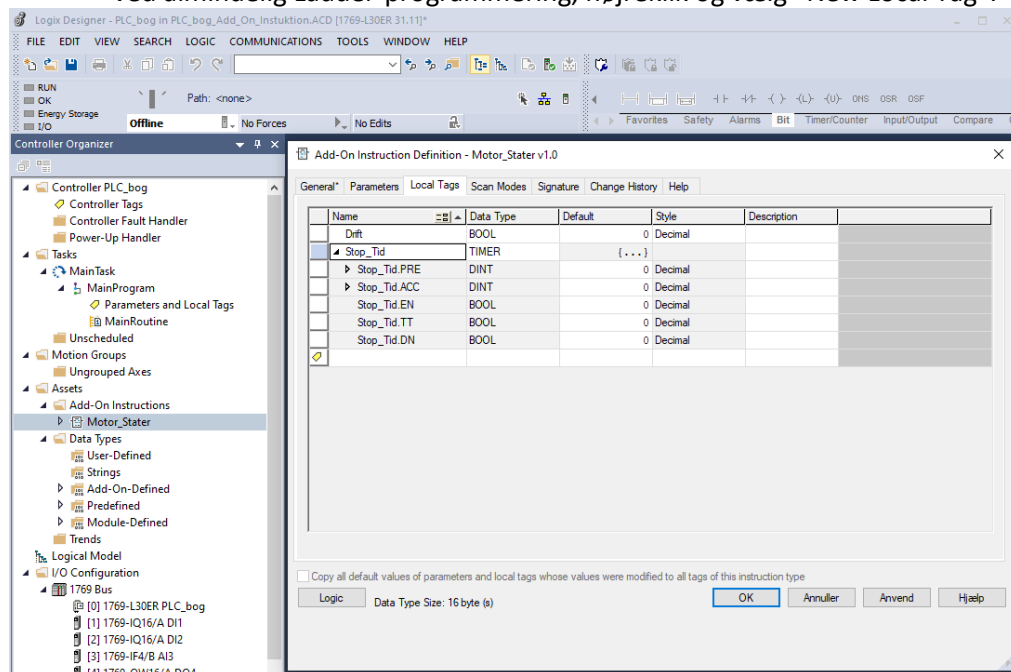
Vælg "parameters" og definer blokkens ind- og udgange som vist herunder:



Opret Local Tags

Vælg "Local Tags" og definer blokkens interne adresser (hjelpeflag). Her et driftsflag og et timer-tag.

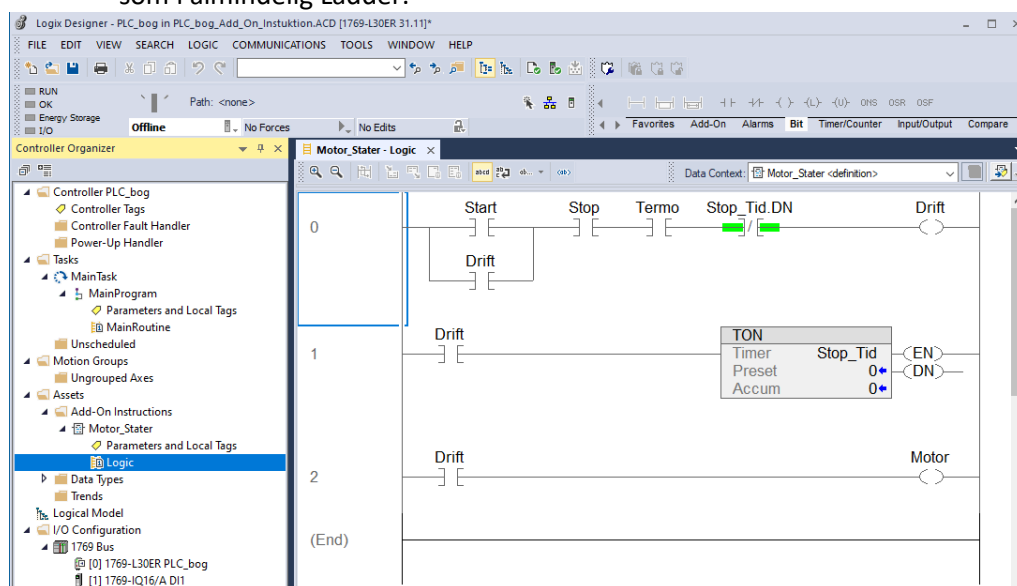
Dette kan også gøres senere på samme måde, som tags oprettes undervejs ved almindelig Ladder-programmering, højreklik og vælg "New Local Tag".



Tryk derefter "OK" for at lukke vinduet.

Opret Ladder-logik

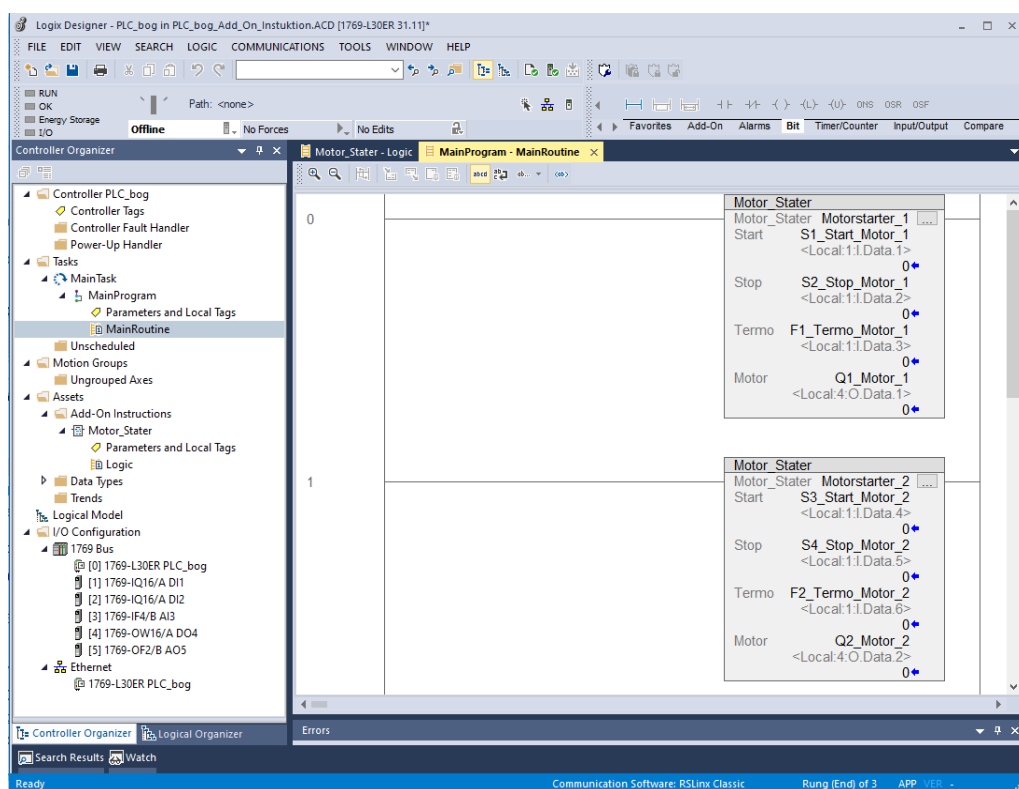
Dobbeltklik på "Logic" og programmer det viste program. På samme måde som i almindelig Ladder.



Programkald af Add-On

Når programmeringerne af Add-On-programmet er færdigt, åbnes "MainRoutine" og Add-On-instruktionen trækkes med musen ind på en tom programlinje. De ønskede adresser fra ind- og udgangskort tilknyttes blokken som vist herunder.

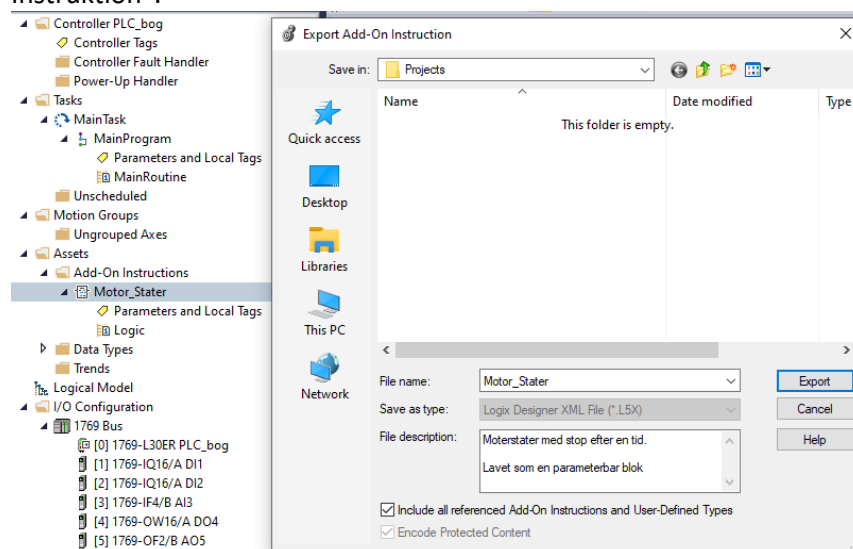
Blokken kan genbruges, så ofte man ønsker det, dog er det vigtigt, at blokken får et nyt tag hver gang. Her hedder taggene "Motorstarter_1" og "Motorstarter_2".



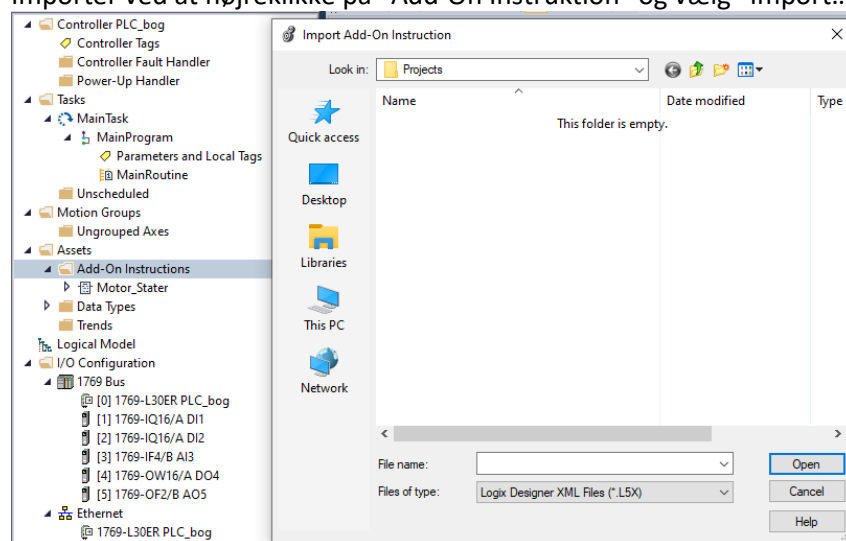
Eksport og import af Add-On

En Add-On-instruktion kan eksporteres fra et program og importeres i et andet program.

Eksporter ved at højreklikke på instruktionen og vælg ”Export Add-On Instruktion”.




Importer ved at højreklikke på ”Add-On Instruktion” og vælg ”Import...”



Allen-Bradley User Defined Type (UDT)

Indledning

Når man laver programmer til store anlæg, kan det være en fordel at kunne organisere eller bundte sine tags. Således at tags af forskellige datatyper, der hører til en specifikt maskinenhed, er samlet. Et simpelt eksempel på en samling kan ses ved at betragte tags til en timer. Tagget hedder overordnet "Tid_1" og består derefter af en række undertags. PLC'en ved, at tagget skal have denne struktur, fordi der er valgt "Data Type" "TIMER"

| Name | Usage | Alias  | Base Tag | Data Type |
|-------------|-------|---|----------|-----------|
| ▲ Tid_1 | | | | TIMER |
| ▶ Tid_1.PRE | | | | DINT |
| ▶ Tid_1.ACC | | | | DINT |
| Tid_1.EN | | | | BOOL |
| Tid_1.TT | | | | BOOL |
| Tid_1.DN | | | | BOOL |

Ved at lave en UDT eller User-Defined datatype kan man selv oprette lignende tags/"Data Type".

Programmeringsvejledning UDT Recepthåndtering



Vi forestiller os, at vi har en maskine, der kan blande dej til en række forskellige cupcakes.

Til hver cupcake er en recept/opskrift.

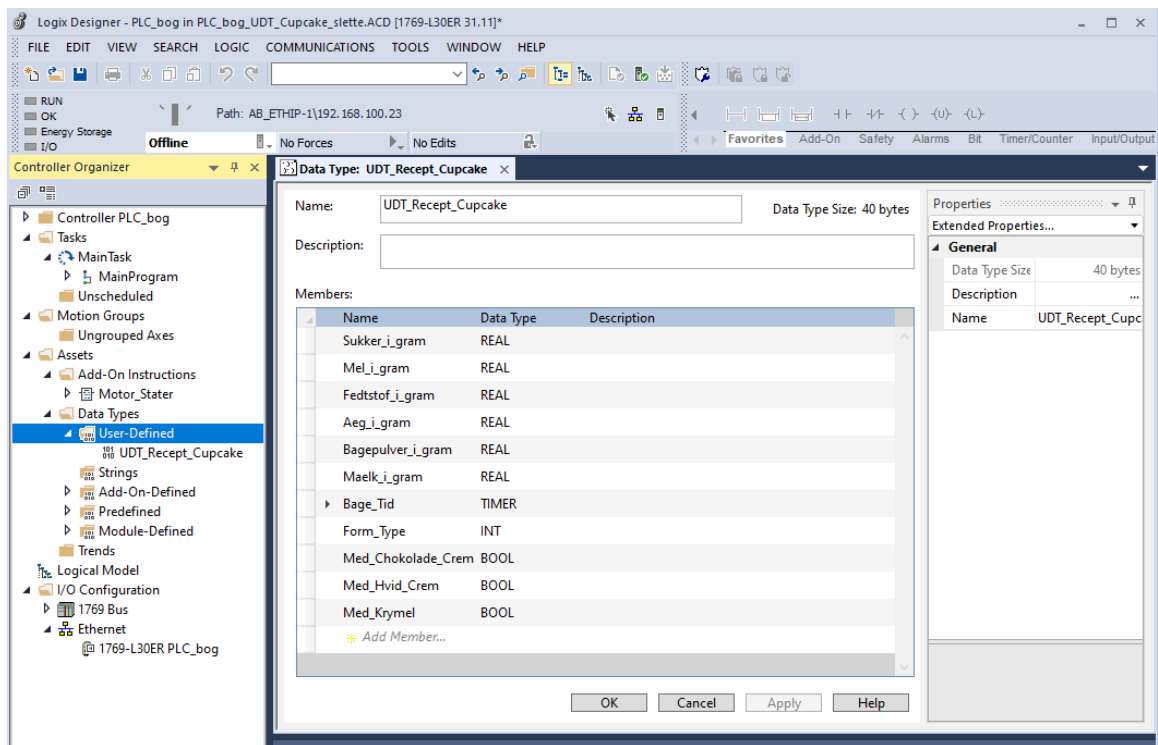
Denne recept kan med fordel gemmes i en UDT eller User-Defined datatype.

UDT kan programmeres som vist herunder.

Opret UDT

I "Controller Organizer" åbnes mappen "Data Type", og der højreklikkes på "User-Defined", og der vælges "New Data Type".

Nu kommer et tomt vindue op. Dette vindue skal gives et navn til UDT'en og udfyldes med de "Members", man ønsker, ens UDT skal indeholde, samt hvilken datatype ens "Members" skal have. Man kan også give hvert "Member"-tags en længere beskrivelse. Når alt er, som man ønsker, trykkes der ok, og vinduet lukker sig selv. UDT vil nu ligge i mappen "User Defined".



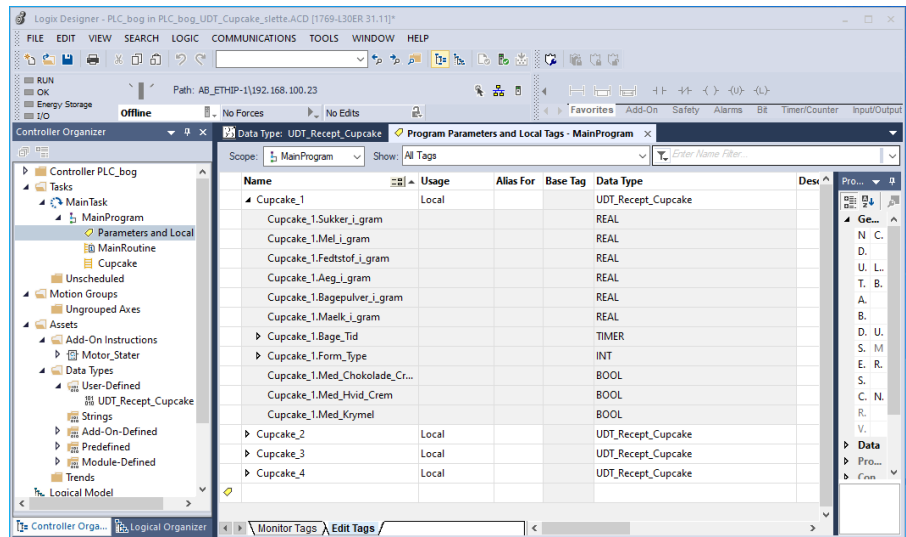
I "properties"-fanen i venstre side kan der defineres yderligere ting for de enkelte "Members", såsom ingeniørenheder og visningsformater, f.eks. decimal, hex, osv.

I eksemplet er der ingen indstillinger lavet i denne fane.

Opret UDT Programtags

Når ens UDT er færdigdefineret skal der oprettes et eller flere programtags. Her skal navnet på ens UDT-fil, i dette tilfælde "UDT_Recept_Cupcake", vælges som datatype. Derefter vil Logix oprette et overordnet tag, her "Cupcake_1", med undertag defineret på den måde, som blev angivet i definitionen af UDT'en.

Denne proces kan gentages, indtil man har lige så mange cupcake-recepter, som man ønsker at bruge.

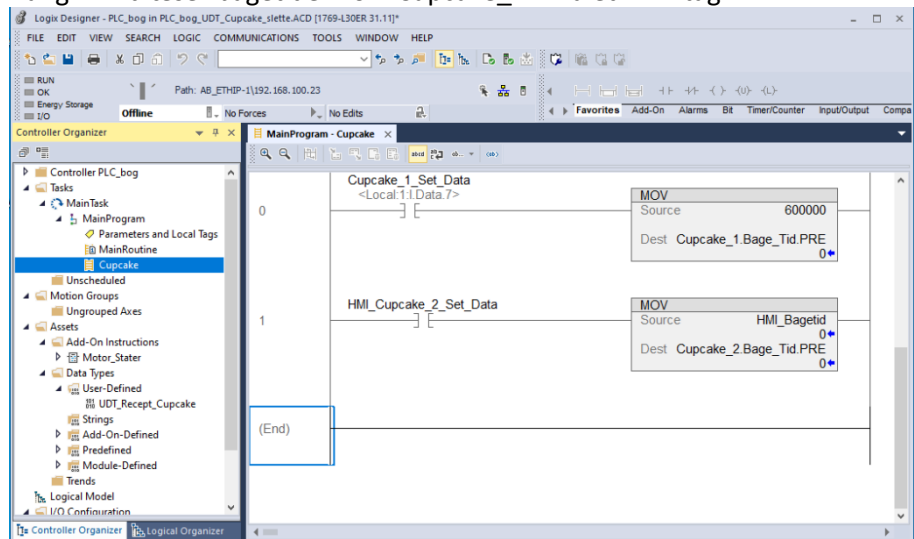


Programmering med UDT-tags

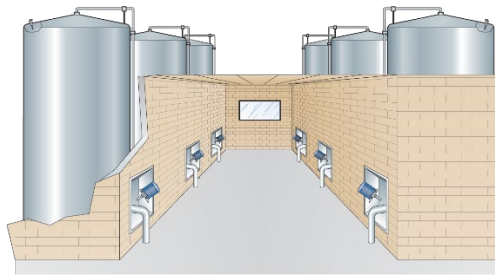
Når programtagene er færdigdefinerede, kan man programmere med dem på almindelig vis. Her er vist et eksempel:

Rung 0: Sætter bagetiden for "Cupcake_1" ved hjælp af en digital indgang.

Rung 1: Indlæser bagetiden for "Cupcake_2" fra et HMI-tag.

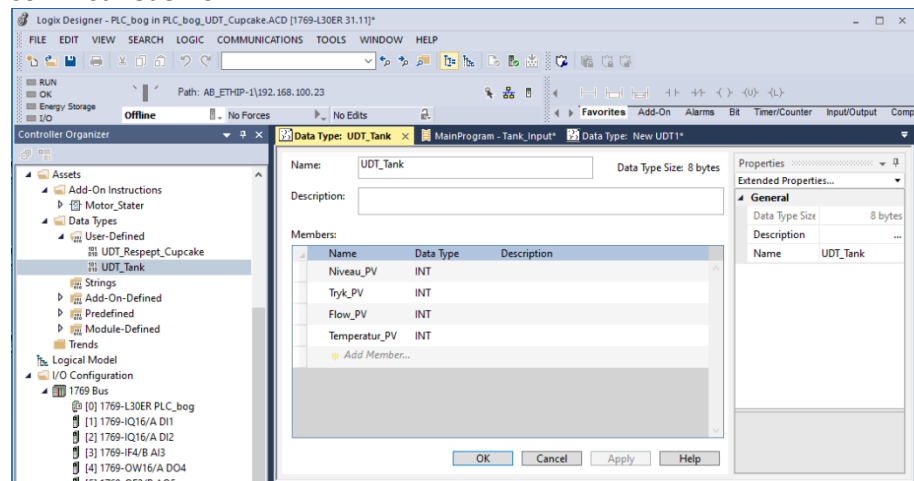


Programmeringsvejledning UDT med COP af I/O-data



Vi forestiller os, at vi har et større tankanlæg med 6 tanke. På hver af disse tanke sidder der 4 analoge procesværditransmittere. Disse 4 transmittere er tilsluttet på et inputkort med 4 analoge indgange.

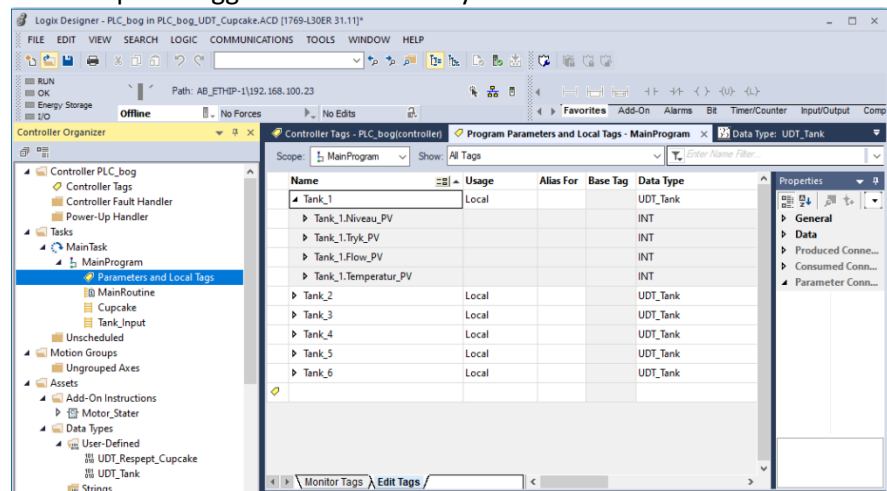
For ikke at skulle oprette de samme tags 4 tags 6 gange, opretter vi en UDT som vist nedenfor.



Opret UDT Programtags

Når ens UDT er færdigdefineret skal der oprettes et eller flere programtags med datatype "UDT_Tank".

Denne proces kan gentages, indtil man har lige så mange tank-tags, som der er tanke på anlægget – her er der 6 styks.



Programmering med UDT-tags COP

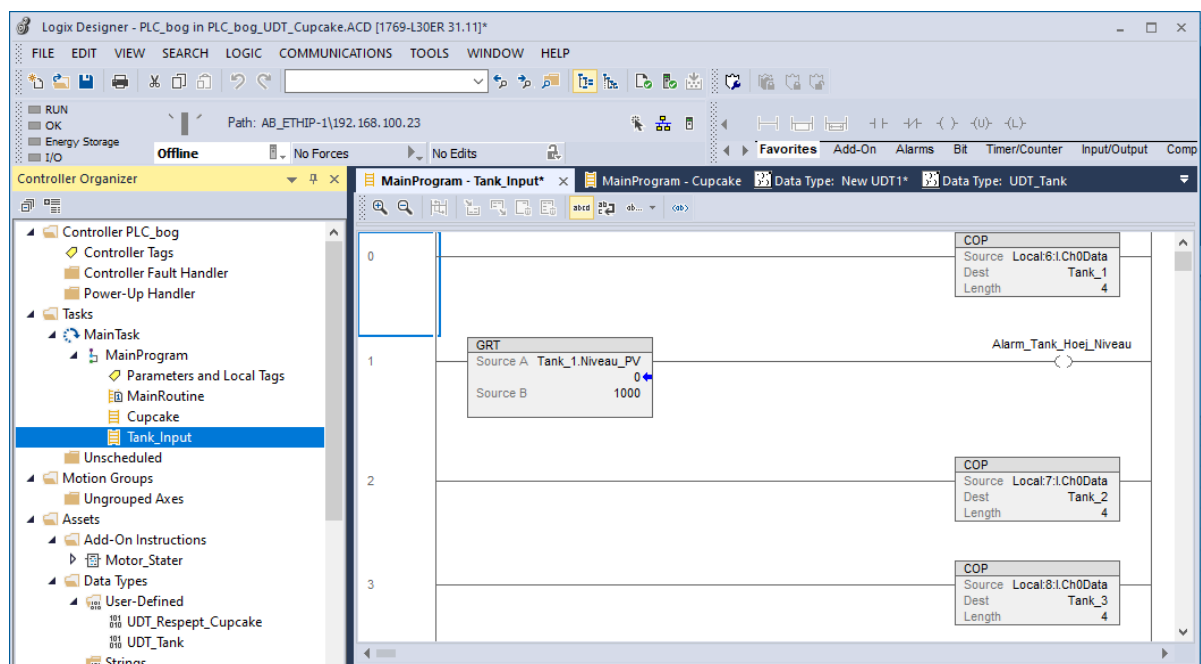
De analoge inputkort har følgende data. Det er værdierne markeret med den orange boks, der skal indlæses i UDT'en.

| Controller Tags - PLC_bog(controller) | | | | | |
|---------------------------------------|-----------|----------|-----------------|-----------------------------|--|
| Name | Alias For | Base Tag | Data Type | Description | |
| Local6:C | | | AB:1769_IF4:C:0 | | |
| Local6:I | | | AB:1769_IF4:I:0 | | |
| Local6:I.Fault | | | DINT | | |
| Local6:I.Ch0Data | | | INT | Her skrives analoge værdier | |
| Local6:I.Ch1Data | | | INT | Her skrives analoge værdier | |
| Local6:I.Ch2Data | | | INT | Her skrives analoge værdier | |
| Local6:I.Ch3Data | | | INT | Her skrives analoge værdier | |
| Local6:I.Status | | | INT | | |
| Local6:I.Ch0Status | | | BOOL | | |

For at indlæse data fra det analoge inputkort til UDT-taggene, bruges instruktionen "COP" Copy File.

Herunder:

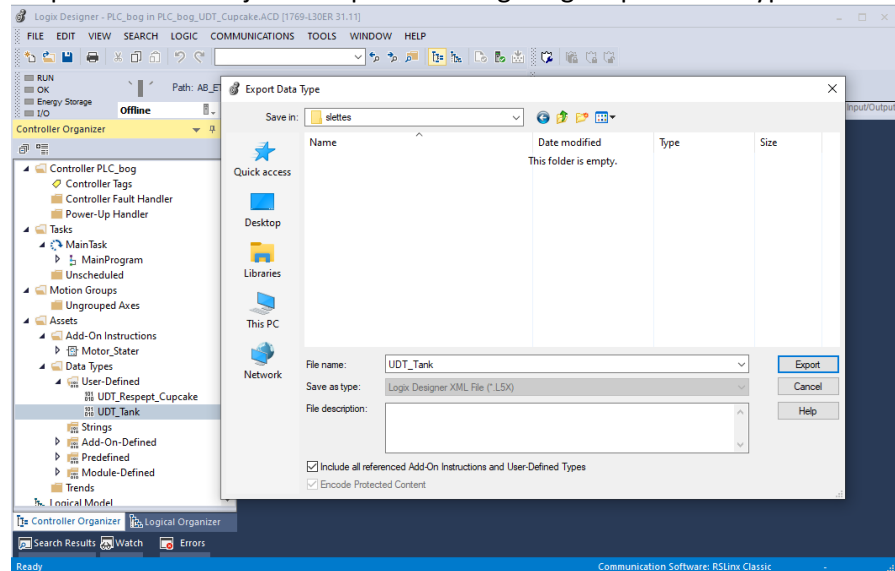
- Rung 0:
Kopieres analogt inputdata fra de 4 analoge input på "Local6:I.Ch0Data" til og med "Local6:I.Ch3Data" til tagget "Tank_1", da "Length" er sat til 4
- Rung 1:
Niveauet i "Tank_1" sætter en alarm, hvis værdien er over 1000
- Rung 2 og 3:
Analogt inputdata fra tank 2 og 3 kopieres ind



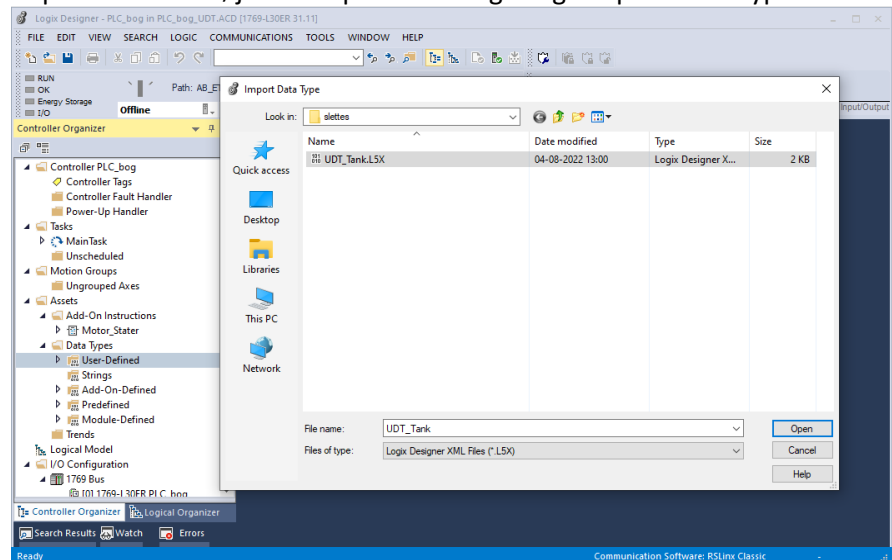
Eksport og import af UDT

En UDT User-Defined datatype kan eksporteres fra et program og importeres i et andet program.

Eksporter ved at højreklikke på UDT'en og vælg "Export Data Type".



Importer ved at højreklikke på UDT'en og vælg "Import Data Type".



Analog PLC-signaler generelt

Generelt

På et PLC-styret anlæg skelnes der mellem to typer af signaler fra processen:

- Digitale signaler
- Analoge signaler

Et digitalt signal er et on/off-signal, som f.eks. kommer fra en startkontakt, endestop eller lignende.

Et analogt signal har et signalområde, der kan antage alle værdier, f.eks. i området fra 4-20 mA eller 0-10 V. Det analoge signal kommer typisk fra en transmitter, der f.eks. anvendes til at måle en temperatur, niveau, tryk eller flow i et PLC-styret procesanlæg. De 2-trådede transmittertyper kan kun bruges ved 4-20 mA-signaler, da de 4 mA, der altid løber i kredsen, bruges til at forsyne transmitters elektronik.

Analoge standardsignaler

For at ensrette brugen af analoge signaler og derved gøre det lettere for dem, der producerer komponenter, såsom f.eks. transmittere og analoge PLC I/O-kort, findes der en række standard signalområder.

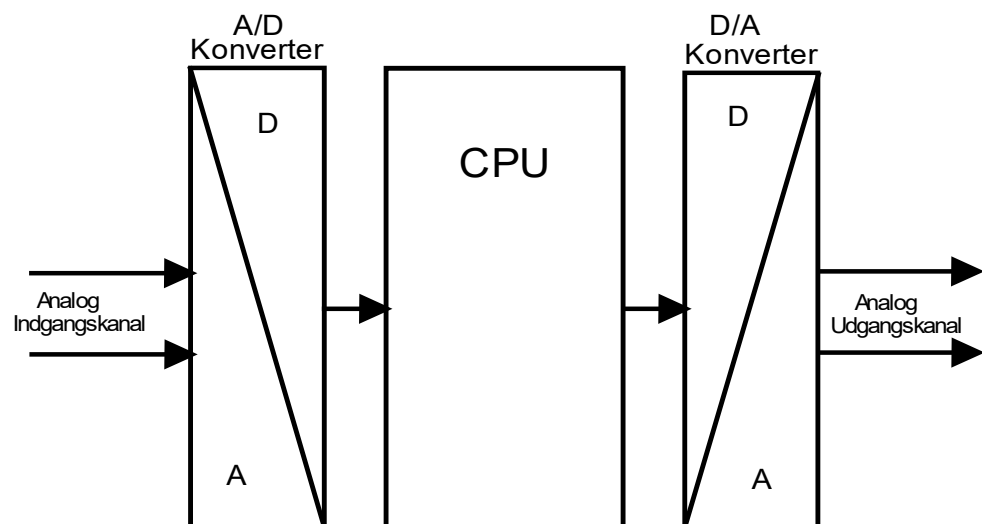
Herunder er en tabel, der viser de mest brugte standardsignaler.

| Signaltype | Fordel | Ulempe |
|-------------------|--|---|
| Spænding generelt | Let at måle på | Følsom over for EMI-støj |
| 0 - 10 V | | |
| ± 10V | Kan styre enheder, der arbejder i to retninger | |
| Strøm generelt | Kan bruges over større afstande end spændings-signaler, da strøm er mere støjimmun | Signalet skal afbrydes ved måling, da et amperemeter skal sidde i serie med kredsen |
| 0 – 20 mA | Stor signalområde / opløsning | Kabelbrud opdages ikke |
| 4 – 20 mA | Kan se kabelbrud Kan forsyne 2-tråds-transmittere | Mindre signalområde / opløsning |

Analog konvertering generelt

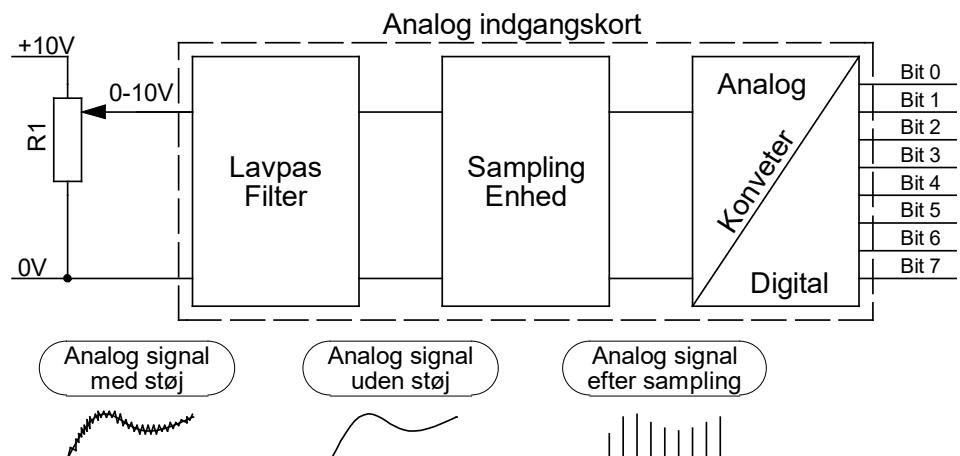
A/D-konverteren omdanner det analoge signal til et digitalt, som kan bearbejdes af PLC'ens CPU.

D/A-konverteren omdanner det digitale signal fra CPU'en til et analogt udgangssignal.



Analog til digital konvertering

Det analoge signal skal omsættes til et digitalt signal for at kunne bearbejdes i PLC'en. Denne Analog/Digital konvertering sker i et analog indgangskort.



Indlæsning af analogværdi

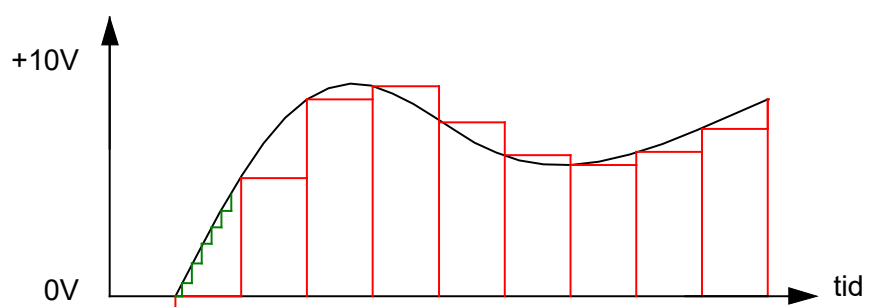
Kortet tilføres en variabel spænding på 0 til 10 VDC. Dette signal vil typisk komme fra en transmitter, men her er det for enkeltheds skyld taget fra en variabel modstand.

Lavpasfilter

Den analoge spænding kan være overlejret af et uønsket støjsignal. Lavpasfilter, som i sin enkleste form består af et RC-led, vil fjerne denne støj og sende et støjfrit analogsignal videre til samplingsenheden.

Samplingsenhed

Et analog inputkort vil ofte have flere analoge indgange, der anvender den samme analog til digitalkonverter. Derfor indlæses værdierne til konverteren på skift ved hjælp af en samplingsenhed, der også bliver kaldet en multiplexer. Den tid, der går imellem to konverteringer af det analoge signal, kaldes samplingstid eller opdateringstid. Bliver denne tid for lang i forhold til, hvor hurtigt den analoge inputværdi ændrer sig, er det ikke godt.



- Den sorte kurve:
 - Viser det faktiske analoge inputsignal
- Den røde kurve:
 - Viser den værdi PLC-programmet modtager, hvis samplingstiden er for lang i forhold til ændringen i analogværdien
- Den grønne kurve:
 - Viser en situation, hvor analogkortet har lidt lettere ved at følge med processen, men det er stadig ikke helt godt

Samplingstiden bør, hvor det er muligt, være ca. 10 gange kortere end analogværdiens ændringstid.

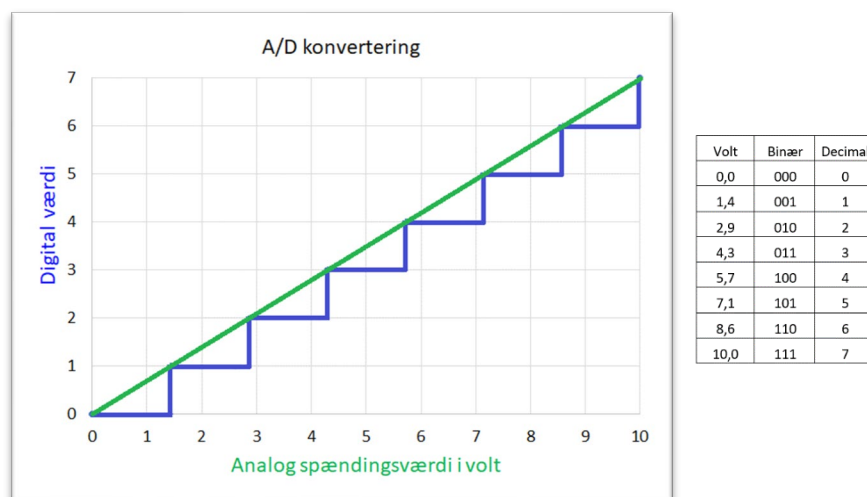
Samplingstiden kan også være oplyst som en samplingsfrekvens. Sammenhængen er således:

$$\text{Samplingsfrekvens [Hz]} = \frac{1}{\text{samplingstid [sekunder]}}$$

A/D-konverteren

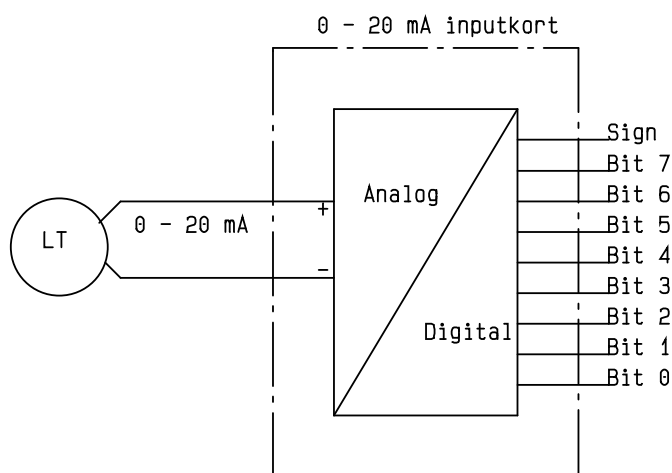
A/D-konverteren omdanner det samplede analoge signal til et digitalt signal, som kan bearbejdes af PLC'ens CPU.

For at forstå hvordan det fungerer, er her vist, hvorledes en analoginputværdi vil blive omregnet til en digital værdi i CPU'en, når konverter har 3 bits til rådighed.



Som det kan ses af eksemplet, er 3 bit for lidt til at give en tilstrækkelig præcis digitalværdi, og derfor bruges der i praksis digital til analoge konvertere med langt flere udgangsbit.

Beregning af opløsning



Diagrammet viser et principdiagram for en analog indgang, der er tilsluttet et 0 - 20 mA signal fra en niveautransmitter (LT).

I dette eksempel omsætter inputkortet det analoge signal til et digitalt signal med 8 bit plus et fortegnssbit (Sign). Fortegnssbittet anvendes til at angive, om signalet er positiv eller negativt.

Opløsningen på kortet kan beregnes ved, at måleområdet divideres med antal af digitale kombinationer.

Ved et 8 bits-opløsning får man 256 kombinationer.

$$\text{Opløsning} = \frac{\text{Måleområde}}{\text{Kombinationer}} = \frac{20}{256} = 0,078125 \text{ mA} = 78,125 \mu\text{A}$$

Dette betyder, at indgangssignalet skal ændres 78,125 μA , for at mindst betydende bit, LSB, skifter.

De digitale bit overføres til en adresse i PLC'en, hvor værdierne kan bearbejdes med PLC'ens ordinstruktioner.

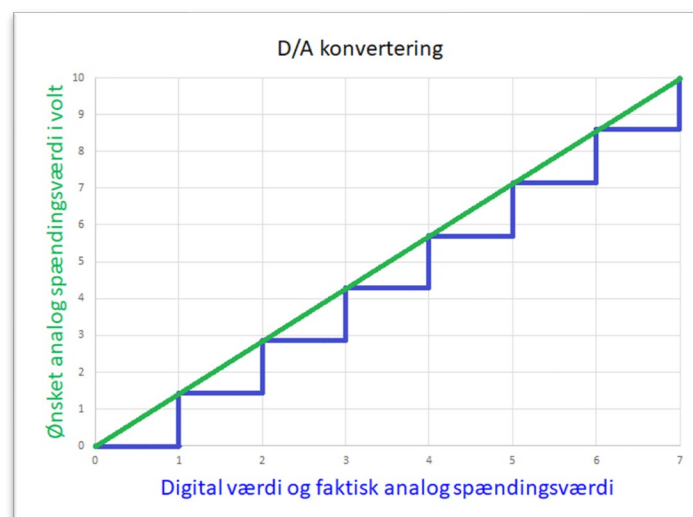
Når man skal vælge et analogkort til en given opgave, kan de forskellige fabrikanten levere kort med forskellige opløsninger, f.eks. kan Siemens levere analogkort med 8, 12, 13, 14 eller 16 bits opløsning.

Digital til Analog konvertering

D/A-konverteren

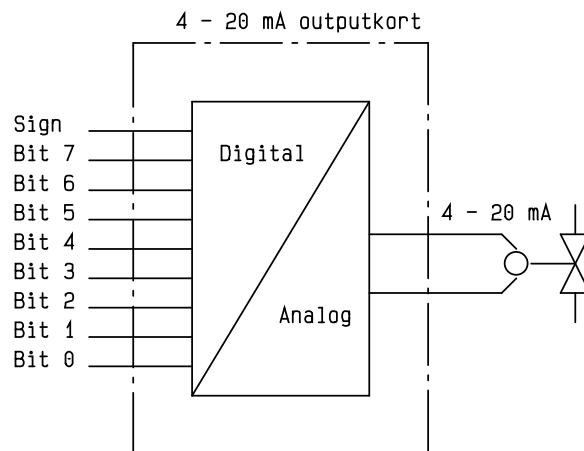
D/A konverteren omdanner det digitale signal fra CPU'en til et analogt udgangssignal.

For at forstå hvordan det fungerer, er her vist hvorledes, en digitalværdi i CPU'en vil blive omregnet til en analogudgangsværdi, når konverter har 3 bits til rådighed.



Som det kan ses af eksemplet, er 3 bit for lidt til at give en tilstrækkelig præcis analogværdi på udgangen, og derfor bruges der i praksis digital til analoge konvertere med langt flere indgangsbit.

Beregning af opløsning



Digital til analog-konverteren (D/A) er grænsefladen mellem PLC'en og de analoge udgangssignaler. Som ved A/D-konverteren gælder det, at jo flere bit, der arbejdes med, jo højere opløsning opnås på den analoge udgang.

Eksempel

Den binære værdi overføres fra en PLC-adresse igennem optokoblere for at opnå galvanisk adskillelse til D/A-konverteren, som afgiver et signal, hvis størrelse svarer til den binære værdi fra PLC'en.

Signalet forstærkes op og styrer en strømgenerator, som afgiver et 0 til 20 mA-signal på udgangsklemmerne.

D/A-konverteren har i det viste eksempel en opløsning på 8 bit, og dette giver $2^8 = 256$ kombinationsmuligheder.

Dermed bliver opløsningen = $\frac{\text{Måleområde}}{\text{Antal kombinationer}} = \frac{20 - 4}{256} = 0,0625$ mA

Dette betyder, at udgangsstrømmen ændres i step med minimum 62,5 µA.

Afhjælpning af elektrisk støj (EMC)

For at reducere problemer med elektrisk støj (EMC) skal der bruges parsnoet skærmet kabel for analoge signaler.

Skærmen skal afsluttes i begge ender.

Hvis der er potentialeforskelle mellem de to kabelender, og hvis det medfører en støjstrøm i skærmen, som forstyrrer det analoge signal, så skal skærmen afbrydes i den ene ende.

Der kan også etableres en jordet udligningsforbindelse mellem de to kabelender for på den måde at fjerne støjstrøm i skærmen. Er dette muligt, bør det være den foretrukne løsning.

Analog signaler Siemens S7-1200

Generelt

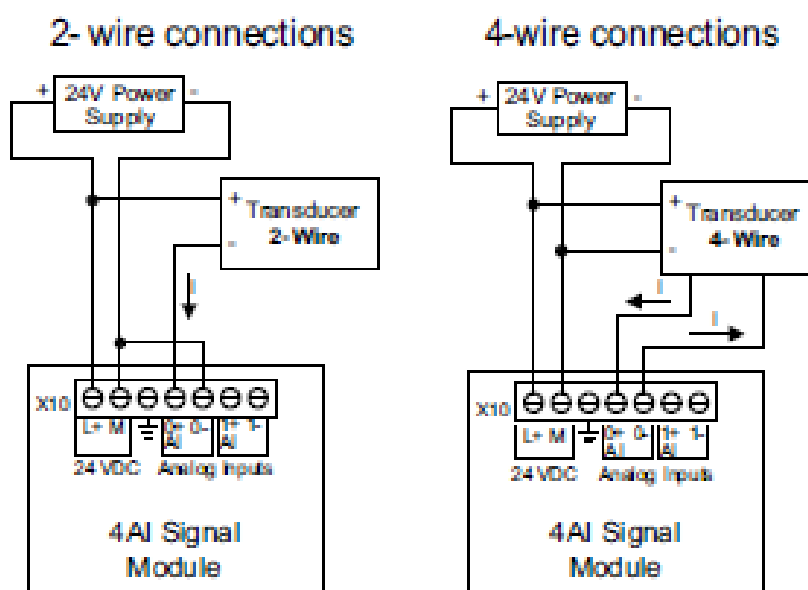


Billedet viser et Siemens SM 1234-analogkort. Dette kort er et kombinationskort med fire analoge indgange og to analoge udgange og en opløsning på for indgange 12 bit plus fortegn og for spændingsudgange 13 bit plus fortegn og 13 bit for strøm.

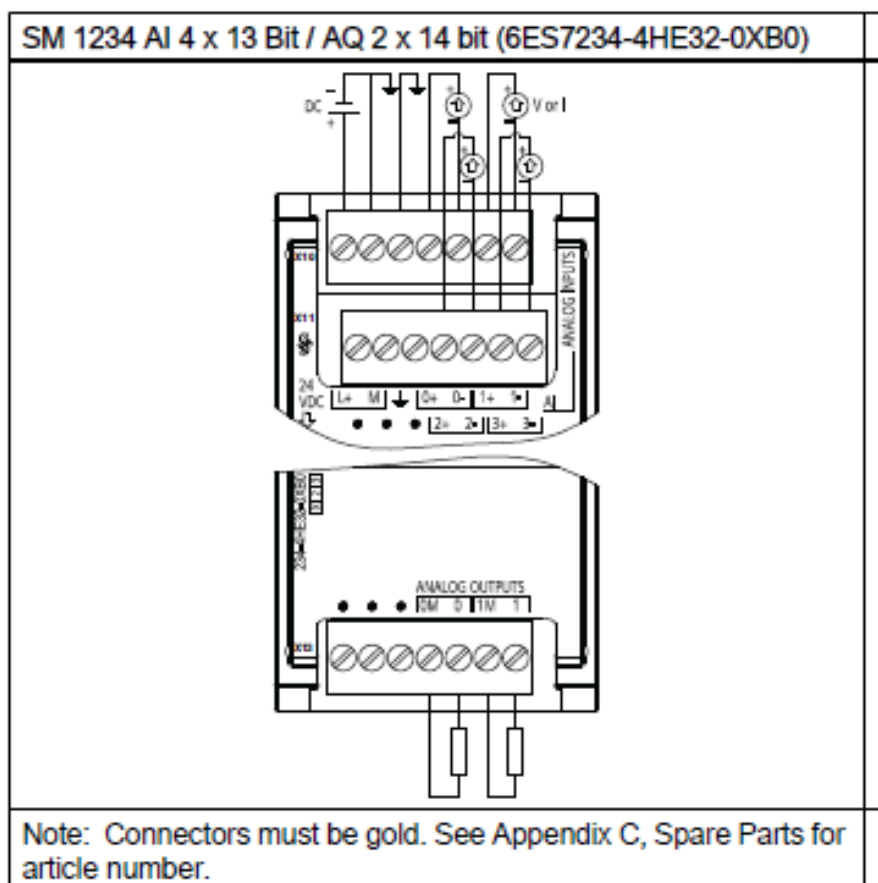
Signalområdet kan vælges mellem ± 10 V eller 0-20 mA, 4-20 mA.

Blokdiagram/tilslutning

Der kan, som vist, tilsluttes transmittere med 2 og 4 ledere.



Diagrammet herunder viser et blokdiagram for det analoge kort. Hvis der skal tilsluttes et strømsignal eller spændingssignal på kanal 0, anvendes klemme 0+ og 0-. Typen indstilles i hardwarekonfigurationen.



Kortet skal forsynes med en 24 V DC på klemme L+ og M.

Data

Table A-155 General specifications

| | |
|-------------------------------|--|
| Technical data | SM 1234 AI 4 x 13 bit / AQ 2 x 14 bit |
| Article number | 6ES7234-4HE32-0XB0 |
| Dimensions W x H x D (mm) | 45 x 100 x 75 |
| Weight | 220 grams |
| Power dissipation | 2.4 W |
| Current consumption (SM Bus) | 80 mA |
| Current consumption (24 V DC) | 60 mA (no load) |

Table A-156 Analog inputs

| | |
|--|--|
| Model | SM 1234 AI 4 x 13 bit / AQ 2 x 14 bit |
| Number of inputs | 4 |
| Type | Voltage or Current (differential): Selectable in groups of 2 |
| Range | ± 10 V, ± 5 V, ± 2.5 V, 0 to 20 mA, or 4 mA to 20 mA |
| Full scale range (data word) | -27648 to 27648 |
| Overshoot/undershoot range (data word) | Voltage: 32511 to 27649 / -27649 to -32512 Current: 32511 to 27649 / 0 to -4864 Refer to the section on input ranges for voltage and current (Page 1409). |
| Overflow/underflow (data word) | Voltage: 32767 to 32512 / -32513 to -32768 Current: 32767 to 32512 / -4865 to -32768 Refer to the section on input ranges for voltage and current (Page 1409). |
| Resolution | 12 bits + sign bit |
| Maximum withstand voltage/current | ± 35 V / ± 40 mA |
| Smoothing | None, weak, medium, or strong Refer to the section on step response times (Page 1408). |
| Noise rejection | 400, 60, 50, or 10 Hz Refer to the section on sample rates (Page 1408). |
| Input impedance | ≥ 9 M Ω (voltage) / ≥ 270 Ω , < 290 Ω (current) |
| Isolation (field side to logic) | None |
| Accuracy (25 °C / -20 to 60 °C) | $\pm 0.1\%$ / $\pm 0.2\%$ of full scale |
| Analog to digital conversion time | 625 μ s (400 Hz rejection) |
| Common mode rejection | 40 dB, DC to 60 Hz |

| | |
|---------------------------------------|--|
| Model | SM 1234 AI 4 x 13 bit / AQ 2 x 14 bit |
| Operational signal range ¹ | Signal plus common mode voltage must be less than +12 V and greater than -12 V |
| Cable length (meters) | 100 m, twisted and shielded |

¹ Voltages outside the operational range applied to one channel may cause interference on other channels.

Table A-157 Analog outputs

| Technical data | SM 1234 AI 4 x 13 bit / AQ 2 x 14 bit |
|--------------------------------------|---|
| Number of outputs | 2 |
| Type | Voltage or current |
| Range | ± 10 V or 0 to 20 mA or 4 mA to 20 mA |
| Resolution | Voltage: 14 bits ; Current: 13 bits |
| Full scale range (data word) | Voltage: -27648 to 27648; Current: 0 to 27648 Refer to the section on output ranges for voltage and current (Page 1410). |
| Accuracy (25 °C / -20 to 60 °C) | $\pm 0.3\%$ / $\pm 0.6\%$ of full scale |
| Settling time (95% of new value) | Voltage: 300 μ s (R), 750 μ s (1 uF) Current: 600 μ s (1 mH), 2 ms (10 mH) |
| Load impedance | Voltage: $\geq 1000 \Omega$ Current: $\leq 600 \Omega$ |
| Maximum output short circuit current | Voltage mode: ≤ 24 mA Current mode: ≥ 38.5 mA |
| Behavior on RUN to STOP | Last value or substitute value (default value 0) |
| Isolation (field side to logic) | none |
| Isolation (24 V to output) | none |
| Cable length (meters) | 100 m twisted and shielded |

Afhjælpning af elektrisk støj (EMC)

De krav, der er beskrevet i afsnittet "Analoge PLC-signaler generelt" vedrørende EMC, skal følges. Hvis dette ikke er nok til at sikre et støjfrit analogsignal, er der mulighed for at anvende signaludglatning.

Signaludglatning

Som en del af hardware-konfigureringen skal der tages stilling til udglatning af analogsignalet. Dette skal gøres for at afhjælpe, at uønskede signaler, der kommer på grund af elektrisk støj, indvirker på den analoge værdi. Det kan gøres ved hjælp af "Smoothing". Det betyder, at PLC'en tager et gennemsnittet af et antal programcyklusser. Det anbefales at vælge det højeste, hvis det kan lade sig gøre.

A.10.4 Step response of the analog inputs

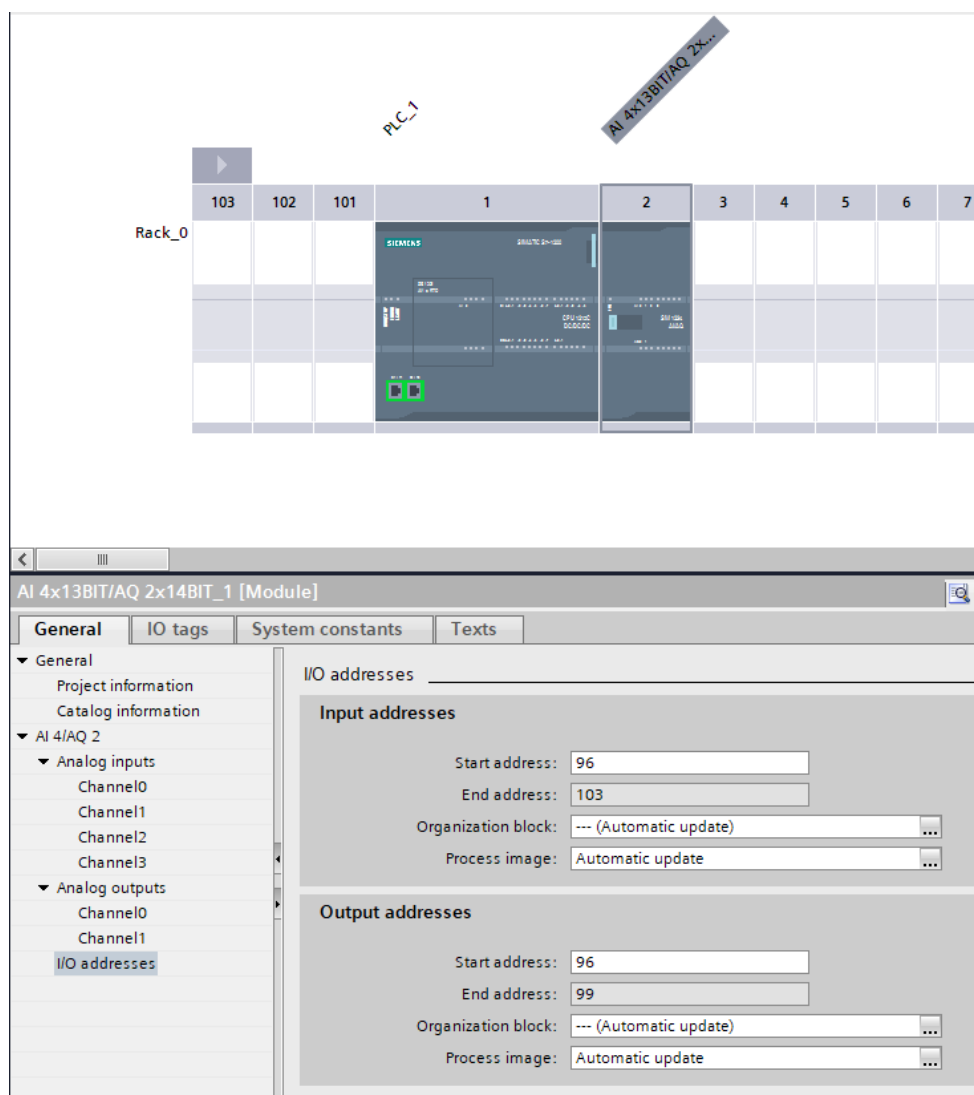
Table A-161 Step response (ms), 0 to full-scale measured at 95%

| Smoothing selection (sample averaging) | Noise reduction/rejection frequency (Integration time selection) | | | |
|--|--|-----------------|---------------|----------------|
| | 400 Hz (2.5 ms) | 60 Hz (16.6 ms) | 50 Hz (20 ms) | 10 Hz (100 ms) |
| None (1 cycle): No averaging | 4 ms | 18 ms | 22 ms | 100 ms |
| Weak (4 cycles): 4 samples | 9 ms | 52 ms | 63 ms | 320 ms |
| Medium (16 cycles): 16 samples | 32 ms | 203 ms | 241 ms | 1200 ms |
| Strong (32 cycles): 32 samples | 61 ms | 400 ms | 483 ms | 2410 ms |

Adressering

Analogkortet indlæser den analoge værdi på en ordadresse. Den tildelte adresse afhænger af kortets placering i racket.

Hvis analogkortet er placeret i rackposition 2, så er standardadressen for kanal 0, adresse IW 96 for indgangen og QW 96 for udgangen. Adressen kan ses og ændres i hardware-konfigurationen.



Analog værdi

Det mest betydende bit (MSB) er et fortegnsbitt, som fortæller om værdien er positiv eller negativ. Hvis fortegnsbittet er lig med logisk 0, så er tallet positivt, mens tallet er negativt, når fortegnsbittet er logisk 1.

A.10.6 Measurement ranges of the analog inputs for voltage (SB and SM)

Table A-163 Analog input representation for voltage (SB and SM)

| System | | Voltage Measuring Range | | | | |
|---------|-------------------|-------------------------|----------|----------|----------|------------------|
| Decimal | Hexadecimal | ±10 V | ±5 V | ±2.5 V | ±1.25 V | |
| 32767 | 7FFF ¹ | 11.851 V | 5.926 V | 2.963 V | 1.481 V | Overflow |
| 32512 | 7F00 | | | | | |
| 32511 | 7EFF | 11.759 V | 5.879 V | 2.940 V | 1.470 V | Overshoot range |
| 27649 | 6C01 | | | | | |
| 27648 | 6C00 | 10 V | 5 V | 2.5 V | 1.250 V | Rated range |
| 20736 | 5100 | 7.5 V | 3.75 V | 1.875 V | 0.938 V | |
| 1 | 1 | 361.7 µV | 180.8 µV | 90.4 µV | 45.2 µV | |
| 0 | 0 | 0 V | 0 V | 0 V | 0 V | |
| -1 | FFFF | | | | | |
| -20736 | AF00 | -7.5 V | -3.75 V | -1.875 V | -0.938 V | Undershoot range |
| -27648 | 9400 | -10 V | -5 V | -2.5 V | -1.250 V | |
| -27649 | 93FF | | | | | |
| -32512 | 8100 | -11.759 V | -5.879 V | -2.940 V | -1.470 V | Underflow |
| -32513 | 80FF | | | | | |
| -32768 | 8000 | -11.851 V | -5.926 V | -2.963 V | -1.481 V | |

¹ 7FFF can be returned for one of the following reasons: overflow (as noted in this table), before valid values are available (for example immediately upon a power up), or if a wire break is detected.

Table A-164 Analog input representation for current (SB and SM)

| System | | Current measuring range | | |
|--------------------|-------------|-------------------------|-----------------|------------------|
| Decimal | Hexadecimal | 0 mA to 20 mA | 4 mA to 20 mA | |
| 32767 | 7FFF | > 23.52 mA | > 22.81 mA | Overflow |
| 32511 | 7EFF | 23.52 mA | 22.81 mA | Overshoot range |
| 27649 | 6C01 | | | |
| 27648 | 6C00 | 20 mA | 20 mA | Nominal range |
| 20736 | 5100 | 15 mA | 16 mA | |
| 1 | 1 | 723.4 nA | 4 mA + 578.7 nA | |
| 0 | 0 | 0 mA | 4 mA | |
| -1 | FFFF | | | |
| -4864 | ED00 | -3.52 mA | 1.185 mA | Undershoot range |
| 32767 ¹ | 7FFF | | < 1.185 mA | |
| -32768 | 8000 | < -3.52 mA | | |

¹ The wire break value of 32767 (16#7FFF) is always returned regardless of the state of the wire break alarm.

A.10.7 Measurement ranges of the analog outputs for voltage and current (SB and SM)

Table A-165 Analog output representation for voltage (SB and SM)

| System | | Voltage Output Range | |
|---------|-------------|----------------------|------------------|
| Decimal | Hexadecimal | ± 10 V | |
| 32767 | 7FFF | See note 1 | Overflow |
| 32512 | 7F00 | See note 1 | |
| 32511 | 7EFF | 11.76 V | Overshoot range |
| 27649 | 6C01 | | |
| 27648 | 6C00 | 10 V | Rated range |
| 20736 | 5100 | 7.5 V | |
| 1 | 1 | 361.7 μ V | |
| 0 | 0 | 0 V | |
| -1 | FFFF | -361.7 μ V | |
| -20736 | AF00 | -7.5 V | |
| -27648 | 9400 | -10 V | |
| -27649 | 93FF | | |
| -32512 | 8100 | -11.76 V | Undershoot range |
| -32513 | 80FF | See note 1 | |
| -32768 | 8000 | See note 1 | Underflow |
| | | | |

¹ In an overflow or underflow condition, analog outputs will take on the substitute value of the STOP mode.

Tabellerne viser den analoge værdi på IW/QW i % af hele måleområdet.

SM 1234-kortet har en opløsning på 12/13 bit og et måleområde fra ± 10 V, 0-20 mA og 4-20 mA.

Dette medfører at ved et signal på 0-20 mA:

0 mA = 0 % er værdien på IW = 0

20 mA = 100 % er værdien på IW = 27648

23,52 mA = 117,589 % er værdien på IW = 32511

Hvis strømmen er større end 23,518 mA, så går kortet i mætning og melder fejl.

Analog universal skaleringsblok

Parameterbar funktionsblok

Global libraries

Vi vil her oprette en FB-blok, der fungerer som en skaleringsblok, hvor vi kan skalere frit mellem forskellige maksimums- og minimumsværdier på in- og outputsignaler.

Vi vil her lave en skaleringsblok, som fungerer direkte på den analoge indgang. Der skal vi bruge værdierne, som indlæses mellem 0 og 27648. Vi kan altså bruge analoge input i områderne 0-10 V, 2-10 V eller 0-20 mA, 4-20 mA som min. og maks. Vi kan således skrive 0/5530 som In min. og 27648 som In maks.

Grundformel for skalering

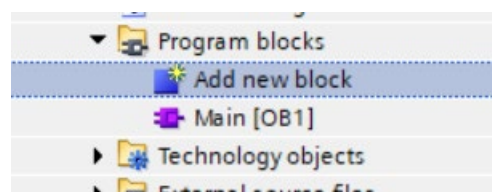
$$OUTværdi = \frac{områdeOUT}{områdeIN} \cdot INværdi$$

Skaleringsformel ved flydende nulpunkt

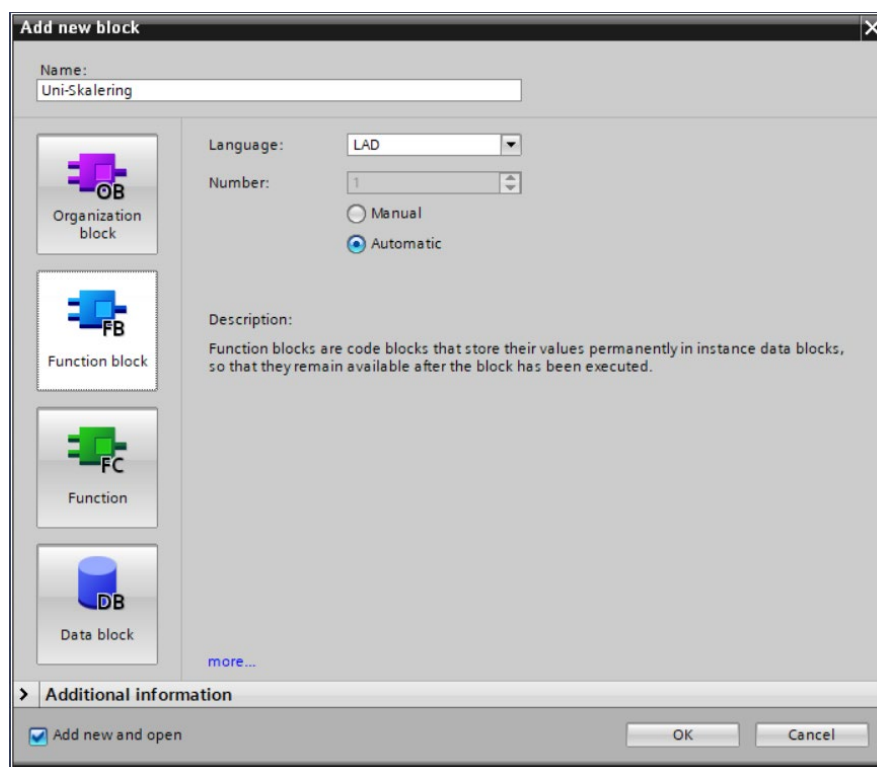
$$OUTsignal = \left(\frac{OUTmax - OUTmin}{INmax - INmin} \cdot (INsignal - INmin) \right) + OUTmin$$

Vi vil her lave et eksempel på en parameterbar funktionsblok, der kan afprøves og eksporteres, så den kan anvendes på en anden PC i et andet PLC-projekt. Denne eksport forklares sidst i afsnittet

Vi starter med at oprette en ny FB-blok.

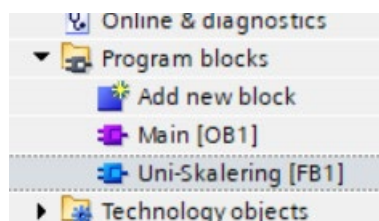

























Vi kalder den her Uni-Skalering, da det bliver vores universal skaleringsblok.



Vi opretter følgende In- og Output, samt et par mellemregninger i "Static"-området. Det analoge IN er her valgt som en INT, da vi altid får dette fra et input-word (IW), som en 16 bitsværdi.

Så åbnes den nye Uni-Skaleringsblok [FB1].



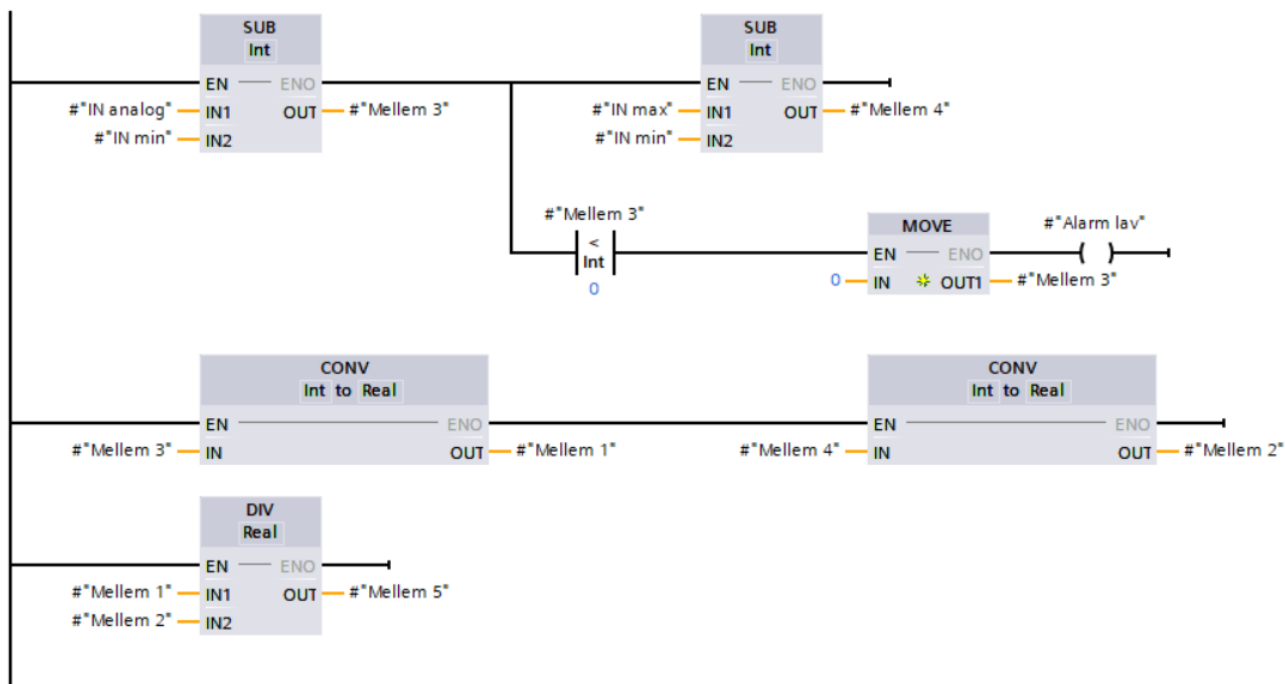
| Uni-Skalering | | | | | |
|---------------|---|-----------|-----------|---------------|------------|
| | | Name | Data type | Default value | Retain |
| 1 |  | ▼ Input | | | |
| 2 |  | IN min | Int | 0 | Non-retain |
| 3 |  | IN analog | Int | 0 | Non-retain |
| 4 |  | IN max | Int | 0 | Non-retain |
| 5 |  | OUT min | Real | 0.0 | Non-retain |
| 6 |  | OUT max | Real | 0.0 | Non-retain |
| 7 |  | <Add new> | | | |
| 8 |  | ▼ Output | | | |
| 9 |  | OUT real | Real | 0.0 | Non-retain |
| 10 |  | OUT int | Int | 0 | Non-retain |
| 11 |  | Alarm lav | Bool | false | Non-retain |
| 12 |  | Alarm høj | Bool | false | Non-retain |
| 13 |  | <Add new> | | | |
| 14 |  | ▼ InOut | | | |
| 15 |  | <Add new> | | | |
| 16 |  | ▼ Static | | | |
| 17 |  | Mellem 1 | Real | 0.0 | Non-retain |
| 18 |  | Mellem 2 | Real | 0.0 | Non-retain |
| 19 |  | Mellem 3 | Int | 0 | Non-retain |
| 20 |  | Mellem 4 | Int | 0 | Non-retain |
| 21 |  | Mellem 5 | Real | 0.0 | Non-retain |
| 22 |  | Mellem 6 | Real | 0.0 | Non-retain |
| 23 |  | Mellem 7 | Real | 0.0 | Non-retain |

Selve skaleringsdelen er lavet med standard regneblokke, og yderligere er der lavet alarmudgange, som sætter en Bool høj ved henholdsvis lavt- og højt signal.

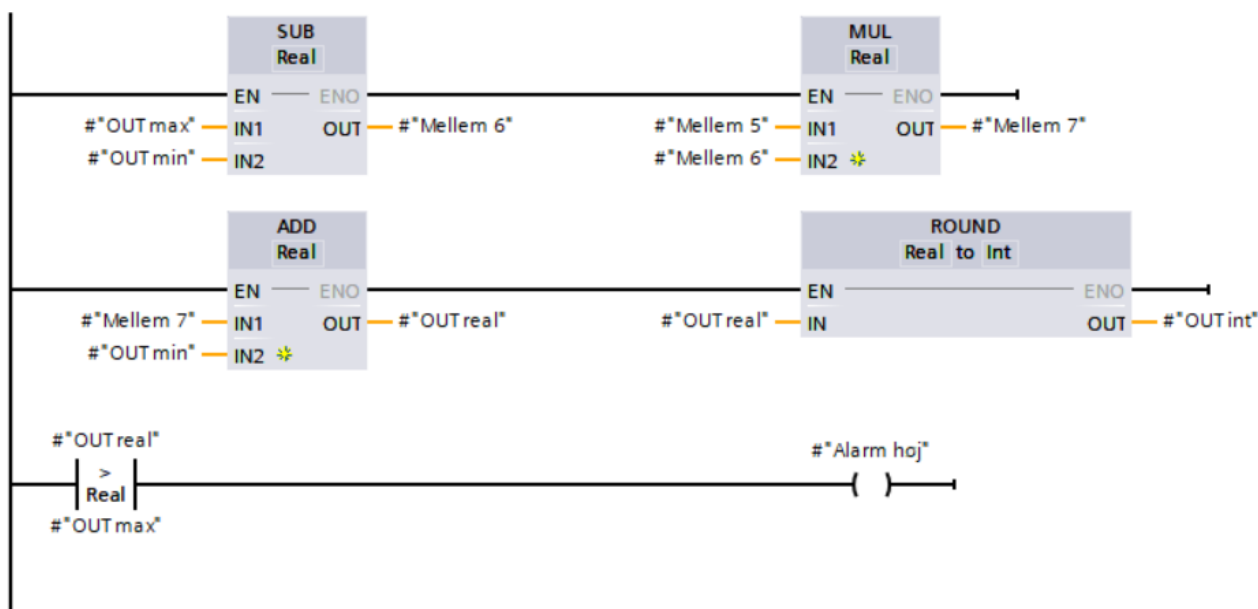
Så leveres den analogt skalerede værdi som henholdsvis en real- og en heltalsværdi.

Network 1: Laver Analog IN om til en værdi mellem 0.0 og 1.0

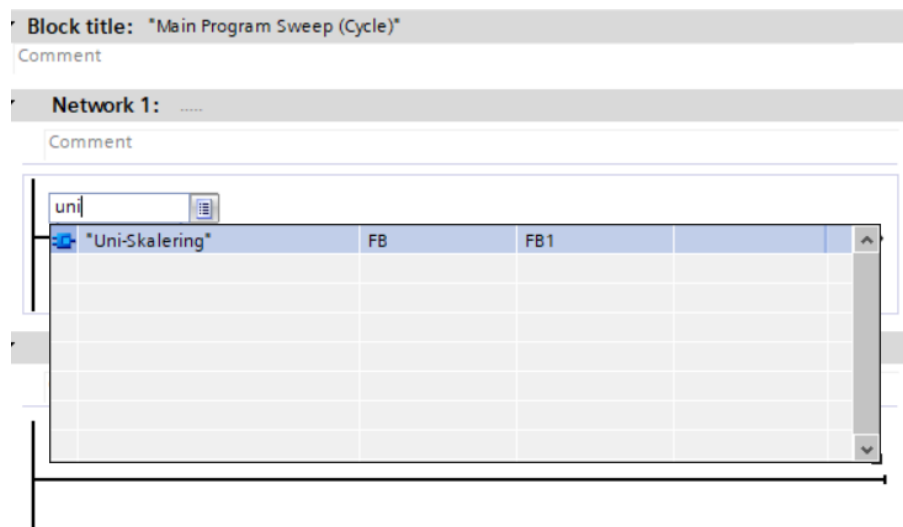
Her laves Int om til Real, således der kan laves en variabel mellem 0.0 og 1.0, samt der sikres mod værdi mindre end 0, samt alarm for lav værdi.

**Network 2:** Her ganges værdien mellem 0.0 og 1.0 op med OUT variabelen.

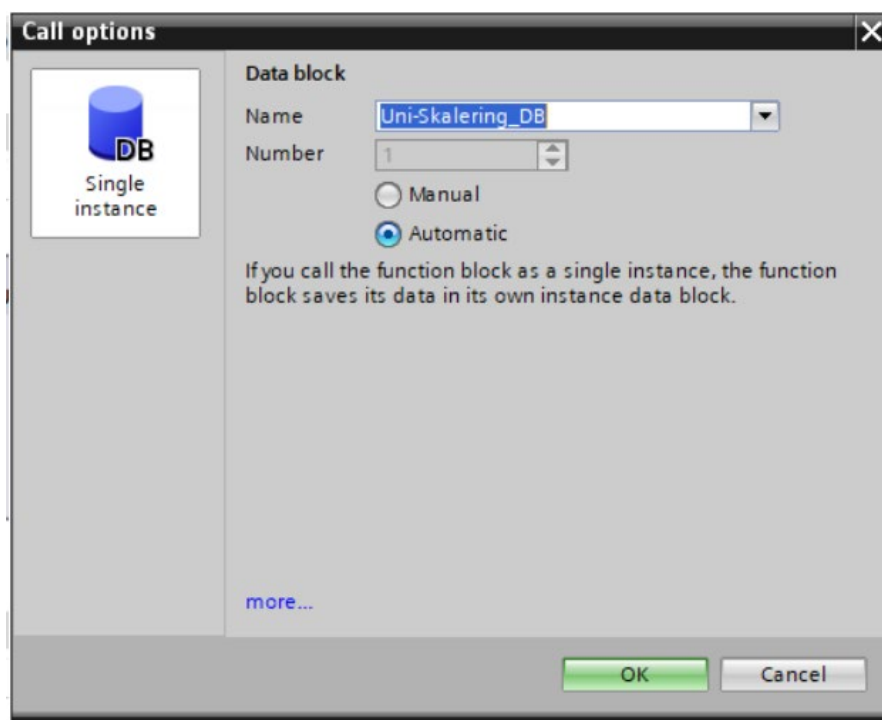
Yderlig lægges OUT min til, samt der laves en alarm for høj værdi.



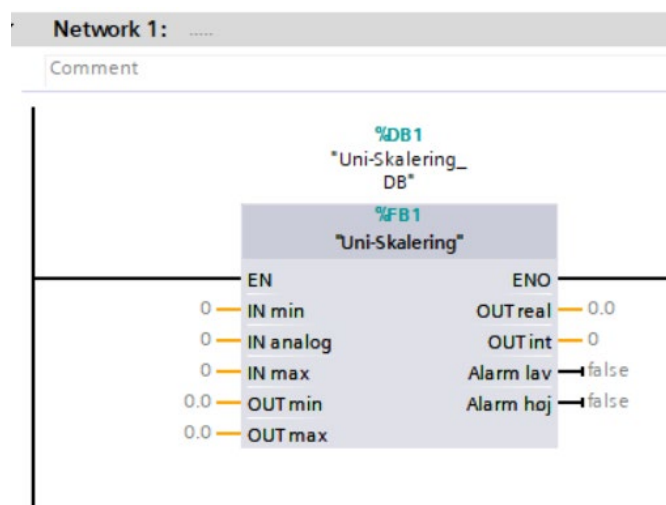
Så kan vi hente vores nyoprettede FB-program i OB1 Main program.



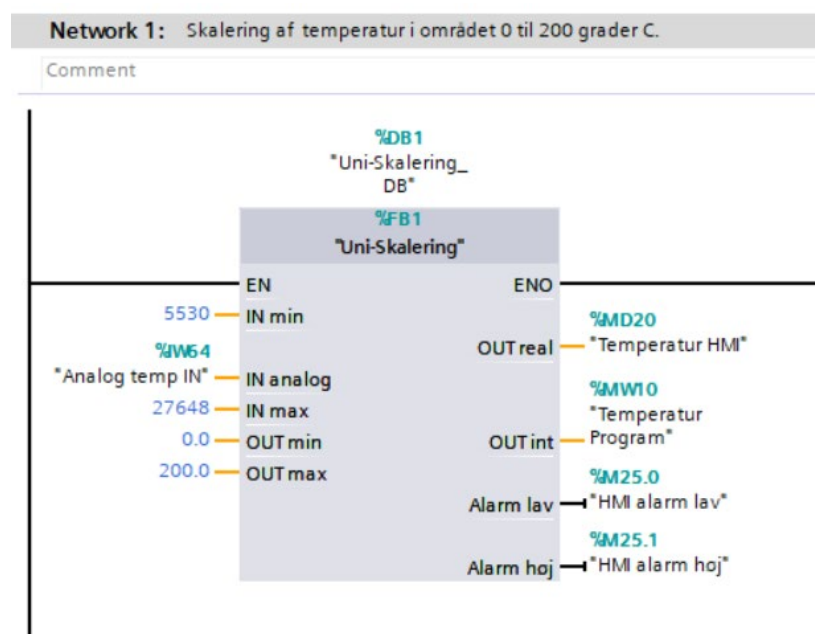
Når vi åbner en FB, skal der tilknyttes en DB-blok, hvor vores data vil ligge for FB-blokkens beregninger, således vi kan bruge FB-blokken igen og igen.



Når vi har hentet vores FB-blok over i Main-programmet, vil den fremstå således med ind- og udgange, der skal defineres til de respektive værdier.

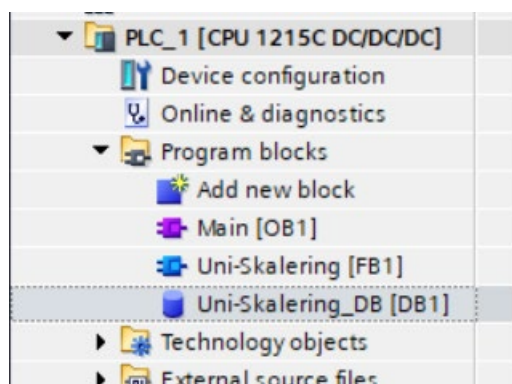


Når FB-blokken er sat korrekt op, burde den se nogenlunde således ud og nu være klar til brug.



Her er den vist som en skaling af et 4-20 mA-signal, der skaleres om til en 0-200 graders værdi.

Datablokken Uni-Skalering_DB [DB1] vil være tilknyttet den skalering, vi lige har lavet, hvor temperaturen skaleres mellem 0 og 200 grader C.



Hvis vi åbner Uni-Skalering DB [DB1], vil vi se, hvorledes denne er opbygget.

PLC bog ▸ PLC_1 [CPU 1215C DC/DC/DC] ▸ Program blocks ▸ Uni-Skalering_DB [DB1]

Keep actual values Snapshot Copy snapshots to start va

Uni-Skalering_DB

| | Name | Data type | Start value | Retain | Accessible f... | Writa... | V |
|----|-----------|-----------|-------------|--------|-------------------------------------|-------------------------------------|---|
| 1 | ▼ Input | | | | | | |
| 2 | IN min | Int | 0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 3 | IN analog | Int | 0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 4 | In max | Int | 0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 5 | OUT min | Real | 0.0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 6 | OUT max | Real | 0.0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 7 | ▼ Output | | | | | | |
| 8 | OUT real | Real | 0.0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 9 | OUT int | Int | 0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 10 | Alarm lav | Bool | false | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 11 | Alarm høj | Bool | false | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 12 | InOut | | | | | | |
| 13 | ▼ Static | | | | | | |
| 14 | Mellem 1 | Real | 0.0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 15 | Mellem 2 | Real | 0.0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 16 | Mellem 3 | Int | 0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 17 | Mellem 4 | Int | 0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 18 | Mellem 5 | Real | 0.0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 19 | Mellem 6 | Real | 0.0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 20 | Mellem 7 | Real | 0.0 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |

Eksport og import af FB

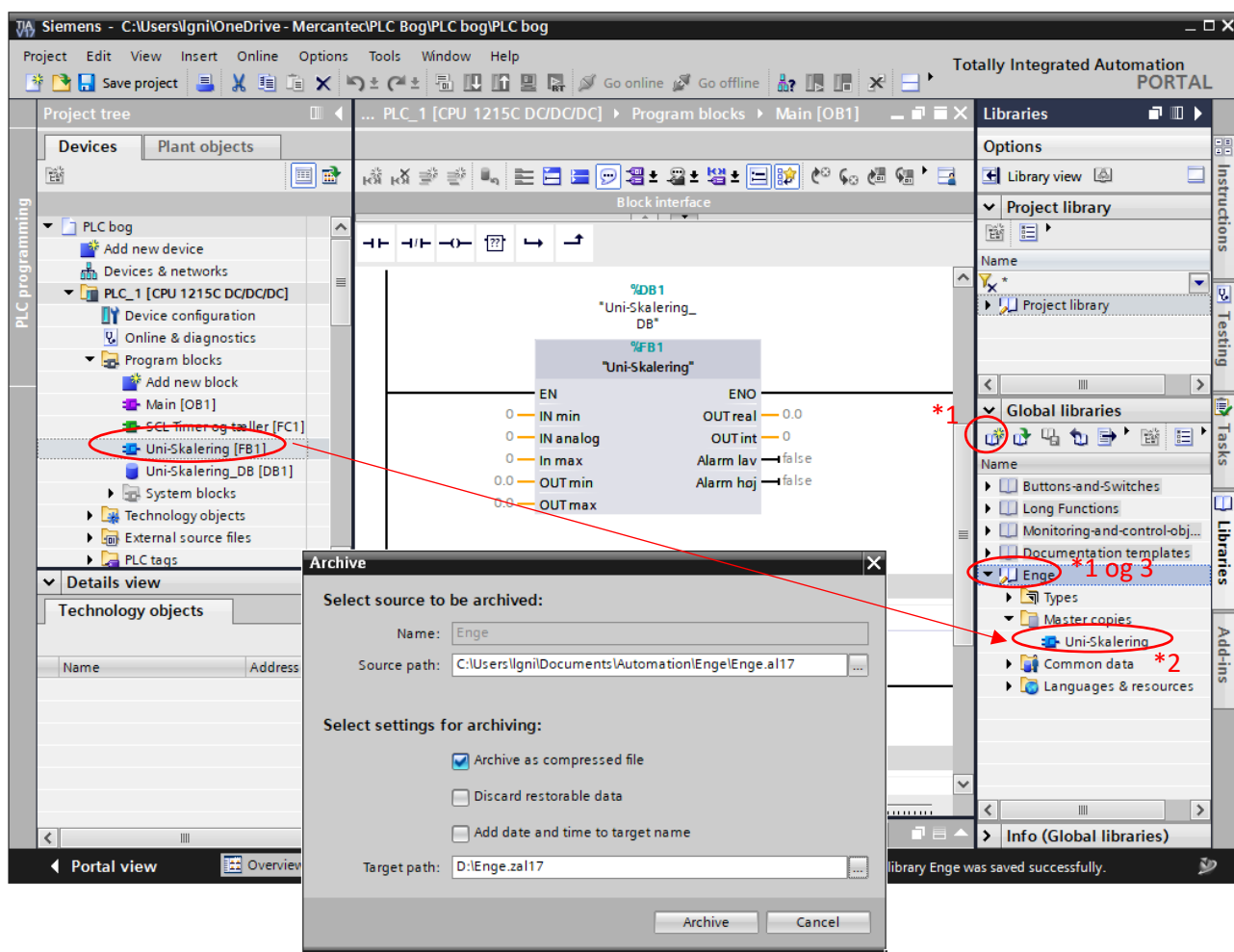
Når den parameterbar funktionblok er helt færdig og afprøvet, kan den exporteres, så den kan anvendes på en anden PC i et andet PLC projekt.

Dette gøres ved at oprette et Global Librari, her kaldt Engne *1.

Der efter trækkes FB-bloken fra program træet over i Engne mappens Master copies *2.

Når dette er gjort kan FB-blokken bruges i andre PLC projekter på denne PC.

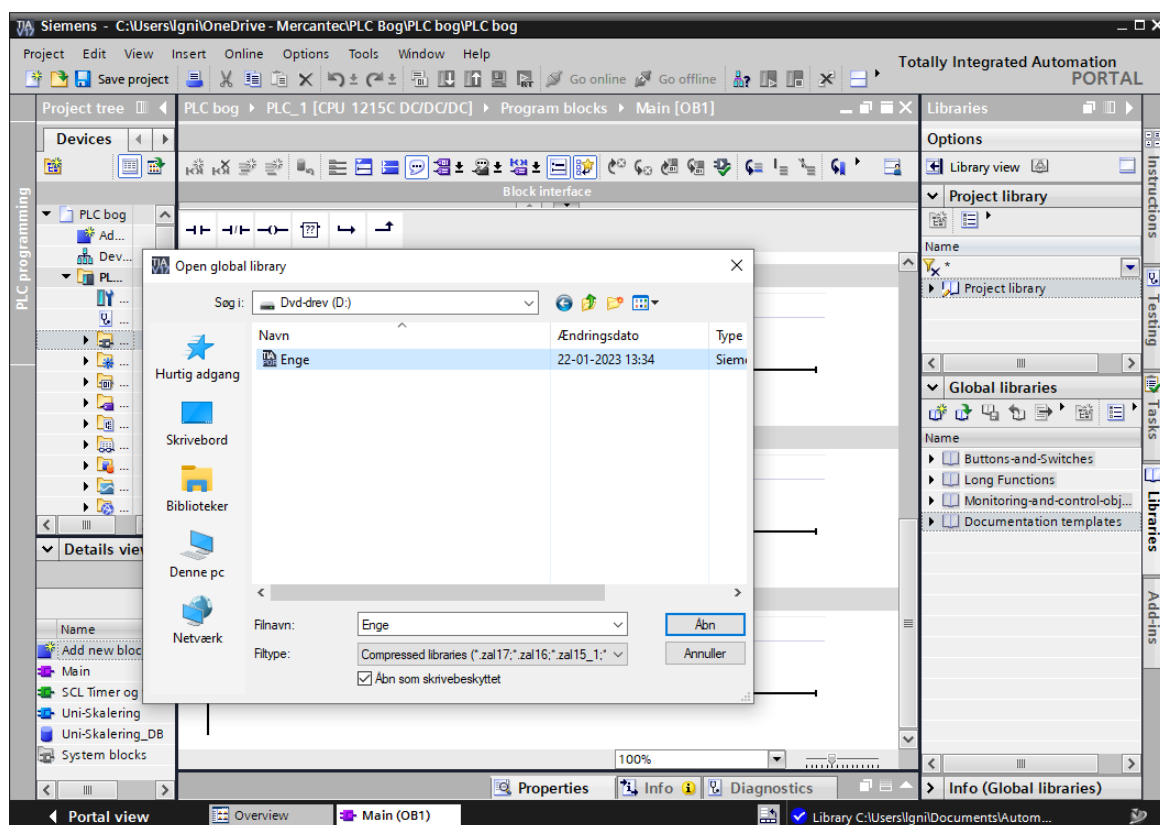
Hvis man ønsker også at bruge FB-blokken på en anden PC højre klikkes der på Engne mappen, og der vælges "Archive library.." *3. Der efter vælges hvor man vil gemme den, her er det på en USB-pen. Tryk "Archive".



Når man så vil bruge den igen på en anden PC, højre klikkes der på det grå område af Global libraries og vælges "Open library..".

Der efter skal man finde filen på USB-pennne, her skal filtypen være Compressed libraries (*.Zal17...osv).

Tryk åben og vælg hvor man vil gemme den og tryk ok.

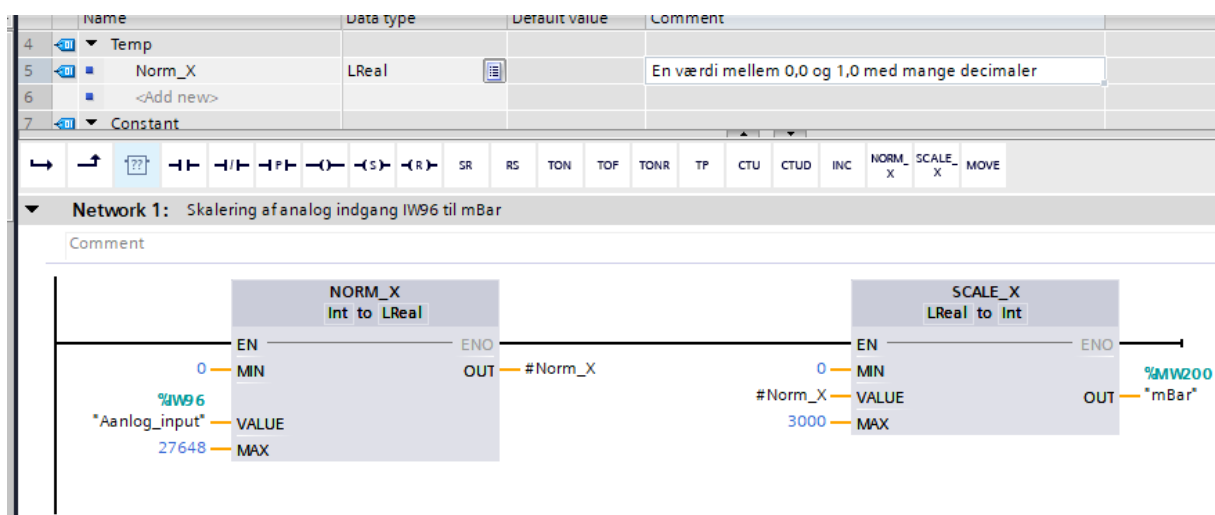


Analog skalering Norm_X og Scale_X Siemens S7-1200

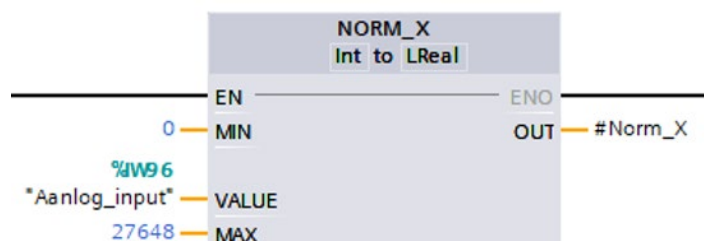
Ved analog signalbehandling på en PLC er der ofte krav om, at de forskellige enheder passer sammen, samt at de er skaleret til ingeniørenheder.

Hvis vi f.eks. har en tryktransmitter, som måler i området fra 0 til 3000 mBar, ønsker vi, at værdien i PLC'en skal angives i området fra 0 til 3000 i stedet for fra 0 - 27648. Dette kræver en skalering. En sådan skalering vil lette programmering og fejlfinding betydeligt, fordi talstørrelser fremtræder mere gennemskuelige, og herudover slipper man også for stadige omregninger.

Man kan anvende de færdige skaleringsblokke, der findes i TIA-Portalen. NORM_X og SCALE_X



NORM_X



Man kan bruge instruktionen "Normaliser" til at normalisere værdien af tagget ved VALUE-inputtet ved at knytte det til en lineær skala.

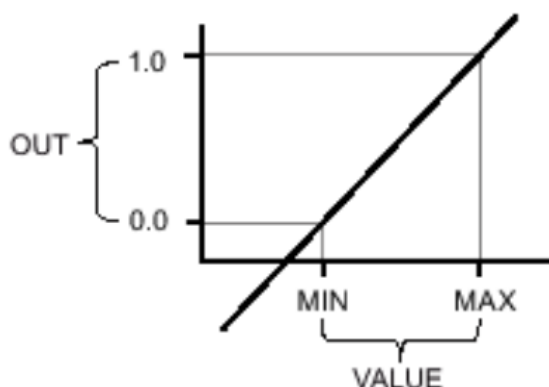
Man kan bruge MIN- og MAX-parametrene til at definere grænserne for det værdiområde, der anvendes på skalaen.

Resultatet på OUT-udgangen beregnes og lagres som Real-tal, der afhænger af den værdi, der skal normaliseres inden for dette værdiområde.

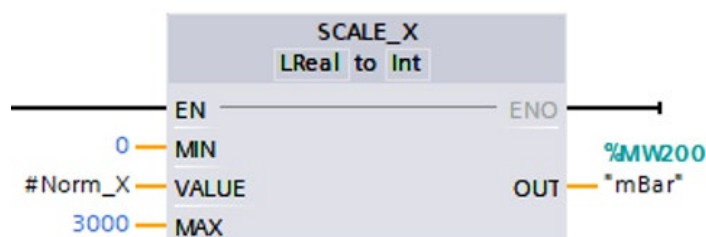
Hvis værdien, der skal normaliseres, er lig med værdien ved MIN-indgangen, har OUT-udgangen værdien "0.0". Hvis værdien, der skal normaliseres, er lig med værdien ved input MAX, returnerer output OUT værdien "1.0".

Følgende figur og formel viser, hvordan værdier normaliseres:

$$OUT = (VALUE - MIN) / (MAX - MIN)$$



SCALE_X



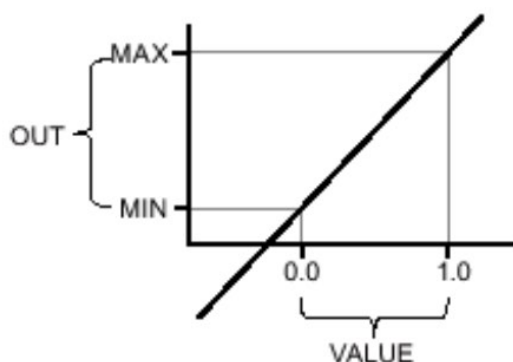
Man kan bruge "Scale"-instruktionen til at skalere værdien på VALUE-inputtet ved at tilknytte den til et specificeret værdiområde.

Når "Scale"-instruktionen udføres, skaleres Real-tallet på VALUE-inputtet til det værdiområde, der blev defineret af MIN- og MAX-parametrene.

Resultatet af skaleringen er et heltal "INT", som er lagret i OUT-udgangen.

Følgende figur og formel viser, hvordan værdier skaleres:

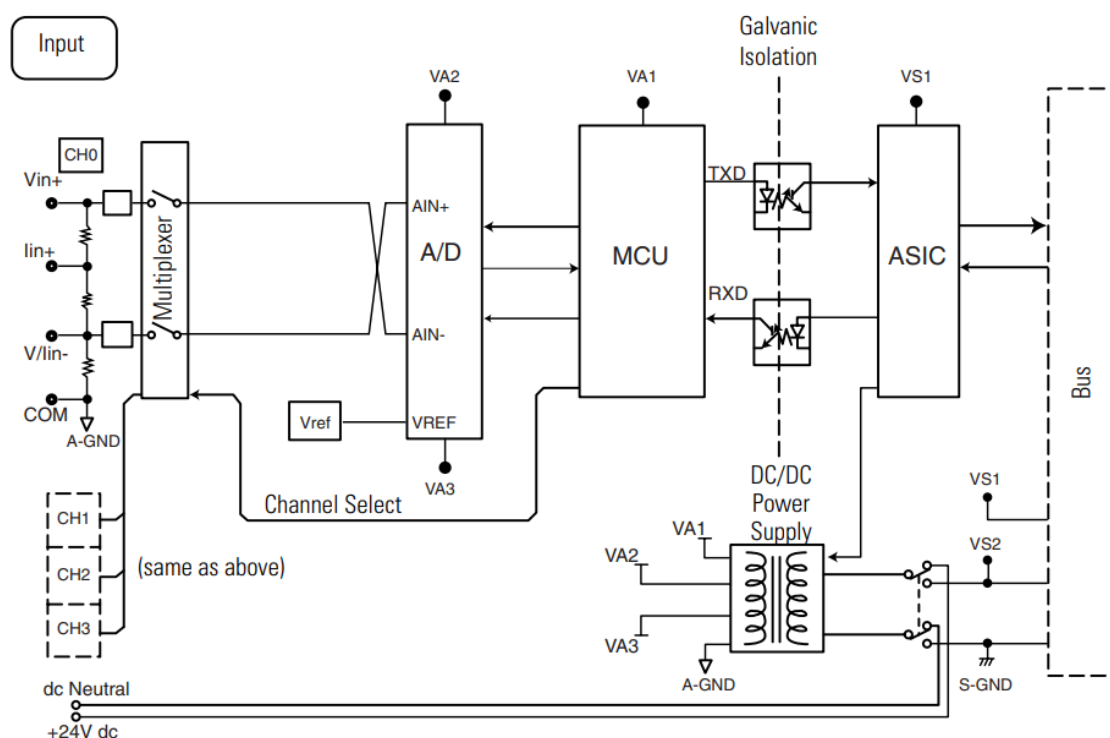
$$\text{OUT} = [\text{VALUE} * (\text{MAX} - \text{MIN})] + \text{MIN}.$$



Analog signaler Allen-Bradley

Analog-inputkort 1796-IF4

Opbygning / blokdiagram



Kortet har 4 analoge input-kanaler (CH0 -3). Man kan i PLC'ens programmeringssoftware vælge, om man vil bruge de enkelte inputkanaler som strøm eller spændingsinput.

Indgangskortets indgangskredsløb består af fire analoge indgange, der via en multiplexer forbindes til en enkelt analog-til-digital-konverter (A/D).

A/D-konverter læser det valgte indgangssignal og konverterer det til en digital værdi, som via den digitale bus overføres til CPU'en.

Multiplexeren skifter konstant mellem de fire indgangskanaler og kortets ene A/D-konverter. Dette betyder, at opdateringstiden på en analog indgang stiger hver gang en indgangskanal aktiveres/enables.

De 4 analoge input er galvanisk adskilt fra CPU-bussen ved hjælp af optokoblerer.

Opdateringstid

Table 3.5 Channel Update Time

| Filter Frequency | Channel Update Time |
|------------------|---------------------|
| 50 Hz | 22 ms |
| 60 Hz | 19 ms |
| 250 Hz | 6 ms |
| 500 Hz | 4 ms |

Table 3.6 Channel Switching and Reconfiguration Times

| | Description | Duration | | | |
|---|--|----------|-------|--------|--------|
| | | 50 Hz | 60 Hz | 250 Hz | 500 Hz |
| Channel Switching Time | The time it takes the module to switch from one channel to another. | 46 ms | 39 ms | 14 ms | 10 ms |
| Channel-to-Channel Reconfiguration Time | The time it takes the module to change its configuration settings for a difference in configuration between one channel and another. | 116 ms | 96 ms | 20 ms | 8 ms |

I følgende eksempel beregnes 1769-IF4-inputkortets opdateringstid:

Eks. 1.

To kanaler aktiveret med identiske konfigurationer og et 500 Hz-filter.

$$\begin{aligned} \text{Kortets opdateringstid} = & [\text{Ch 0 Update Time} + \text{Ch 0 Switching Time}] \\ & + [\text{Ch 1 Update Time} + \text{Ch 1 Switching Time}] \end{aligned}$$

$$\text{Kortets opdateringstid} = [4 \text{ ms} + 10 \text{ ms}] + [4 \text{ ms} + 10 \text{ ms}] = \underline{28\text{ms}}$$

Eks. 2.

Tre kanaler aktiveret med forskellige konfigurationer:

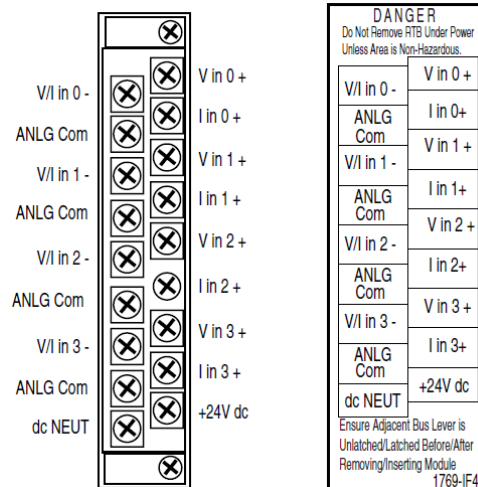
- Kanal 0: $\pm 10\text{V}$ jævnstrøm med 60 Hz-filter
- Kanal 1: $\pm 10\text{V}$ jævnstrøm med 500 Hz-filter
- Kanal 2: 4 til 20 mA med 250 Hz-filter

$$\begin{aligned} \text{Kortets opdateringstid} = & [\text{Ch 0 Reconfiguration Time} + \text{Ch 0 Update Time} + \text{Ch 0 Switching Time}] \\ & + [\text{Ch 1 Reconfiguration Time} + \text{Ch 1 Update Time} + \text{Ch 1 Switching Time}] \\ & + [\text{Ch 2 Reconfiguration Time} + \text{Ch 2 Scan Time} + \text{Ch 2 Switching Time}] \end{aligned}$$

$$\text{Kortets opdateringstid} = [96 \text{ ms} + 19 \text{ ms} + 39 \text{ ms}] + [8 \text{ ms} + 4 \text{ ms} + 10 \text{ ms}] + [20 \text{ ms} + 6 \text{ ms} + 14 \text{ ms}] = \underline{216\text{ms}}$$

Tilslutning

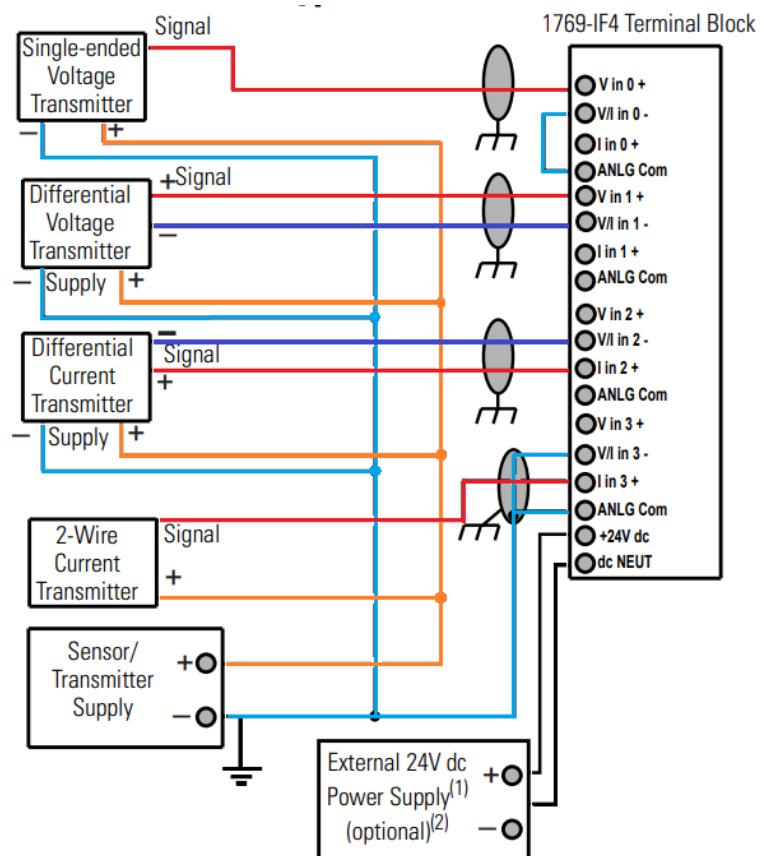
Det analoge inputkort 1769-IF4 har følgende tilslutningsterminaler:



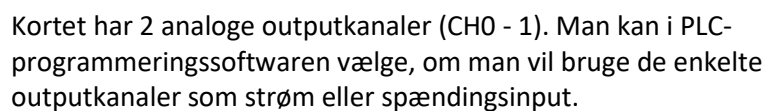
Der er flere måder, man kan tilslutte en analog transmitter/signalgiver til kortet. Hvilken måde, man vælger, afhænger af, om man bruger et strøm- eller spændingssignal.

Det afhænger også af, om transmitteren er en 2- eller 4-trådet type.

Herunder ses et forbindelsesdiagram, hvor der er brugt alle muligheder. OBS: Alle "ANLG Com" er internt forbundene.



Opbygning / blokdiagram

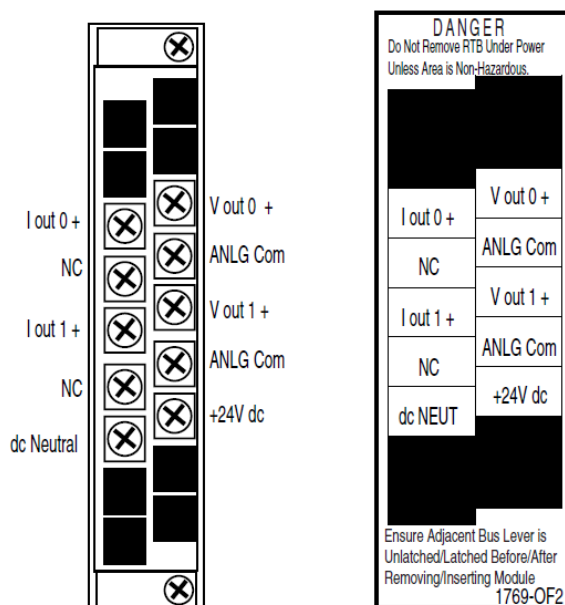


Udgangskortet bruger en digital-til-analog (D/A)-konverter til at læse det digitale datasignal fra CPU'en og konvertere det til et analogt udgangssignal.

De 2 analoge output er galvanisk adskilt fra CPU-bussen ved hjælp af optokoblerer.

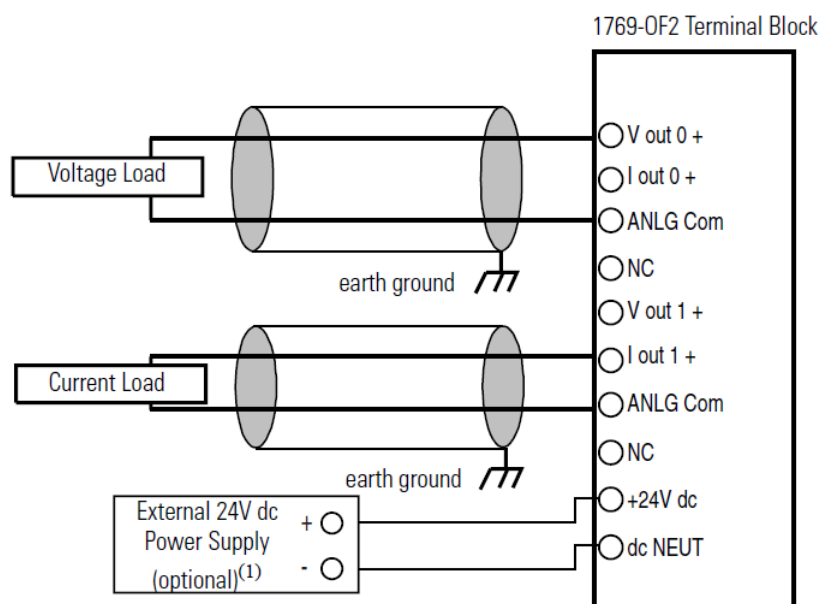
Tilslutning

Det analog outputkort 1769-OF2 har følgende tilslutningsterminaler:



Der er flere måder, man kan tilslutte et analogt handleorgan eller aktuator til kortet. Hvilken måde, man vælger, afhænger af, om man bruger strøm eller spændingssignal.

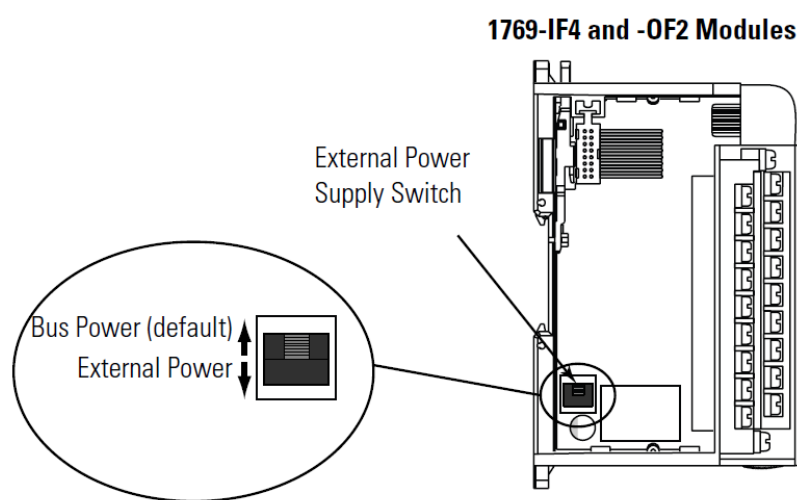
Herunder ses et forbindelsesdiagram, hvor der er brugt alle muligheder. OBS: Alle "ANLG Com" er internt forbundene.



Fælles for 1769-IF4 og -OF2 Spændingsforsyning

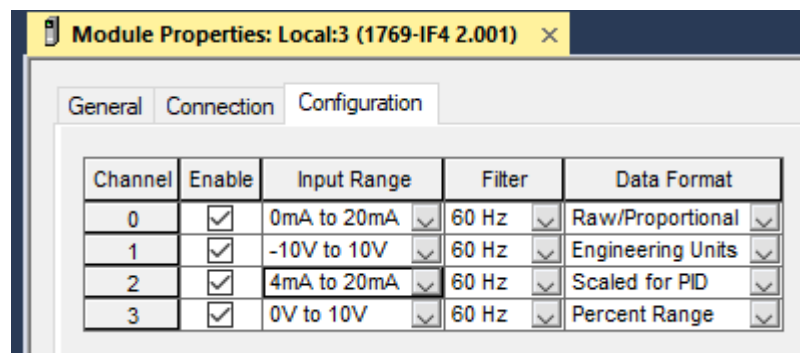
På en del Allen-Bradley analoge in- og outputkort skal man beslutte, om man vil bruge PLC'ens spændingsforsyning til at forsyne de analoge signaler, dette kaldes "Bus Power", eller om man vil bruge en ekstern, separat spændingsforsyning.

På siden af de analoge kort sidder der en omskifter, der bruges til at vælge dette.



Valg af PLC dataformat / skalering

Dataformatet bestemmes af, hvordan analogkortet skal fortolke den digitale data til og fra CPU'en:



Her under er eksempler på data formater og hvordan de kommer til udstyk:

- Raw/Proportional Data;
 - Det fulde område for ± 10.5 VDC er -32767 til +32767 (15bit+sine).
 - eks. ± 10.0 VDC er -31207 til +31207 (14bit+sine).
 - eks. 4-20 mA er 6241 til 31207
- Engineering Units:
 - eks. 4-20 mA er 4.000 til 20.000
 - eks. ± 10.0 VDC er -10.00 til +10.000
- Scaled-for-PID:
 - eks. ± 10.0 VDC er -10 V = 0, 0 V = 8192, +10V = 16383
 - eks. 0-20 mA, 4-20 mA og 0-10 VDC er 0 til 16383
- Percent Full Range:
 - eks. 0-20 mA, 4-20 mA og 0-10 VDC er 0 til 10.000
 - eks. ± 10.0 VDC. Procentskalering kan ikke bruges.

På de næste to sider ses to uddybende tabeller over dataformater. Tabellen for det analoge inputkort 1769-IF4 er komplet, hvorimod der for det analoge outputkort 1769-OF2 kun er vist de mest brugte.

Table 3.7 Valid Input Data

| 1769-IF4 Input Range | Input Value | Example Data | Input Range Condition | Raw/Proportional Data | Engineering Unit | Scaled-for-PID | Percent Full Range |
|----------------------|---------------------|--------------|-----------------------|-----------------------|------------------|----------------|--------------------|
| | | | | Decimal Range | Decimal Range | Decimal Range | Decimal Range |
| -10V to +10V dc | Over 10.5V dc | +11.0V dc | Over-range | 32767 (max.) | 10500 (max.) | 16793 (max.) | N/A |
| | +10.5V dc | + 10.5V dc | Over-range | 32767 (max.) | 10500 (max.) | 16793 (max.) | N/A |
| | -10V to +10V dc | +10.0V dc | Normal | 31206 | 10000 | 16383 | N/A |
| | | 0.0V dc | Normal | 0 | 0 | 8192 | N/A |
| | | -10.0V dc | Normal | -31206 | -10000 | 0 | N/A |
| | -10.5Vdc | -10.5V dc | Under-range | -32767 (min.) | -10500 (min.) | -410 (min.) | N/A |
| 0V to 5V dc | Under -10.5V dc | -11.0V dc | Under-range | -32767 (min.) | -10500 (min.) | -410 (min.) | N/A |
| | Over 5.25V dc | 5.5V dc | Over-range | 32767 (max.) | 5250 (max.) | 17202 (max.) | 10500 (max.) |
| | 5.25V dc | 5.25V dc | Over-range | 32767 (max.) | 5250 (max.) | 17202 (max.) | 10500 (max.) |
| | 0.0V dc to 5.0V dc | 5.0V dc | Normal | 31206 | 5000 | 16383 | 10000 |
| | | 0.0V dc | Normal | 0 | 0 | 0 | 0 |
| | -0.5V dc | -0.5V dc | Under-range | -3121 (min.) | -500 (min.) | -1638 (min.) | -1000 (min.) |
| 0V to 10V dc | Under -0.5V dc | -1.0V dc | Under-range | -3121 (min.) | -500 (min.) | -1638 (min.) | -1000 (min.) |
| | Over 10.5V dc | 11.0V dc | Over-range | 32767 (max.) | 10500 (max.) | 17202 (max.) | 10500 (max.) |
| | +10.5V dc | 10.5V dc | Over-range | 32767 (max.) | 10500 (max.) | 17202 (max.) | 10500 (max.) |
| | 0.0V dc to 10.0V dc | 10.0V dc | Normal | 31206 | 10000 | 16383 | 10000 |
| | | 0.0V dc | Normal | 0 | 0 | 0 | 0 |
| | -0.5V dc | -0.5V dc | Under-range | -1560 (min.) | -500 (min.) | -819 (min.) | -500 (min.) |
| 4 mA to 20 mA | Under -5.0V dc | -1.0V dc | Under-range | -1560 (min.) | -500 (min.) | -819 (min.) | -500 (min.) |
| | Over 21.0 mA | 22.0 mA | Over-range | 32767 (max.) | 21000 (max.) | 17407 (max.) | 10625 (max.) |
| | 21.0 mA | 21.0 mA | Over-range | 32767 (max.) | 21000 (max.) | 17407 (max.) | 10625 (max.) |
| | 4.0 mA to 20.0 mA | 20.0 mA | Normal | 31206 | 20000 | 16383 | 10000 |
| | | 4.0 mA | Normal | 6241 | 4000 | 0 | 0 |
| | 3.2 mA | 3.2 mA | Under-range | 4993 (min.) | 3200 (min.) | -819 (min.) | -500 (min.) |
| 1.0V to 5V dc | Under 3.2 mA | 0.0 mA | Under-range | 4993 (min.) | 3200 (min.) | -819 (min.) | -500 (min.) |
| | Over 5.25V dc | 5.5V dc | Over-range | 32767 (max.) | 5250 | 17407 | 10625 |
| | +5.25V dc | 5.25V dc | Over-range | 32767 (max.) | 5250 | 17407 | 10625 |
| | 1.0V to 5.0V dc | 5.0V dc | Normal | 31206 | 5000 | 16383 | 10000 |
| | | 1.0V dc | Normal | 6243 | 1000 | 1 | 1 |
| | 0.5V dc | 0.5V dc | Under-range | 3121 (min.) | 500 | -2048 | -1250 |
| 0 mA to 20 mA | Under 0.5V dc | 0.0V dc | Under-range | 3121 (min.) | 500 | -2048 | -1250 |
| | Over 21.0 mA | 22.0 mA | Over-range | 32767 | 21000 | 17202 | 10500 |
| | 21.0 mA | 21.0 mA | Over-range | 32767 | 21000 | 17202 | 10500 |
| | 0.0 mA to 20.0 mA | 20.0 mA | Normal | 31206 | 20000 | 16383 | 10000 |
| | | 0.0 mA | Normal | 0 | 0 | 0 | 0 |
| | Under 0.0 mA | 0.0 mA | Under-range | 0 | 0 | 0 | 0 |

Table 4.5 1769-OF2 Valid Output Data Table

| OF2 Output Range | Input Value | Example Data | | Output Range State | Raw/Proportional Data | | Engineering Unit | | Scaled-for-PID | | Percent Full Range | |
|------------------|---------------------|--------------------|------------|--------------------|-----------------------|---------------------|--------------------|---------------------|--------------------|---------------------|--------------------|---------------------|
| | | | | | Decimal Range | | Decimal Range | | Decimal Range | | Decimal Range | |
| | | Controller Ordered | OF2 Output | | Controller Ordered | OF2 Output and Echo | Controller Ordered | OF2 Output and Echo | Controller Ordered | OF2 Output and Echo | Controller Ordered | OF2 Output and Echo |
| 0V to 10V dc | Over 10.5V dc | 11.0V dc | +10.5V dc | Over | N/A | N/A | 11000 | 10500 | 18021 | 17202 | 11000 | 10500 |
| | +10.5V dc | +10.5V dc | +10.5V dc | Over | 32767 | 32767 | 10500 | 10500 | 17202 | 17202 | 10500 | 10500 |
| | 0.0V dc to 10.0V dc | +10.0V dc | +10.0V dc | Normal | 31207 | 31207 | 10000 | 10000 | 16383 | 16383 | 10000 | 10000 |
| | | 0.0V dc | 0.0V dc | Normal | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | -0.5V dc | -0.5V dc | -0.5V dc | Under | -1560 | -1560 | -500 | -500 | -819 | -819 | -500 | -500 |
| | Under -5.0V dc | -1.0V dc | -0.5V dc | Under | -3121 | -1560 | -1000 | -500 | -1638 | -819 | -1000 | -500 |
| 4 mA to 20 mA | Over 21.0 mA | +22.0 mA | +21.0 mA | Over | N/A | N/A | 22000 | 21000 | 18431 | 17407 | 11250 | 10625 |
| | 21.0 mA | +21.0 mA | +21.0 mA | Over | 32767 | 32767 | 21000 | 21000 | 17407 | 17407 | 10625 | 10625 |
| | 4.0 mA to 20.0 mA | +20.0 mA | +20.0 mA | Normal | 31207 | 31207 | 20000 | 20000 | 16383 | 16383 | 10000 | 10000 |
| | | +4.0 mA | +4.0 mA | Normal | 6241 | 6241 | 4000 | 4000 | 0 | 0 | 0 | 0 |
| | 3.2 mA | +3.2 mA | +3.2 mA | Under | 4993 | 4993 | 3200 | 3200 | -819 | -819 | -500 | -500 |
| | Under 3.2 mA | 0.0 mA | +3.2 mA | Under | 0 | 4993 | 0 | 3200 | -4096 | -819 | -2500 | -500 |
| 0 mA to 20 mA | Over 21.0 mA | +22.0 mA | +21.0 mA | Over | N/A | N/A | 22000 | 21000 | 18201 | 17202 | 11000 | 10500 |
| | 21.0 mA | 21.0 mA | +21.0 mA | Over | 32767 | 32767 | 21000 | 21000 | 17202 | 17202 | 10500 | 10500 |
| | 0.0 mA to 20.0 mA | 20.0 mA | +20.0 mA | Normal | 31207 | 31207 | 20000 | 20000 | 16383 | 16383 | 10000 | 10000 |
| | | 0.0 mA | 0.0 mA | Normal | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Under 0.0 mA | -1.0 mA | 0.0 mA | Under | -1560 | 0 | 0 | -1000 | -819 | 0 | -500 | 0 |

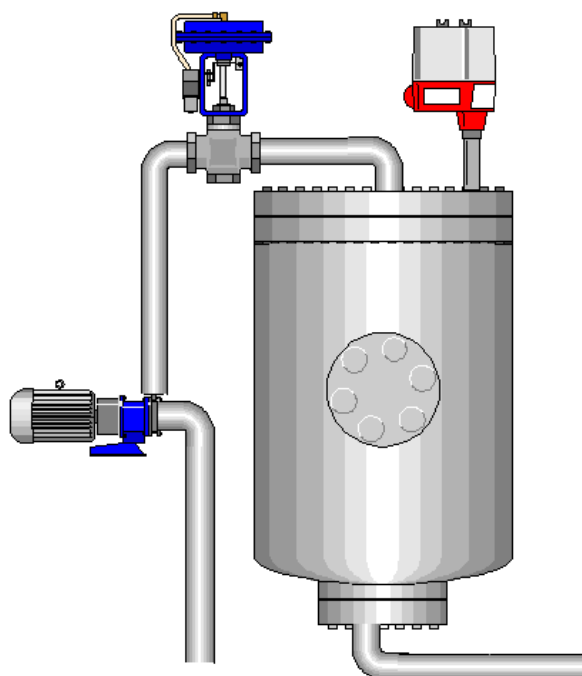
PID-regulering generelt

Princip

Et procesreguleringssystem er et lukket system, i hvilken faktiske værdi (PV-procesværdi) af den regulerede størrelse, f.eks. niveauet, sammenlignes med den ønskede værdi (SP-setpunkt), og en korrektion udføres automatisk, således at afvigelsen bliver mindre.

Indgrebet i processen sker på grundlag af en procesværdi en måling af den regulerede størrelse. Man vil derfor ved en automatisk regulering altid have en lukket reguleringskreds eller -sløjfe.

Niveauproces



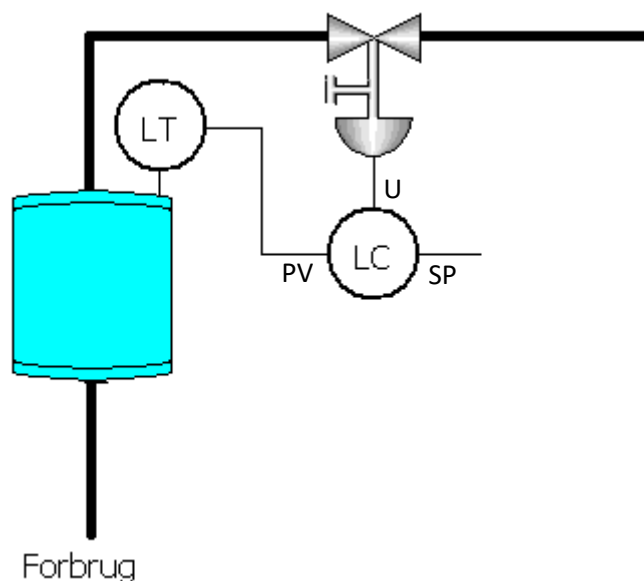
Tegningen viser en beholder, hvor niveauet skal holdes konstant, uanset hvor stort forbrug, der er på udgangsflowet.

Vandet pumpes frem til beholderen, hvor ventilen bruges til at regulere, hvor stor den tilførte vandmængde er.

Niveauet i beholderen måles med en niveaumåler, som er placeret i toppen.

For at holde et konstant niveau i beholderen skal den tilførte vandmængde være lige så stor som den mængde, der bortledes.

Reguleringskreds



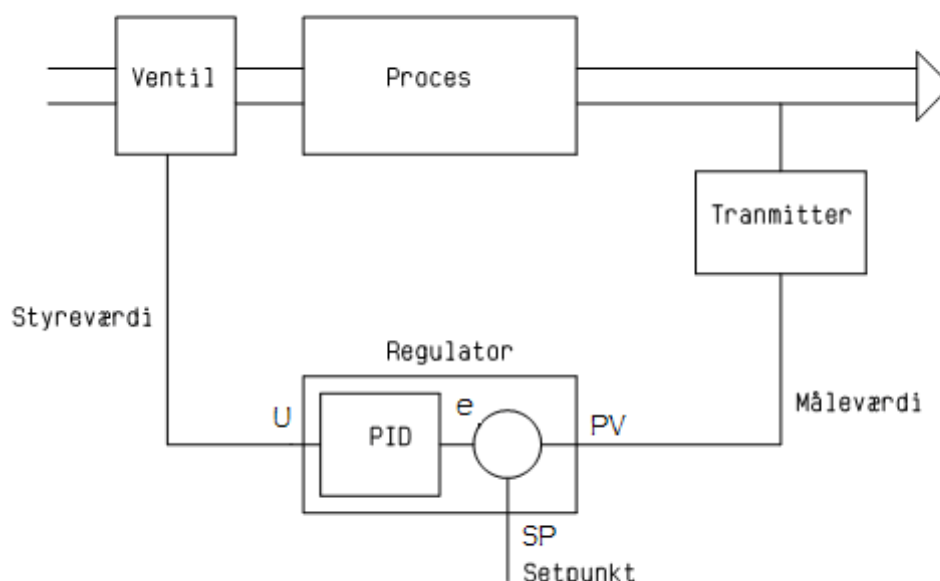
For at sikre et konstant niveau i tanken skal processen forsynes med en niveau-transmitter, der måler vandstanden i beholderen og sender dette PV-signal til regulatoren. PV-signalet sammenlignes med setpunktet SP som er det niveau man ønsker sig i tanken. Er PV og SP forskellig griber regulatoren ind på følgende måde: Regulatorens udgangssignal U styrer ventilen. Herved opnås mulighed for at ændre på den tilførte vandmængde.

Hvis vandstanden falder i tanken, giver niveaumålingen (LT) en tilbagemelding til regulatoren (LC). Regulatoren registrerer en afvigelse, og på grundlag af denne ændres udgangssignalet U

Udgangssignalet påvirker en ventil, som dermed ændrer på den tilførte vandmængde. Ventilen åbner, så den tilførte mængde igen er lig med den, der bortledes.

Et procesreguleringskredsløb er et lukket kredsløb, som indeholder en tilbageføring.

Blokdiagram



Diagrammet viser et blokdiagram for processen. Beholderens niveau måles med en transmitter, og måleværdien benævnes med bogstaverne PV.

Det ønskede setpunkt angives med bogstaverne SP, og regulatorens udgangssignal, styreværdien, benævnes med bogstavet U.

En regulator indeholder et sammenligningsled og en regulatoralgoritme. Sammenligningsleddet sammenligner måleværdien PV og setpunktet SP. Forskellen benævnes reguleringsafvigelse e for error.

PID-regulatoren indeholder nogle regneregler, som bestemmer, hvorledes regulatoren skal reagere på en reguleringsafvigelse. Der findes en række forskellige typer af regulatorer. I dette eksempel er der valgt en PID-regulator. PID-regulatoren modtager signalet fra sammenligningen, og ud fra afvigelsens størrelse ændrer PID-regulatoren udgangssignalet og dermed styreværdien U.

En PID-regulator indeholder et forstærkerled benævnt P og to tidsforholdsled, der benævnes henholdsvis I og D. De enkelte leds indflydelse på regulatorens reaktion indstilles med en række parametre på PID-regulatoren:

- P-leddets indflydelse indstilles med forstærkningen K_p (antal gange)
- I-leddets indflydelse indstilles med I-tiden (T_i eller T_n i sek.)
- D-leddets indflydelse indstilles med D-tiden (T_d eller T_v i sek.)

Det er vigtigt, at parametrene for de enkelte led er indstillet korrekt, ellers reagerer PID-regulatoren ikke korrekt, og processen kan gå i selvsving.

Indstilling af regulator

Optimal indstilling

En PID-regulator skal være i stand til at:

- Indregulere til en ønsket setpunktsværdi efter en forstyrrelse
- Indregulere til en ny værdi efter ændring af setpunkt

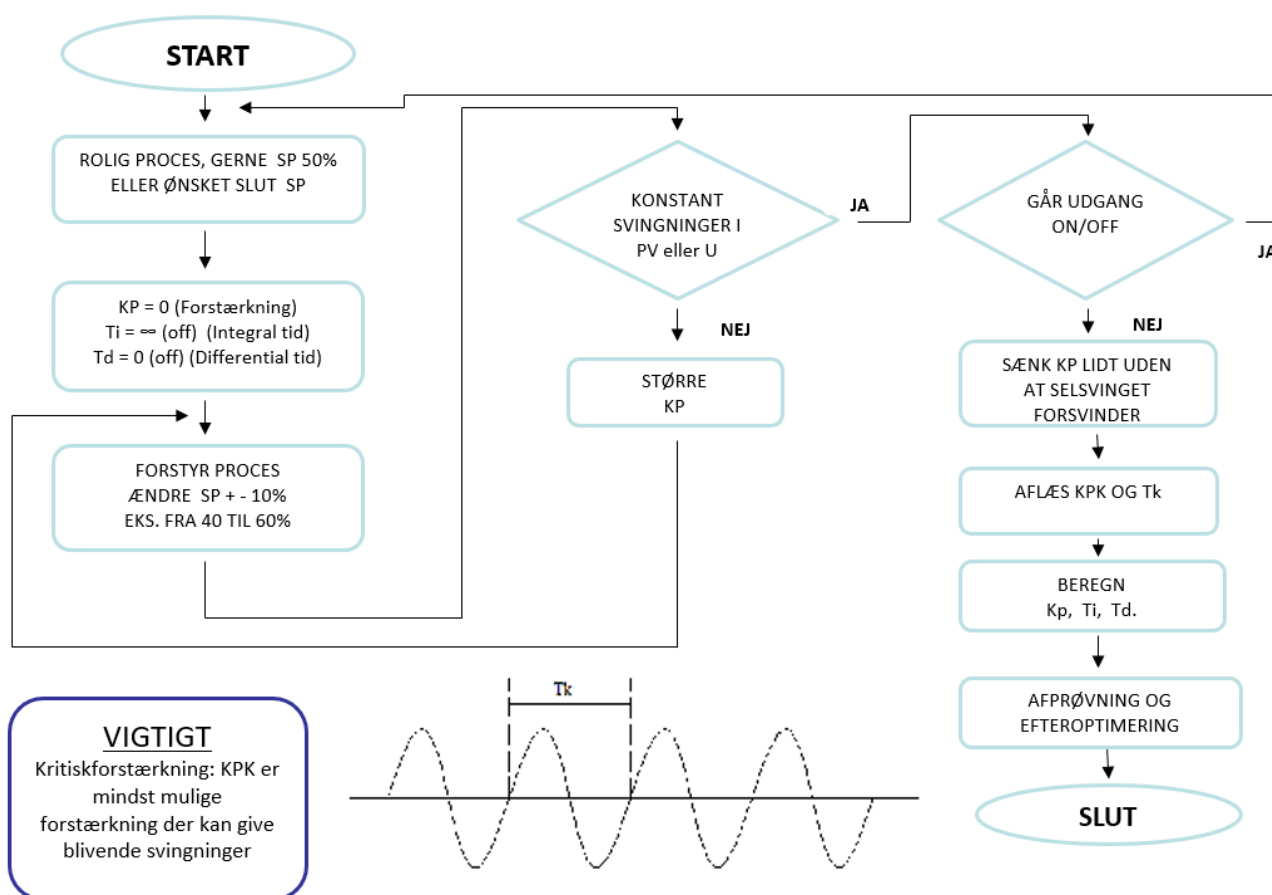
I en proces skal dette ske så hurtigt som muligt og helst uden for mange oversving omkring setpunktet.

Det efterfølgende er en beskrivelse af et par metoder, der kan anvendes ved indstilling af regulator. Disse metoder er kun vejledende. Der skal typisk foretages en efterjustering.

De parametre, der skal indstilles på en PID-regulator, er K_p forstærkningen, I -tiden og D -tiden.

Ziegler & Nichols

Indstilling efter Ziegler & Nichols-metoden er udarbejdet på grundlag af, at processen overholder nogle nøje specifikke egenskaber, som vi ikke kommer nærmere ind på i dette afsnit. Herunder er metoden vist som flowchart. På næste side beskrives den som tekst.



Ziegler & Nichols-metoden må kun anvendes som et udgangspunkt sammen med de øvrige håndregler for regulatorindstilling i en proces. Indstillingen sker i praksis ved, at man forsøger sig frem til den bedste indstilling.

Fremgangsmåden kan beskrives i en række trin:

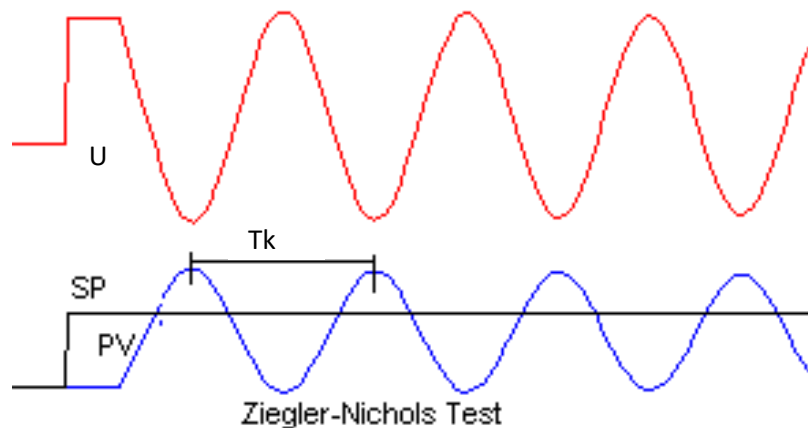
- Indstil regulatoren som ren P-regulator
- Foretag en hurtig forstyrrelse i systemet og betragt måleværdien PV

Forstyrrelsen kan f.eks. foretages ved at ændre setpunktet.

Hvis processen ikke går i selvsving, øges forstærkningen, og der laves en ny forstyrrelse.

Forstærkningen ændres, indtil systemet bliver ustabilt og går i selvsving.

Måleværdien skal svinge med en konstant amplitude.



Forstærkningen må ikke forøges så meget, at udgangen svinger on/off.

- Den kritiske forstærkning K_{PK} , hvor systemet går i sving, aflæses på regulatoren
- Tiden T_k for en svingning bestemmes
- Ud fra de fundne værdier beregnes K_p , T_i og T_d ved hjælp af efterfølgende tabel

| Regulator type | Proportionalbånd | Forstærkning | I-tiden | D-tiden |
|----------------|------------------|---------------------|------------------|-------------------|
| P | $X_{pk}/0,5$ | $0,5 \cdot K_{PK}$ | | |
| PI | $X_{pk}/0,45$ | $0,45 \cdot K_{PK}$ | $0,83 \cdot T_k$ | |
| PID | $X_{pk}/0,6$ | $0,6 \cdot K_{PK}$ | $0,5 \cdot T_k$ | $0,125 \cdot T_k$ |

Ziegler & Nichols-metoden kan være farlig at benytte, fordi forsøget foretages på et procesanlæg, som skal gå i selvsving. Situationen kan løbe løbsk, uventede ulineariteter kan vise sig, når svingningerne begynder, og katastrofale udsving kan blive følgen.

Forsøgsmetoden

Regulatoren kobles i automatisk mode. Regulatoren indstilles til ren P-regulator.

- Forstærkningen K_p forøges, indtil regulatorens udgangssignal bliver uroligt.
 - K_p indstilles herefter på halvdelen af værdien.
- Hvis der ønskes en I-virkning, skrues T_i ned, indtil styresignalet bliver uroligt.
 - T_i indstilles herefter på en værdi, der er lidt større.
- Hvis der ønskes en D-virkning, skrues T_d op, indtil styresignalet bliver uroligt.
 - T_d indstilles herefter på en værdi, der er lidt mindre.

Efter indstilling skal der foretages en optimering af de enkelte parametre.

PID-regulator S7-1500 CONT_C

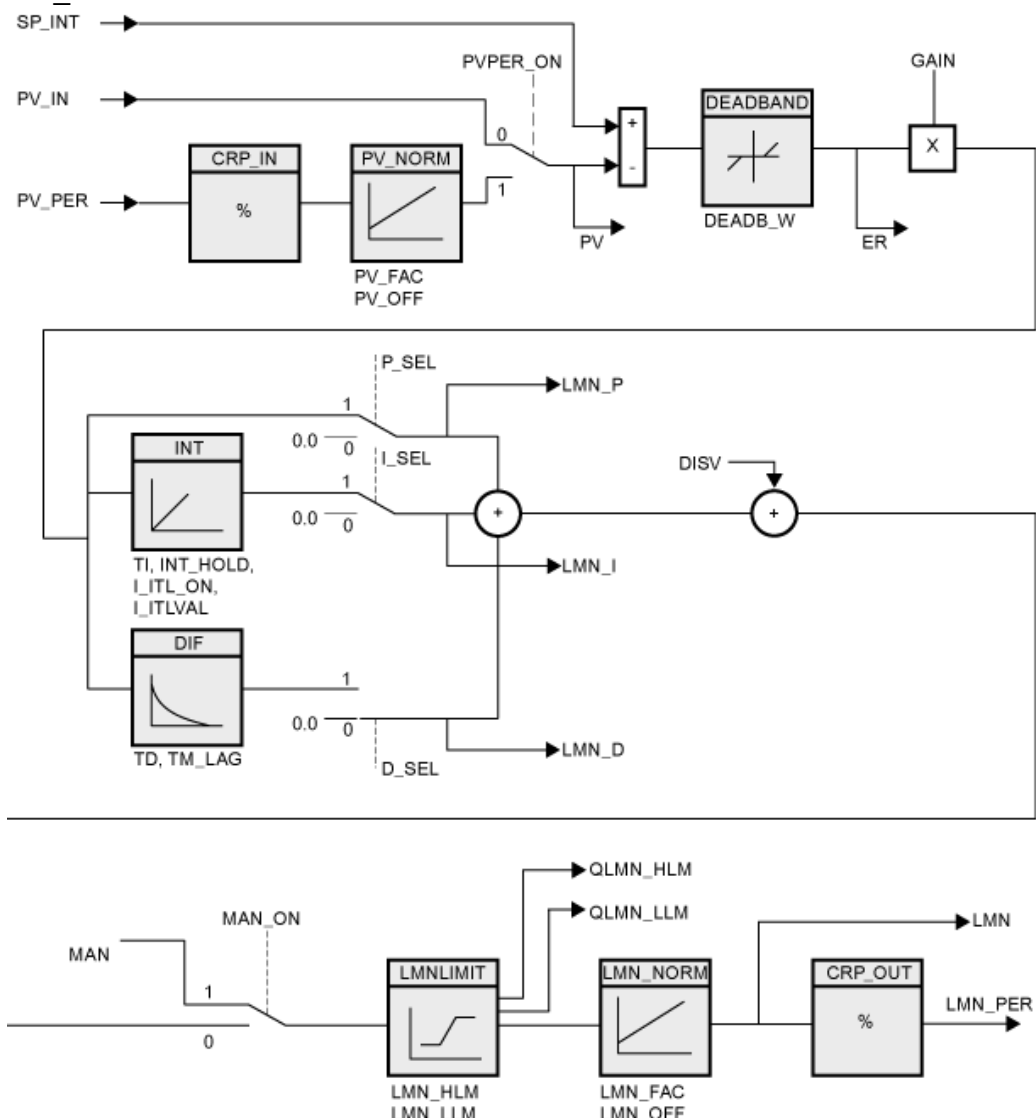
PLC med PID-regulator

Eksempel

Det efterfølgende er en kort beskrivelse af en programmerbar PID-regulator på en Siemens S7-1500 PLC.

I PLC'en anvendes en FB 41 (Cont_C) som PID-regulator. FB 41 er et færdigt program, der leveres fra Siemens. Programmet indeholder de algoritmer, som bruges til en PID-regulator.

Blokdiagram CONT_C FB 41



Blokdiagrammet viser en PID-regulator. I det efterfølgende er der en beskrivelse af de vigtigste parametre på regulatoren.

SP_INT = setpunkt, angives i real 0-100 %
 PV_IN = procesværdi, angives i real 0-100 %
 PV_PER = procesværdi fra periferi f.eks. analogtinput, angives 0-27648

Ved programmering vælges hvilken procesværdiindgang, som skal anvendes. Dette vælges med variablen PVPER_ON.

PVPER_ON = skift mellem procesværdiindgange, angives med boolsk værdi.

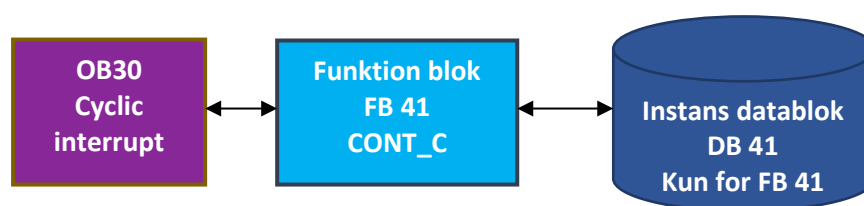
Gain = forstærkning (P-led), antal gange angives i real tal
 DISV = Bais U_0 eller feedforward input i real tal
 TI = I-tiden (I-led), sek. angives i Time
 TD = D-tiden (D-led), sek. angives i Time
 TM_LAG = tidsforsinkelse til D-ledet, sek. angives i Time

P-SEL = P-led on/off, angives med boolsk værdi.
 I-SEL = I-led on/off, angives med boolsk værdi.
 D-SEL = D-led on/off, angives med boolsk værdi.

MAN_ON = manuel/automatic, angives med boolsk værdi.

LMN = Udgangssignal, angives i real 0-100 %
 LMN_PER = Udgangssignal, via periferi f.eks. Analogudgang, angives 0-27648

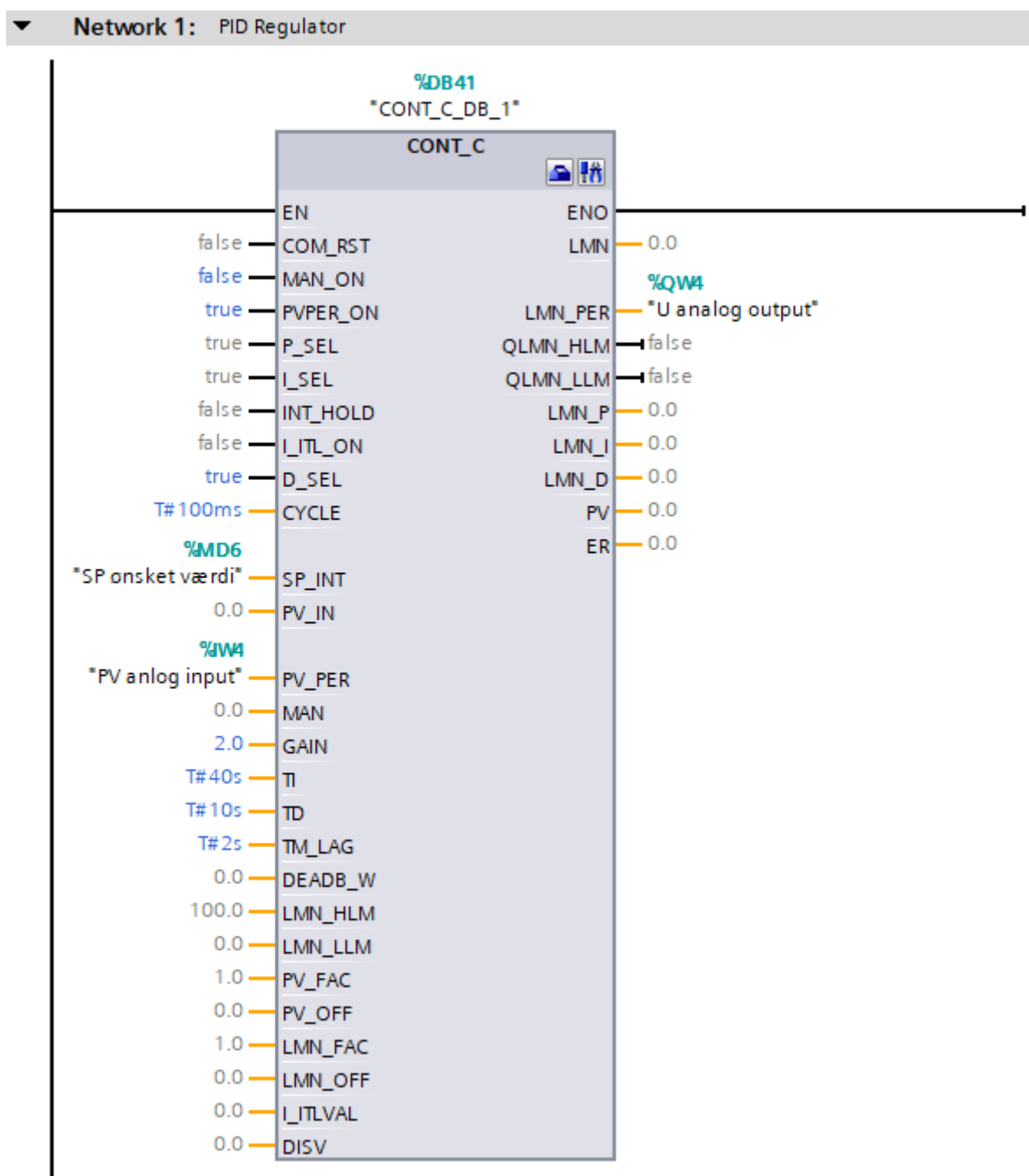
Kald af regulator



PID-regulatoren CONT_C FB 41 skal kaldes med et bestemt tidsinterval. OB 30 kalder FB 41 cyklisk med et tidsinterval på 100 mS. Dette medfører, at regulatoren kaldes hver gang, der er gået 100 mS.

| PLC bog ▶ PLC_3 [CPU 1516-3 PN/DP] ▶ Program info | | | | | |
|---|---------------------|------------|-------------------------|----------------------|-------------------------|
| Call structure | | | | | |
| Call structure of PLC_3 | | | | | |
| | Call structure | Address | Details | Local data (in path) | Local data (for blocks) |
| 1 | Main | OB1 | | 0 | 0 |
| 2 | ▼ Cyclic interrupt | OB30 | | 0 | 0 |
| 3 | CONT_C, CONT_C_DB_1 | FB41, DB41 | @Cyclic interrupt ▶ NW1 | 0 | 0 |

PLC-programeksempel



Programeksemplet viser kald af en PID-regulator, der placeres i OB 30.

Parametre

De regulatorparameter, der angives nedenfor, er dem, det er nødvendigt at ændre for at idriftsætte regulatoren.

Den grå tekst ud for benene angiver regulatorens grundindstillinger. De steder, hvor der er ændret på disse, er det angivet med blå.

MAN_ON = "false" bruges til at skifte mellem manuel og automatik. Logisk 0 er lig med automatik.

PVPER_ON = "true" skal være lig med logisk 1 for indlæsning af procesværdi via periferi.

D_SEL = "true" skal være lig med logisk 1 for at aktivere D-ledet.

CYCLE = her angives cyklostiden, som i dette eksempel er 100 mS. (pga. kald fra OB30)

SP_INT = setpunkt, som i dette eksempel hentes fra MD6.

PV_PER = processignal, der hentes fra en analog indgang IW4.

GAIN = forstærkning $K_p = 2$

DISV = Bais U_0 bruges kun ved ren P- eller feedforward- regulering = 0

TI = I-tiden = 40 sek.

TD = D-tiden = 10 sek.

TM_LAG = der erfaringsvist sætte til 1/5 af TD = 2 sek.

LMN_PER = Udgangssignal, der overføres til det analoge output QW4.

Siemens FB41 CONT_C regulatorer, følger denne regulatorligning:

$$LMN = K_p * \left(e + \frac{1}{T_i} \int_0^t e * dt + T_d * \frac{de}{dt} \right) + DISV$$

Det betyder, at værdier beregnet med Z & N, kan bruges direkte.

PID-regulator Compact S7-1200 og 1500

PID Compact Proces Regulatorer

I et lukket reguleringsystem, hvor der er en tilbagekobling af måleværdien af den størrelse, der skal reguleres, ser man brugen af softregulatorer i PLC'er.

Brugen af softregulatorer i programmer har den fordel, at det skaber overblik. Man kan simpelt og billigt have flere regulatorer i et program og skal ikke udføre fortrådning mellem dem.

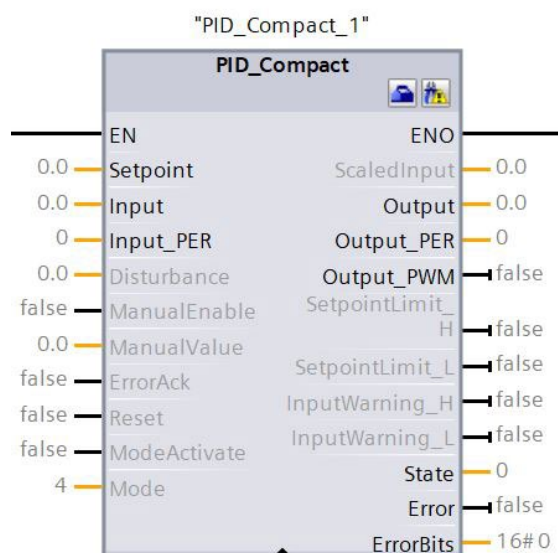
Brugerfladen kan være mere overskuelig, da der kan indgå idriftsættelse gennem programmeringsværktøjet.

Begrænsningen kan ligge i fysiske indgange, som kan løses med udvidelsesmoduler.

PID Compact har seks forskellige driftstilstande.

Driftstilstande:

- Inactive
- Pretuning
- Fine tuning
- Automatic mode
- Manual mode
- Substitute output value with error monitoring



PID-algoritme

Algoritmen for Siemens PID Compact softregulator er beskrevet nedenfor. Man kan se proportional forstærkning er trukket uden for parentesen og påvirker alle tre led proportionalt. Dette betyder, at vi har en standardkoblet PID-regulator, hvor værdier beregnet med Z & N kan bruges direkte.

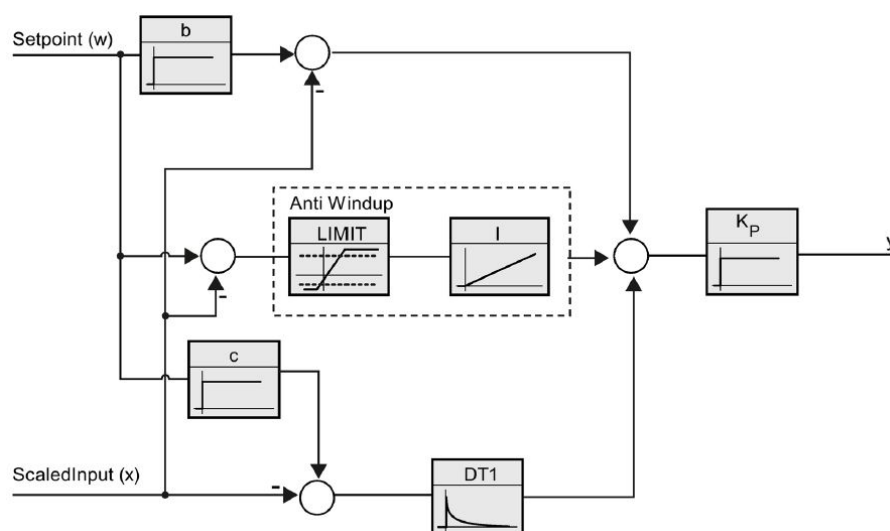
De forskellige variabler, som ikke er PID-parametrene, kan indstilles af regulatorens autotuner eller manuelt.

$$y = K_P \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

| | |
|------------------|---|
| y: | Output værdi af PID-algoritme |
| K _P : | Proportional forstærkning |
| s: | Laplace operator |
| b: | Proportional koefficient |
| w: | Setpoint |
| x: | Procesværdi |
| T _I : | Integraltid |
| a: | Differentieltid forsinkelseskoefficient |
| T _D : | Differentieltid |
| c: | Differentiel koefficient |

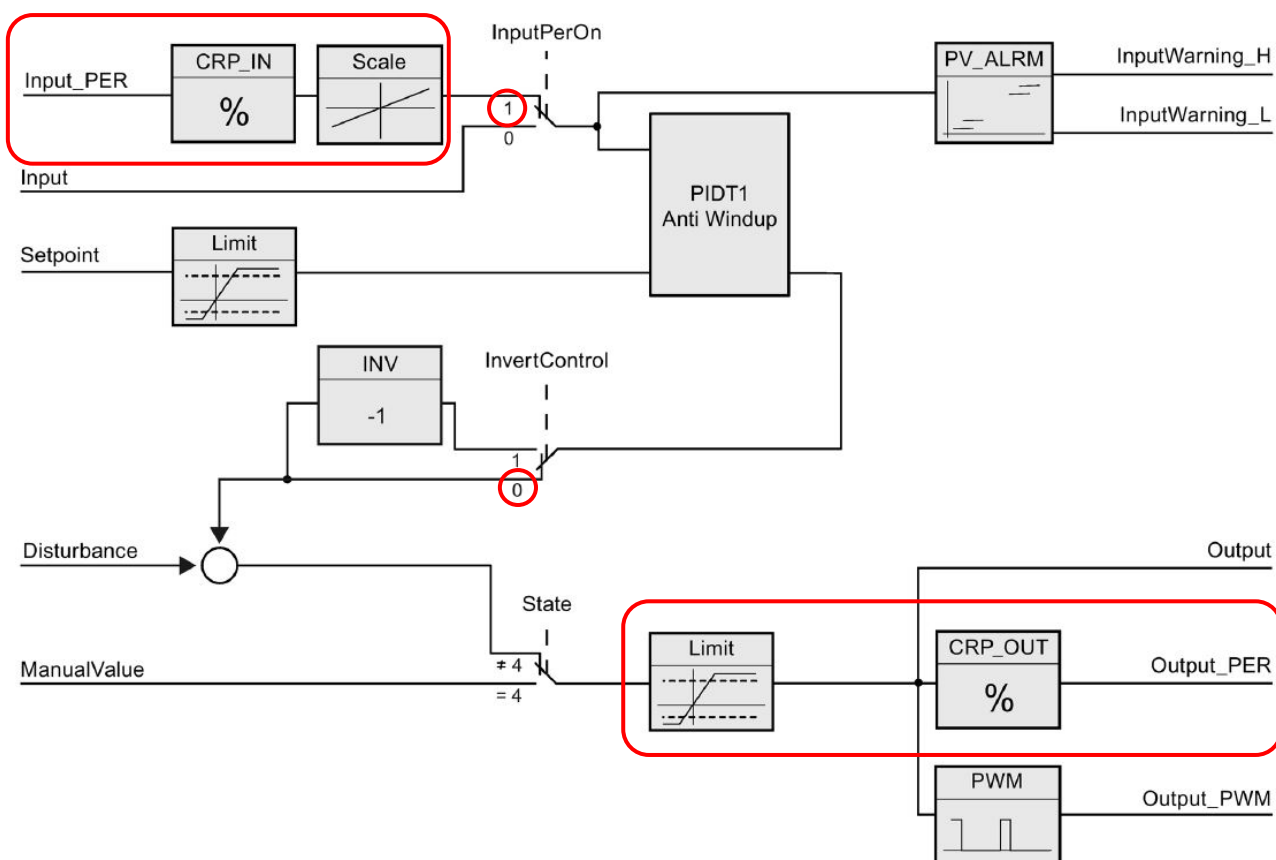
Blokdiagrammet for algoritmen viser, at der er indbygget en anti windup-funktion i integralblokken. Dette er for at modvirke over- eller undersving efter store trinændringer på procesværdien PV.

Det er også muligt at se de to vægtninger af P- og D-leddene (b, c). Disse fungerer på samme måde som Proportional forstærkningen.



Som default skal Anti windup blive resat, når regulatoren går fra Inactive til Automatic mode. Hvis den aktuelle værdi er lavere end SP, vil outputtet stige. Dette er uafhængigt af proportional-vægtningen og regulerings-differensen.

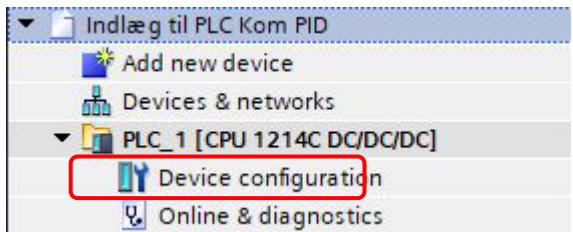

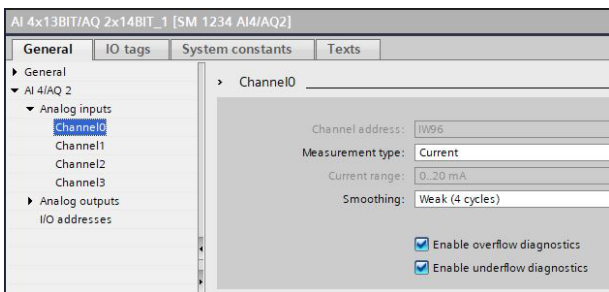
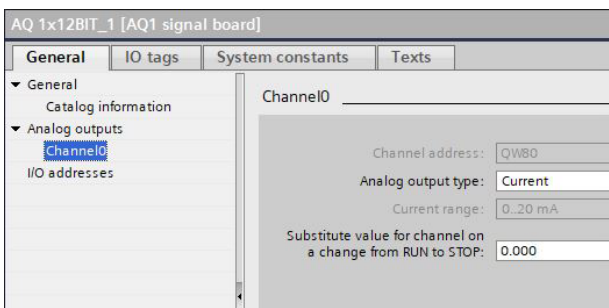
Blokdiagrammet for regulatoren er vist nedenfor. I denne opstilling benyttes de markerede funktioner.

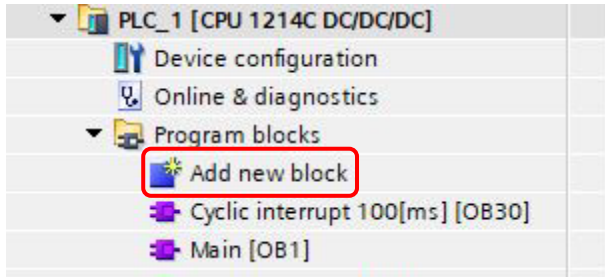
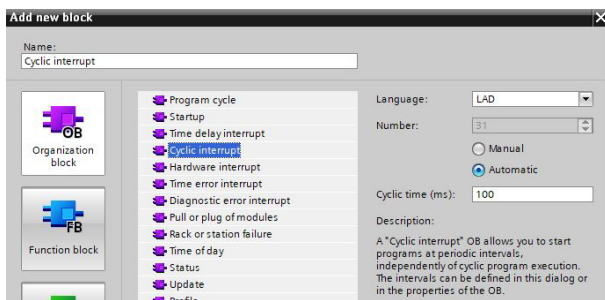


Hardware & Organization block

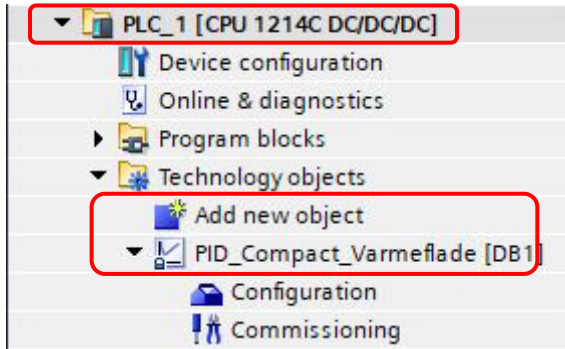
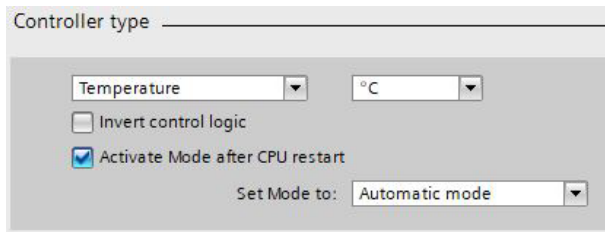
Fremgangen nedenfor viser, hvordan periferien kan sættes op til regulatoren.

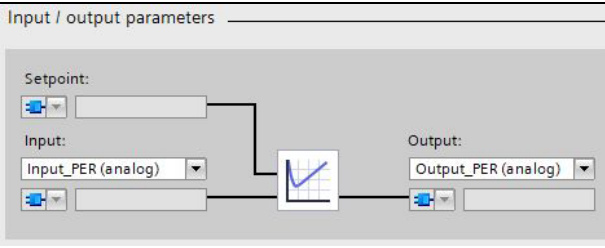
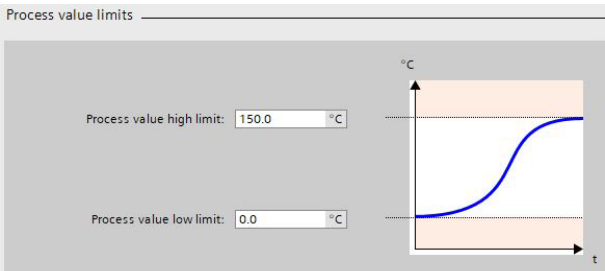
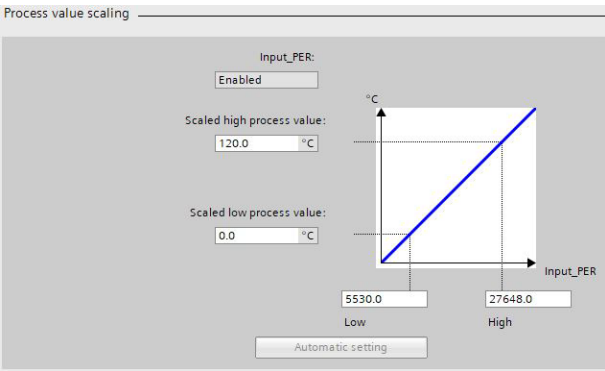
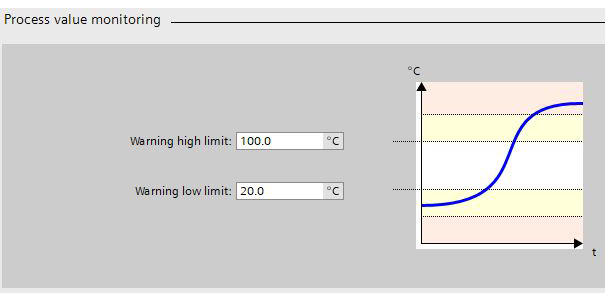

Samtidigt er det vist, hvor regulatoren skal blive kaldt i programmet.

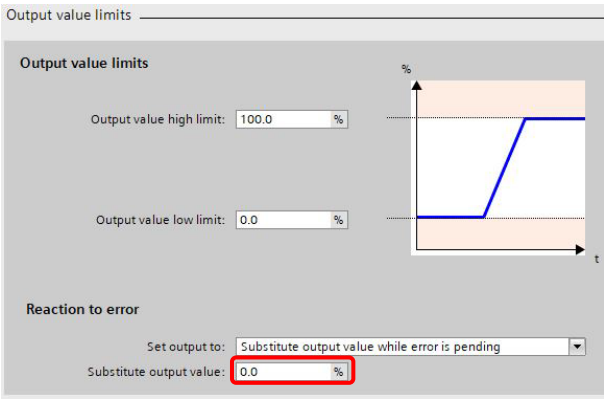
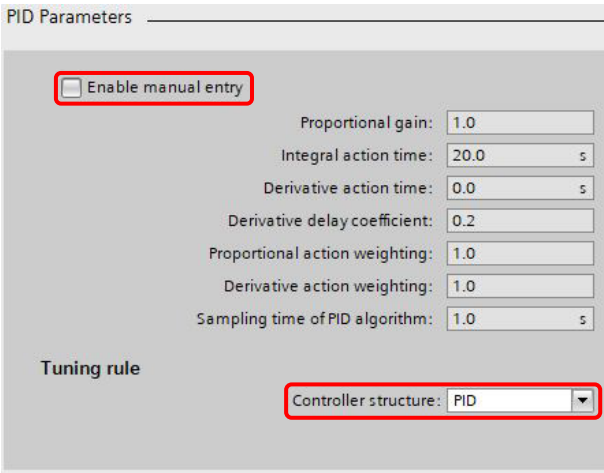
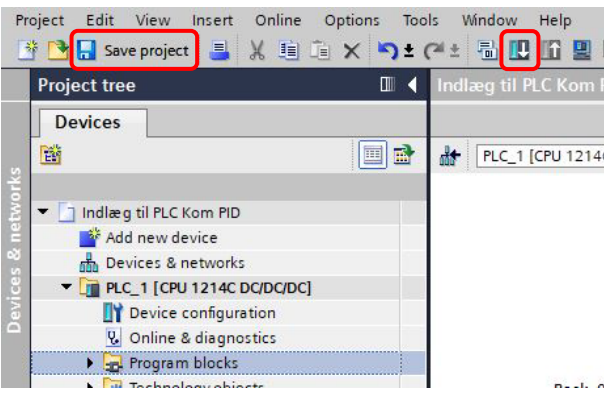
| Nr. | Handling | Note |
|-----|--|--|
| 1. | Åbn device configuration til CPU-opsætningen. |  |
| 2. | <p>Til denne opsætning bliver der brugt en CPU 1214C med et SM1234 AI/AQ og et Signal board AQ.</p> <p>Feedback-signaler er et 4 – 20[mA] og outputsignalet er et 4 – 20[mA].</p> |  |
| 3. | <p>En 2-trådsstrømforbindelse er monteret til SM-modulet på periferiindgangen "kanal 0" med adresse IW96.</p> <p>Der er ikke behov for at glatte signalet ud med "Smoothing"-funktionen.</p> |  |
| 4. | <p>Styresignaler til varmefladen er monteret til Signal board AQ på periferiudgangen "kanal 0" med adresse QW80.</p> |  |

| | | |
|----|---|--|
| 5. | <p>Under PLC_1-mappen skal der oprettes en ny organiseringsblok.</p> <p>Dette udføres under Program blocks "Add new block".</p> |  |
| 6. | <p>Under organization block kan der indføres en "Cyclic interrupt".</p> <p>Den cykliske tid mellem to OB-kald sættes til 100[ms] af hensyn til bekvemmelighed ved gennemgang.</p> |  |
| 7. | | |

Technology object & Configuration

| Nr. | Handling | Note |
|-----|---|--|
| 1. | <p>Under Technology objects tilføjes PID-regulator. Under den laves konfigurationen.</p> |  |
| 2. | <p>Det skal være en temperaturregulator, og PV skal holdes op mod SP i grader celsius.</p> <p>Der skal reguleres direkte, og efter en CPU-restart skal regulatoren gå i automatisk styring.</p> |  |
| 3. | <p>Som vist tidligere under analog tilslutning, bruges der (IW96 & QW80).</p> | |

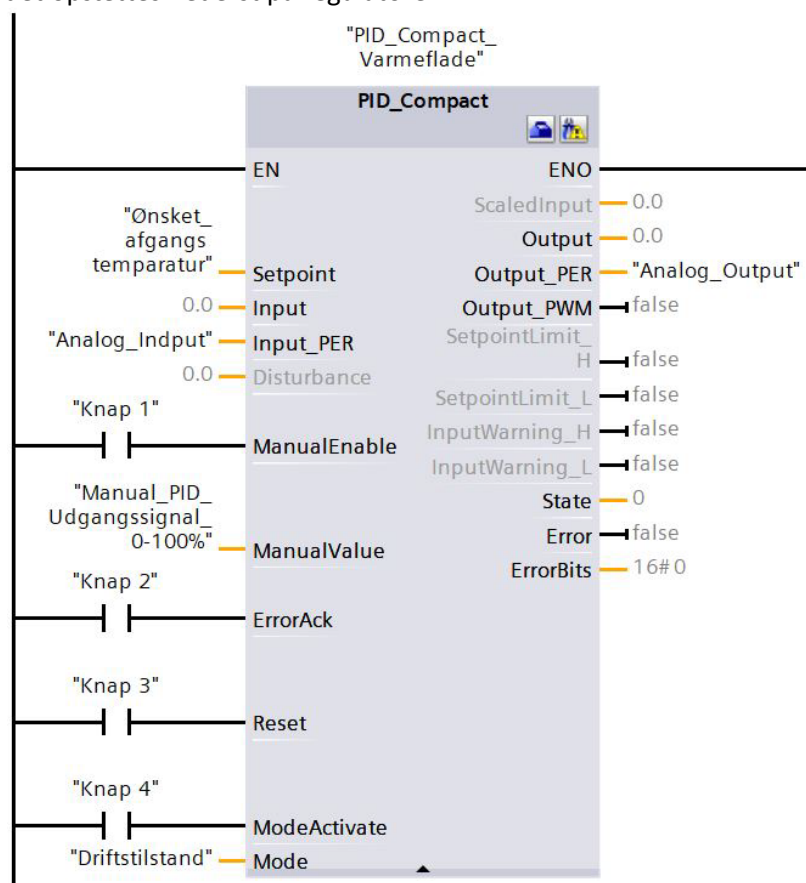
| | | |
|----|--|--|
| | Derfor vælges der Input PER (analog) & Output PER (analog). |  |
| 4. | <p>Processen skal arbejde indenfor 20 – 100 grader celsius.</p> <p>Grænserne skal være maks. 150.0 & min. 0.0 grader celsius.</p> |  |
| 5. | <p>Feedbacksignalet er et 4 – 20 mA-signal og skal skaleres ud fra periferien til den fysiske målestørrelse, som er temperaturtransmitterens måleområde.</p> <p>Der bliver målt med en PT-100-føler og transmitterens feedbacksignal svarer til 0.0 – 120.0 grader celsius.</p> <p>Periferien skal skaleres fra 5530.0 – 27648.0</p> |  |
| 6. | <p>Advarselsgrænser er lagt på minimum og maksimum af arbejdsområdet.</p> <p>20.0 – 100.0 grader celsius.</p> |  |
| 7. | Hvis der bliver valgt et styreelement, som er digitalt styret og output-parameter er sat til PWM, kan man manuelt indstille minimum ON tid og OFF-tid på det digitale signal. |  |

| | | |
|-----|--|--|
| 8. | <p>Output-signalet kan blive begrænset, hvis dette er nødvendigt.</p> <p>Hvis regulatoren går i fejl, kan output blive sat til en prædefineret værdi.</p> |  |
| 9. | <p>Parametrene kan blive manuelt indstillet med "Enable manual entry".</p> <p>Hvis man bruger autotuning, kan beregningsmetode for PI & PID vælges i "Controller structure".</p> |  |
| 10. | <p>Herefter er konfigurationen færdig. Gem projektet og download konfigurationen.</p> |  |

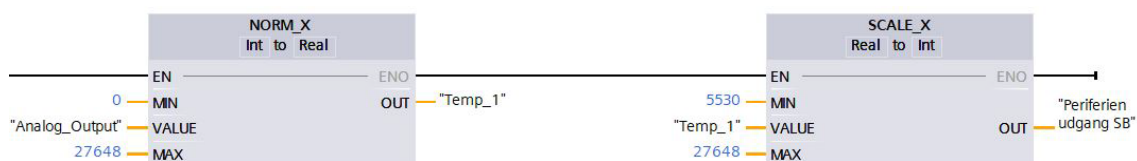
Programming

I den Cyclic interrupt OB skal PID Compact kaldes. Nedenfor ses de forbindelser, der er lavet til PID Compact FB'en:

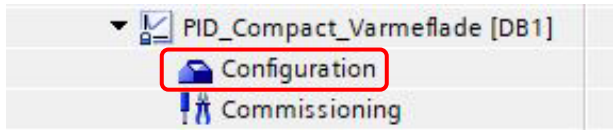
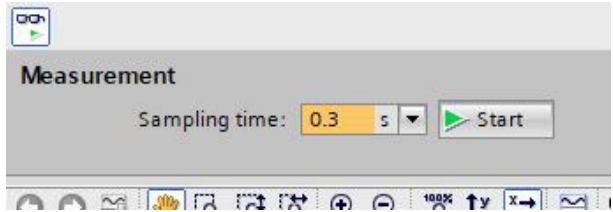
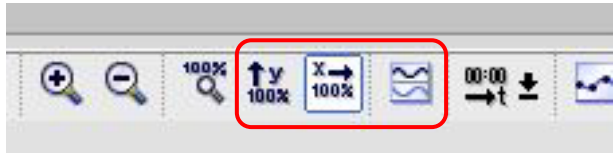
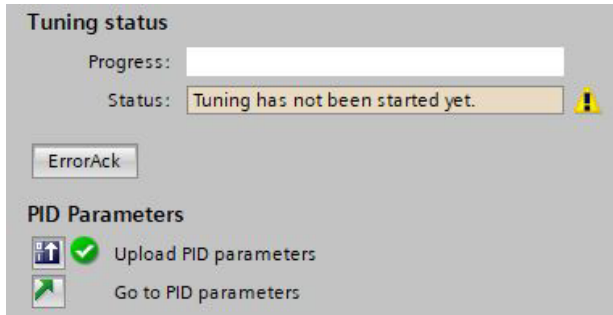
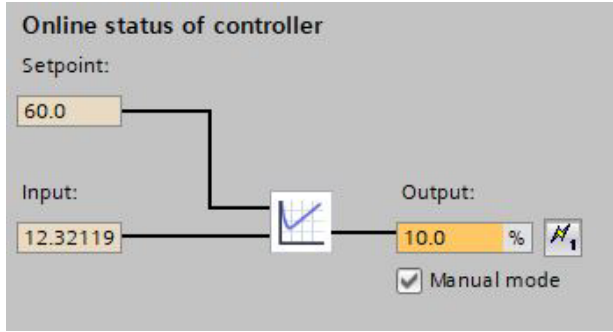
- Det ønskede setpunkt til varmefladen
- Regulatorens feedback gennem periferiindgangen IW96 på SM1234
- Der er opsat knapper til manuel styring af udgangssignalet ved hjælp af et potentiometer (*opsætningen af denne er ikke vist*)
- Der er opsat øvrige knapper til regulatoren
- Ønsker man ikke at benytte "Commissioning" til at skifte mode, kan det opsættes nederst på regulatoren

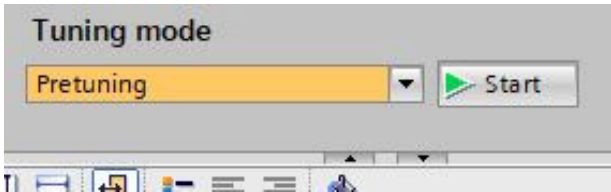
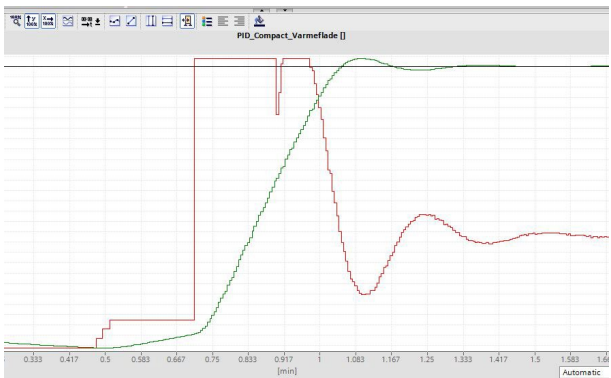
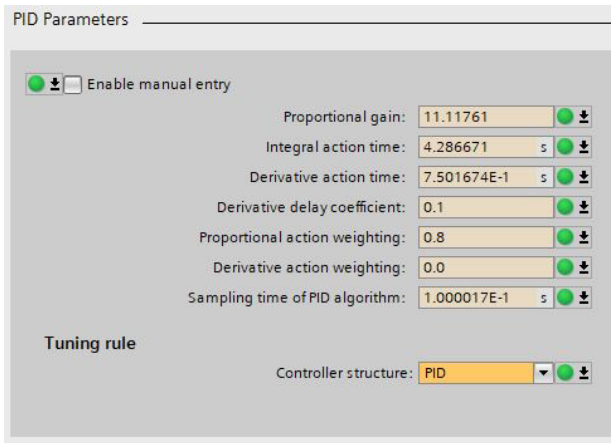



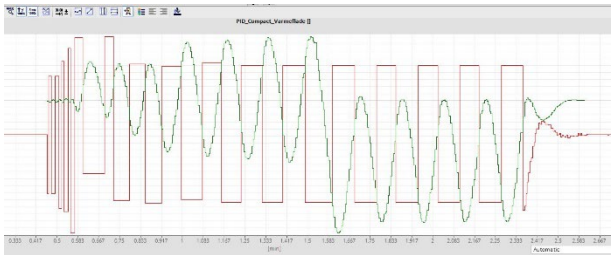
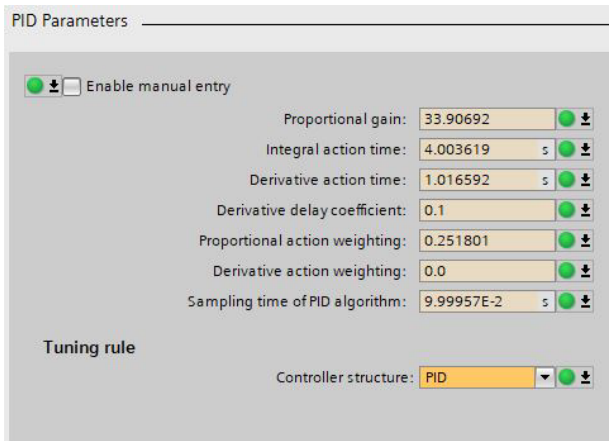
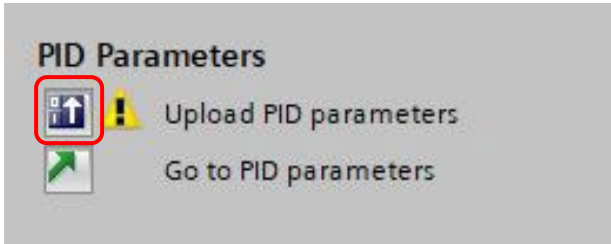
På outputtet var det periferi-udgang QW80. Styresignalet fra regulatoren skal skaleres fra regulatoren, da opstillingens analog-signal til styreenheden er 4 - 20[mA]



Idriftsættelse

| Nr. | Handling | Note |
|-----|--|--|
| 1. | Under "Commissioning" i teknologi objekt kan man idriftsætte regulatoren. |  |
| 2. | Regulatoren har en indbygget måler og trend-kurve for SP, PV & Y. Det er muligt at opskalere disse i reel time. |  |
| 3. | I værktøjslinjen kan man autoskalere Y- og X-aksen. Der er også mulighed for at samle Outputtet med PV & SP. |  |
| 4. | For at pretune kan starte, skal regulatoren være i inactive, manual eller automatic mode ManualEnable = FALSE Reset = FALSE |  |
| 5. | Forskellen mellem SP & PV skal være større end $0,3 * (\text{Input Upper} - \text{Lower Limit})$ & $0,5 * \text{Setpoint}$ Sæt regulatoren i manual mode (via ModeActivate) og lav forskellen med regulatorens udgangssignal. |  |

| | | |
|----|--|--|
| 6. | Når alle krav er opfyldt, start pretuning. |  |
| 7. | <p>Under pretuningen laver regulatoren en step-respons på processen (varme fladen) og søger efter vendepunktet på svaret fra step-responsen.</p> <p>Dødtiden i sløjfen bliver også målt.</p> |  |
| 8. | Under konfiguration kan man se de parametre, som pretuningen er kommet frem til efter beregningen. |  |
| 9. | Det anbefales efter en pretuning, at man laver en finetuning. |  |

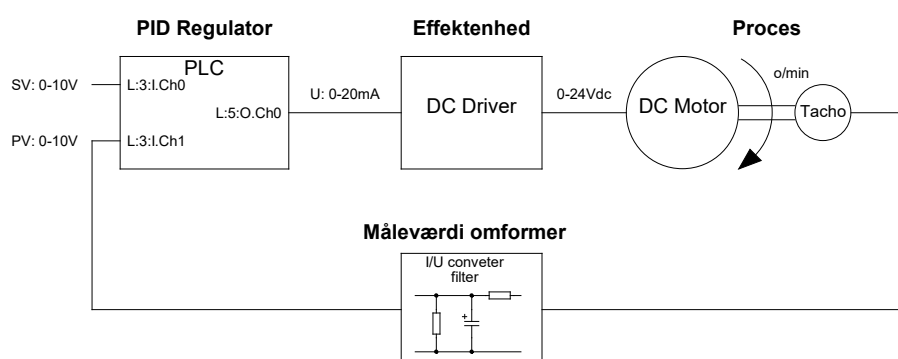
| | | |
|-----|---|--|
| 10. | <p>Under finetuningen vil regulatoren skifte udgangssignalet fra høj til lav for at sætte processen i svingninger.</p> <p>Herunder aflæser regulatoren den kritiske forstærkning og svingningstid i reguleringssløjfen.</p> |  |
| 11 | <p>Når regulatoren har den kritiske forstærkning og svingningstid i reguleringssløjfen, beregner den de nye parametre.</p> |  |
| 12 | <p>Hvis man ønsker de parametre, autotuningen er kommet frem til, kan de uploades direkte under "commissioning".</p> <p>Husk at gemme projektet herefter.</p> |  |

PID-regulator Allen-Bradley

Generelt

Allen-Bradley PLC har en indbygget PID-regulator. I dette afsnit beskrives, hvordan man programmerer denne regulator, når man ønsker, at den skal styre en motors hastighedsreguleringskreds som vist på blokdiagrammet.

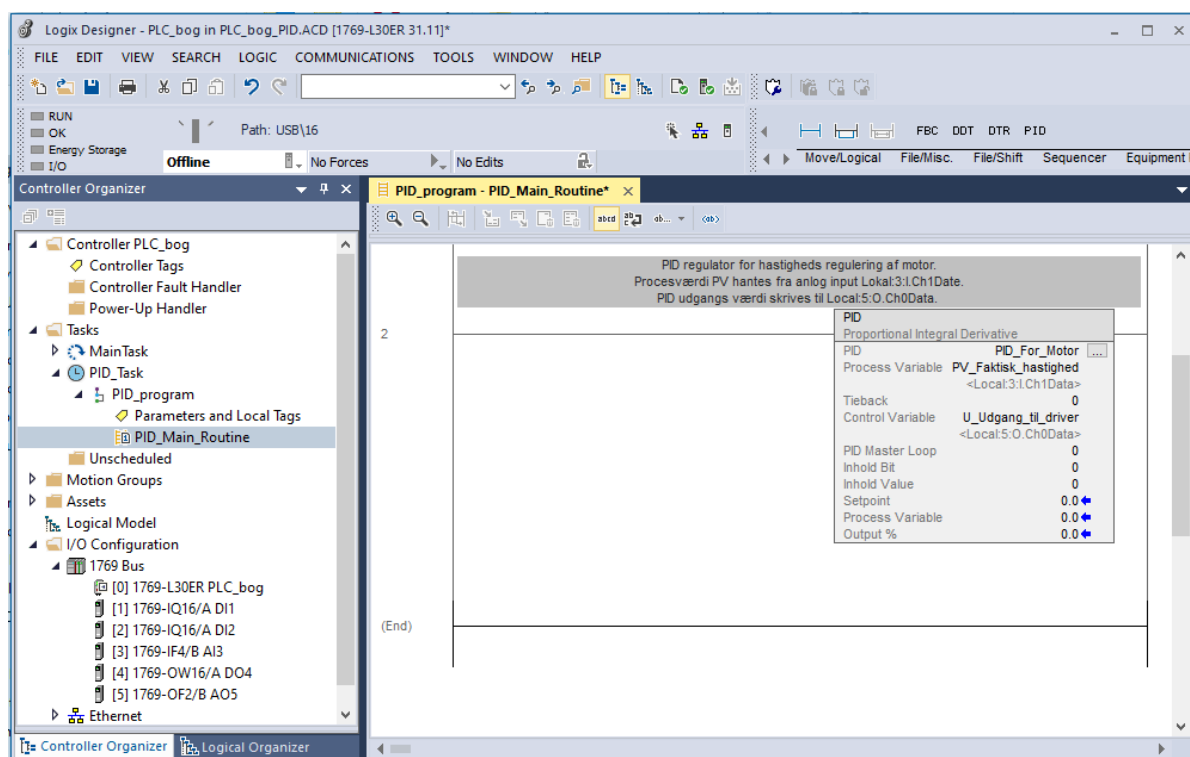
Blokdiagram motor hastighedsregulering



PID-regulatoren

Nedenfor er der vist, hvordan PID-regulatoren ser ud, når den er klar til drift.

Inden den kan idriftsættes, er der dog en del opsætninger. Disse bliver forklaret på de næste sider.

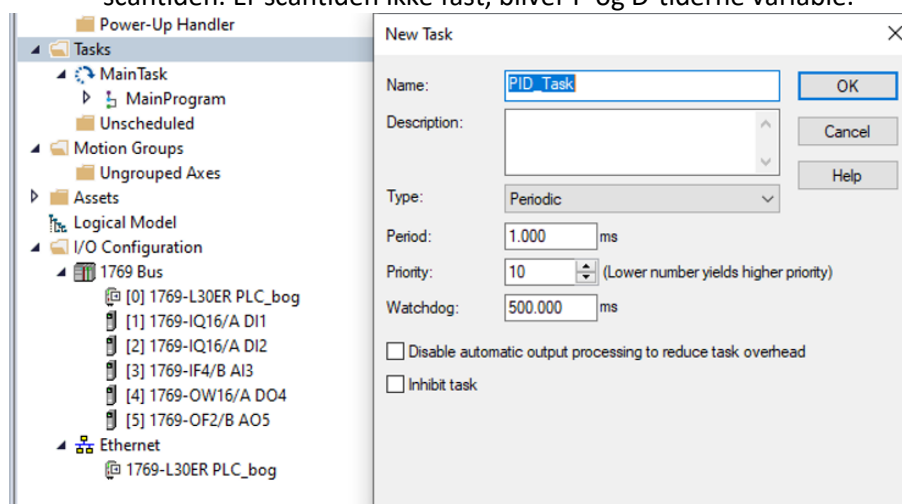


Programopsætninger

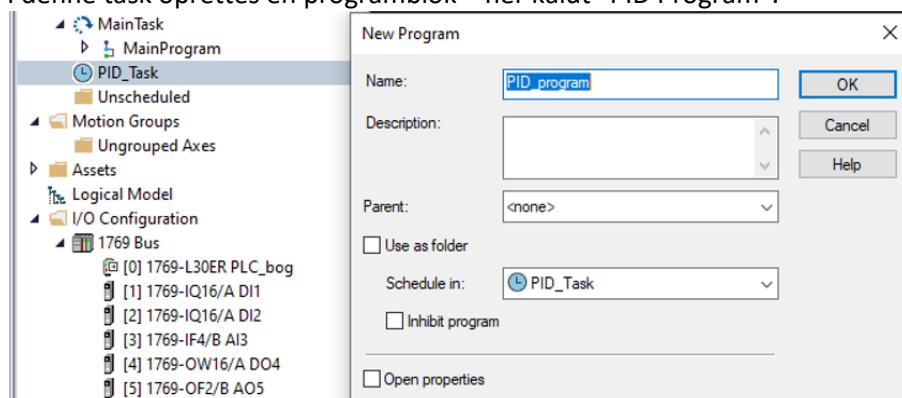
Følgende opsætninger er en forudsætning for, at PID-regulatoren kan virke.

Opret en Periodisk task (kaldt PID Task) med en periodetid på 1 ms.

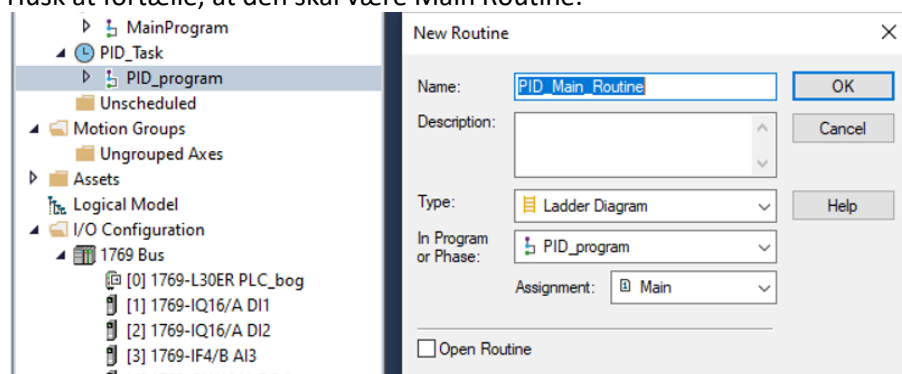
- PID-regulatoren skal afvikles fra en programblok med fast scantid, da PLC'en beregning af I- og D-leddets tilskud til U afhænger af scantiden. Er scantiden ikke fast, bliver I- og D-tiderne variable.



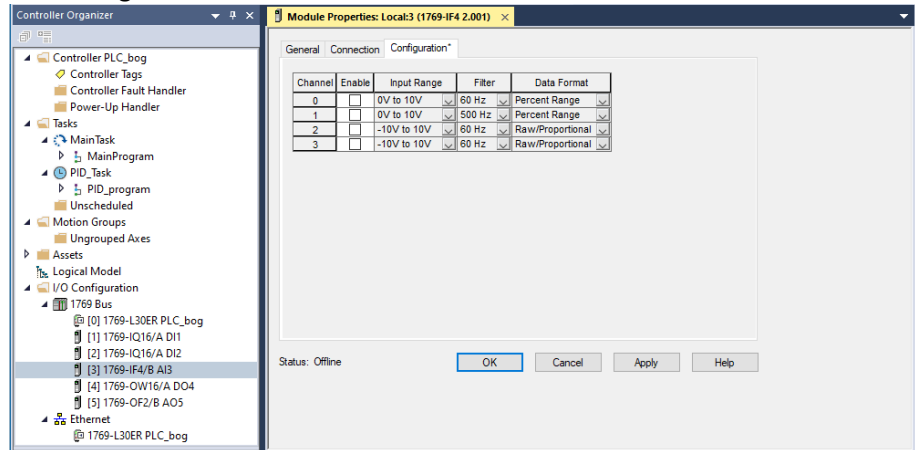
I denne task oprettes en programblok – her kaldt "PID Program".



I PID-programblokken oprettes en Routine – her kaldt "PID Main Routine". Husk at fortælle, at den skal være Main Routine.

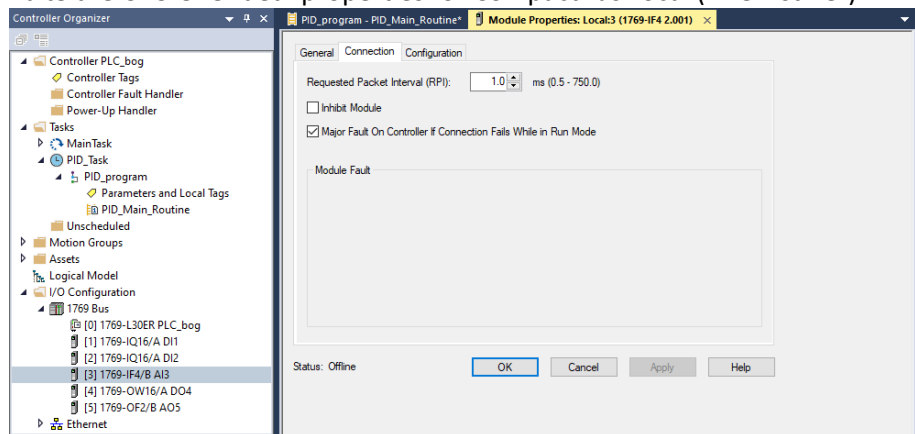


Opsæt det analoge input for PV, så det står til 0 V til 10 V og "Percent Range", som vil sige, at de 0-10 V svarer til 0 – 10.000 i PLC'en. Vælg herefter filterfrekvens til 500 Hz, da vi vil have så kort opdateringstid som muligt.

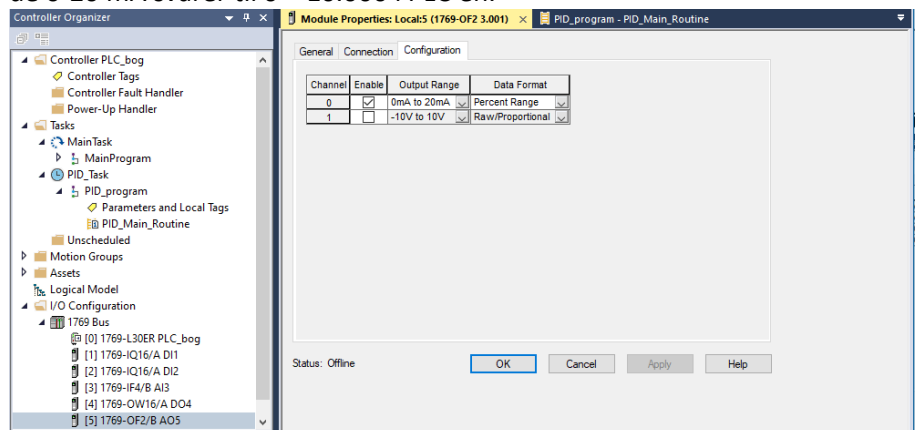


Ændrer, ved hjælp af properties på det analoge inputkort, opdateringstiden (RPI) til 1ms. Denne meget hurtige opdateringstid er nødvendig, fordi det er en meget hurtig motorregulering.

På ældre CPU'er er det i properties for CompactBus Local (ikke vist her).

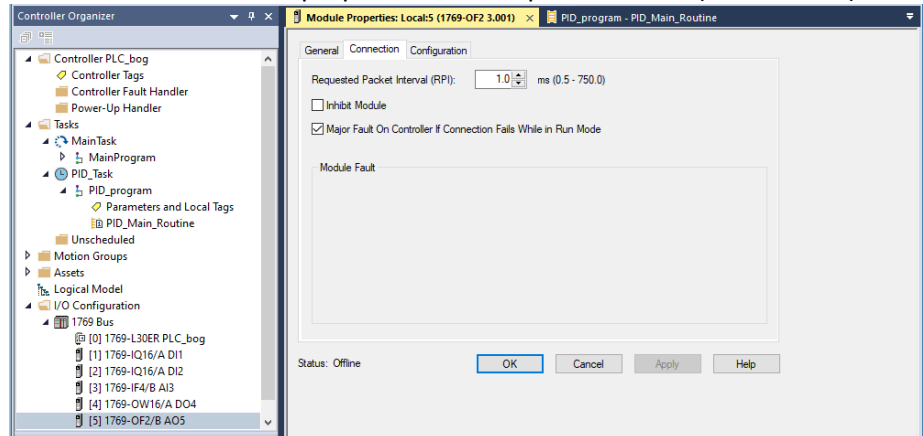


Analogudgangen sættes til = 0 til 20 mA og "Percent Range", som vil sige, at de 0-20 mA svarer til 0 – 10.000 i PLC'en.

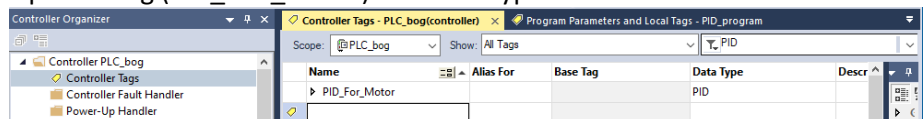


Ændrer, ved hjælp af properties på det analoge outputkort, opdateringstiden (RPI) til 1 ms. Denne meget hurtige opdateringstid er nødvendig, fordi det er en meget hurtig motorregulering.

På ældre CPU'er er det i properties for CompactBus Local (ikke vist her).

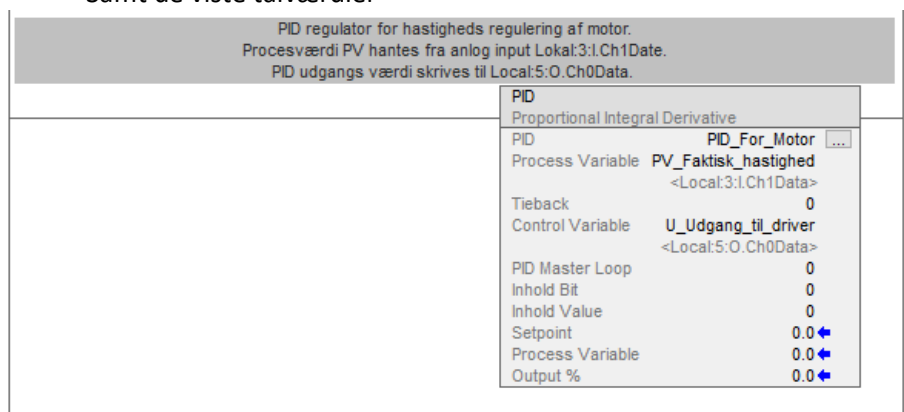


Opret et tag (PID_For_Motor) med datatype PID.




Tilknyt PID-blokkens talværdier og tags som vist nedenfor

- Tag for "PID" – her er brugt tagget "PID_For_Motor"
- Tag for "Process Variable" – her er brugt tagget "PV_Faktisk_hastighed"
- Tag for "Control Variable" – her er brugt tagget "U_Udgang_til_Driver"
- Samt de viste talværdier

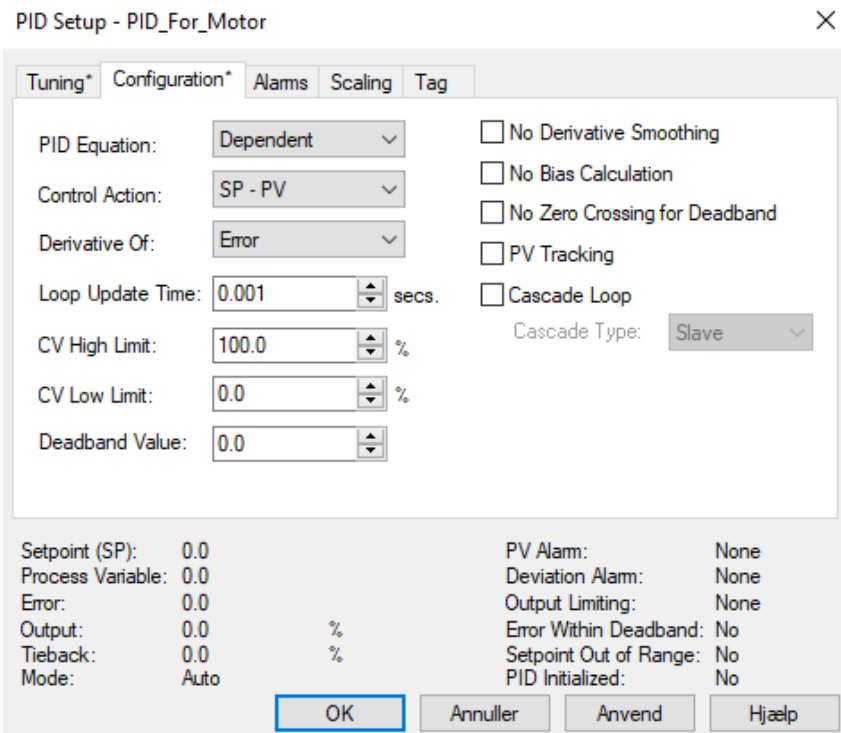


Indstilling af PID-regulatoren

Inden PID-regulatorblokken er klar til idriftsætning, skal der laves nogle indstillinger i den.

Dette gøres ved at trykke på knappen med de tre prikker 

Configuration



PID Setup - PID_For_Motor

Tuning* Configuration* Alarms Scaling Tag

PID Equation: ☐ No Derivative Smoothing

Control Action: ☐ No Bias Calculation

Derivative Of: ☐ No Zero Crossing for Deadband

Loop Update Time: secs. ☐ PV Tracking

CV High Limit: % ☐ Cascade Loop

CV Low Limit: % Cascade Type:

Deadband Value:

Setpoint (SP): 0.0 PV Alarm: None

Process Variable: 0.0 Deviation Alarm: None

Error: 0.0 Output Limiting: None

Output: 0.0 % Error Within Deadband: No

Tieback: 0.0 % Setpoint Out of Range: No

Mode: Auto PID Initialized: No

OK Annuller Anvend Hjælp

Vælg PID Equation Dependent, således at regulatoren følger denne ligning:

$$CV = K_c * \left(e + \frac{1}{T_i} \int_0^t e * dt + T_d * \frac{de}{dt} \right) + \text{BIAS}$$

Det betyder, at værdier beregnet med Z & N kan bruges direkte.

- Vælg "SP -PV" for invers regulering
- Sæt Loop "Update Time" til 1 millisekund, da den skal være lig med den opdateringstid, vi valgte til den periodiske task. Endvidere skal opdateringstiden af den analoge indgang, der bruges til PV, også være den samme
- Sæt "CV High Limit" til 100% og "CV Low Limit" til 0%, således at CPU'en ikke går i fejl, hvis regulatorudgangen bliver for stor eller lille
- Sæt "Derivation Of" til "Error", således at D-ledet arbejder med fejlen og ikke kun PV
- Resten skal blive stående

Scaling

På denne fane skal der angives skaleringer, der passer til dem, der er valgt til de analoge in- og out-put.

PID Setup - PID_For_Motor

Tuning Configuration Alarms **Scaling*** Tag

Process Variable (PV)

Unscaled Max.: 10000.0 Engineering Unit Max.: 10000.0

Unscaled Min.: 0.0 Engineering Unit Min.: 0.0

Control Variable (CV)

Max. (at 100 %): 10000.0

Min. (at 0 %): 0.0

Tieback

Max. (at 100 %): 0.0

Min. (at 0 %): 0.0

☐ PID Initialized

Setpoint (SP): 0.0 PV Alarm: None

Process Variable: 0.0 Deviation Alarm: None

Error: 0.0 Output Limiting: None

Output: 0.0 % Error Within Deadband: No

Tieback: 0.0 % Setpoint Out of Range: No

Mode: Auto PID Initialized: No

OK Annuller Anvend Hjælp

- Sæt "process Variable (PV)" til 10.000, da der er valgt "Percent Range" til den analoge indgang, som PV hentes fra
- Sæt "Engineering Unit Max" til 10.000 og min. til 0, således at SV bliver skaleret på samme måde som PV. Dette betyder, at hvis der ønskes 50 % omdrejninger på motoren, skal SV være lig med 5.000
- Sæt "Control Variabel (CV)" Max til 10.000 og Min. til 0, således at vi kan udnytte hele regulatorudgangens arbejdsområde. Og så det passer til "Percent Range", som blev valgt til det analoge output
- Resten skal blive stående

Tuning

På fanen "Tuning" kan ønsket SP- og PID-parametrene indstilles.

PID Setup - PID_For_Motor

Tuning Configuration Alarms Scaling Tag

Setpoint (SP): 5000.0

Set Output: 0.0 %

Output Bias: 0.0 %

Manual Modes

☐ Manual

☐ Software Manual

Tuning Constants

Proportional Gain (Kc): 2.0

Reset Time (Ti): 0.5 min/repeat

Derivative Rate (Td): 0.0 min

Reset Tuning Constants to the values they had upon entry into the PID Setup dialog

Reset

Setpoint (SP): 5000.0

Process Variable: 0.0

Error: 0.0

Output: 0.0 %

Tieback: 0.0 %

Mode: Auto

PV Alarm: None

Deviation Alarm: None

Output Limiting: None

Error Within Deadband: No

Setpoint Out of Range: No

PID Initialized: No

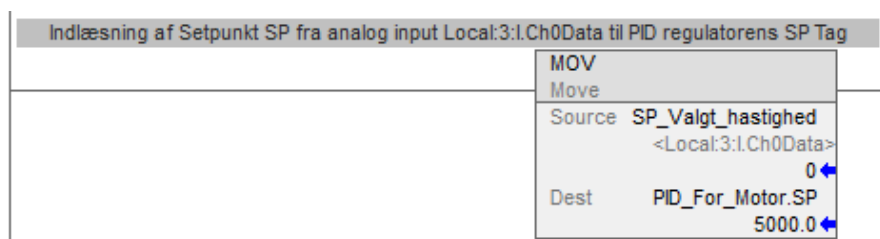
OK Annuller Anvend Hjælp

- "Setpoint (SP):" Sætpunkt eller den værdi, som du ønsker, at PV skal have. Ovenfor er valgt et SP på 50 %, der pga. skaleringer er lig med 5.000
- "Output Bias: %" Kan bruges som udgangstilskud ved ren P-regulering eller i forbindelse med feed-forward regulering, hvor en feed-forward værdi indlæses i tagget (PID_For_Motor.BIAS)
- "Proportional Gain:" Forstærkning - Kp eller Kc. Ovenfor er der valgt en forstærkning på 2 gange
- "Reset Time (Ti):" I-tiden i minutter. Ovenfor er der valgt en I-tid på 0.5 minut, det vil sige 30 sekunder. Ti = 0.0 sætter I-leddet off.
- "Derivative Time (Td):" D-tiden i minutter. Ovenfor er D-leddet sat på off, da der er valgt 0.0
- "Manual Modes" Vælg enten "Manuel" eller "Software Manual". PID-regulatorens udgang kan i disse modes styres ved hjælp af "Output Bias:" eller "Set Output:"
- "Reset" Klik på denne knap for at nulstille forstærkningsværdierne til de værdier, der var gældende, da dialogboksen blev åbnet. Gem værdierne ved at trykke "Anvend"

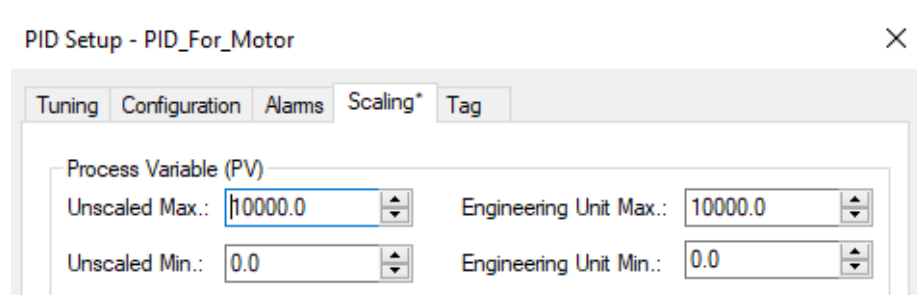
Eksternt setpunkt

I nogle tilfælde kan der være ønske om af hente setpunkt fra et HMI-panel eller en analog indgang.

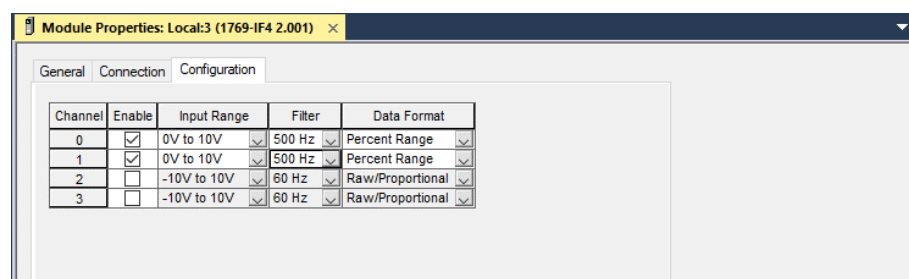
Dette kan gøres ved at "Move" en værdi ind i PID-blokkens tag "?SP". Herunder er en analog indgang flyttet ind i tagget "PID_For_Motor.SP".



Når man bruger en analog indgang til setpunkt skal man være opmærksom på, at den skal være skaleret, så den passer med den skaleringsværdi, man valgte på PID-blokkens "Scaling"-fane under "Engineering Unit".



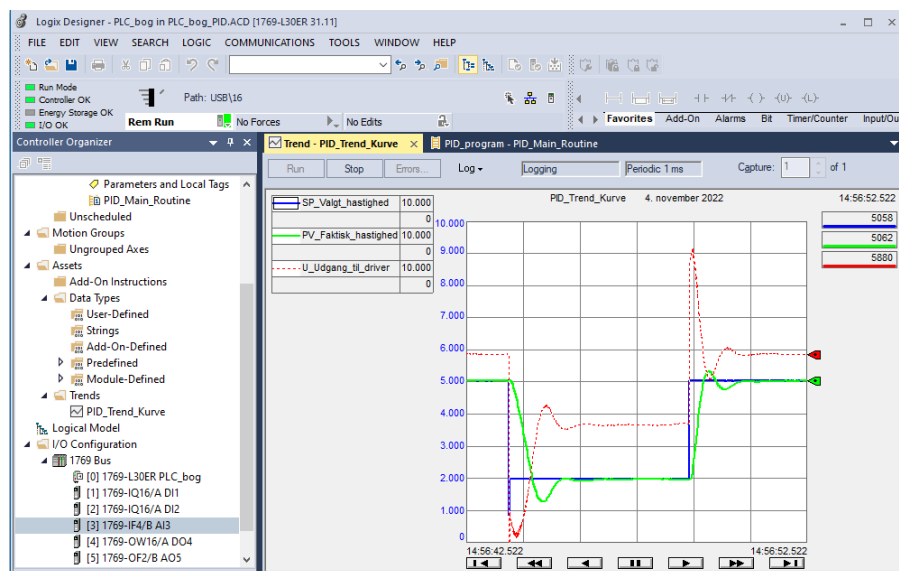
I dette eksempel blev der angivet 0 til 10.000 som "Engineering Unit", og derfor skal den analoge indgang opsættes i dens "Configuration"-fane til at benytte "Data format" → "Percent Range", da "Percent Range" også arbejder i området fra 0 til 10.000.



Henter man setpunkt værdien fra en HMI-skærm, kan andre skaleringsværdier bruges, når man blot ændrer "Engineering Unit".

PID-trendkurver

Når man skal optimere en PID-regulator er trendkurver over setpunkt "SP", procesværdi "PV" og evt. fejlen "e" et værdifuldt værktøj. "Studio 5000 Logix Designer" har en indbygget funktion, der gør det muligt at se trendkurver.

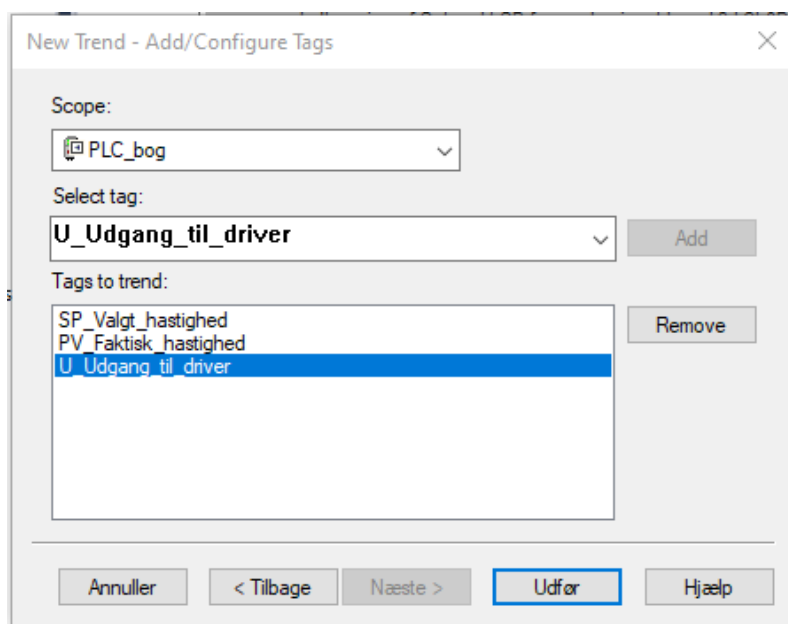


Opret Trends

Højreklik på mappen "Trend" i venstre side af skærmen og vælg "New Trend"

- Giv den et navn her "PID_Trend_Kurve"
- Sæt en "Sample Period", der er noget kortere end processens reaktionstid.
- Da denne kurve skal bruges til en meget hurtig motorhastighedsregulering, er den her sat til 1 millisekund. Tryk næste.

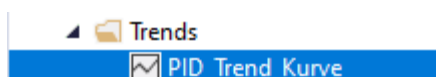
Vælg tag-navnene på de værdier, man vil have vist på trendkurven. Når dette er gjort, trykkes der udfør.



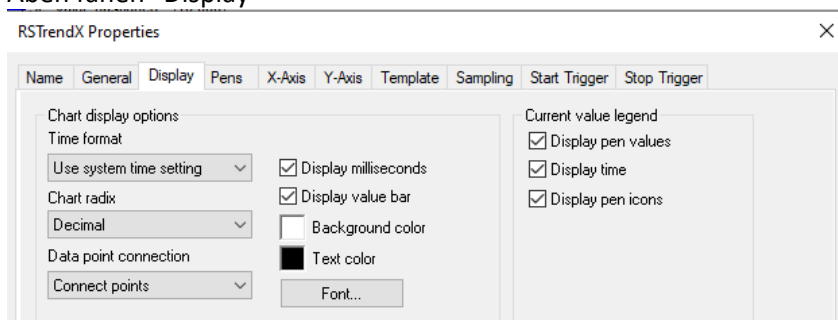
Tilpasninger af trendkurven til proces

Inden trendkurver kan idriftsættes, er der lige lidt tilpasninger, der skal laves.

- Højreklik på den nye PID_Trend_Kurve i "Trends"-mappen i venstre side af skærmen. Vælg "Properties"



Åben fanen "Display"



- Vælg "Display milliseconds", så det passer til den meget hurtige motorproces. Er det en langsom proces, er dette ikke nødvendigt
- Vælg evt. også hvid som baggrundsfxve

Vælg fanen "X-Axis".

RSTrendX Properties

Name General Display Pens **X-Axis** Y-Axis Template Sampling Start Trigger Stop Trigger

Chart time range

Start date: 03-11-2022

Start time: 11:40:46

Time span: 10 Second(s)

Display options

☒ Display scale

☐ Display date on scale

☒ Display grid lines

5 Major grid lines

0 Minor grid lines

Grid color

OK Annuller Anvend Hjælp

- Sæt "Time span" til 10 sekunder – da denne kurve skal bruges til en meget hurtig motorhastighedsregulering, er det nok.
- Indstil X-aksens tidsvisning så den er let at læse på skærmen. Dette gøres ved at prøve sig frem med forskellige indstillinger for "Major grid lines" og trykke anvend.

Vælg fanen "Y-Axis"

RSTrendX Properties

Name General Display Pens X-Axis **Y-Axis** Template Sampling Start Trigger Stop Trigger

Minimum / maximum value options

☐ Automatic (best fit based on actual data)

☐ Preset (use min/max setting from Pens tab)

☒ Custom

Minimum value

☒ Actual minimum value: 0

Maximum value

☒ Actual maximum value: 10000

Display options

☐ Isolated graphing

☒ Display scale

☒ Display grid lines

Grid color

Scale options

☐ All pens on same scale

☒ Each pen on independent scale

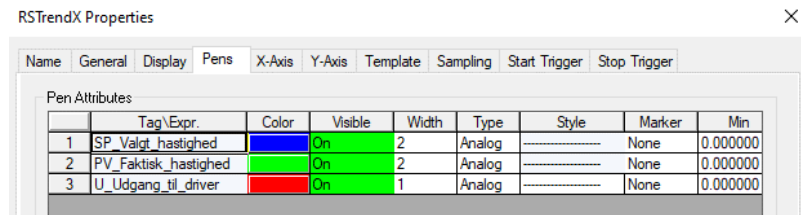
☐ Scale using pen

☒ Scale as percentage

OK Annuller Anvend Hjælp

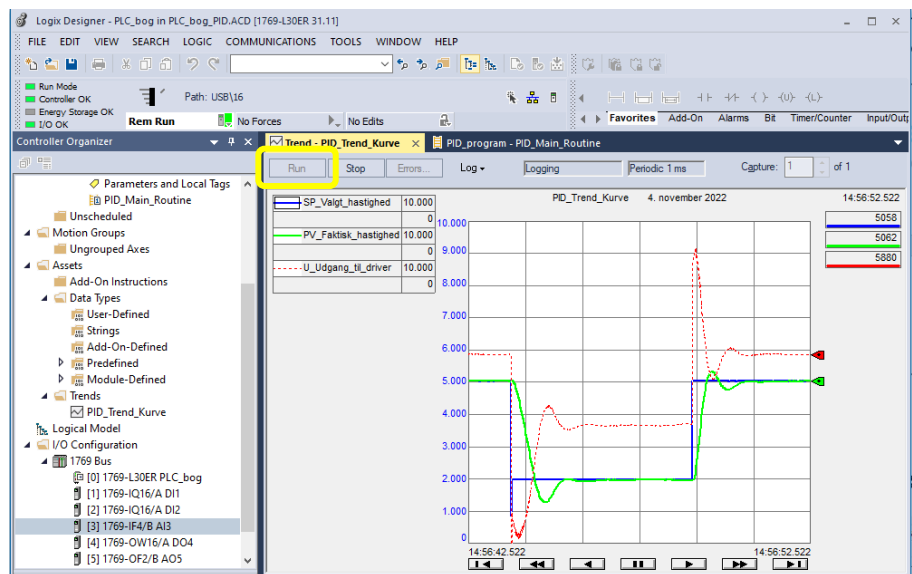
- Vælg "Custom" og angiv min. og maks. til 0 til 10.000
- Sæt et ✓ i "Scale as percentage"

Vælg evt. passende pen-farver og streg -trykkelse og -typer i fanen "Pens".



Online trendvisning

Gå online på PLC'en og tryk på "Run" for at optage trendkurver.





PLC-Programmering

Denne bog kan anvendes til undervisning af automatiktekniker- og elektrikerlærlinge, AMU-kursister og andre i PLC.

Bogen tager udgangspunkt i programmering af:
Siemens S7-1200 ved hjælp af TIA Portal og Allen-Bradley CompactLogix ved hjælp af Studio 5000 Logix Designer.

Bogen startet helt fra bunden med spørgsmålet: "Hvad er en PLC". Bogen dykker herefter ned i emner som:

- Bestykning og adressering
- Grundlæggende programmering
- Sekvensteknik
- Ord-instruktioner
- UDT'er og parametrerbare blokke
- Analog signalbehandling
- PID-regulatoren

Bogen berører også emnerne: Talsystemer, I/O-kredsløb, digitale følere, EMC, PLC- installation, idriftsætning og fejlfinding.

Bogen kan hentes gratis på www.videncenterportalen.dk/arn og må frit kopieres og redigeres.