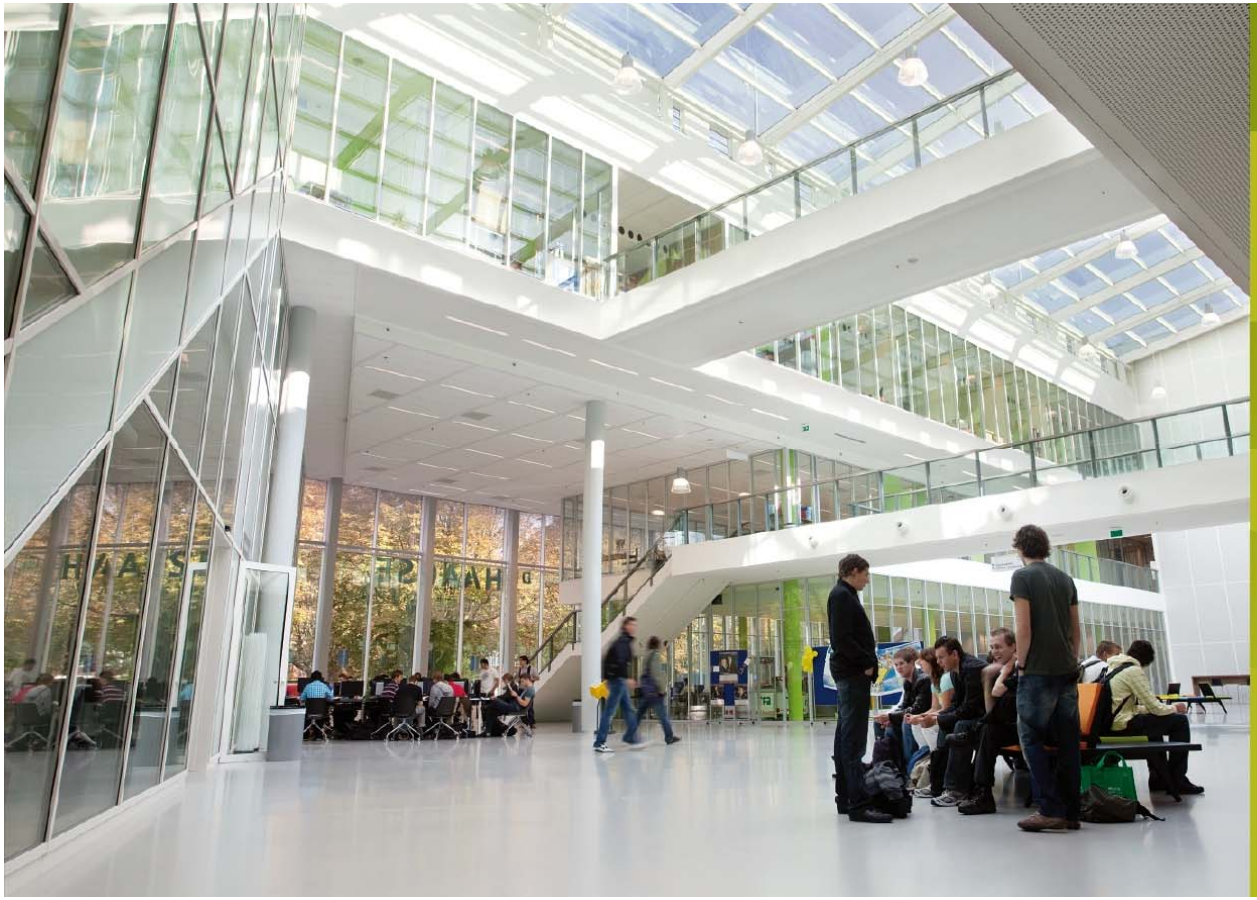


Academie voor Technology, Innovation & Society Delft
Academie voor ICT & Media

Besturingstechniek

Inleiding S7-300 PLC en programmeren

J.E.J. op den Brouw/B. Kuiper

PLCTEC/2018-2019

DE HAAGSE HOGESCHOOL

PLCTEC (1)

- PLCTEC-co1 + PLCTEC-pr1 = 2 ECTS = 56 SBU
- 1 lesuur per week theorie
- 2 lesuren per week practicum
- Theorie:
  - Schriftelijke toets met m.c. vragen beoordeeld met cijfer.
- Practicum:
  - 2 opdrachten voor het bekend worden met Siemens apparatuur en software.
  - Andere en laatste opdracht: Automatiseren productiestraat
  - Beoordeeld met Voldoende/Onvoldoende.
  - Aanwezigheid verplicht.

PLCTEC (2)

- Boeken:
  - Er wordt geen boek gebruikt, maar wel:
  - Deze slides
  - Een aantal pdf’s die in deze slides staan vermeld.

Les 1

- Introductie PLC
- Siemens S7 serie
- PLC “Programmeertalen”
- Data types
- Adressering

PLC

- PLC = Programmable Logic Controller
- Een PLC is een computer met gespecialiseerde hardware voor aansturing van o.a. industriële productieprocessen
  (= industriële automatisering)

PLC (2)

- PLC’s zijn real-time systemen.
- Een PLC kan relatief eenvoudig worden geprogrammeerd m.b.v. verschillende talen die weer bepaalde standaarden volgen.
- Een PLC kan gekoppeld worden aan andere (ERP) systemen.

PLC (3)

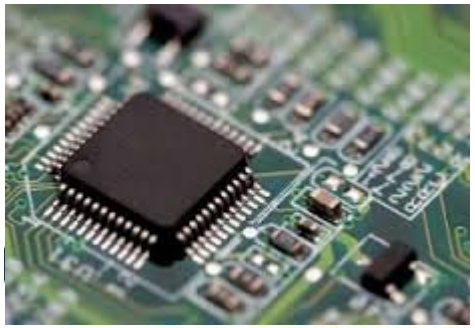
- Er zijn meerdere fabrikanten
  - Allen Bradley
  - Omron
  - Siemens
  - Hitachi
  - Modicon
  - ABB

- Allen Bradley wordt veel in Amerika en Azië gebruikt

- Siemens wordt veel in Europa gebruikt

PLC (4)

- PLC versus microcontroller:
  - PLC hardware en software is duurder.
  - Ontwikkeling van PLC software gaat daarentegen relatief snel en is dus vrij goedkoop.
  - Minder hardware-ontwikkeling nodig bij een PLC, omdat vele kant-en-klare modules te koop zijn voor allerlei sensoren en actuatoren.
  - PLC hardwaremodules zijn al gecertificeerd (en dus uitgebreid getest). Bij een microcontroller niet, want daar zul je veel van de hardware zelf moeten ontwerpen.




SIEMENS
SIMATIC PANEL PC
SIEMENS
SIMATIC C7
Start
Metal Press
Stop
SIEMENS
SIEMENS

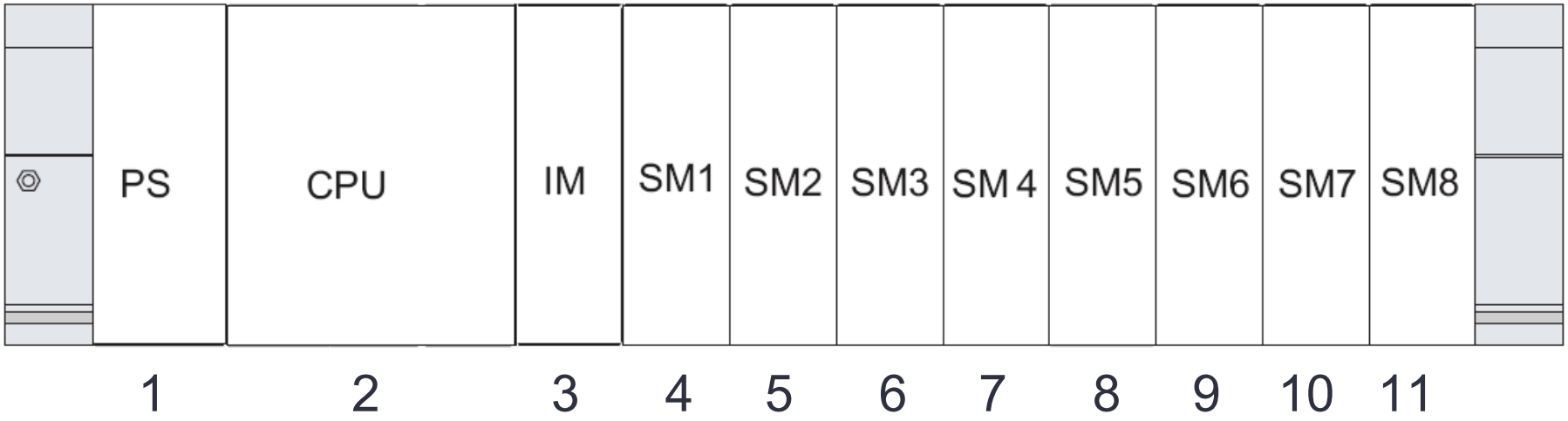
Siemens S7 Modular Controller serie

- De S7 serie bestaat weer uit o.a. de volgende series:
  - S7-1200 (voor onderkant van de markt)
  - S7-1500 (voor bovenkant van de markt)
- Iets oudere series zijn o.a. S7-200, S7-300 en S7-400
  - S7-300 was een aantal jaren geleden bedoeld voor het middensegment
- PLC's zijn modulair opgebouwd rond een CPU-module
- In het lab:
  - CPU 315F-2 PN/DP
  - CPU 313C-2 DP (voor leuke projecten)

CPU315 met I/O

DE HAAGSE HOGESCHOOL

Module/slot-adressing



Slot 1: Power supply
Slot 2: CPU module
Slot 3: Interface module
Slot 4-11: Signaalmodules

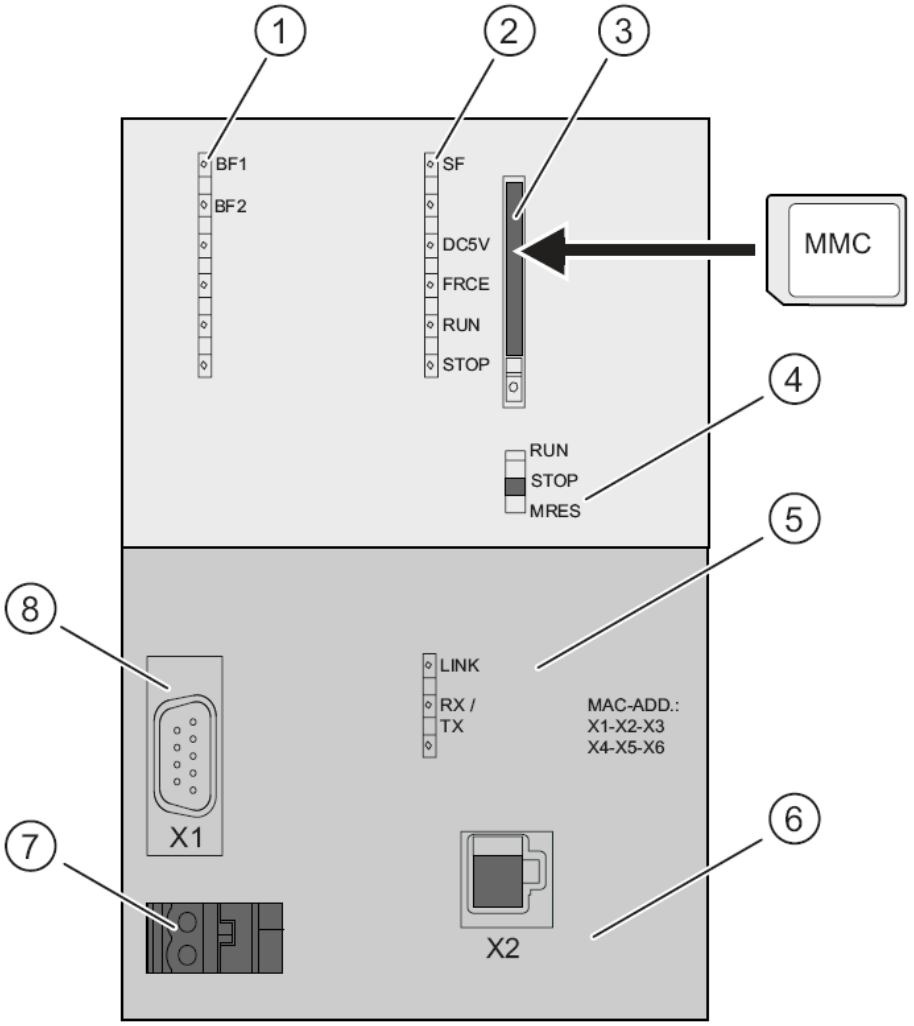
NB: Soms zijn CPU en IM gecombineerd.

Siemens S7 serie

- CPU315F-2 PN/DP
  - 256 KB werkgeheugen
  - 512 KB flashgeheugen
  - 0.1 µs instructietijd voor bitoperaties, 3 µs voor floating point
  - 2048 I/O adressen
  - Losse DI/DO (16/16 kanalen), AI/AO (4/2 kanalen) en simulatiemodule
  - Profibus/Profinet aansluiting
  - Failsafe mogelijkheden

- CPU313C-2 DP
  - 32 KB werkgeheugen
  - 512 KB flashgeheugen
  - 0.1 µs instructietijd voor bitoperaties, 3 µs voor floating point
  - 1024 I/O adressen (16/16 geïntegreerd), max 128 in PI
  - Losse 230V output module (8 kanalen)
  - Profibus aansluiting

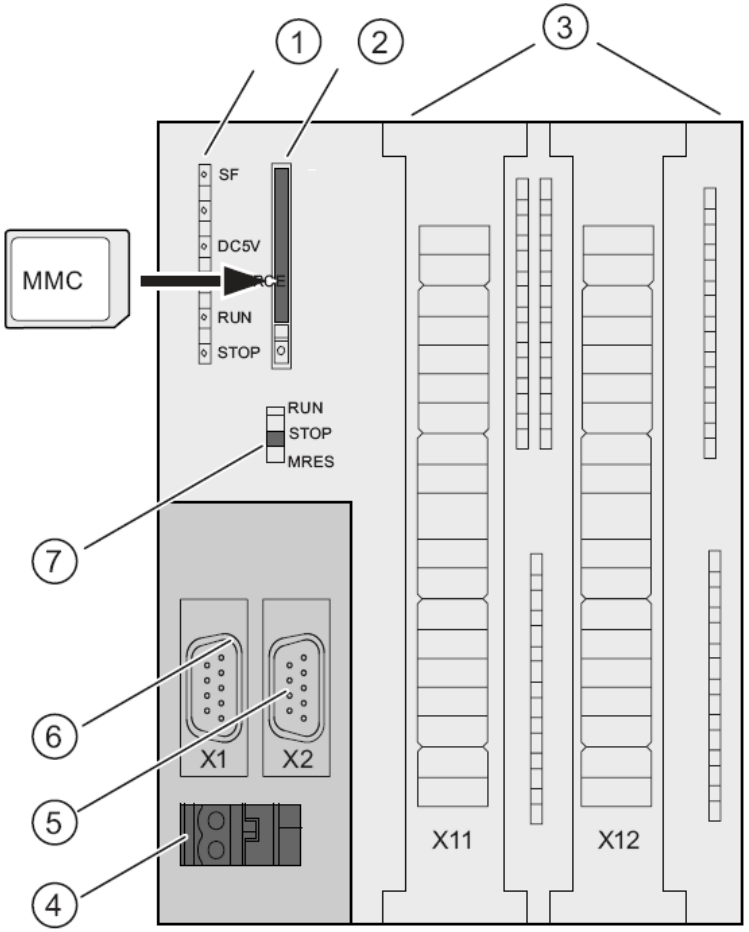
Bedieningselementen 315F-2 PN/DP



- 1 Busfoutmeldingleds
- 2 Status- en foutmeldingleds
- 3 MMC slot
- 4 Werkmodus-selector
- 5 Statusleds Profinet interface
- 6 Profinet interface
- 7 Voedingaansluiting
- 8 MPI/Profibus interface

DE HAAGSE HOGESCHOOL

Bedieningselementen 313C-2 DP



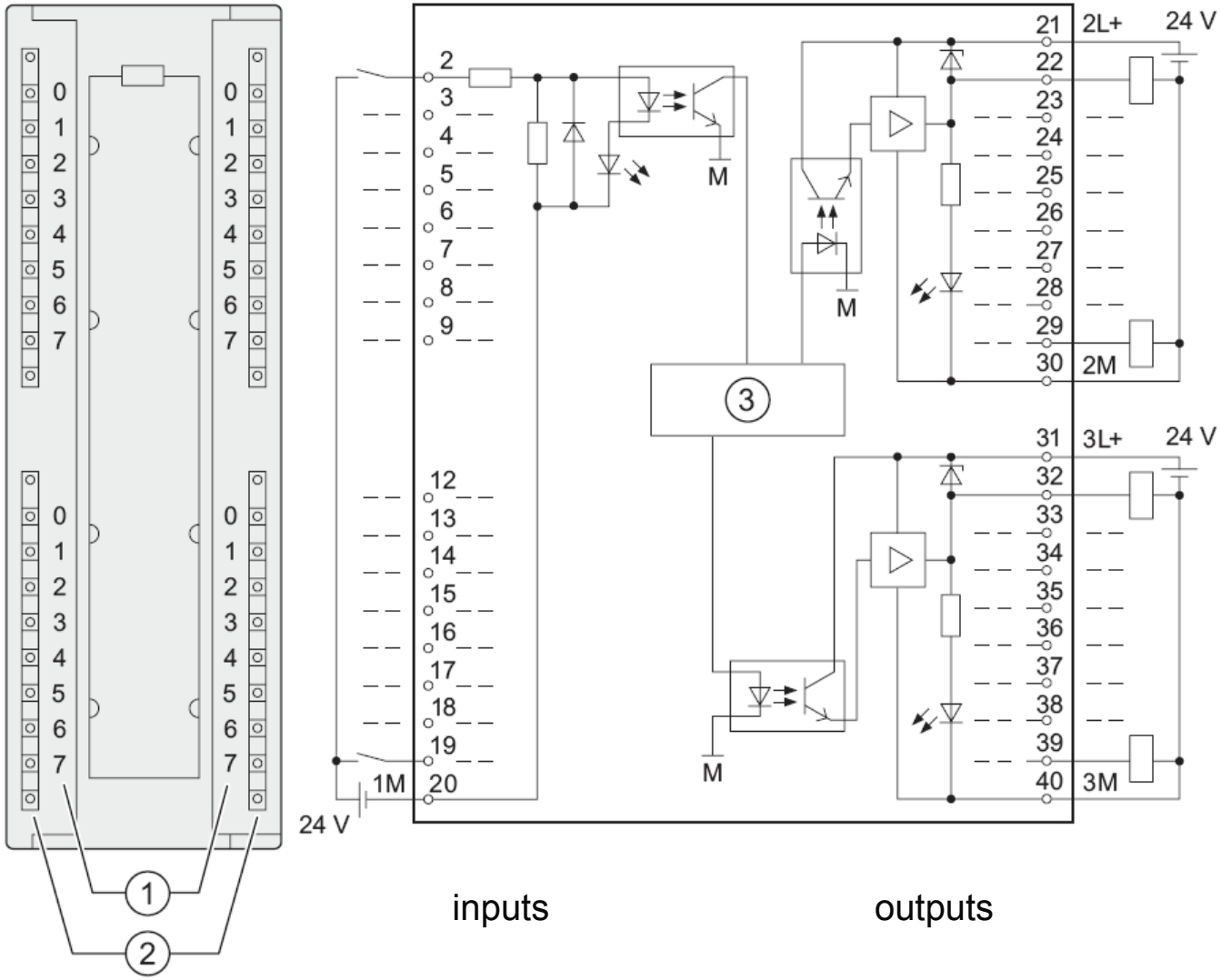
- 1 Status- en foutmeldingleds
- 2 MMC slot
- 3 DI/DO
- 4 Voedingaansluiting
- 5 Profibus interface
- 6 MPI/Profibus interface
- 7 Werkmodus-selector

DE HAAGSE HOGESCHOOL

Digitale I/O module SM323 DI16/DO16xDC24V

- 16x digitaal in (0 = 0V, 1 = 24V)
- 16x digitaal uit (0 = 0V, 1 = 24V)
  - 0,5 A per uitgang
- Geen diagnostische interrupts

Digitale I/O module SM323 DI16/DO16xDC24V



1 = kanaalnummer

2 = leds

3 = backplane bus

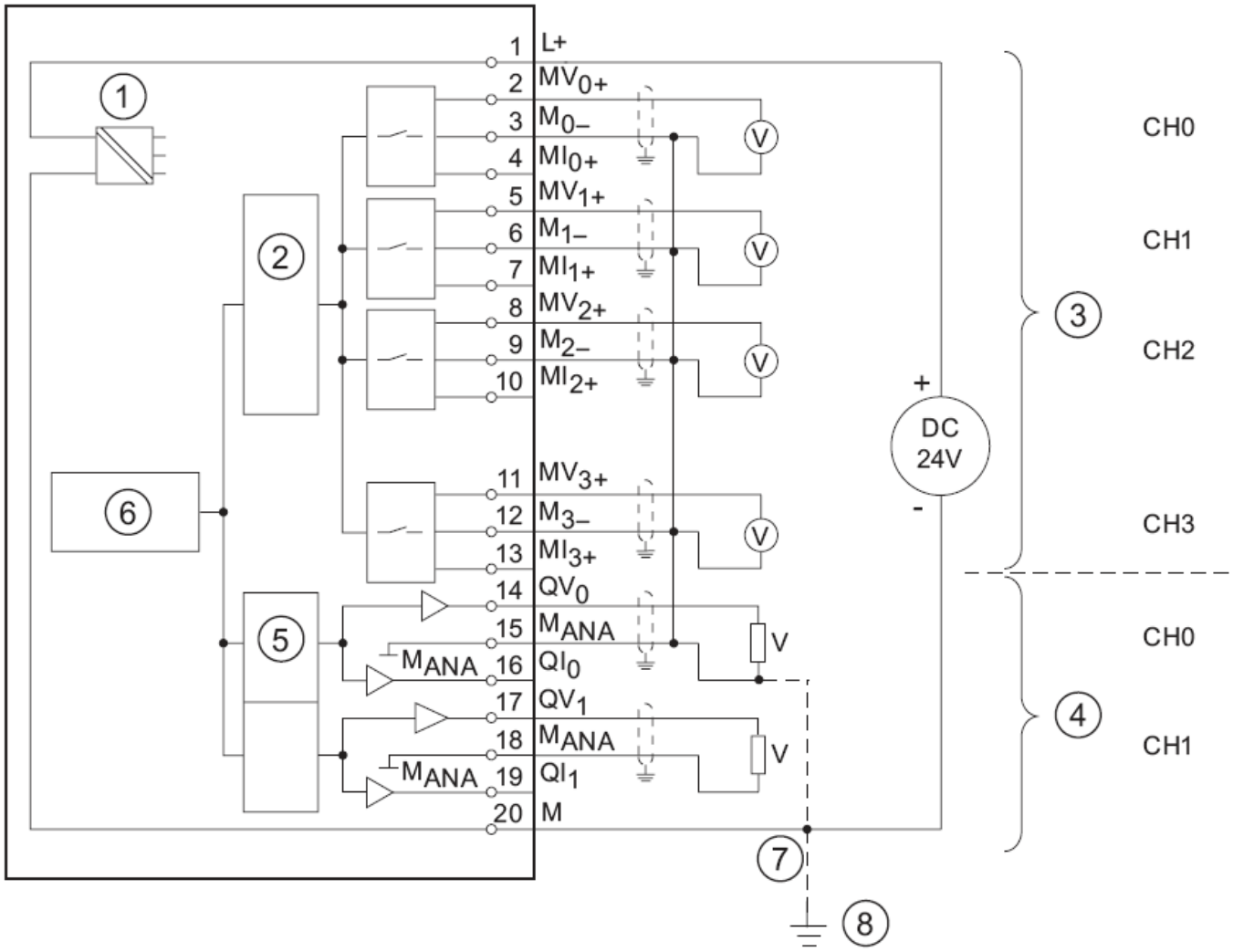
inputs

outputs

Analoge I/O module SM 334 AI4/A02x8BIT

- 8 bits resolutie, 256 stappen
- 4x analoog in, 2x analoog uit
- Spanningsingang/uitgang: 0 V – 10 V
- Stroomingang/uitgang: 0 mA – 20 mA
- Geen diagnostische interrupts

Analoge I/O module SM 334 AI4/A02x8BIT

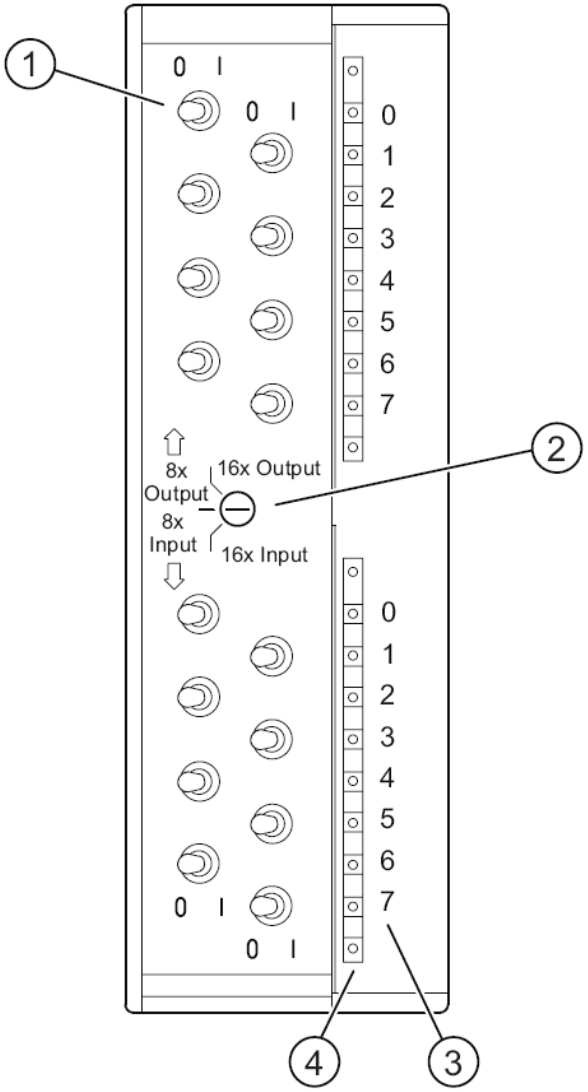


1 = internal supply
2 = ADC
3 = voltage inputs
4 = voltage outputs
5 = DAC
6 = backplane bus
7 = equi. bonding
8 = functional earth

Simulation Module SM 374

- Wordt gebruikt voor handmatige invoer en uitvoer.
- 8x digitaal in, schakelaars
- 8x digitaal uit, leds
- Geen diagnostics
- Er zit een mode selector op de SM 374, daarmee kan de werkmodus worden ingesteld
  - 16x input
  - 16x output
  - 8x input, 8x output, tijdens practicum

Simulation Module SM 374



1 = schakelaars

2 = mode selector

3 = kanaalnummer

4 = leds

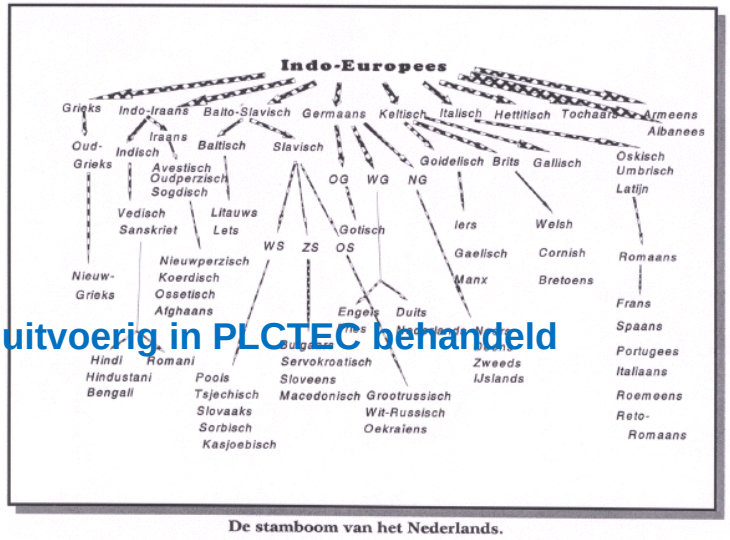
Noot: outputs zitten boven!

Programmeertalen

- De PLC (of eigenlijk CPU) kan geprogrammeerd worden met verschillende “programmeertalen”.

Volgens IEC 61131-3:

- Ladderdiagram (LAD, LD)
- Functieblokken (FBD, FBD)
- Assembler (STL*, IL)
- Grafisch, toestandsmachine (Graph, SFC)
- Gestructureerde programmeertaal (SCL, ST)



De stamboom van het Nederlands.

Wordt uitvoerig in PLCTEC behandeld

Andere programmeertalen worden kort toegelicht/behandeld

* Niet volgens IEC-norm

DE HAAGSE HOGESCHOOL

Voorkennis voor programmeren

Voordat we gaan programmeren in een taal gaan we eerst kijken naar een aantal zaken die in zijn algemeenheid gelden:

- Datatypes
- Adressering
- Software blokken
- PLC operating modes
- Cyclische uitvoering programma
- Opbouw PLC programma

Volgende les

DE HAAGSE HOGESCHOOL

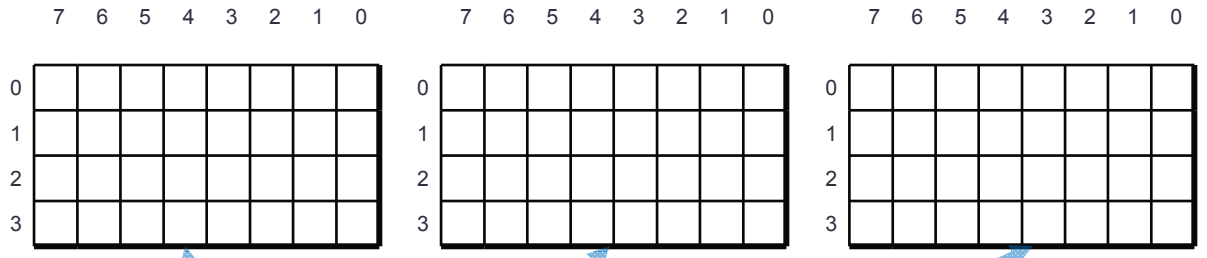
Datatypen

- De PLC kan de volgende simpele datatypen aan:
- Bit (bool)
- Byte
  - 8 bits, 0 – 255
- Woord (word)
  - 16 bits, 0 – 65535
- Dubbelwoord (dword)
  - 32 bits, 0 - 4294967295

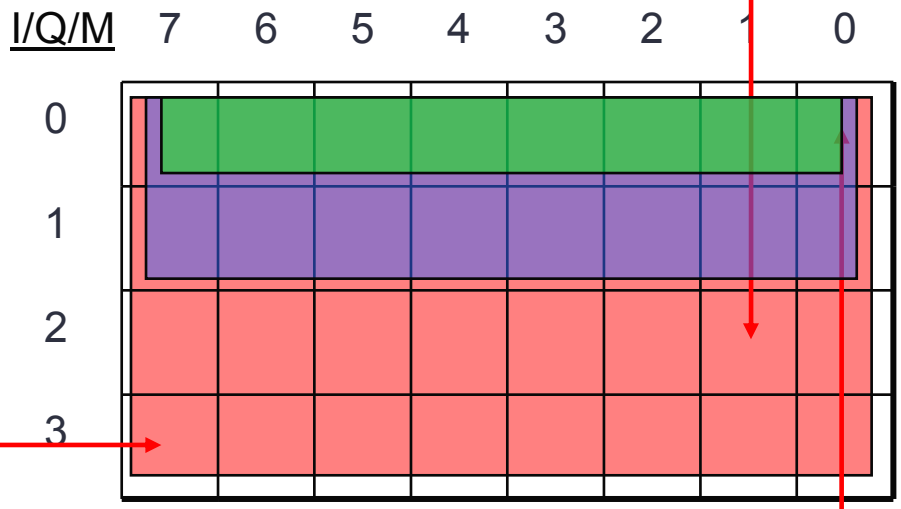
Char, integer, dubbel integer, floating point

- Character (char)
  - 8 bits, ASCII-teken
- Integer (int)
  - 16 bits, -32768 - 32767
- Dubbel integer (dint)
  - 32 bits, -2147483648 - 2147483647
- Floating point (real)
  - 32 bits, -3.402822E+38 - 3.402822E+38
  - Kleinste getal anders dan 0.0: ±1.175495E-38

Adressering



- De PLC heeft diverse soorten I/O:
- Ingangsregister (I)
  - I 2.1
  - IB 37
- Uitgangsregister (Q)
  - Q 3.7
  - QW 18
- Merker (M)
  - M 0.0
  - MD 64



Adressering (2)

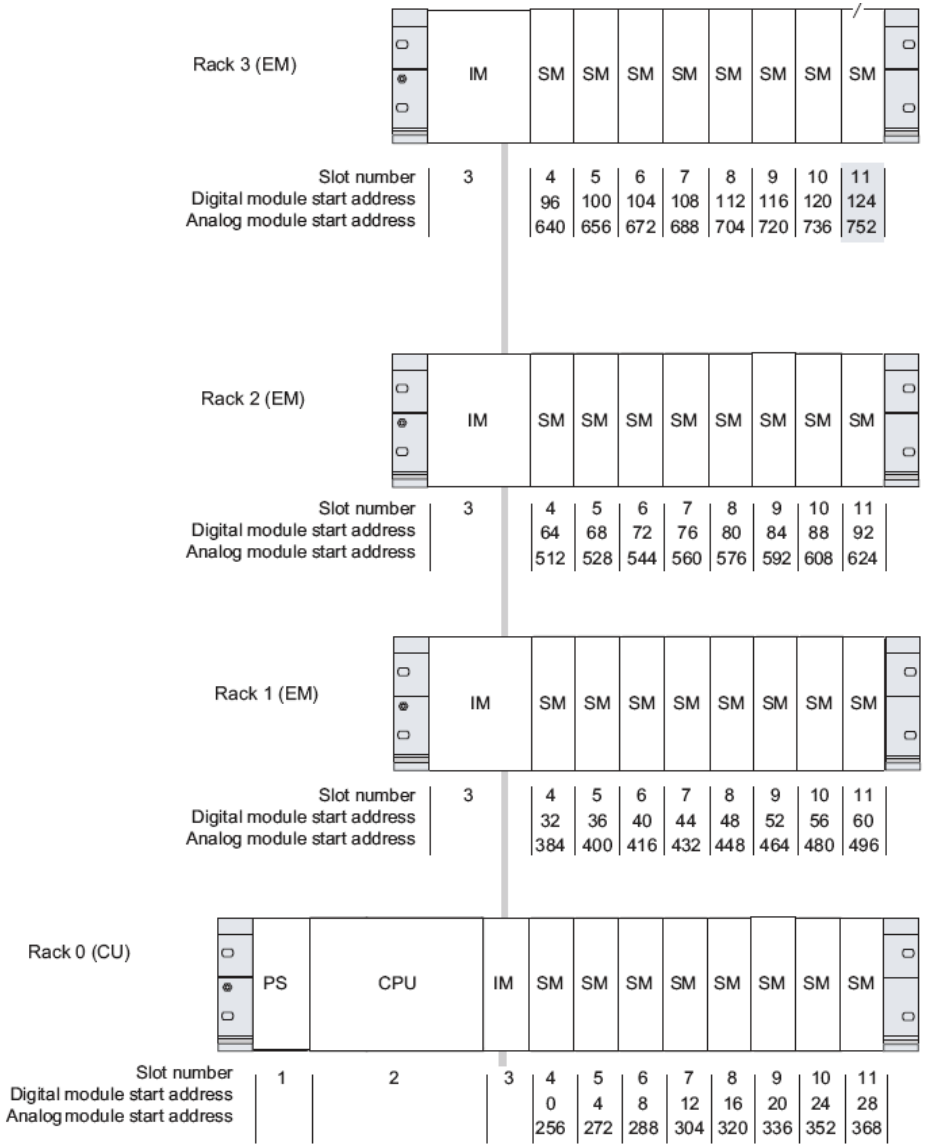
- Counter (C), timer (T)
  - C1, T2
- Peripheral (P)
  - Peripheral access is een vorm waarbij direct met de I/O modules wordt gewerkt, zonder gebruik te maken van de *Process Image.*
  - Process Image: aan het begin van de scan cycle wordt een kopie gemaakt van de huidige ingangswaarden op de I/O modules en consistent gehouden tijdens de scan cycle. Outputs die gewijzigd worden, zullen pas aan het einde van de scan cycle uitgevoerd worden.
    - De I en Q geheugens zijn dus eigenlijk het process image
- Voorbeeld gebruik Peripheral (P):
  - PIW512 – deze moet je bv. gebruiken bij analoge modules
  - PQW512

Byte order

- De S7-300 PLC maakt gebruik van Big Endian byte order.
- Van een variabele is de meest significante byte op de laagste geheugenplaats afgebeeld.

vb: MD20, bit 0 = MW22, bit 0 = MB23, bit0 = M23.0

Rack- en I/O-adressering



- S7-300 heeft maximaal 4 racks.
- Eerste SM in rack 0 heeft adressen 0.0 t/m 3.7, tweede 4.0 t/m 7.7 voor digitale modules.
- Eerste SM in rack 0 heeft adressen 256 t/m 270, tweede 272 t/m 286 voor analoge modules.

DE HAAGSE HOGESCHOOL

Zelfstudie

- Hoofdstuk 2 en 3 uit *Tutorial Siemens PLC,* J.E.J. op den Brouw
- Achtergrondinformatie: Hoofdstuk 3 uit http://www.patchn.com/images/articles/plc/plcbook/plcbook5_0.pdf
- En natuurlijk deze slides…

Les 2

- Programmeerblokken
- Operating modes
- Scan cycle
- Communicatieprotocollen

Programmeerblokken

- Organization Block (OB)
- Function (FC)
- Function Block (FB)
- Data Block (DB)
- System Function (SFC)
- System Function Block (SFB)

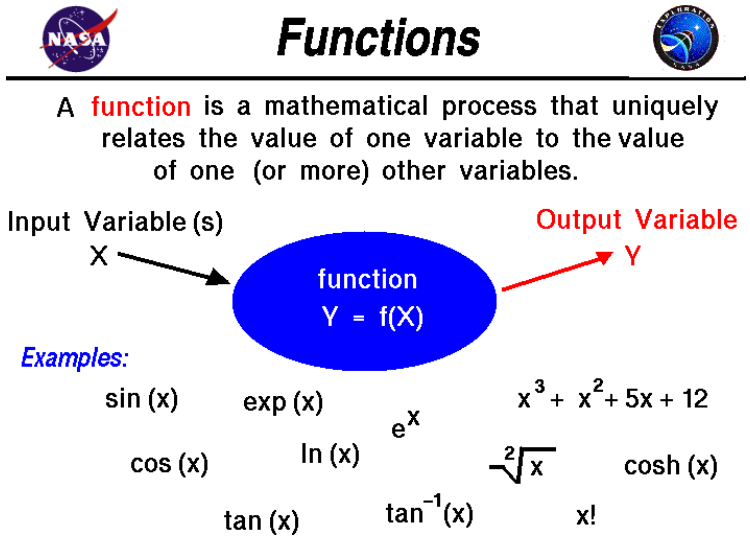
Organization Block (OB)

- Organization Blocks zijn de koppelingen tussen het O.S. van de PLC en het gebruikersprogramma.

- Worden aangeroepen door het O.S. voor diverse soorten handelingen:
  - Main program scan (OB1)
  - Time-of-day interrupts (bijv. OB10)
  - Time-delay interrupts (bijv. OB20)
  - Cyclic interrupts (bijv. OB35)
  - Hardware interrupts (OB40 t/m OB47)
  - Asynchronous errors (OB80 t/m OB87)
  - Synchronous errors (bijv. OB120)
  - Startup (bijv. OB100)

Function (FC)

- Functions bevatten routines/netwerken
- Functions:
  - Hebben een return-waarde.
  - Kunnen parameters meekrijgen.
  - Hebben *geen* eigen data blok.



- Een FC is dus *enigszins* vergelijkbaar met een C-functie, maar dan een functie zonder static/statische variabelen.

Function Block (FB)

- Function Blocks bevatten routines, net als Functions.
- Function Blocks:
  - Hebben geen return-waarde, wel een out of in-out
  - Kunnen gegevens behouden over aanroepen heen
    - Statische variabelen
  - Zijn gekoppeld aan een Data Block
    - Wordt een Instance Data Block genoemd
    - Wordt gebruikt bij parameter-overdracht
    - Bij aanroep van FB dus DB opgeven

Data Block (DB)

- Een Data Block wordt gebruikt voor data-opslag.
- DB's kunnen ook complexe datatypen aan:
  - Array's
  - Structures
  - Date_and_time
- Twee verschillende type DB's:
  - Shared DB
    - Gemeenschappelijk gebruik, andere programma-bouwstenen kunnen hier data uit halen of data inzetten.
  - Instance DB
    - Gekoppeld aan een FB

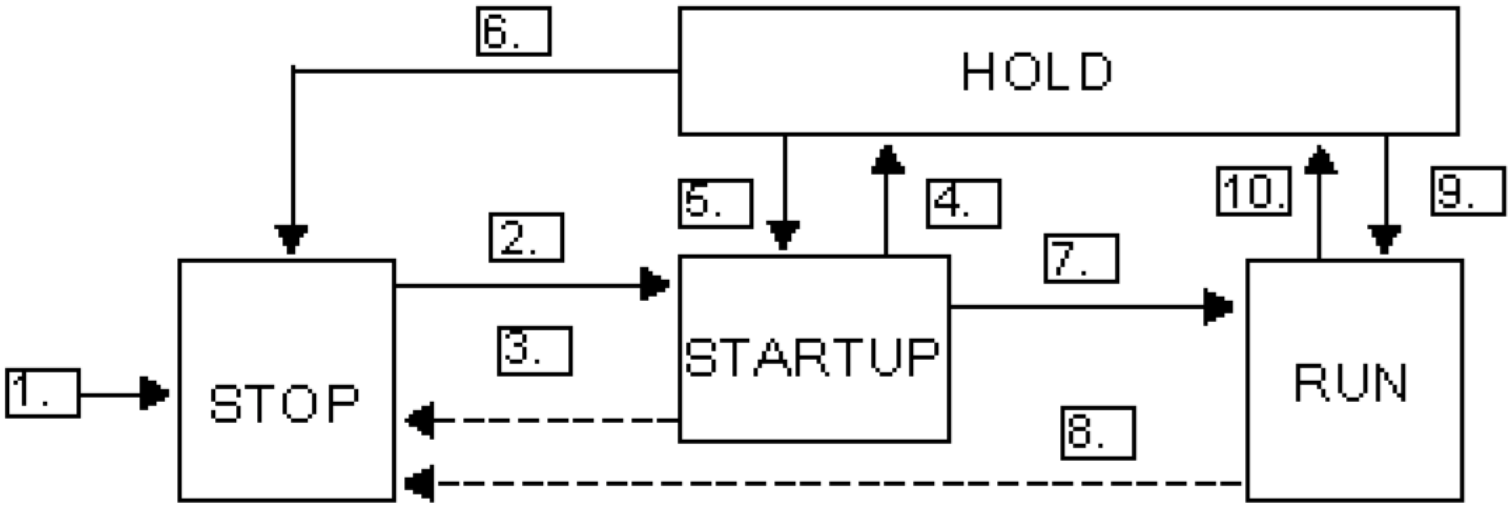
System Function (SFC)

- SFC is vergelijkbaar met FC's.
- Dit is een FC die in het O.S. van de PLC ingebakken zit.
- Deze SFC's worden gebruikt voor OS specifieke calls of veel voorkomende routines.
- Voorbeeld: SFC 1 – read PLC clock
  - Je kan hiermee de tijd en datum opvragen.

System Function Block (SFB)

- SFB's zijn vergelijkbaar met FB's
- Dit is een FB die in het O.S. van de PLC ingebakken zit.
- Deze SFB's worden gebruikt voor OS specifieke calls of veel voorkomende routines.
- Voorbeeld: SFB 2 – IEC up/down counter
  - IEC counters kunnen veel verder tellen dan de standaard Siemens S5 tellers
- Aan een SFB wordt een DB gekoppeld.

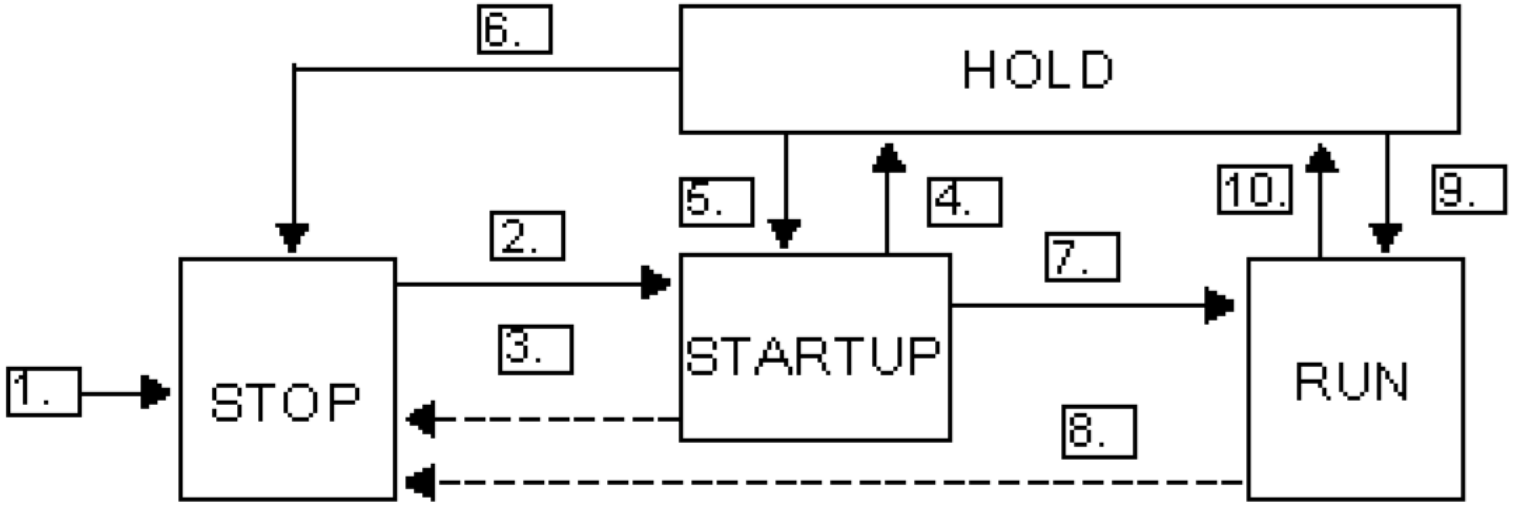
PLC operating modes



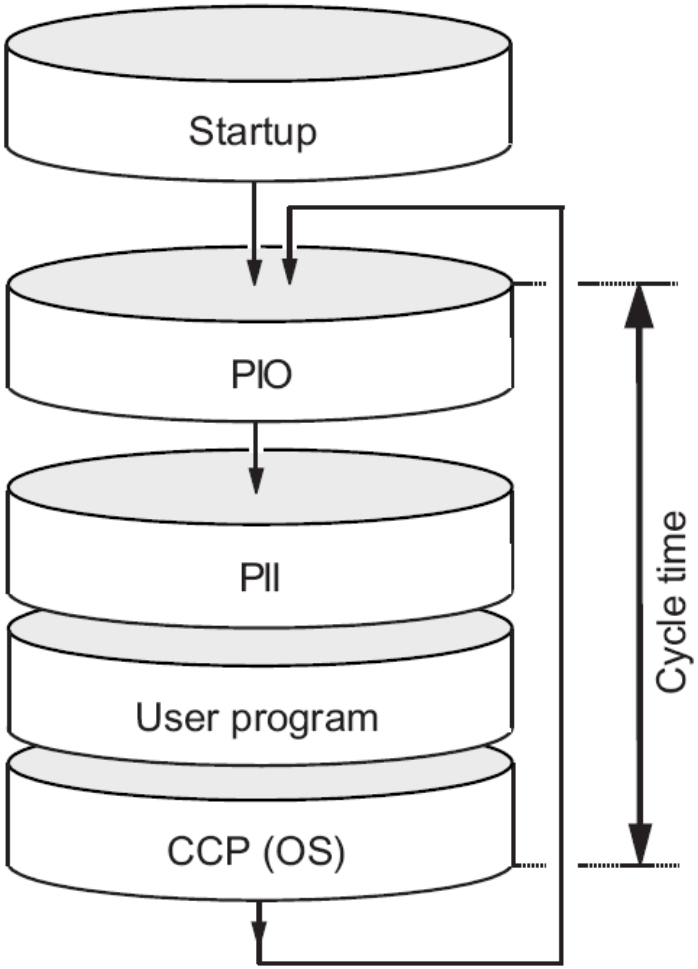
- STOP: de PLC test de configuratie en zet de I/O op een opgeven standaard waarde
- STARTUP: warm en koud, I/O wordt gewist
- RUN: het programma draait, alle I/O wordt bijgewerkt
- HOLD: debug mode, alleen via PG

PLC operating modes

1. Na power-up
2. De key-switch wordt op RUN gezet
3. Indien er een onherstelbare fout plaats vindt
4. PLC gaan in HOLD na een break point
5. Terugkeer uit HOLD
6. De key-switch op STOP wordt op STOP gezet
7. Na de STARTUP-fase gaat de PLC in RUN
8. Indien er een onherstelbare fout plaats vindt of key-switch op STOP
9. Zie 5
10. Zie 4



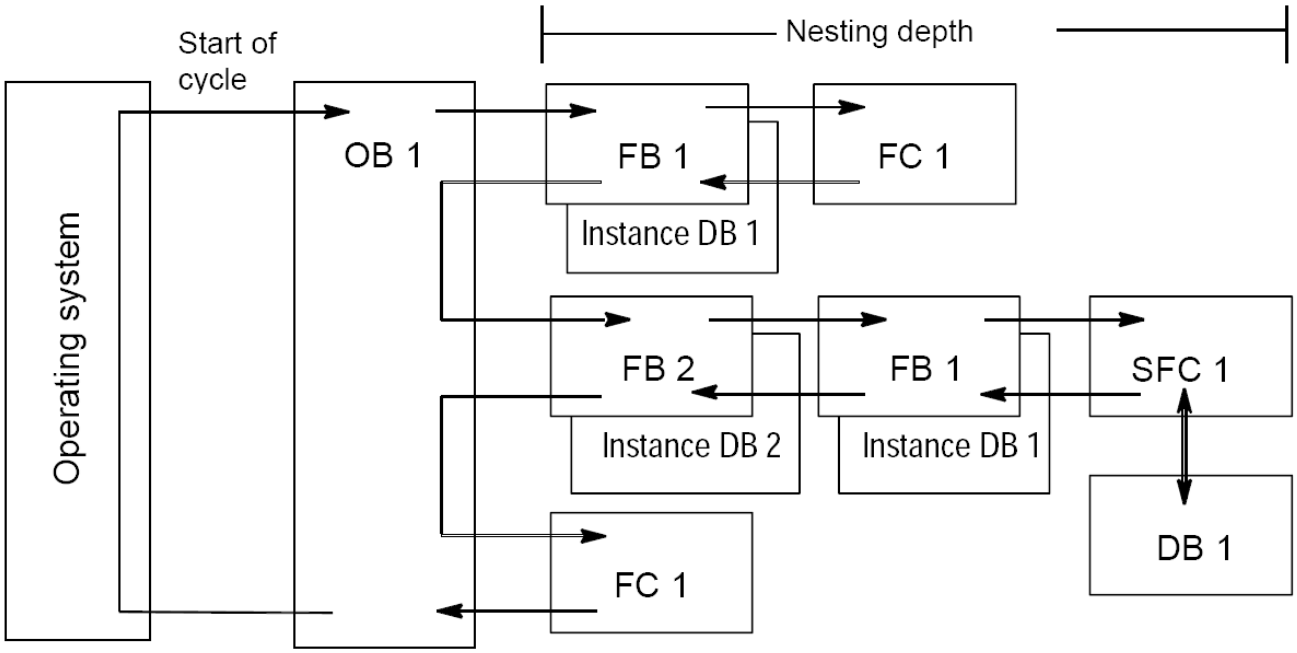
Programma-uitvoer



- Na opstarten wordt het PLC programma cyclisch uitgevoerd.
- Eerst worden de outputs bijgewerkt, dan de inputs.
- Het gebruikersprogramma draait daarna **één keer**.
- Het O.S. van de PLC doet daarna nog allerlei zaken.
- De cyclus moet binnen een bepaalde tijd afgerond zijn.

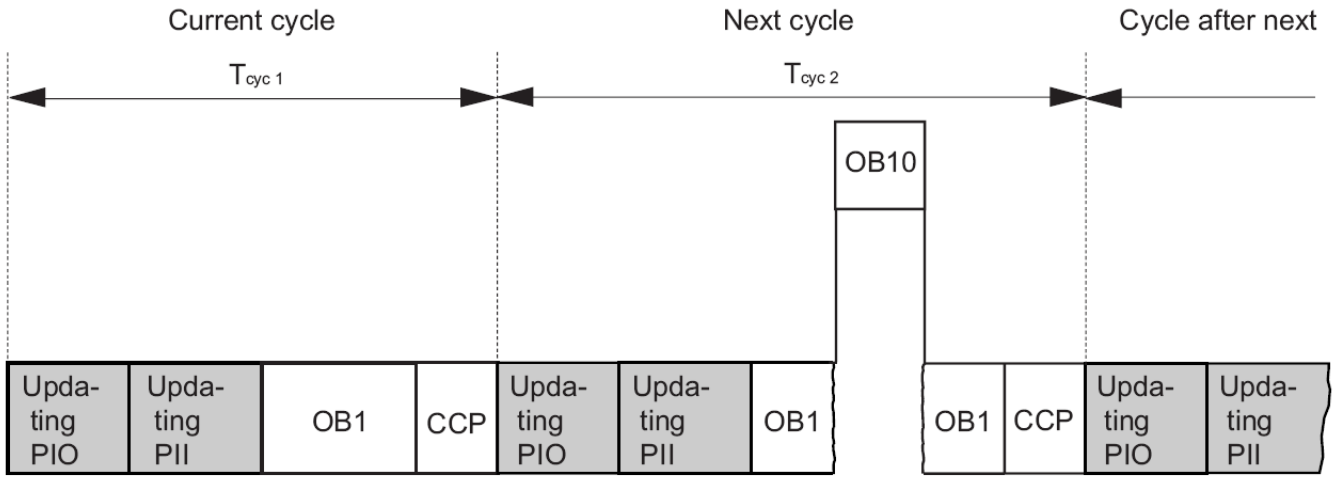
DE HAAGSE HOGESCHOOL

Opbouw PLC-programma



- Elk PLC-programma heeft minstens de bouwsteen OB1.
- OB1 roept daarna FC's en FB's aan.
- Nesting depth is 8.

Uitvoer Scan Cycle



- De Scan Cycle moet binnen een bepaalde tijd afgerond zijn anders volgt een een Scan Cycle Time Error en zal de PLC in STOP gaan (bij S7-300 is dat 150 ms).

- OB1 processing kan onderbroken worden door andere OB's, bv. OB10 (time-of-day interrupt).

Geheugen PLC

- Load memory
  - Dit is het geheugen op de MMC. Het wordt gebruikt voor opslag van codeblokken, datablokken en systeemdata (configuratie, verbindingen, ..). Blokken die niet runtime gerelateerd zijn, worden hier opgeslagen. Je kan ook alle configuratiedata op de MMC zetten.
- System memory
  - De RAM system memory is geintegreerd in de CPU-module en kan niet uitgebreid worden. Het bevat:
    - de adresgebieden voor merkers, timers en counters,
    - de process image van de I/O,
    - lokale data.
- Work memory
  - De RAM work memory wordt gebruikt om de programma's te draaien. Programma's draaien alleen in work memory.

MPI

- Multi-Point Interface.
- Wordt vooral gebruikt voor communicatie tussen een programmeer-apparaat (PG) en de PLC.
- Wordt gebruikt voor communicatie tussen een PLC en een HMI-station (OP = Operator Panel).
- De standaard transmissie-snelheid is 187,5 kbps.
- Adressering, dus mix van PLC’s, OP’s en PG’s mogelijk.

Profibus

- Profibus (Process Field Bus) is een populaire *fieldbus* die gebruikt wordt in de industriële automatisering.
- Het is gebaseerd op een serieel protocol: RS-485.
- PLC's kunnen hiermee I/O op afstand bedienen (*distributed I/O*).
- In principe fabrikant-onafhankelijk.
- Snelheid: 9,6 kbps - 12 Mbps.

Profinet

- Profibus via industrial ethernet
- Gebruikt de bekende netwerk-infrastructuur.
- Speciale voorzieningen voor Real Time applicaties.
- Snelheid: 10 Mbps, 100 Mbps, 1Gbps etc.

Zelfstudie

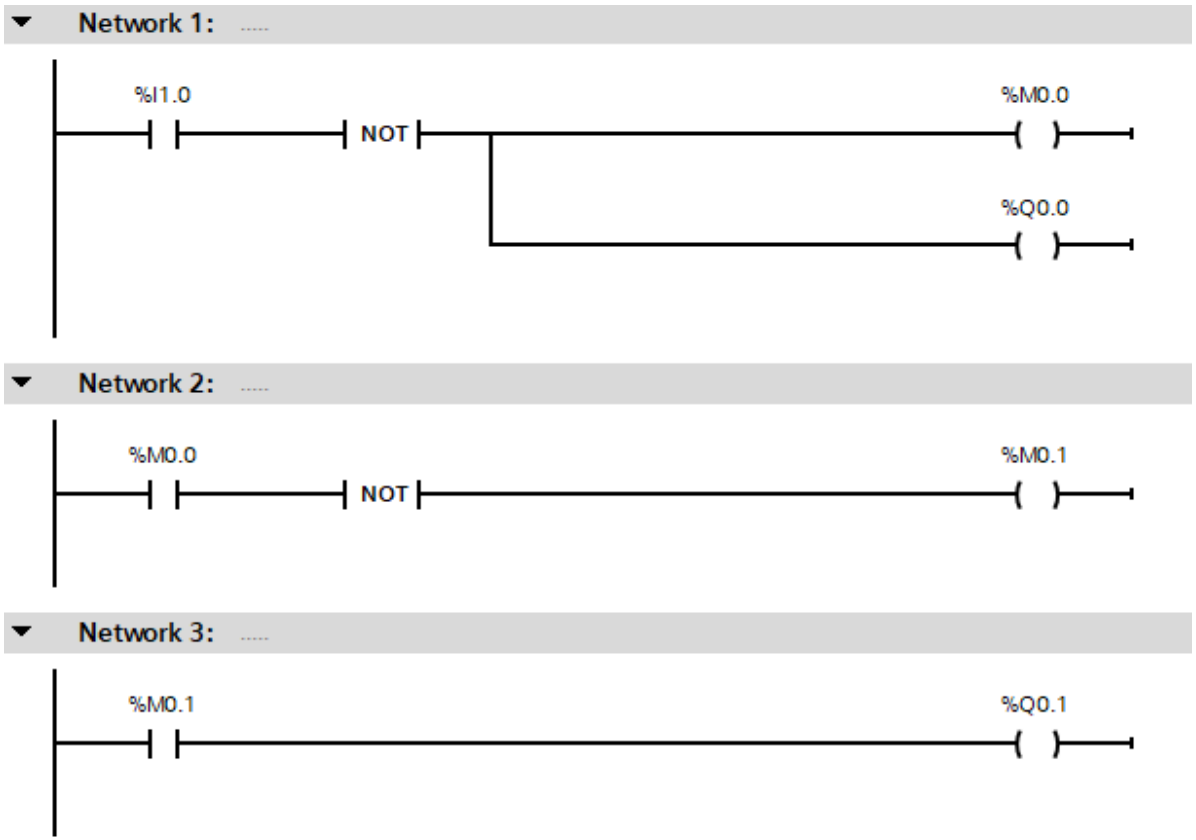
- Deze slides
- 8.2 t/m 8.5 uit
  http://www.patchn.com/images/articles/plc/plcbook/plcbook5_0.pdf
- Verdere achtergrondinfo is o.a. te vinden in:
  http://www.acro.be/Downloads/BasisPLC/Hoofdstuk%2013_Organisatie bouwstenen.pdf
  (Helaas zijn de screenshots wel van een oude Siemens "TIA portal" versie)

Les 3

- Netwerken
- Ladderdiagrammen (LAD)
- Timers en Counters
- Function Block Diagram (FBD)

Netwerken

- Ieder OB-, FC-, FB-blok bestaat uit verschillende netwerken.
- Ieder netwerk bevat een (LAD/FBD) diagram of code.
- Ieder netwerk wordt in een scan cycle **één keer** uitgevoerd.
- Volgorde van uitvoer is van boven naar beneden. (bovenste netwerk eerst, daarna netwerk daaronder, etc.)



Programmeertalen

Ter herinnering: De PLC (of eigenlijk CPU) kan geprogrammeerd worden met verschillende “programmeertalen”.

Volgens IEC 61131-3:

- Ladderdiagram (LAD, LD)
- Functieblokken (FBD, FBD)
- Assembler (STL*, IL)
- Grafisch, toestandsmachine (Graph, SFC)
- Gestructureerde programmeertaal (SCL, ST)

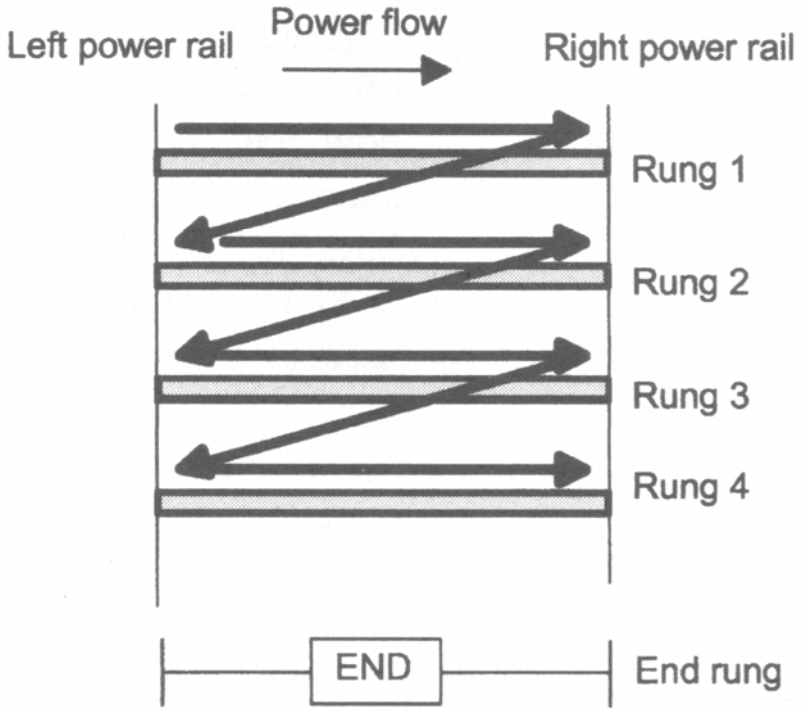
Wordt uitvoerig in PLCTEC behandeld

Andere programmeertalen worden kort toegelicht/behandeld

* Niet volgens IEC-norm

DE HAAGSE HOGESCHOOL

Ladderdiagrammen (LAD)

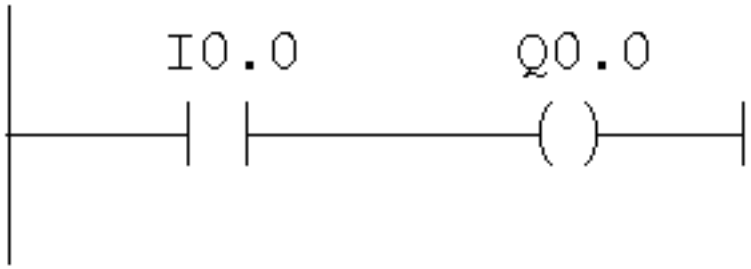


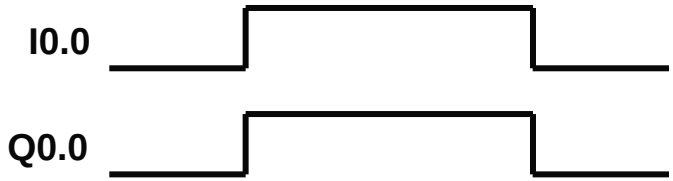
Ladderdiagrammen (LAD):

- Zijn een veel gebruikte methode voor programmering.
- Bestaan uit twee verticale *power rails.*

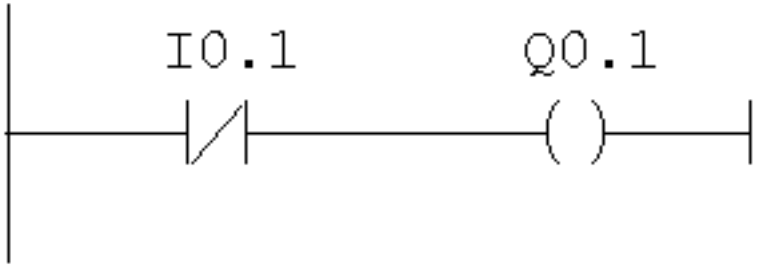
- Schakelingen worden weergegeven als horizontale lijnen (zogenaamde *rungs*) tussen de twee power rails.

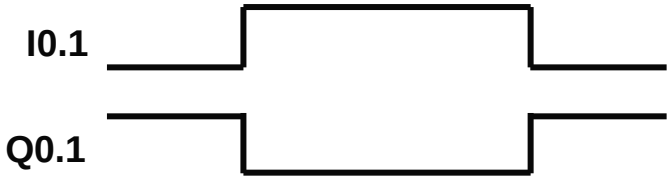
Input / output





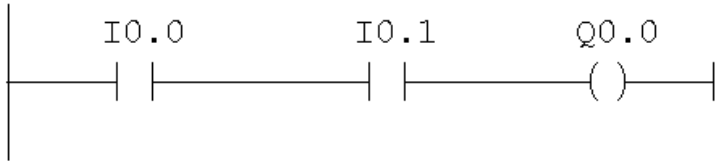
- Als ingang I0.0 gesloten wordt, zal uitgang Q0.0 geactiveerd worden. Dit is een normally open contact.



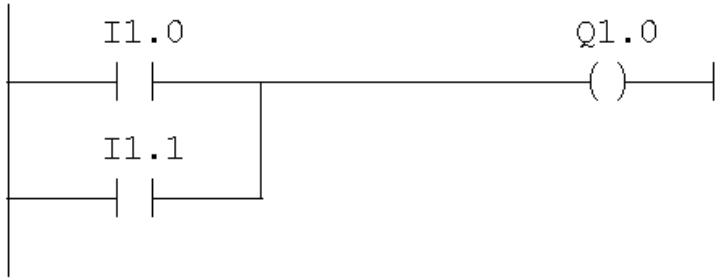


- Als ingang I0.1 geopend wordt, zal uitgang Q0.1 geactiveerd worden. Dit is een normally closed contact.

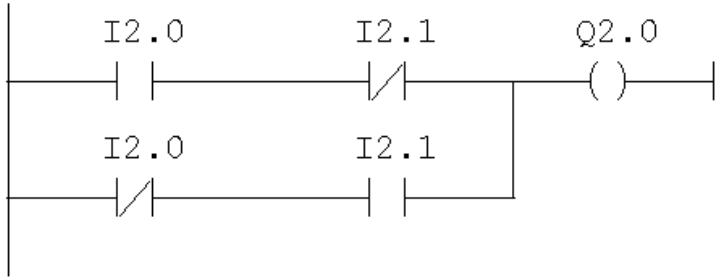
Logische constructies



- AND constructie



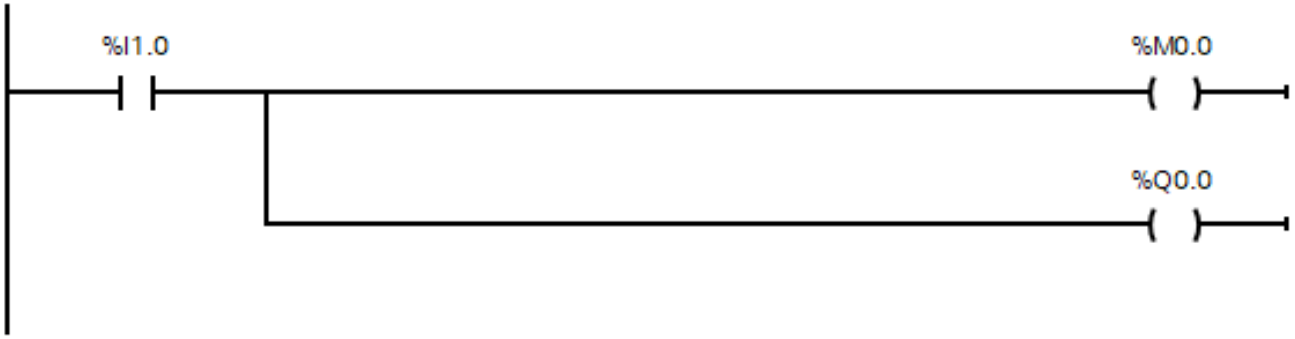
- OR constructie



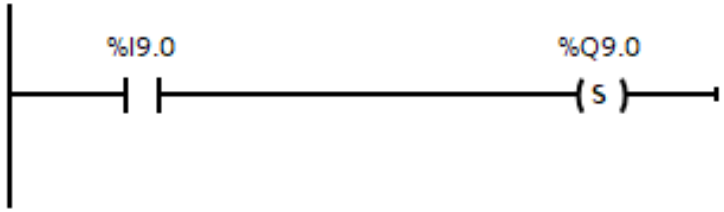
- EXOR
  $Q2.0 = I2.0 \bullet \overline{I2.1} + \overline{I2.0} \bullet I2.1$
  $Q2.0 = I2.0 \oplus I2.1$

%

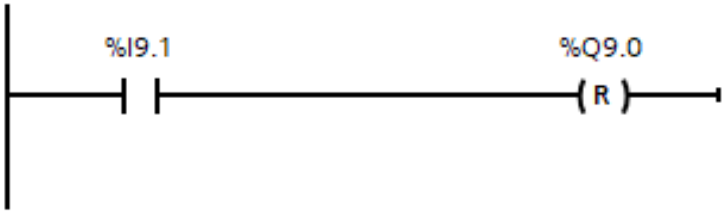
- In TIA portal wordt de adresaanduiding voorafgegaan met een ‘%’-teken. (absolute addressering)



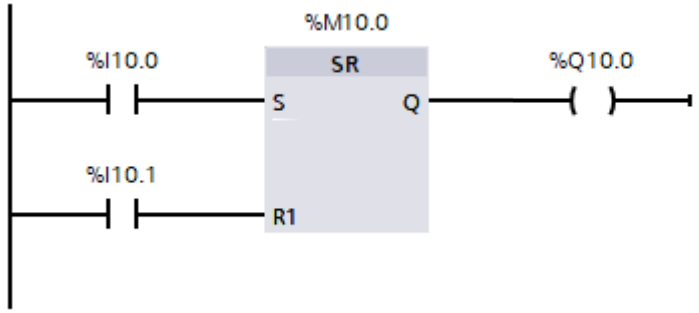
Set en Reset



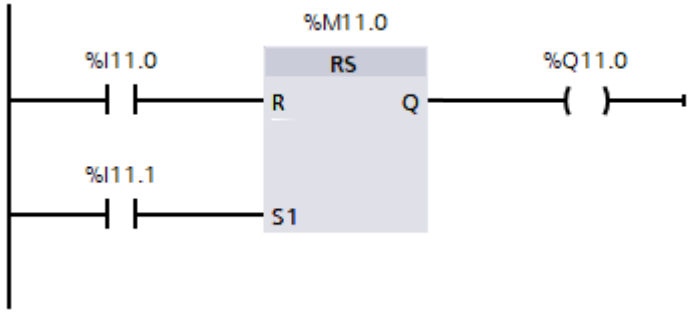
- SET-actie



- RESET-actie



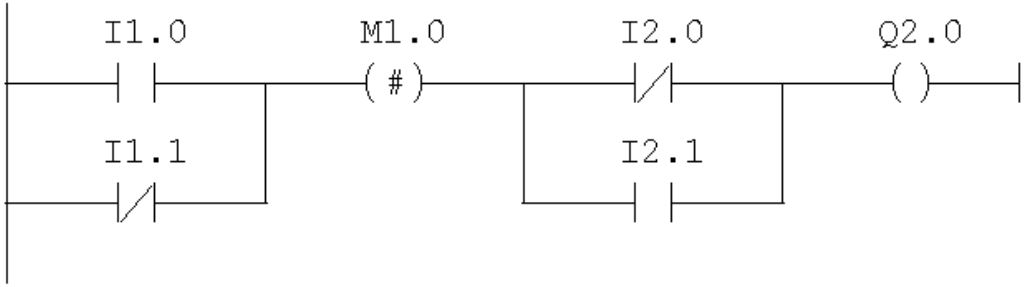
- SR-element met overheersende *reset*



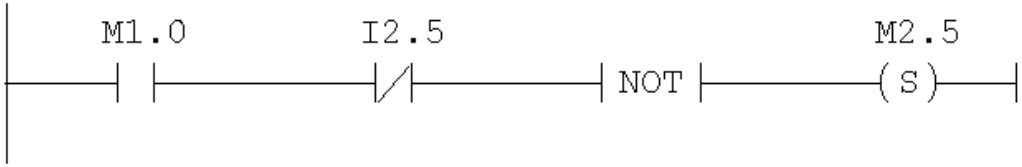
- SR-element met overheersende *set*

DE HAAGSE HOGESCHOOL

Tussenmerker, NOT

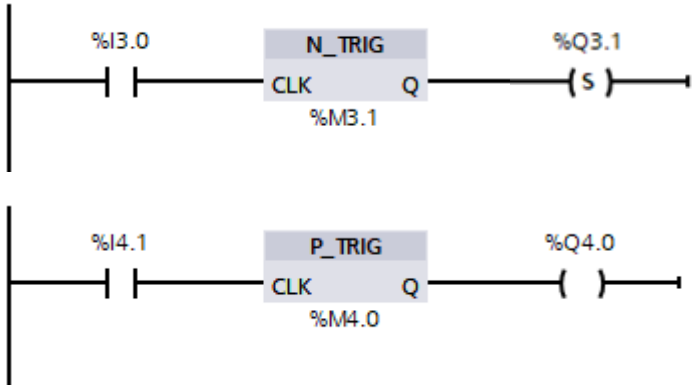


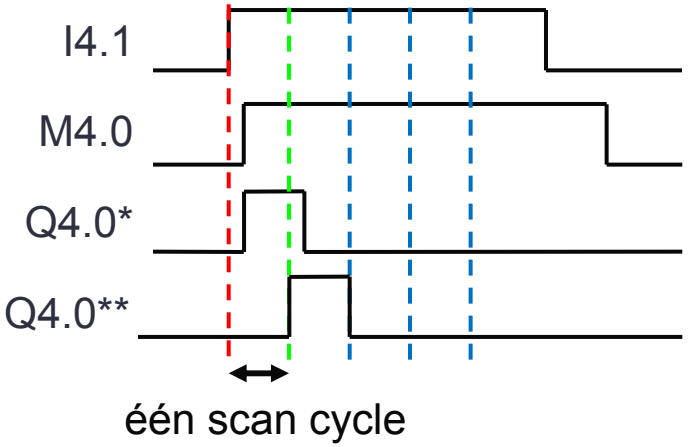
- Een tussenmerker wordt gebruikt als opslag van een deel van een logische bewerking. Deze tussenmerker is in andere bewerkingen weer te gebruiken.



- NOT inverteert de uitkomst van een logische bewerking.

Flankdetectie

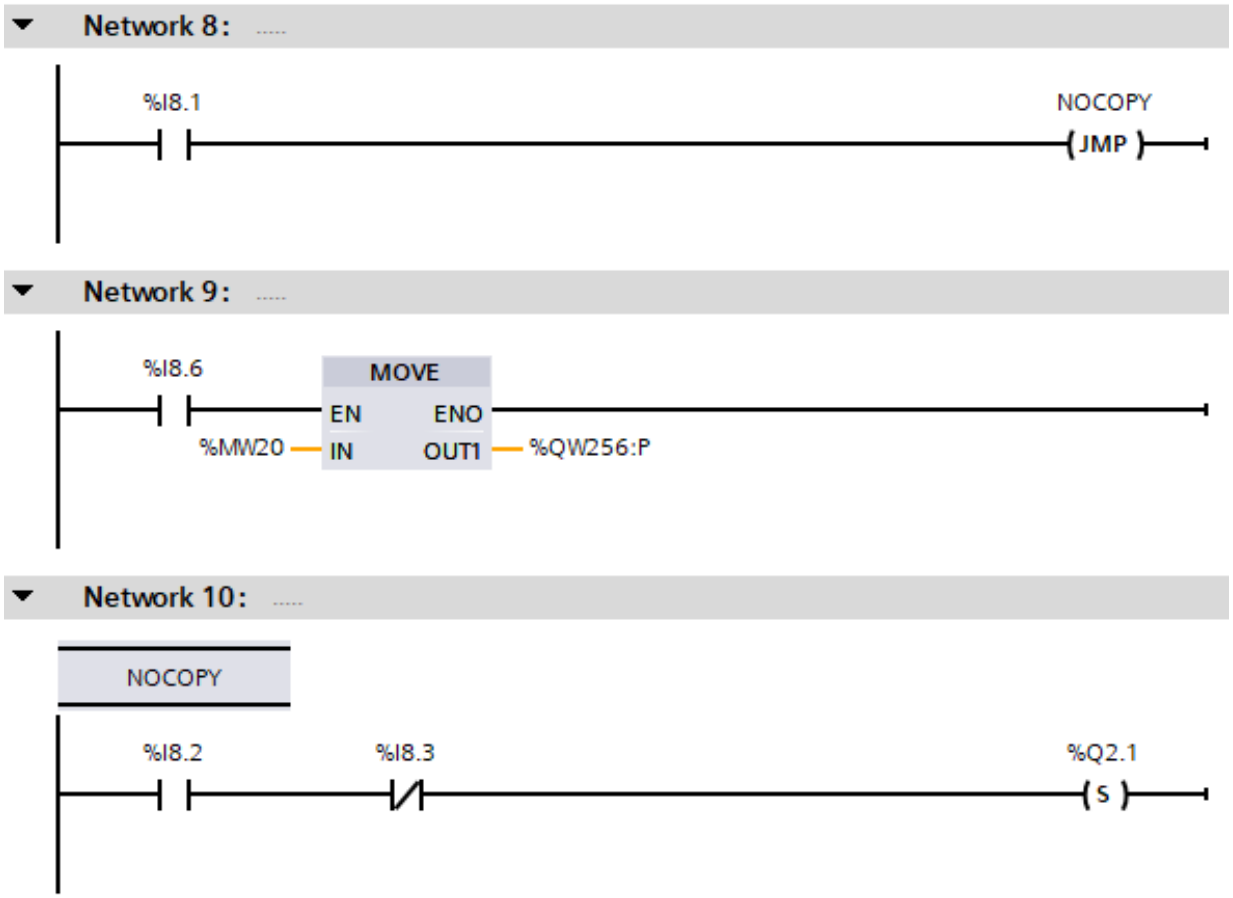




- Als ingang I3.0 van 1 naar 0 verandert, dan zal Q3.1 geset worden.
- Als ingang I4.1 van 0 naar 1 verandert, zal uitgang Q4.0 precies één scan cycle 1 zijn.
- M3.1 en M4.0 zijn nodig voor tussenopslag, namelijk de waarden van de ingangen tijdens de vorige scan cycle. Ze worden aangepast tijdens het uitvoeren van de rung.

Q4.0*: in de process image, Q4.0**: op de uitgang.

Jump en label

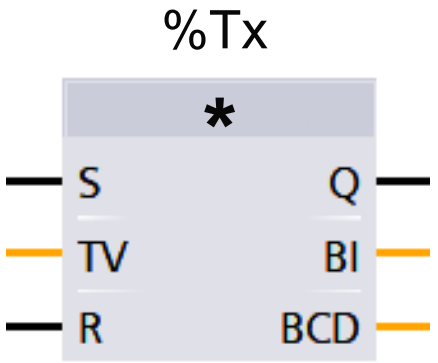


- Indien I8.1 een logische 1 is, zal gesprongen worden naar label NOCOPY. De tweede rung wordt overgeslagen.
- PQW256 = QW256:P in TIA portal!

Timers

- Een veel voorkomende taak tijdens het draaien van een PLC-programma is het meten van tijd.
- Denk hierbij aan: het openhouden van een klep, het mixen van brooddeeg.
- LAD biedt hiervoor een vijftal timers:
  - Pulse timer
  - Extended pulse timer
  - On-delay timer
  - Retentive on-delay timer
  - Off-delay timer

Timers



- Hiernaast staat het symbool van een timer.
- Tx is het timer nummer (T0, T3, T6 etc).
- Op de plaats van de asterisk (*) staat het type van de timer.
- Aansluitingen:
  - S = start input
  - R = reset input
  - TV = timer value
  - Q = status bit
  - BI = remaining time in integer format
  - BCD = remaining time in BCD format

Timers

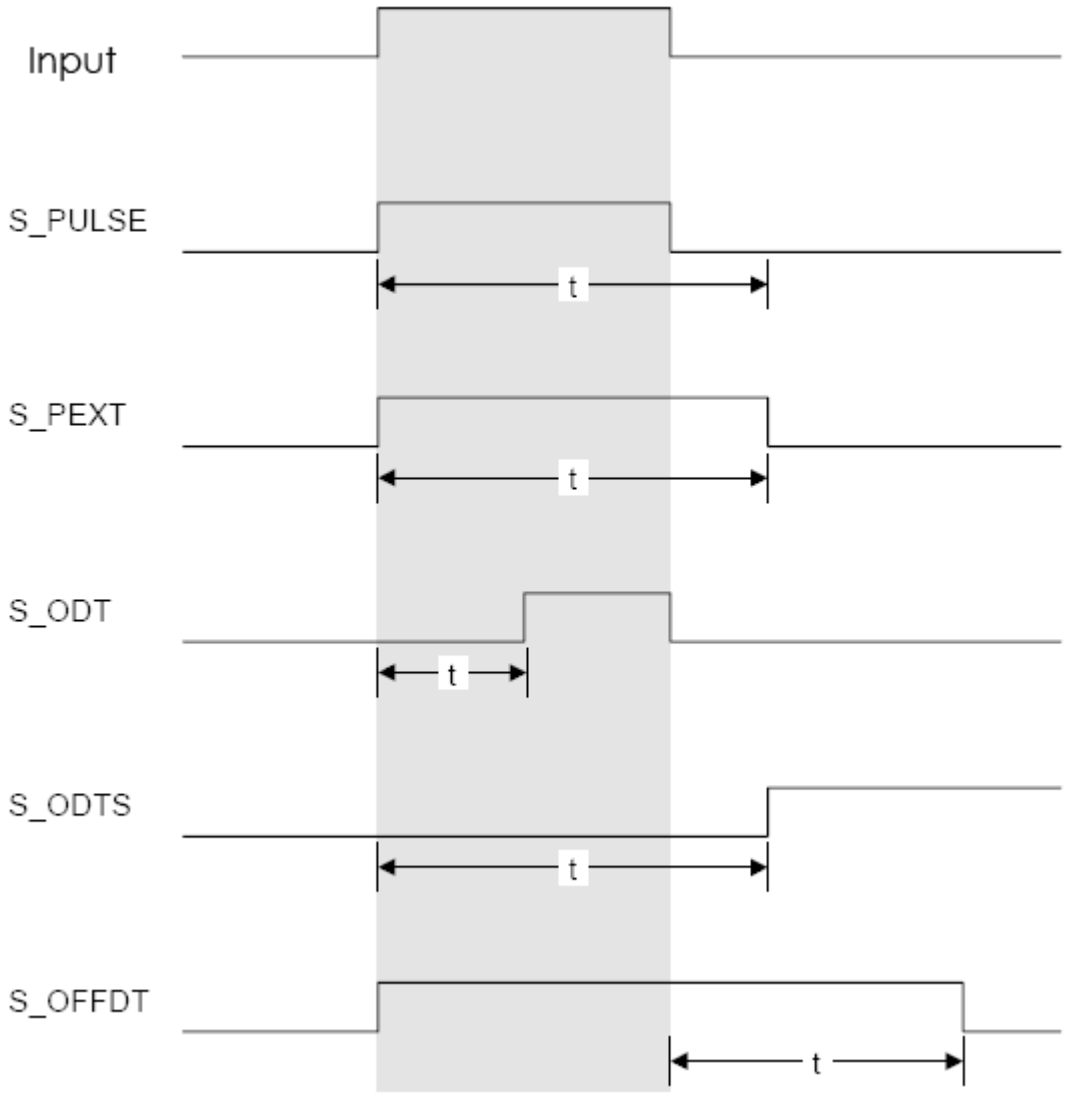
De vijftal timer-soorten:

- S_PULSE
  - De maximale tijd dat de uitgang 1 blijft, is hetzelfde als de geprogram-meerde tijd. De uitgang wordt (eerder) 0 als de ingang (eerder) 0 wordt.
- S_PEXT
  - De uitgang blijft de geprogrammeerde tijd 1, ook al wordt de ingang eerder 0.
- S_ODT
  - De uitgang wordt pas 1 als de geprogrammeerde tijd verstreken is en de ingang is nog steeds 1.
- S_ODTS
  - De uitgang wordt pas 1 als de geprogrammeerde tijd verstreken, onafhankelijk de tijd dat de ingang 1 is.
- S_OFFDT
  - De uitgang wordt 1 wanneer de ingang 1 wordt of zolang de timer loopt. De tijdmeting wordt gestart wanneer de ingang verandert van 1 naar 0.

Timers

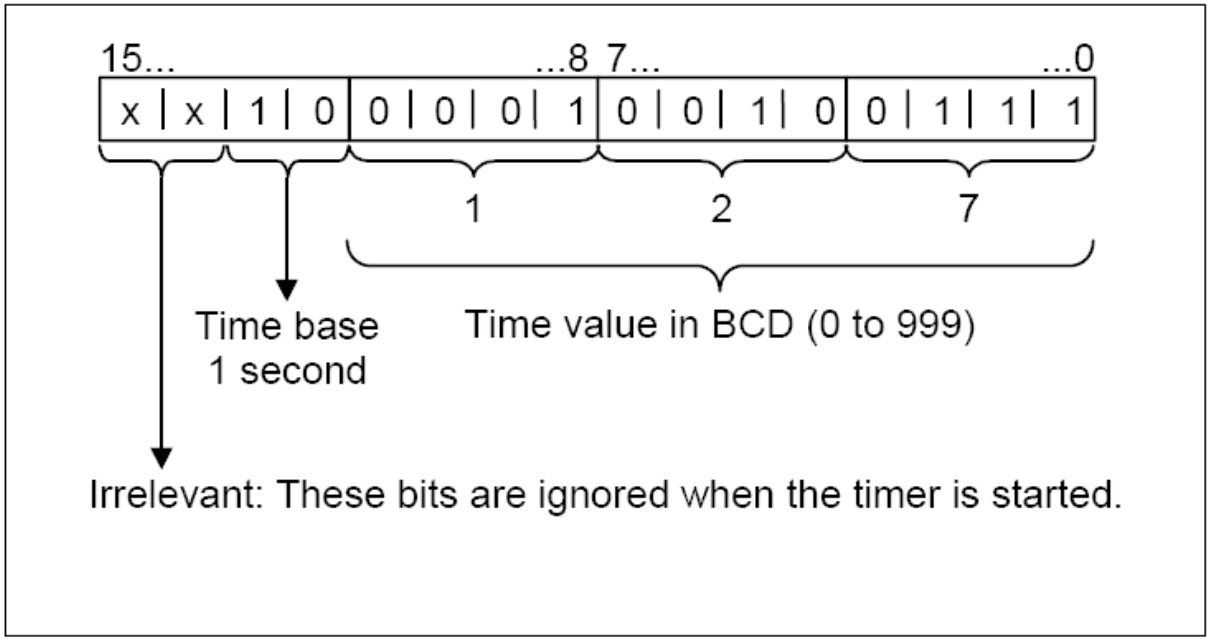
- Hiernaast de diverse timers en de puls-vormen.
- Deze tijddiagrammen zijn niet volledig.
- Zie de documentatie voor meer info.

(https://support.industry.siemens.com/cs/document/45523822/simatic-step-7-v5-5-ladder-logic-%28lad%29-for-s7-300-and-s7-400-programming?dti=0&pnid=14342&lc=en-WW HS13)



Timers

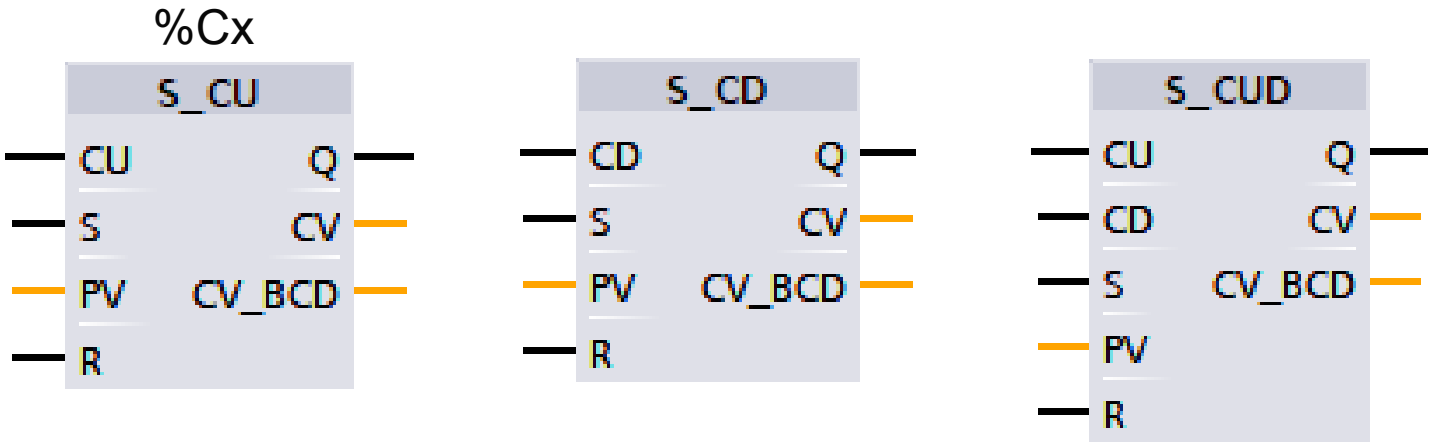
- Tijdwaarde wordt opgeslagen in 14 bits, 12 bits voor een waarde tussen 0 en 999, 2 bits voor de resolutie.
- Voor tijden langer dat 2:45 uur moeten andere voorzieningen worden getroffen.



Code	Resolutie	Tijdduur instelbaar
00	10 ms	10 ms to 9s 990ms
01	100 ms	100 ms to 1m 39s 900ms
10	1 s	1 s to 16 m 39s
11	10 s	10 s to 2h 46m 30s

Counters

- Naast timers zijn er ook counters (tellers)



- Cx is het counter nummer (C0, C1, C2, etc).
- Aansluitingen:

CU = count up input

CD = count down input

S = set the preset value

R = reset counter

PV = preset value

CV = current count value

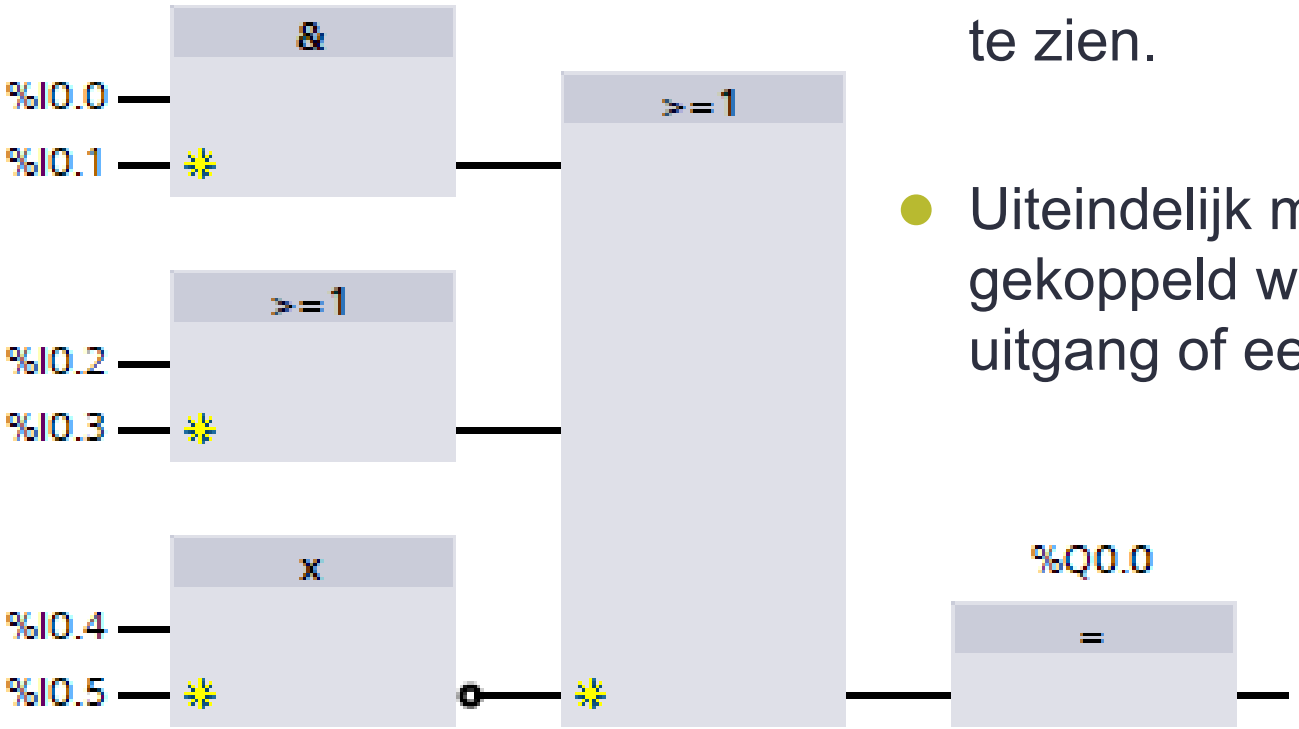
CV_BCD = counter value (BCD format)

Q = 1 if count value > 0

Function Block Diagram

- Deze vorm van programmeren komt overeen met het programmeren van digitale bouwstenen.
- Logische constructies worden door middel van poorten weergegeven.
- Ladderdiagrammen en FBD’s zijn eenvoudig in elkaar om te zetten.

Function Block Diagram

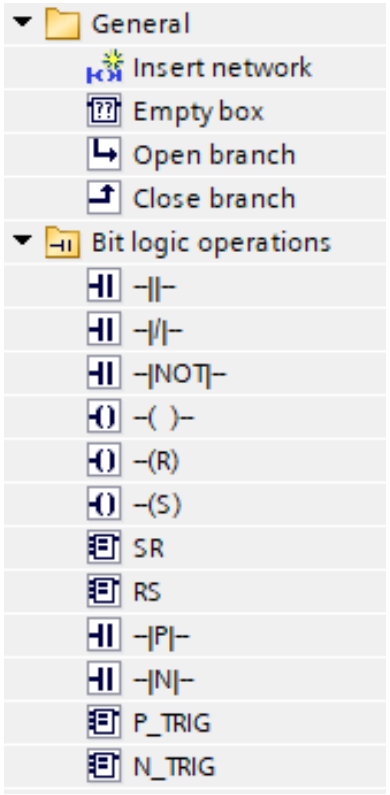


- Hiernaast is een mengeling van AND, OR, NOT en XOR te zien.
- Uiteindelijk moet de uitkomst gekoppeld worden aan een uitgang of een merker.

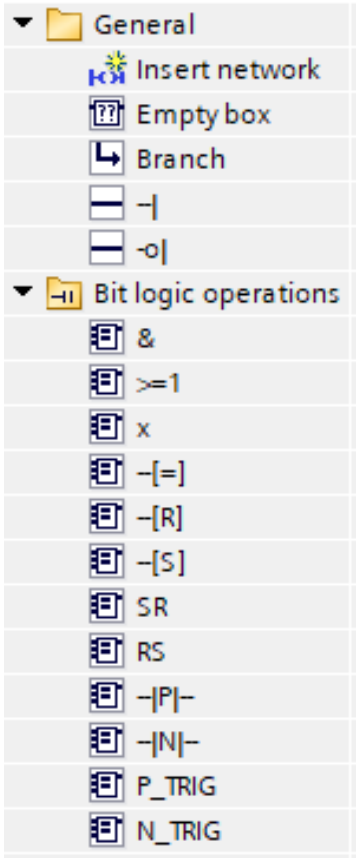
Function Block Diagram

- Vergelijk LAD en FBD.
- De overige componenten, zoals SR-elementen, timers en counters, zijn hetzelfde.

LAD

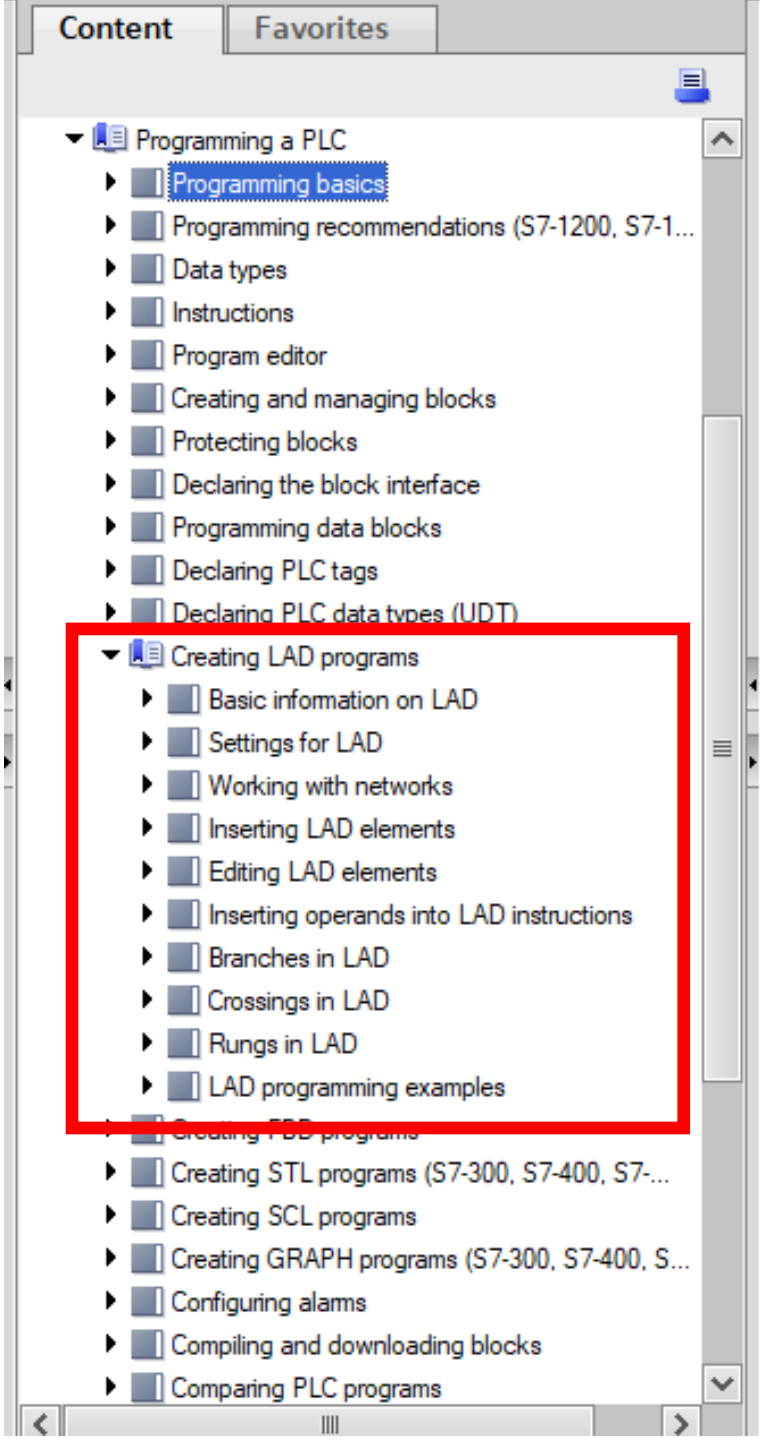


FBD



Zelfstudie

- Zelfstudie: Deze slides
- Bestudeer werking timers in 13.2 t/m 13.7 uit https://support.industry.siemens.com/cs/document/45523822/simatic-step-7-v5-5-ladder-logic-%28lad%29-for-s7-300-and-s7-400-programming?dti=0&pnid=14342&lc=en-WW
  (Let op: notatiewijze van ladderdiagrammen is volgens oude Siemens software)
- Wanneer iets niet duidelijk is kijk dan in de TIA portal help (zie hiernaast). Of kijk op blackboard voor enkele kopieën uit de TIA portal help.

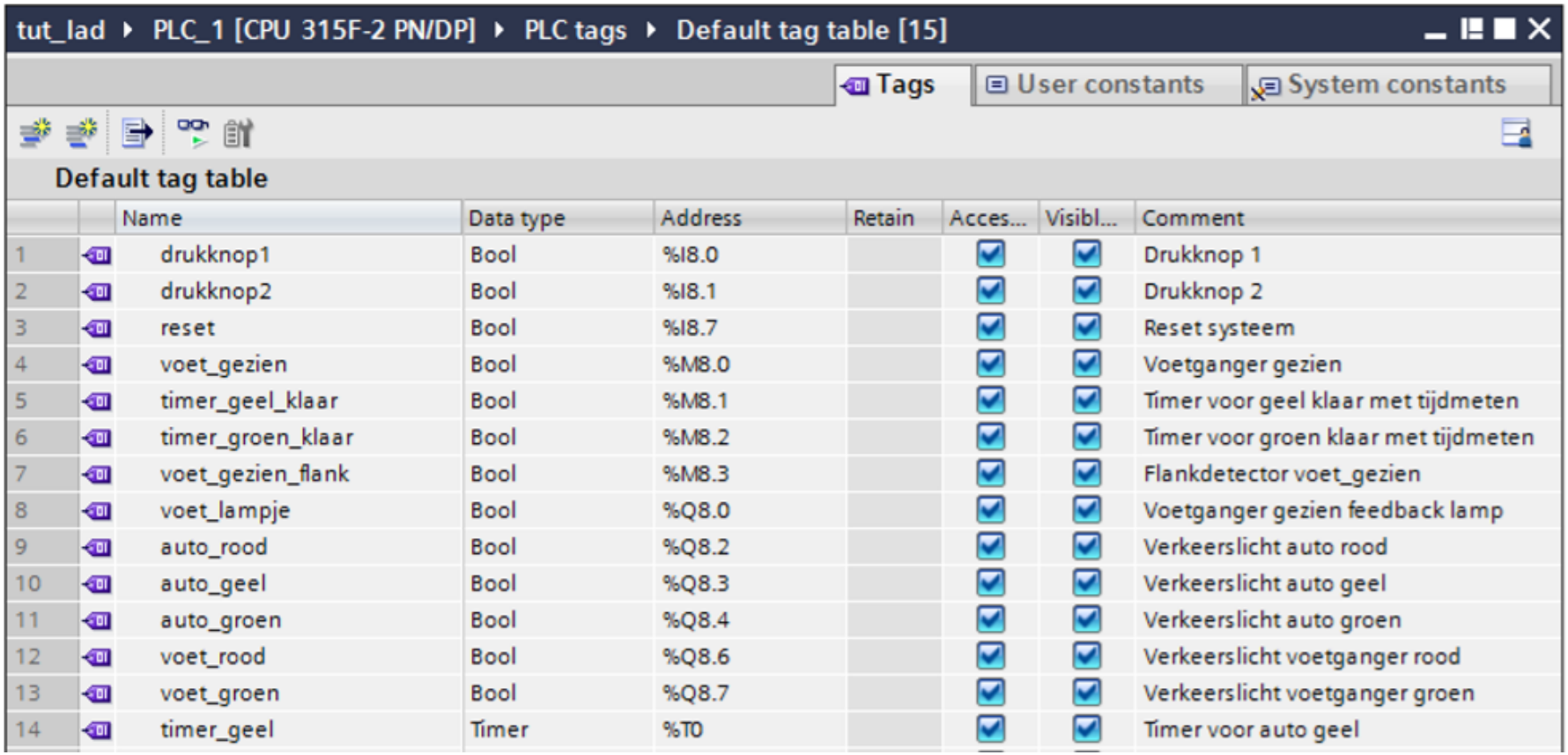


Les 4

- Symbol table
- S7-Graph

Symbol table

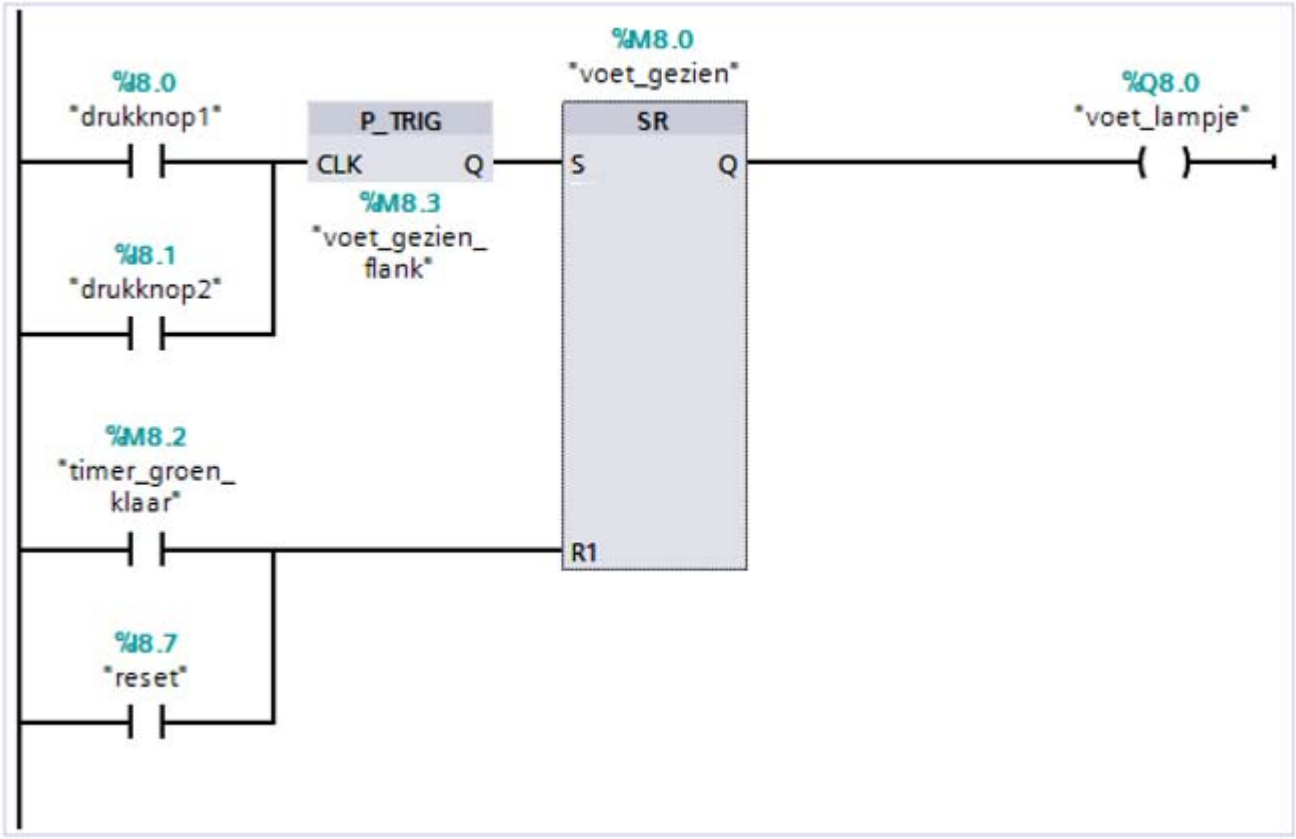
- Het gebruik van directe adressen zoals M0.0 en Q2.1 leidt makkelijk tot verwarring. Een tag table vergemakkelijkt het programmeren.



	Name	Data type	Address	Retain	Acces...	Visibl...	Comment
1	drukknop1	Bool	%I8.0		☑	☑	Drukknop 1
2	drukknop2	Bool	%I8.1		☑	☑	Drukknop 2
3	reset	Bool	%I8.7		☑	☑	Reset systeem
4	voet_gezien	Bool	%M8.0		☑	☑	Voetganger gezien
5	timer_geel_klaar	Bool	%M8.1		☑	☑	Timer voor geel klaar met tijdmeten
6	timer_groen_klaar	Bool	%M8.2		☑	☑	Timer voor groen klaar met tijdmeten
7	voet_gezien_flank	Bool	%M8.3		☑	☑	Flankdetector voet_gezien
8	voet_lampje	Bool	%Q8.0		☑	☑	Voetganger gezien feedback lamp
9	auto_rood	Bool	%Q8.2		☑	☑	Verkeerslicht auto rood
10	auto_geel	Bool	%Q8.3		☑	☑	Verkeerslicht auto geel
11	auto_groen	Bool	%Q8.4		☑	☑	Verkeerslicht auto groen
12	voet_rood	Bool	%Q8.6		☑	☑	Verkeerslicht voetganger rood
13	voet_groen	Bool	%Q8.7		☑	☑	Verkeerslicht voetganger groen
14	timer_geel	Timer	%T0		☑	☑	Timer voor auto geel

Voorbeeld netwerk

- Hieronder is netwerk 1 van het verkeerslichtsysteem weergegeven.



- Het is mogelijk om alleen de symboolnamen weer te geven.

S7-Graph

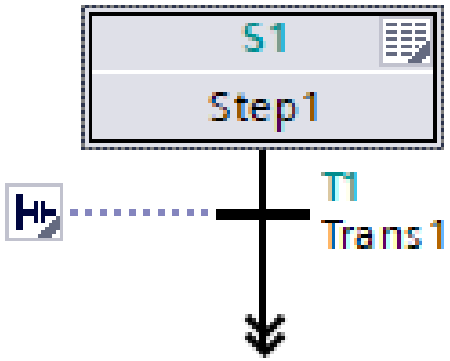
- Met S7-Graph kan je toestandsdiagrammen programmeren.
- Veel systemen kan je programmeren als een *afloop*. Het systeem loopt van toestand naar toestand. Denk hierbij aan een verkeerslichtsysteem of een lopende band.
- S7-Graph volgt de IEC-norm SFC met hier en daar een uitbreiding.
- Een S7-Graph programma wordt geprogrammeerd in een FB met bijbehorende DB (*Instance DB*).

S7-Graph

- Een S7-Graph toestandsdiagram bestaat uit:
  - Sequencers
- Een sequencer bestaat uit:
  - Stappen (vergelijkbaar met toestanden in een FSM)
  - Overgangscondities
  - Acties
- Een sequencer kan bevatten:
  - Gewone vertakkingen
  - Simultane vertakkingen
  - Sequencer stop
  - Lussen (jump)

Beginstap

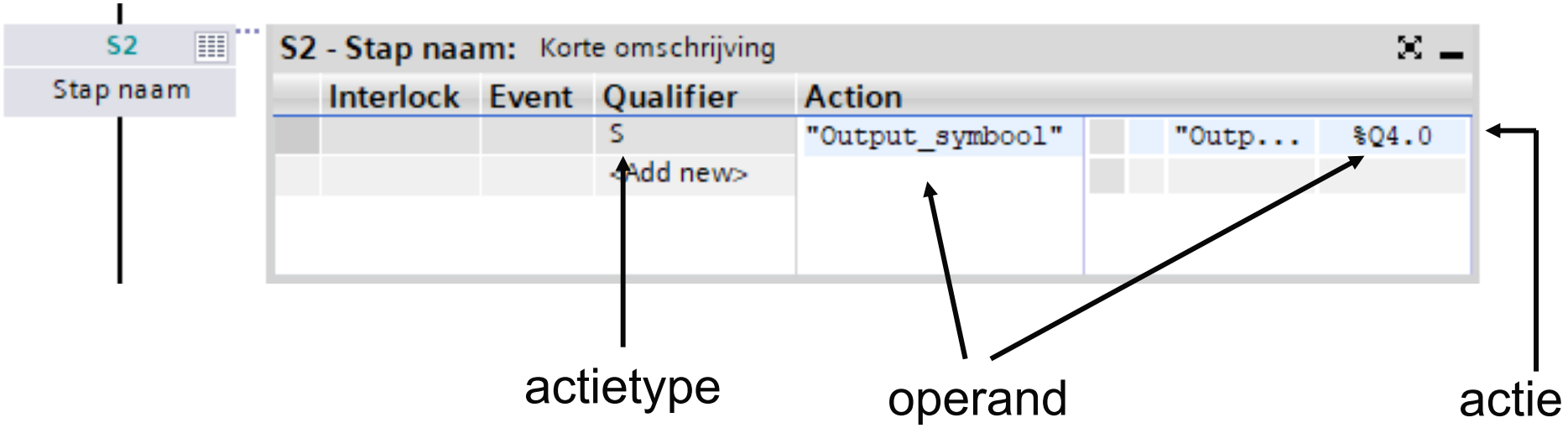
- In S7-Graph wordt een toestand een stap (*step*) genoemd. Er is altijd één beginstap (*initial step*), waar de sequencer begint na een reset. Deze is te herkennen aan de dubbele ring.



- Noot: het is mogelijk om géén initial step in een S7-Graph te hebben; dan wordt de sequencer niet gestart!

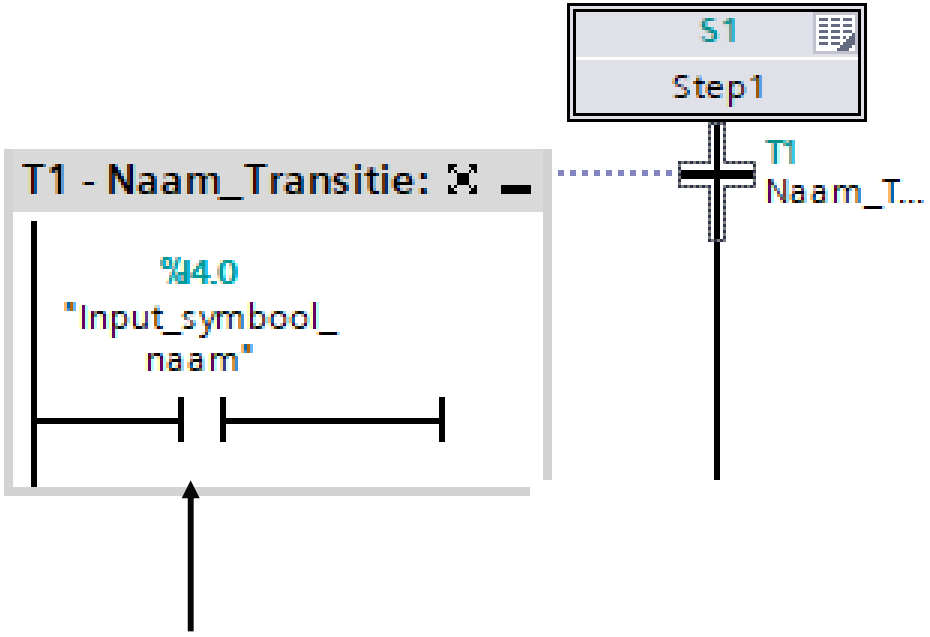
Stappen

- Een stap heeft een nummer, een naam, een korte omschrijving en een lijst met acties. Nummer en naam moeten uniek zijn.



Overgangen

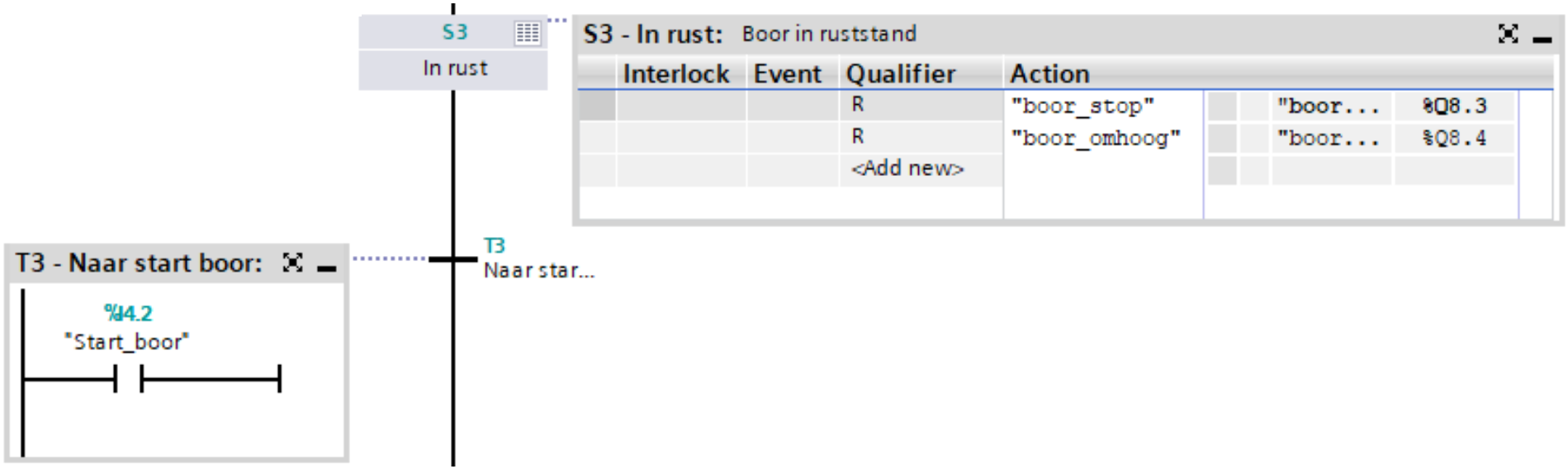
- Een overgang (*transition*) heeft een nummer, een naam en een overgangsconditie. Nummer en naam moeten uniek zijn. De overgangsconditie kan in LAD of FBD gespecificeerd worden.



overgangsconditie

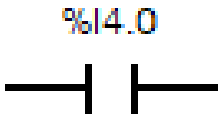
Stap en overgang

- Een stap en een overgang komen alleen gecombineerd voor.

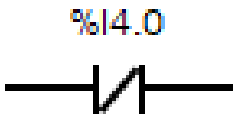


Overgangscondities in LAD (1)

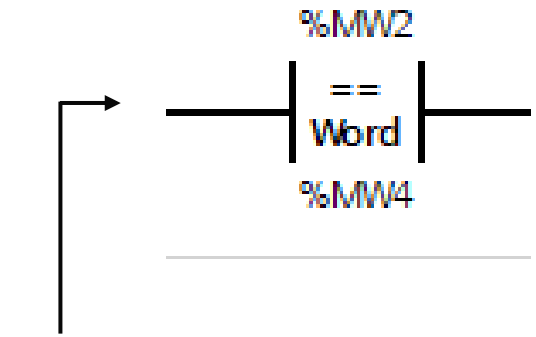
- Standaard ladder (LAD) onderdelen kunnen worden gebruikt om overgangscondities mee te construeren.
- Onder andere de volgende onderdelen kunnen dus worden gebruikt:



Normally open contact, levert 1 als signaal 1 is.



Normally closed contact, levert 0 als signaal 1 is.

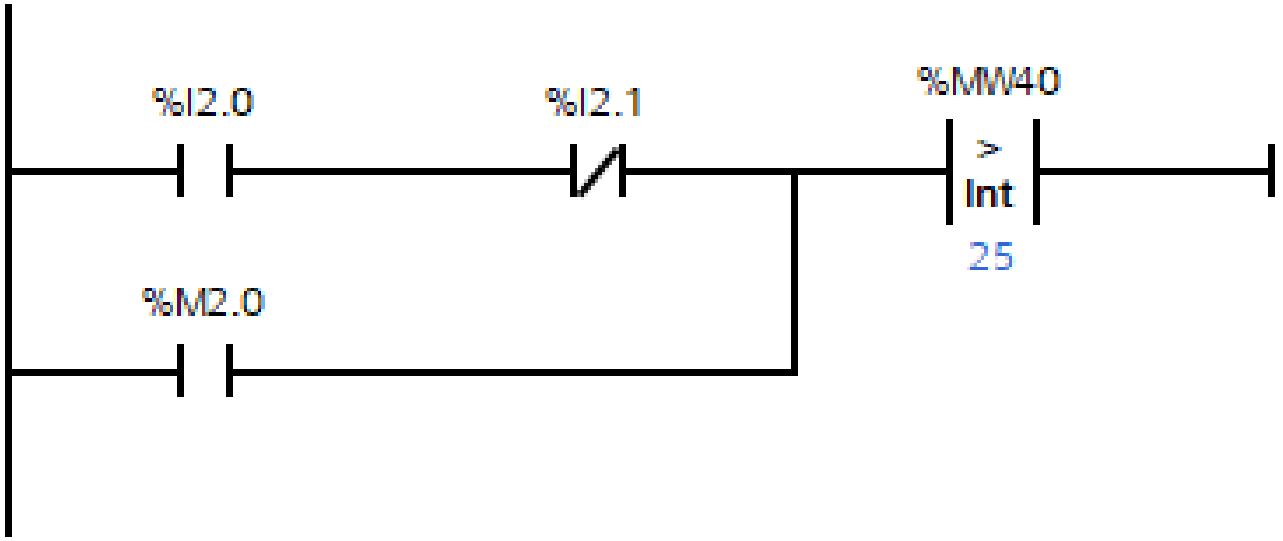


Vergelijker: levert een 1 als het resultaat van de vergelijking waar is én de enable-input is 1. Vergeleken wordt de inhoud van de twee aangegeven adressen.

Overgangscondities in LAD (2)

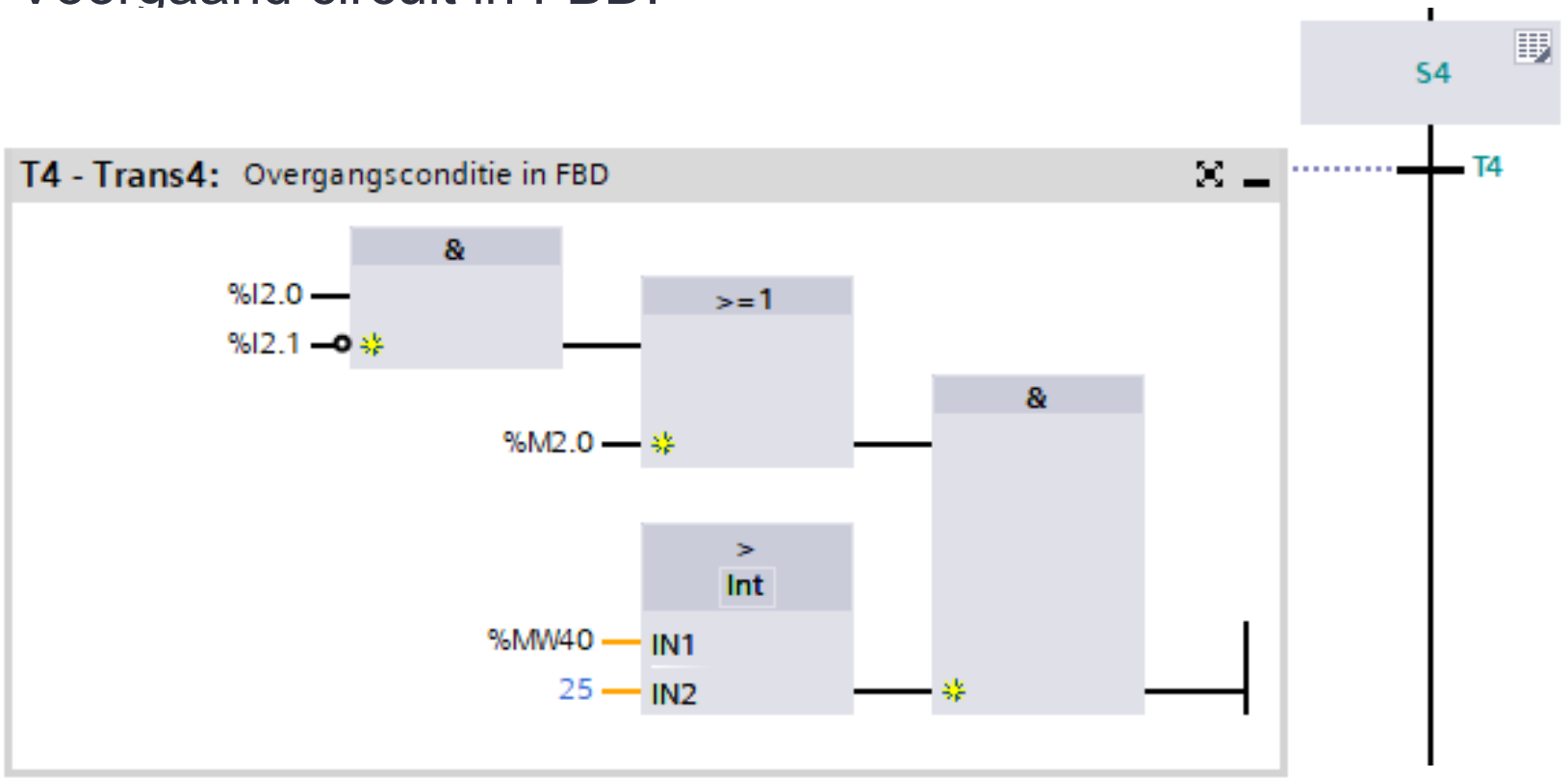
- Er wordt naar een volgende stap gesprongen als de uitkomst van het ladder circuit 1 is.
- Voorbeeld:
  - Conditie is waar als I2.0 = 1 en I2.1 = 0 of M2.0 = 1 en MW40 > 25:

$$T = \left(\left(I2.0 \bullet \overline{I2.1}\right) + M2.0\right) \bullet (MW40 > 25)$$



Overgangscondities in FBD

- Overgangscondities kunnen ook in FBD beschreven worden.
- Voorgaand circuit in FBD:



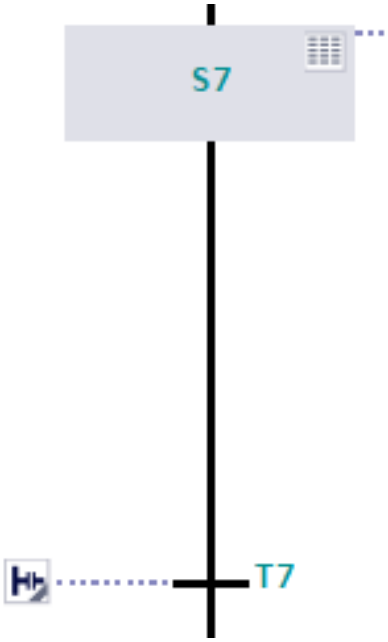
Acties

- De volgende acties zijn mogelijk:

S	Zet uitgang of merker op 1.
R	Zet uitgang of merker op 0.
N	Non-holding: zolang de stap actief is wordt de uitgang of merker op 1 gezet.
D	Delay: de uitgang of merker wordt na een bepaalde tijd op 1 gezet, nadat de stap actief geworden is en wordt 0 nadat de stap niet (meer) actief is.
L	Limited pulse: als de stap actief is wordt de uitgang of merker voor een bepaalde tijd op 1 gezet. Uitgang of merker wordt 0 als de stap niet (meer) actief is of de tijd verstreken is.
CALL	Zolang de stap actief is wordt een FC, FB, SFC of SFB aangeroepen.

Acties

- Hieronder een voorbeeld. Bij D en L moet een tijd worden opgegeven.

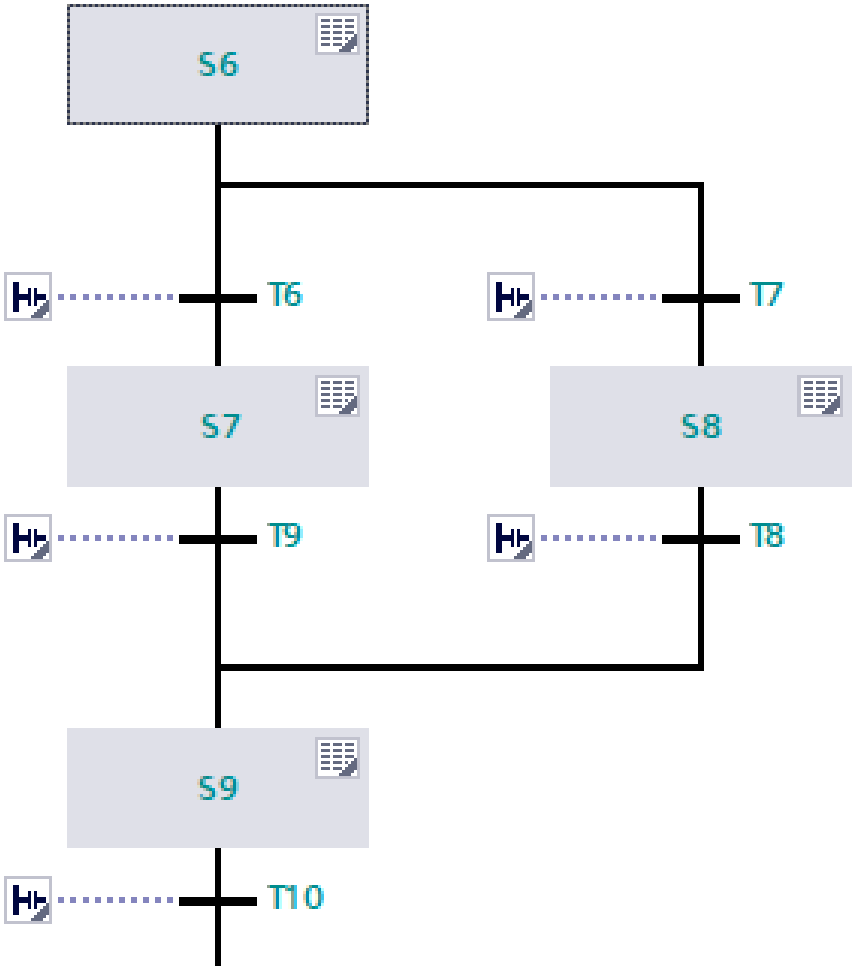


S7 - Step7: Voorbeeld acties

Interlock	Event	Qualifier	Action
		S	%Q2.0
		R	%M2.0
		N	%Q2.1
		D	%Q2.2,T#5s
		L	%M2.2,T#1m10s
		N	CALL %FC1 ()
		N	CALL %FB2, %DB2 (OFF_SQ := FALSE

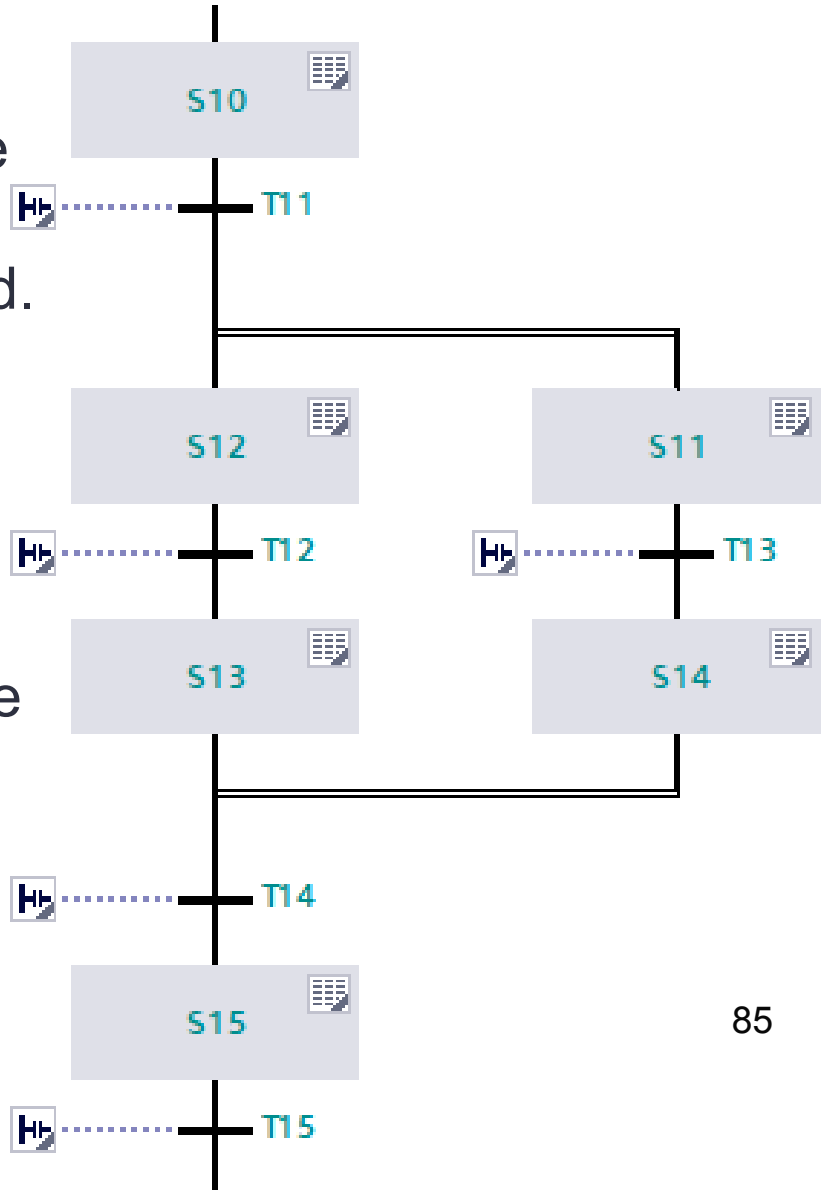
Vertakking

- Met een vertakking (*alternative branch*) is een beslissing te programmeren.
- Transities worden van links naar rechts geevalueerd, T6 gaat voor T7, ook als T6 en T7 beide waar zijn.



Parallelle vertakking (1)

- Bij een parallelle vertakking (*simultaneous branch*) worden twee of meer takken parallel (en dus onafhankelijk van elkaar) uitgevoerd.
- Er zijn meerdere stappen tegelijkertijd actief.
- De stap na het samenkomen van de takken wordt pas uitgevoerd als beide takken afgewerkt zijn.



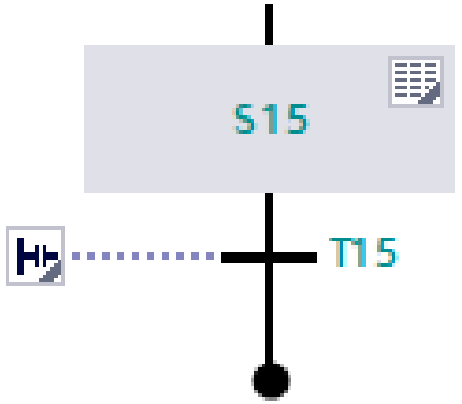
Parallelle vertakking (2)

- Meerdere sequencers in een S7-Graph programma kunnen dus parallel (*simultaneous*) worden uitgevoerd.

- Let op: de volgorde van de interne uitvoer van de sequencers is niet vastgelegd. Let hierbij op met gebruik van gemeenschappelijk geheugenplaatsen.

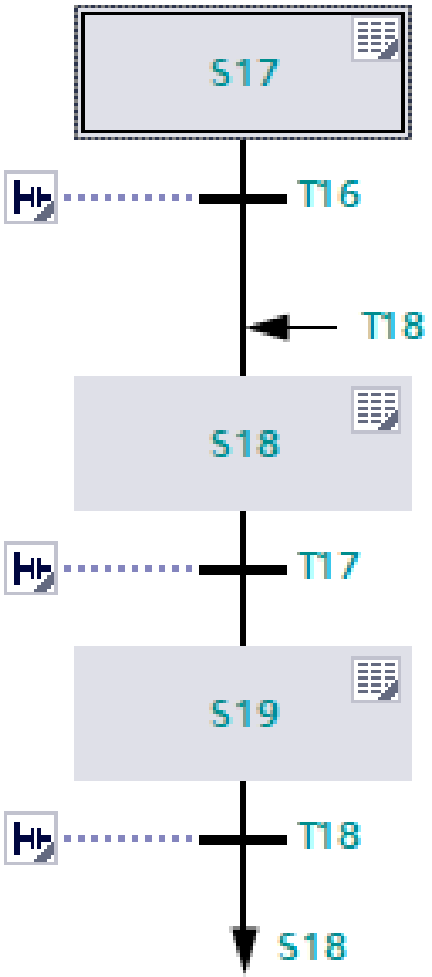
Sequencer stop

- Een sequencer stop (*branch stop*) zorgt ervoor dat een lineaire sequencer stopt.
- Als de stop zich in een vertakking bevindt wordt de sequencer gestopt.
- Als de stop zich in een parallelle vertakking bevindt wordt alleen die tak gestopt. De rest van de sequencer gaat verder.



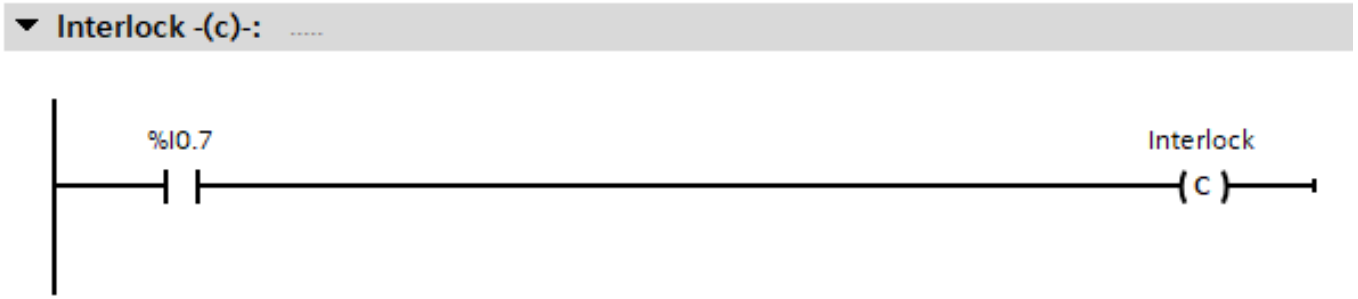
Springen

- Het is mogelijk om onvoorwaardelijk te springen (*jump*).
- Hierdoor is het mogelijk om lussen te programmeren.
- In het voorbeeld wordt na stap 19 gesprongen naar stap 18.



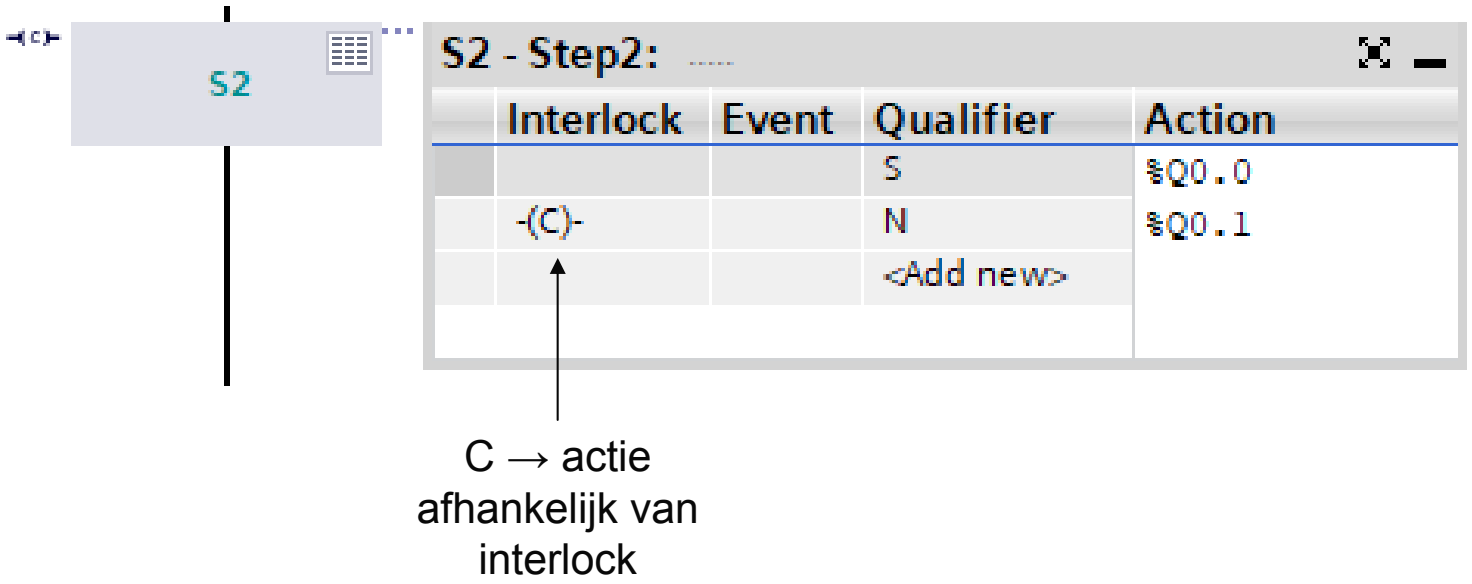
Interlock (1)

- Met een interlock kan de werking van de acties tijdelijk worden opgeschort. Het testen van de overgangsconditie gaat gewoon verder.
- Een interlock kan ingevoerd worden bij de stap details (dubbelklikken op stap):



Interlock (2)

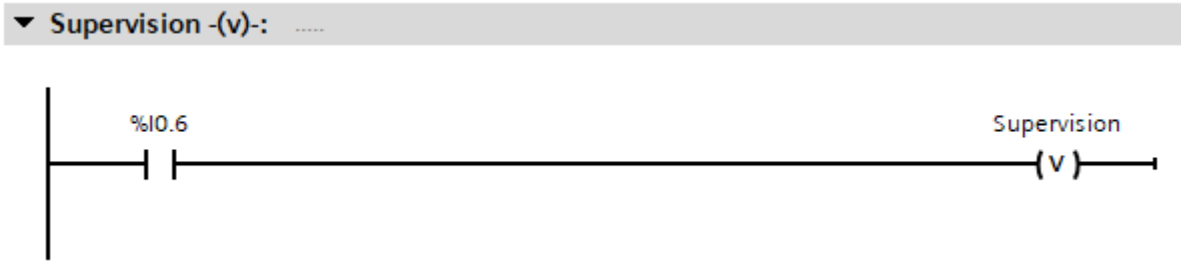
- Bij de acties kan de interlock worden toegepast:



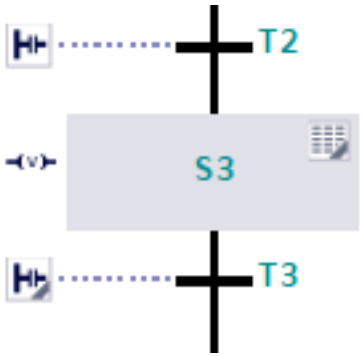
- Als de conditie voor de interlock waar is, gaat de werking verder.

Supervision (1)

- Met een supervision kan het testen van de overgangsconditie worden opgeschort. De sequencer blijft in de huidige stap als de supervision-conditie waar is.
- In de stap-details:



- In de sequence view:



Supervision (2)

- Als de supervision-conditie onwaar wordt, zal de sequencer in de huidige stap blijven, ook als de overgangsconditie waar is. Eerst moet namelijk een acknowledge worden gegeven. (later meer)
  - Bij tabblad General -> Attributes -> Sequence Properties -> “Acknowledgement required for supervision errors” kan er evt. voor worden gekozen om geen acknowledge te hoeven geven.

Permanente instructies (1)

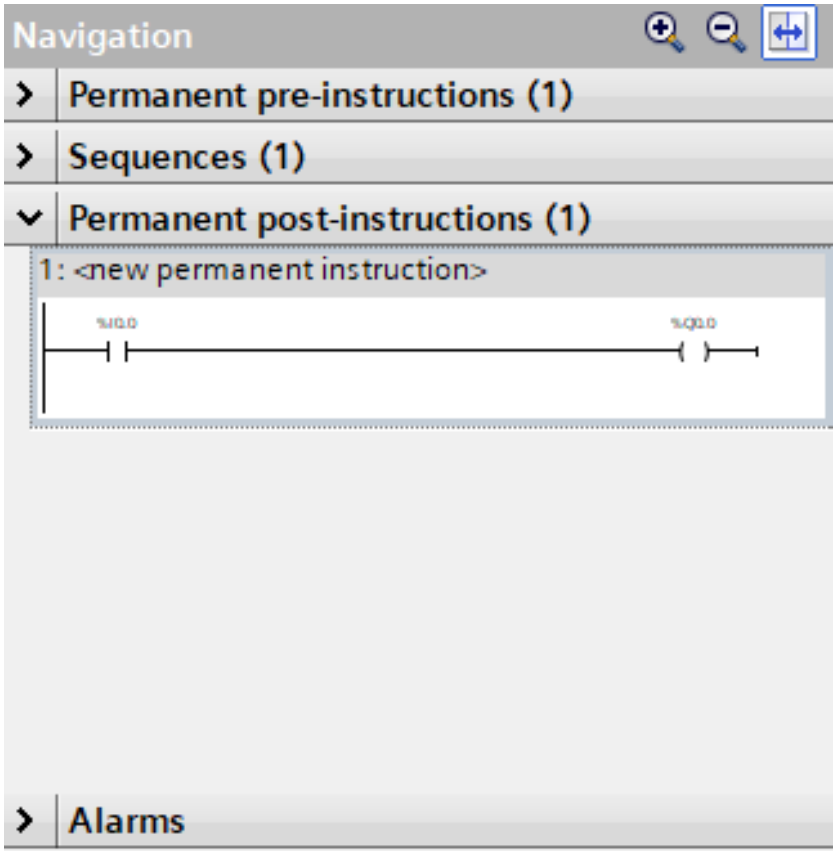
- Het is mogelijk om permanente instructies op te nemen in het S7-Graph programma. Dit zijn instructies die voor of na de sequencer-code worden geplaatst.
- Deze instructies worden ook uitgevoerd voor of na de sequencer overeenkomstig met de plaatsing.

- De instructies zijn in LAD of FBD die elke keer als de FB wordt aangeroepen worden uitgevoerd.
- Hiermee kan je bijv. signalen geschikt maken voor verwerking in de sequencer.

- Zie ook Hs 6.11 uit https://support.industry.siemens.com/cs/document/1137630/s7-graph-v5-3-for-s7-300-400-programming-sequential-control-systems-?dti=0&lc=en-WW

Permanente instructies (2)

- Plaats van permanente instructies in Graph editor:



- Zie ook Hs 6.11 uit https://support.industry.siemens.com/cs/document/1137630/s7-graph-v5-3-for-s7-300-400-programming-sequential-control-systems-?dti=0&lc=en-WW

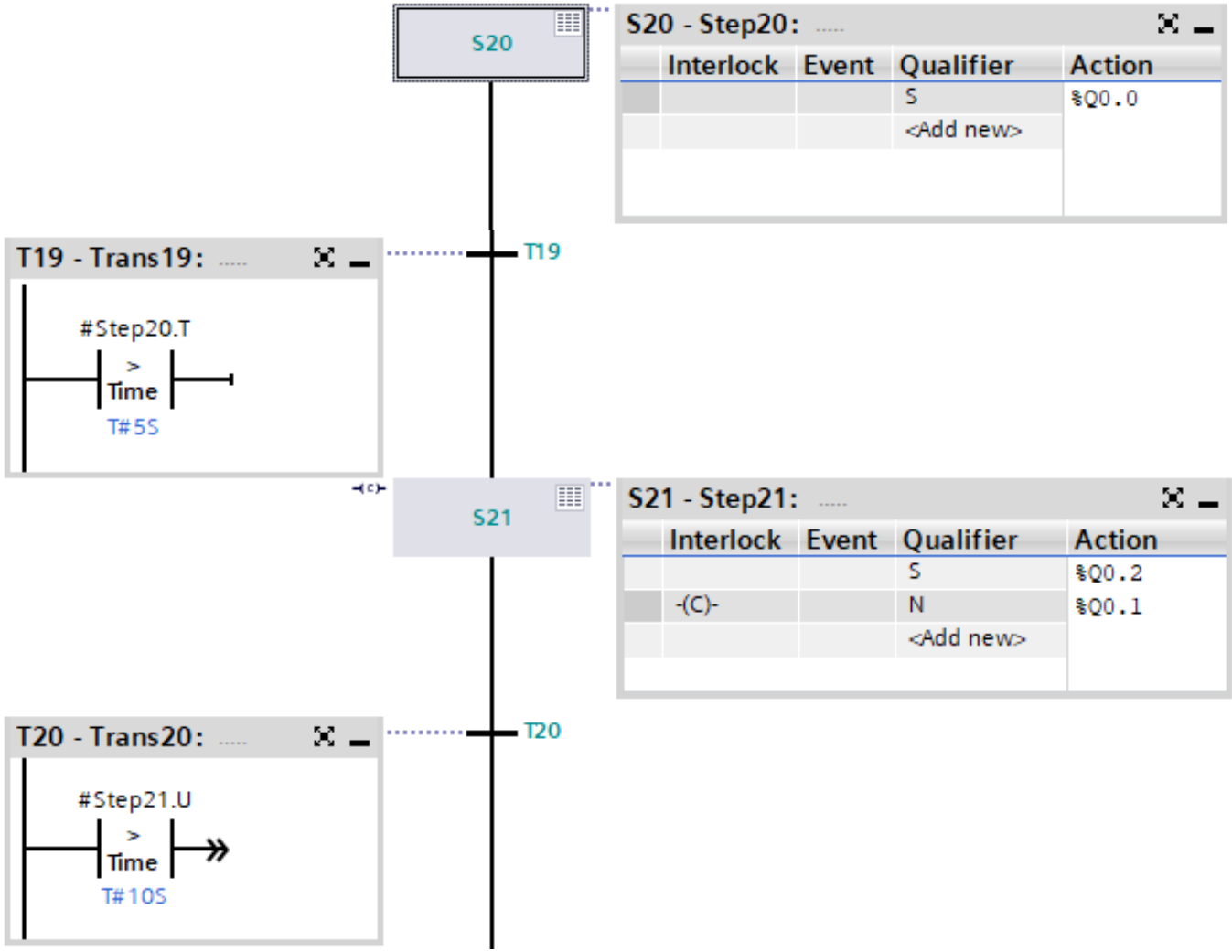
Timers (1)

- Bij iedere stap worden twee timers gebruikt:
  - Stap_naam.T: Geeft de tijd weer sinds de laatste activatie van de stap.
  - Stap_naam.U: Geeft de tijd weer sinds de laatste activatie van de stap, maar dan zonder de tijd van een ‘disturbance’
- Er is sprake van een disturbance indien een interlockconditie niet waar is of een supervision waar is.
- Beide timers stoppen met tellen wanneer een volgende stap wordt geactiveerd.

DE HAAGSE HOGESCHOOL

Timers (2)

- Voorbeeldtoepassing:



Event afhankelijke acties

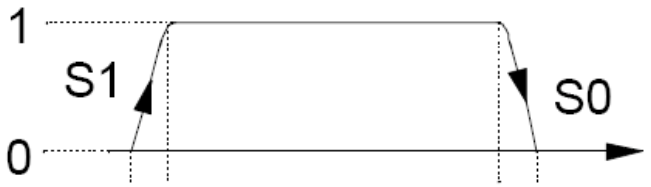
- Een actie kan logisch gecombineerd worden met een event.

- Een event kan zijn:
  - Een verandering in de signaaltoestand van een stap.
  - Een interlock
  - Een supervision
  - Een bevestiging van een supervision (*acknowledgement*).

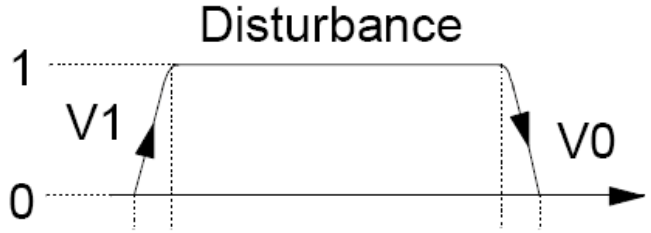
- Noot: events zijn slechts één scan cycle actief.
  - Hierdoor kunnen D en L acties trouwens niet gecombineerd worden met een event.

Event afhankelijke acties

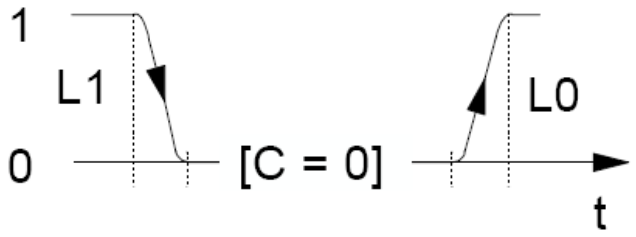
- S1: stap wordt actief
- S0: stap wordt inactief



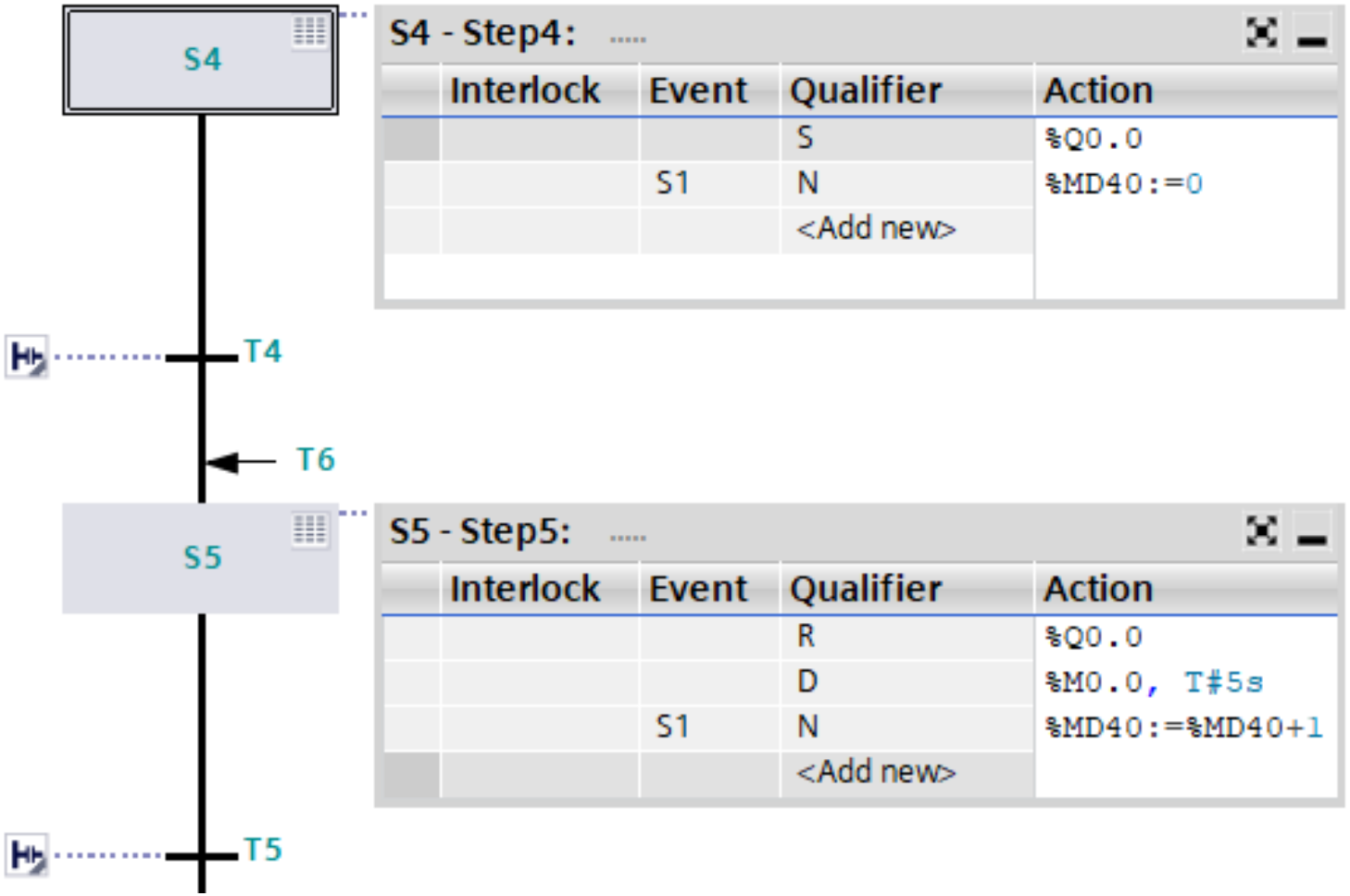
- V1: Supervision fout wordt actief
- V0: Supervision fout wordt opgeheven



- L1: Interlock wordt inactief
- L0: Interlock wordt actief
- C: Interlockconditie

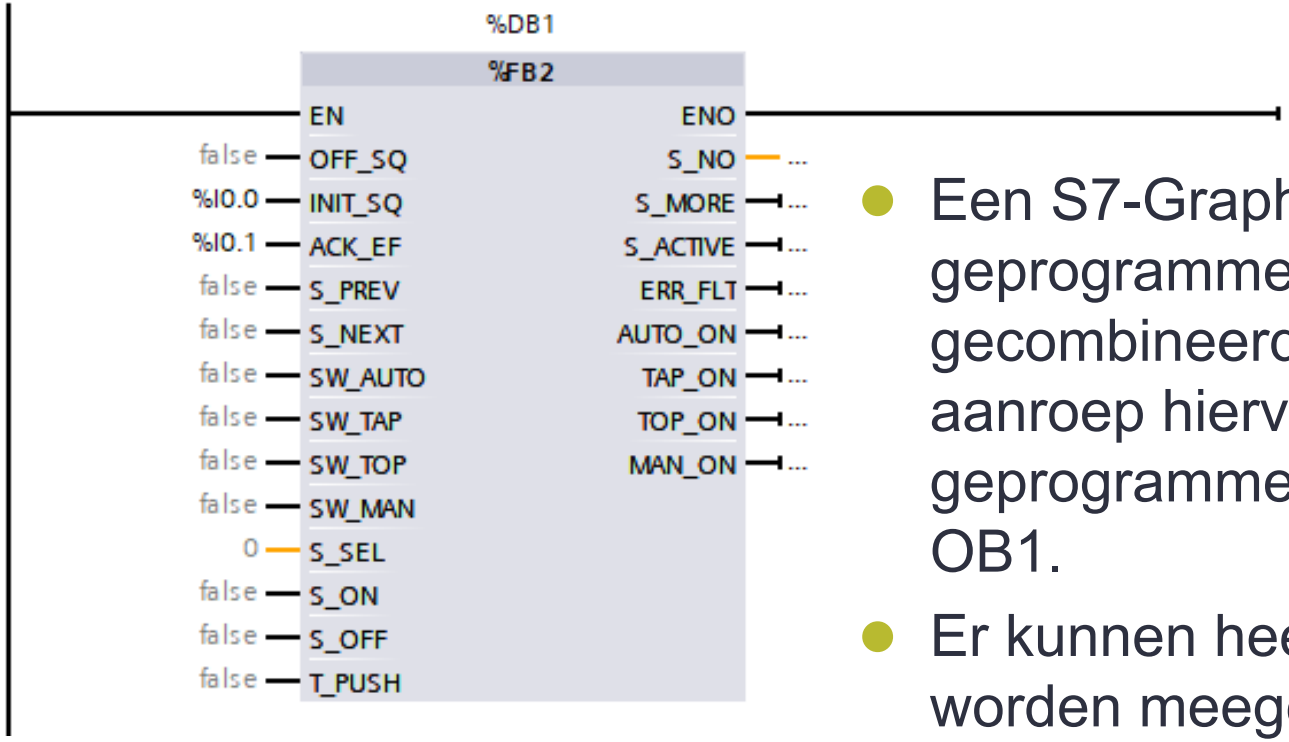


Event afhankelijke acties



- In stap 4 wordt eerst MD40 op 0 gezet.
- Elke keer als stap 5 wordt geactiveerd, wordt MD40 met 1 verhoogd.

Aanroep S7-Graph



- Een S7-Graph wordt geprogrammeerd in een FB, gecombineerd met een DB. De aanroep hiervan kan geprogrammeerd worden in bijv. OB1.
- Er kunnen heel veel parameters worden meegegeven. Slechts twee worden besproken:
  - INIT_SQ: een positieve flank op deze ingang reset de sequencer.
  - ACK_EF: een positieve flank op deze ingang geeft een acknowlegde op een supervision fout.

Zelfstudie

- Zelfstudie: Deze slides

- Indien iets niet duidelijk is, bestudeer het betreffende in: https://support.industry.siemens.com/cs/document/1137630/s7-graph-v5-3-for-s7-300-400-programming-sequential-control-systems-?dti=0&lc=en-WW
- Helaas wordt in bovenstaande bron Step7 v5.4 software gebruikt. Voor toepassing van GRAPH in TIA Portal zie *TUTORIAL STEP7 TIA PORTAL V14 MET S7-300*, J.E.J. op de Brouw op Blackboard. De help van TIA portal kan uiteraard ook gebruikt worden.

Les 5

- STL
- Rekenkundige operaties
- Vergelijken
- Schuiven

Statement List (STL)

- Ladder en FBD zijn symbolische programmeertalen: met behulp van symbolen zoals kontakten en poorten is de bedoelde functionaliteit te programmeren.
- Naast deze twee is er nog een taal: Statement List. (STL)
- STL is de assemblertaal voor de S7 PLC's.
- De programmeur is vrijer in het gebruik van instructies: constructies zijn mogelijk die in ladder of FBD niet kunnen.
- We geven hier slechts een introductie tot STL.

Verknoping

- Een groep logische operaties noemt men een *verknoping*.
- Een verknoping loopt van de eerste *afvraag*-instructie tot eerste opslag-instructie.
- Voorbeeld in STL:

```
A     I 0.0     // vraag I0.0 af
A     I 0.1     // vraag I0.1 af, AND met I0.0
=     Q 1.1     // schrijf weg in Q1.1 (RLO!)
```

Result of Logic Operation

- Het resultaat van een onderdeel van een logische bewerking wordt op geslagen in het RLO-bit. Het RLO-bit wordt na elke logische instructie bewerkt. Voorbeelden zijn AND, OR, EXOR.

- Voorbeeld in STL:

```
A     I 0.0     // vraag I0.0 af, zet in RLO
A     I 0.1     // vraag I0.1 af, AND met RLO
A     I 0.2     // vraag I0.2 af, AND met RLO
A     I 0.3     // vraag I0.3 af, AND met RLO
=     Q 1.1     // schrijf weg in Q1.1 (RLO!)
```

RLO en BR

- Er zijn instructies die direct op het RLO-bit werken.
- Het BR-bit is een vrij bit dat door het programma gebruikt kan worden als tussenopslag.
- Voorbeeld in STL:

```
SET       // Zet RLO op 1
CLR       // Zet RLO op 0
NOT       // Inverteer RLO
SAVE      // Schrijf RLO-bit naar BR-bit
A BR      // Afvragen BR-bit
```

Logische constructies

- Er zijn ook logische constructies te maken zoals AND-OR. De AND gaat voor de OR.

- Voorbeeld in STL:

```
A     M 0.0     // vraag M0.0 af
A     M 0.1     // vraag M0.1 af, AND met M0.0
O               // tussenopslag OR
A     M 0.2     // vraag M0.2 af
A     M 0.3     // vraag M0.3 af, AND met M0.2
=     M 1.1     // schrijf weg in M1.1 (RLO!)
```

Meer logische operaties

- Voorbeeld van OR-AND met SR-element in STL:

```
A(              // Begin van AND
O     M 0.0     // Afvragen M0.0
O     M 0.1     // Afvragen M0.1, OR met M0.0
)               // Einde AND
A(              // Begin nieuwe AND
O     M 0.2
O     M 0.3
)               // Einde AND
S     M 1.0     // Als RLO=1, zet M1.0 op 1
A     M 0.4     // Afvragen M0.4
R     M 1.0     // Als RLO=1, zet M1.0 op 0
```

Rekenkundige operaties

- Naast logische operaties kan een PLC ook rekenen.

- Gerekend kan worden met de volgende typen op een Siemens PLC:
  - Integer (16 bits)
  - Double Integer (32 bits)
  - Real (32 bits)

- Rekenkundige operaties voor integers (geldt voor iedere taal):
  - add, sub, mul div, mod

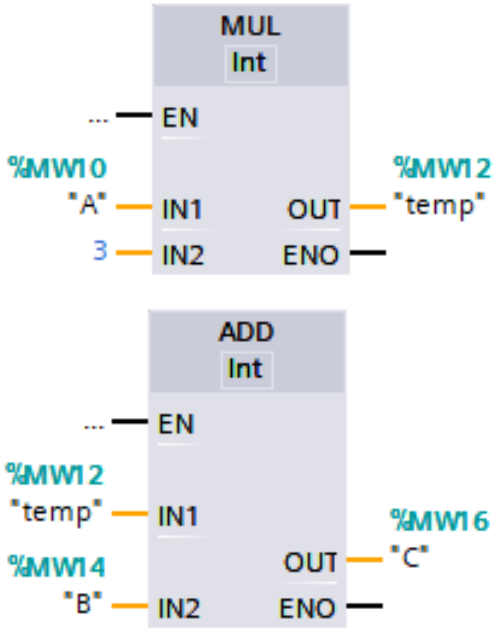
- Rekenkundige operaties voor reals (geldt voor iedere taal):
  - add, sub, mul, div, abs, sqrt, sqr, ln, exp, sin, cos, tan, asin, acos, atan

Voorbeeld rekenkundige functie

- Als voorbeeld de uitwerking van de eenvoudige functie:

  C = 3*A + B

- Let op: het datatype is integer maar het staat in memory words!



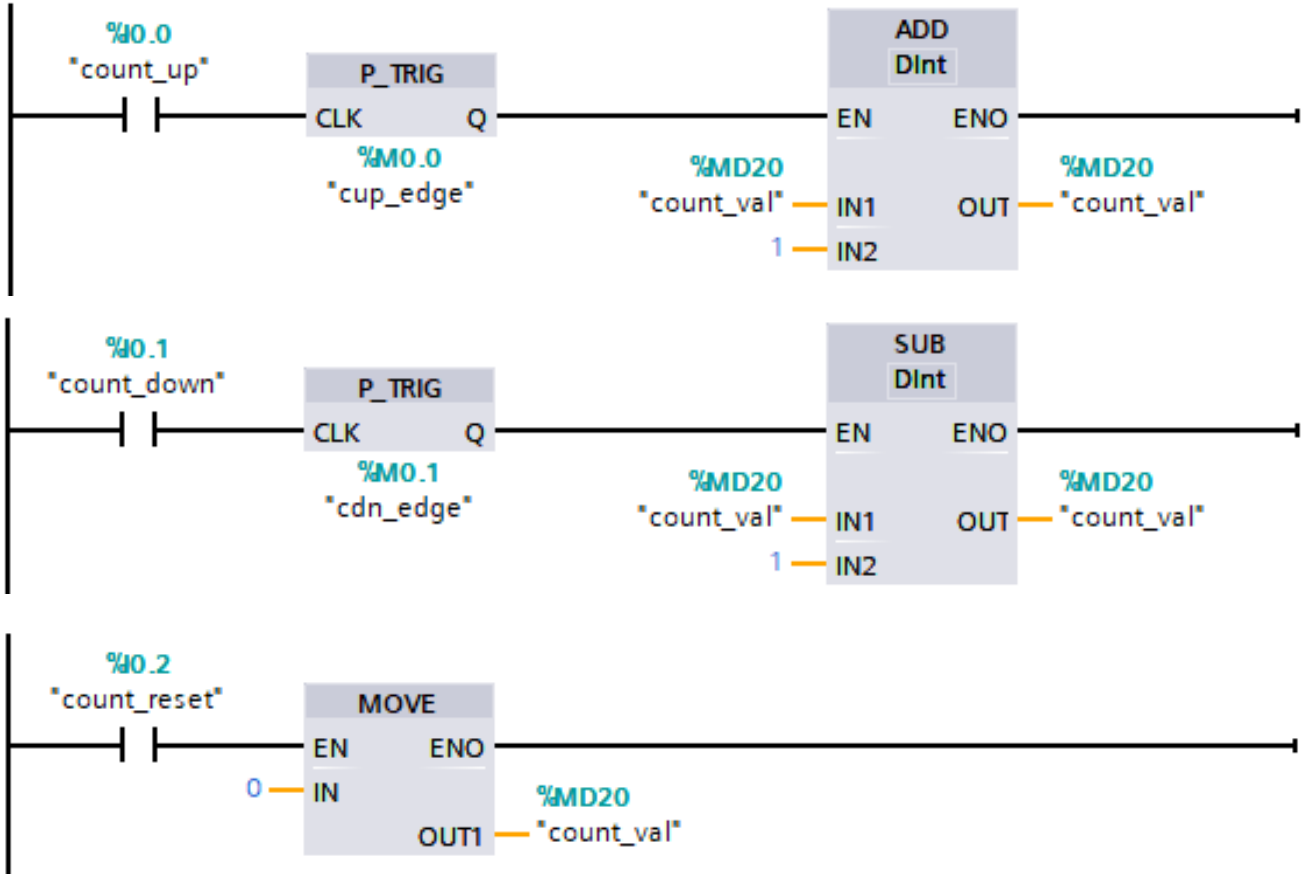
PLC tags

		Name	Tag table	Data type	Address
65		A	Default tag table	Int	%MW10
66		temp	Default tag table	Int	%MW12
67		B	Default tag table	Int	%MW14
68		C	Default tag table	Int	%MW16
69		<Add new>			

DE HAAGSE HOGESCHOOL

Eigen teller maken

- Het is eenvoudig een eigen teller te maken die grote waarden kan bijhouden én negatieve waarden kan aangeven.
- Noot: reset overheerst!

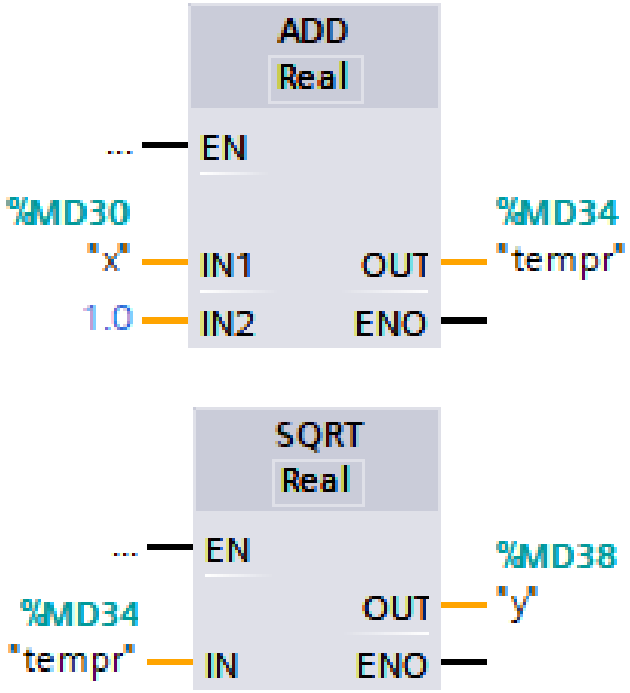


Voorbeeld rekenkundige functie

- Als voorbeeld hier de uitwerking van de functie:

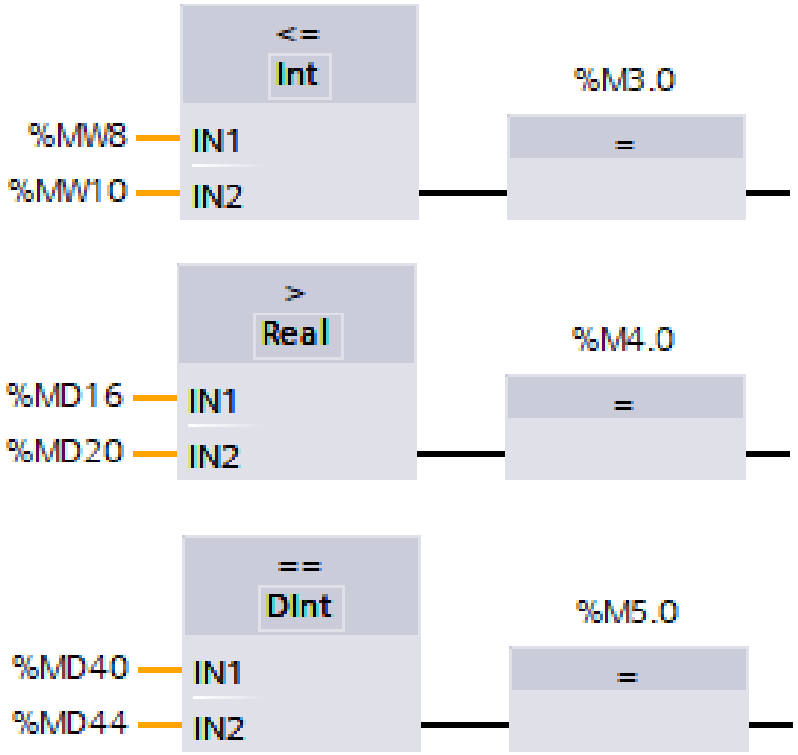
$$y = \sqrt{x+1}$$

- Wat gebeurt er nu als x < -1 is?



Vergelijken

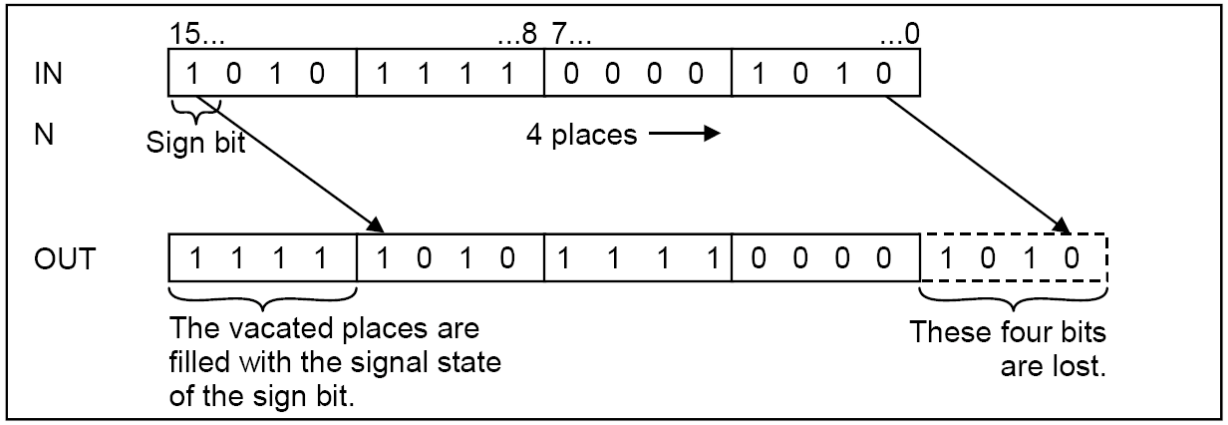
- De PLC kent ook een aantal vergelijkingen:

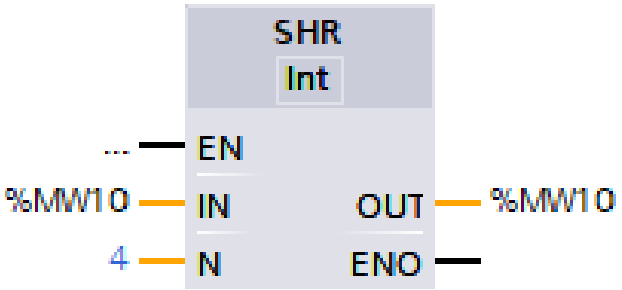


- Vergeleken kan worden Int, Dint, Real en Time

Schuiven

- Er zijn vier schuifopdrachten:
  - SHR, SHL, ROR, ROL
- SHR kan met volgende datatypes gecombineerd worden:
  - Integer en Double Integer (rekenkundige schuifopdrachten, het tekenbit blijft behouden)
  - Word en Double Word
- SHL met:
  - Word en Double Word
- ROR en ROL werken alleen met een Double Word.





Zelfstudie

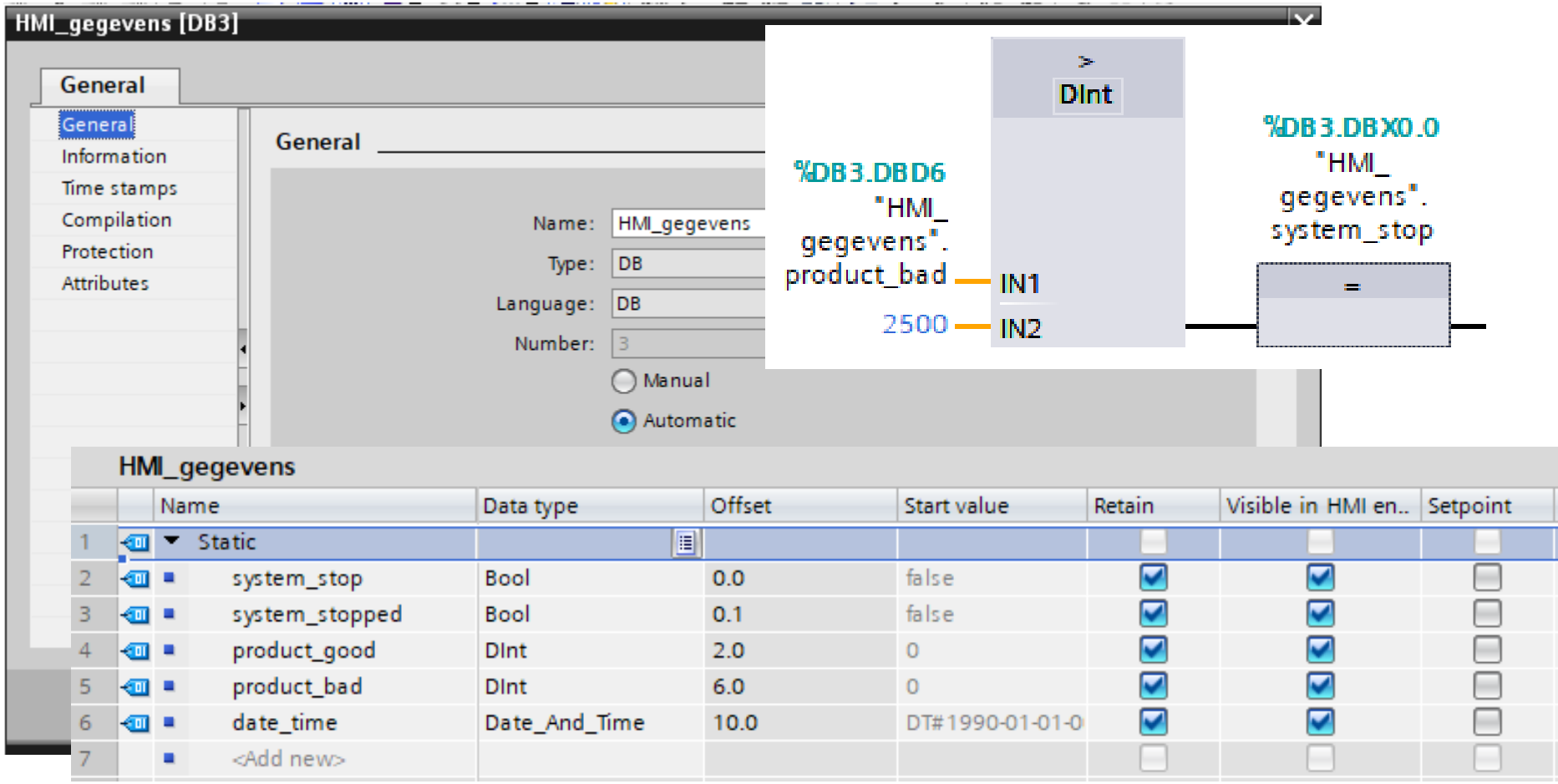
- Deze slides
- Hoofdstuk 1 uit: https://support.industry.siemens.com/cs/document/45523446/simatic-step-7-v5-5-statement-list-%28stl%29-for-s7-300-and-s7-400-programming?dti=0&pnid=14341&lc=en-WW
- Verdere informatie over rekenkundige functies en vergelijkers kan worden gevonden in: https://support.industry.siemens.com/cs/document/45522487/simatic-step7-v5-5-function-block-diagram-%28fbd%29-for-s7-300-and-s7-400-programming?dti=0&pnid=14342&lc=en-WW

Les 6

- Shared Data Block
- Parameters bij FC en FB
- TEMP variabele
- Startup OB

Global Data block

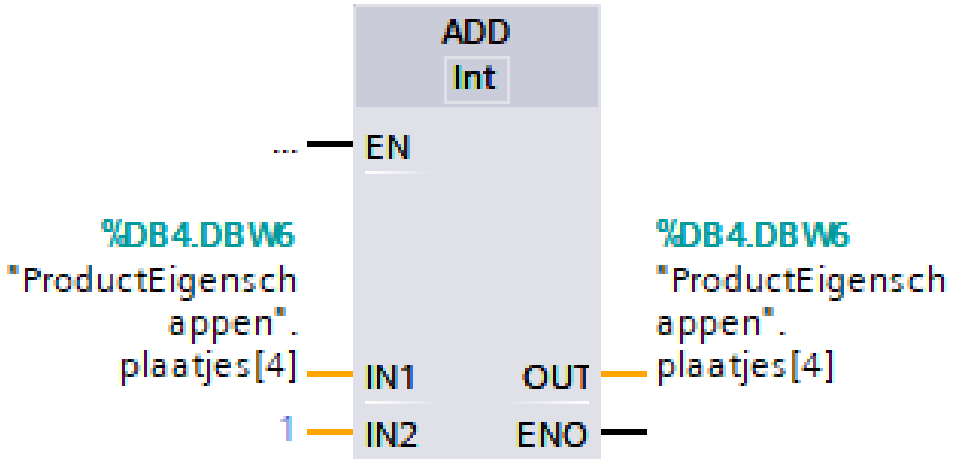
- In een Shared/Global Data Block kan je gemeenschappelijke informatie opslaan en gebruiken

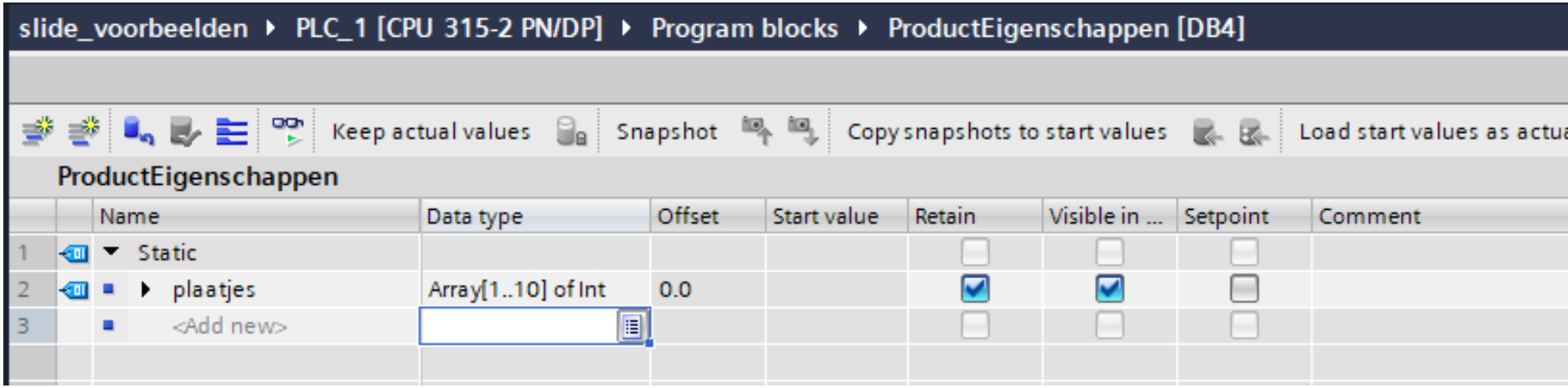


Array’s

- Het is mogelijk om in DB’s array’s op te slaan:

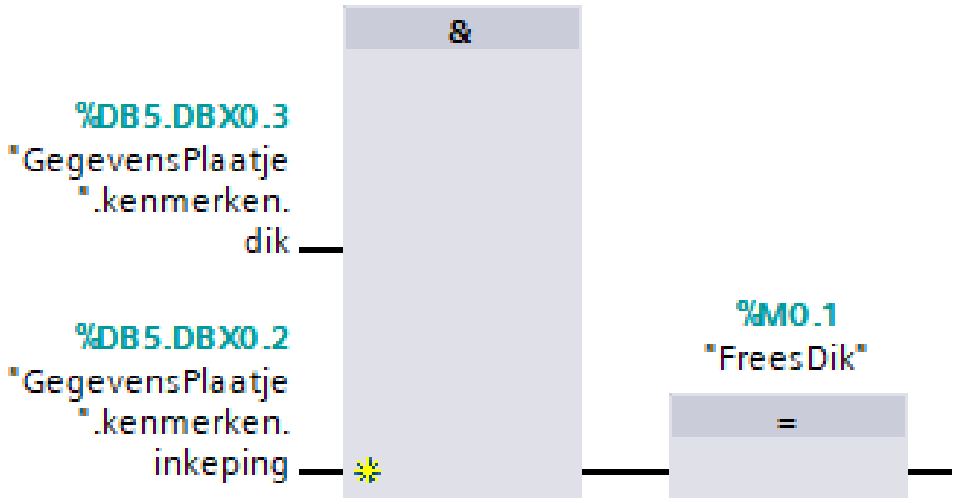
```
L   DB24.plaatjes[4]
L   1
+I

T   DB24.plaatjes[4]
```





Structures

- Het is mogelijk om structures te gebruiken. Hiermee worden bij elkaar horende gegevens gegroepeerd:



slide_voorbeelden ▸ PLC_1 [CPU 315-2 PN/DP] ▸ Program blocks ▸ GegevensPlaatje [DB5]

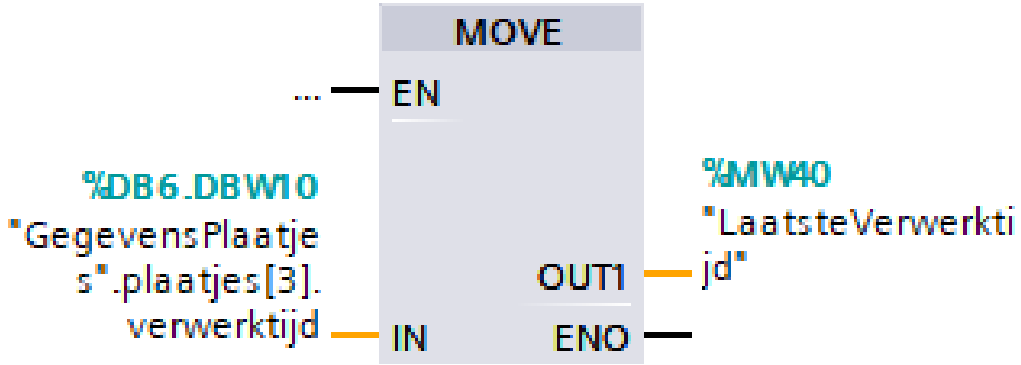
Keep actual values | Snapshot | Copy snapshots to start values | Load start values as actua

GegevensPlaatje

		Name	Data type	Offset	Start value	Retain	Visible in ...	Setpoint	Comment
1		Static							
2		kenmerken	Struct	0.0		☑	☑		Kenmerken van pla
3		alleGaatjes	Bool	0.0	false	☑	☑		
4		sleuf	Bool	0.1	false	☑	☑		
5		inkeping	Bool	0.2	false	☑	☑		
6		dik	Bool	0.3	false	☑	☑		

Array van structures

- En dit is ook mogelijk:



slide_voorbeelden ▸ PLC_1 [CPU 315-2 PN/DP] ▸ Progra

Keep actual values | Snapshot

GegevensPlaatjes

	Name	Data type	Offset	Start value	Retain	Visible in ...	Setpoint	Comment
1	Static							
2	plaatjes	Array[1..10] of Struct	0.0		☑	☐	☐	
3	plaatjes[1]	Struct	0.0		☑	☐	☐	
4	heeftGaatjes	Bool	0.0	false	☑	☐	☐	
5	heeftSleuf	Bool	0.1	false	☑	☐	☐	
6	heeftInkeping	Bool	0.2	false	☑	☐	☐	
7	isDik	Bool	0.3	false	☑	☐	☐	
8	verwerktijd	Int	2.0	0	☑	☐	☐	
9	plaatjes[2]	Struct	4.0		☑	☐	☐	
10	plaatjes[3]	Struct	8.0		☑	☐	☐	
11	plaatjes[4]	Struct	12.0		☑	☐	☐	
12	plaatjes[5]	Struct	16.0		☑	☐	☐	
13	plaatjes[6]	Struct	20.0		☑	☐	☐	
14	plaatjes[7]	Struct	24.0		☑	☐	☐	
15	plaatjes[8]	Struct	28.0		☑	☐	☐	
16	plaatjes[9]	Struct	32.0		☑	☐	☐	
17	plaatjes[10]	Struct	36.0		☑	☐	☐	
18	<Add new>				☐	☐	☐	

Parameters en variabelen FC & FB

- Interface FC
  - IN = ingangsparameter (read-only)
  - OUT = uitgangsparameter (write-only)
  - IN_OUT = in- en uitgangs-parameter (read-write)
  - TEMP = tijdelijke opslag
  - RETURN = uitgangsparameter (volgens IEC-norm)
- Parameters moeten bij de bouw-steenoproep voorzien zijn van een adres.

- Interface FB
  - IN = ingangsparameter (read-only)
  - OUT = uitgangsparameter (write-only)
  - IN_OUT = in- en uitgangs-parameter (read-write)
  - STAT = DB lokaal geheugen (gebufferd)
  - TEMP = tijdelijke opslag
- Parameters mogen bij de bouw-steenoproep voorzien worden van een adres, maar hoeft niet.

DE HAAGSE HOGESCHOOL

Lokale stack

- De lokale stack is een geheugengebied waar een OB, FB of FC gegevens kan aanpassen.

- Bij een FC staan alleen TEMP-parameters op de lokale stack.

- Bij een FB staan ook alleen TEMP-parameters in de lokale stack.
  - Overige parameters staan in de Instance Data Block

- Bij een OB staan OB-specifieke gegevens in de lokale stack.

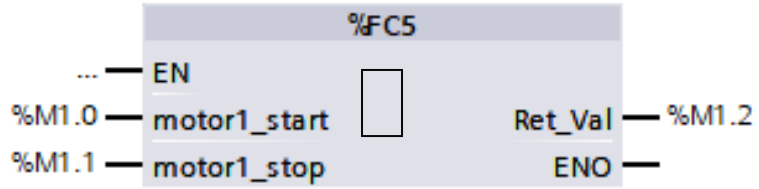
Parameteroverdracht FC

slide_voorbeelden ▸ PLC_1 [CPU 315-2 PN/DP] ▸ Program bloc

MotorStarter

		Name	Data type	Offset
1		Input		
2		motor1_start	Bool	
3		motor1_stop	Bool	
4		<Add new>		
5		Output		
6		<Add new>		
7		InOut		
8		<Add new>		
9		Temp		
10		xyz	Bool	0.0
11		<Add new>		
12		Constant		
13		<Add new>		
14		Return		
15		MotorStarter	Bool	

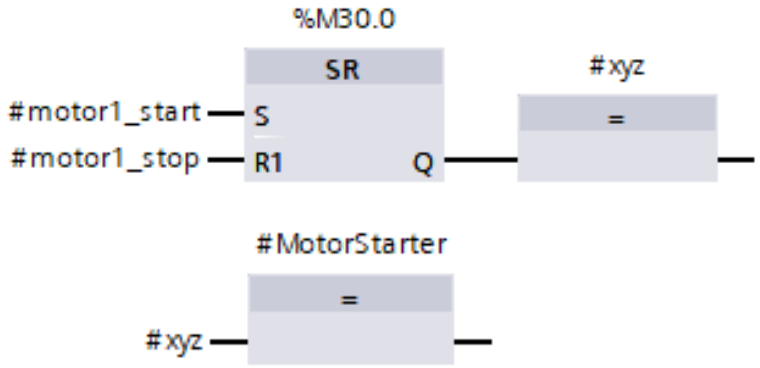
OB1



EN = enable
ENO = enable out

EN en ENO mogen open gelaten worden

FC5



Parameteroverdracht FB (1)

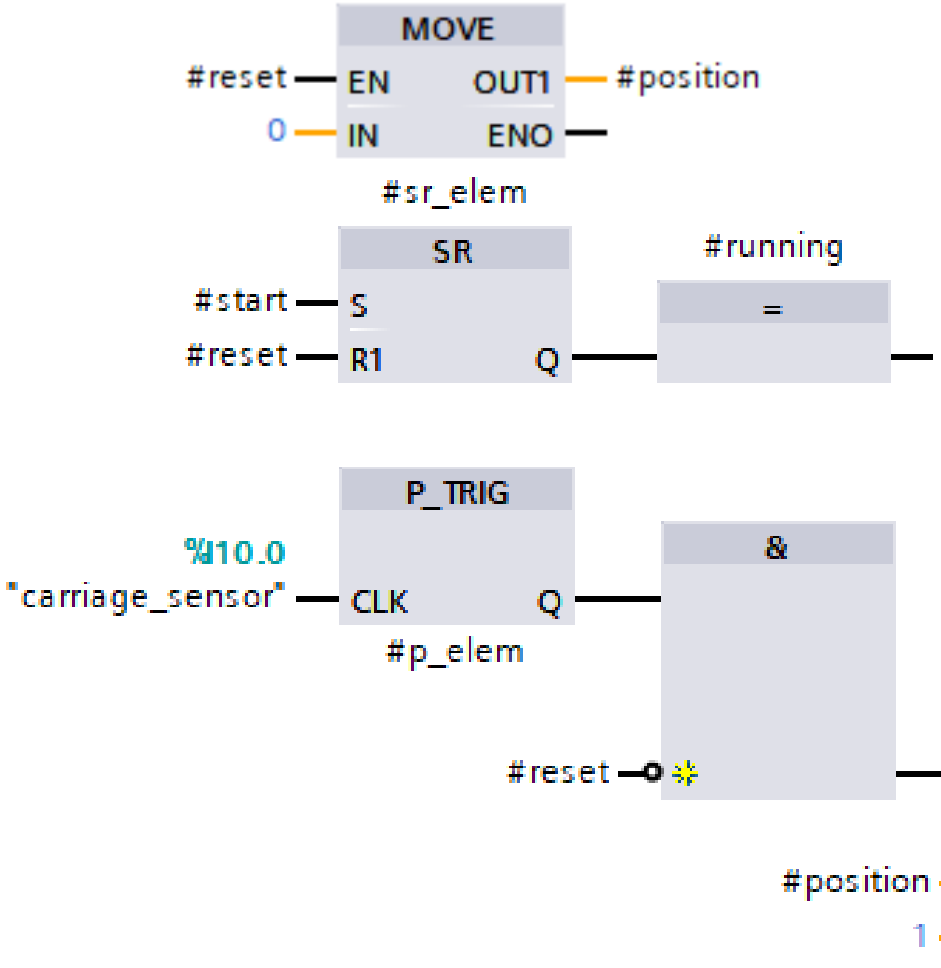
- Parameters bij een FB worden doorgegeven via ‘pass by value’. (waardes bij het aanroepende netwerk worden gekopiëerd naar interne variabelen) Dit in tegenstelling tot bij een FC!
- Inhoud van STAT (static) variabelen blijft behouden en is dus weer beschikbaar in volgende scan-cycle.
- Voorbeeld:

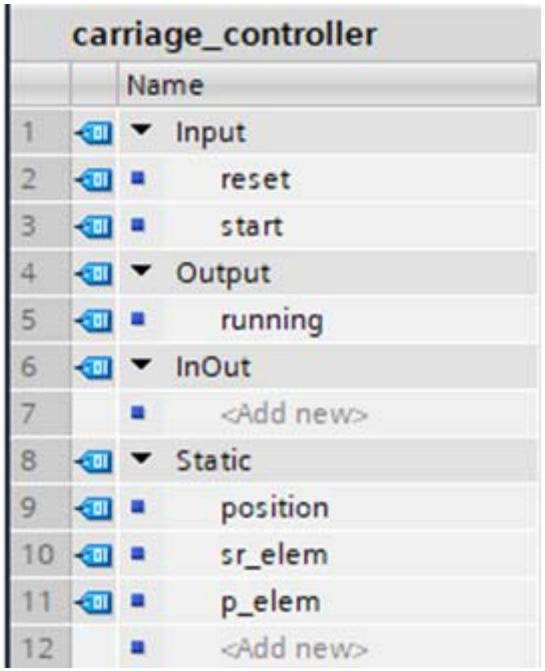
carriage_controller

	Name	Data type	Offset	Default v..
1	Input			
2	reset	Bool	0.0	false
3	start	Bool	0.1	false
4	Output			
5	running	Bool	2.0	false
6	InOut			
7	<Add new>			
8	Static			
9	position	Int	4.0	0
10	sr_elem	Bool	6.0	false
11	p_elem	Bool	6.1	false
12	<Add new>			
13	Temp			
14	<Add new>			
15	Constant			
16	<Add new>			

Parameteroverdracht FB (2)

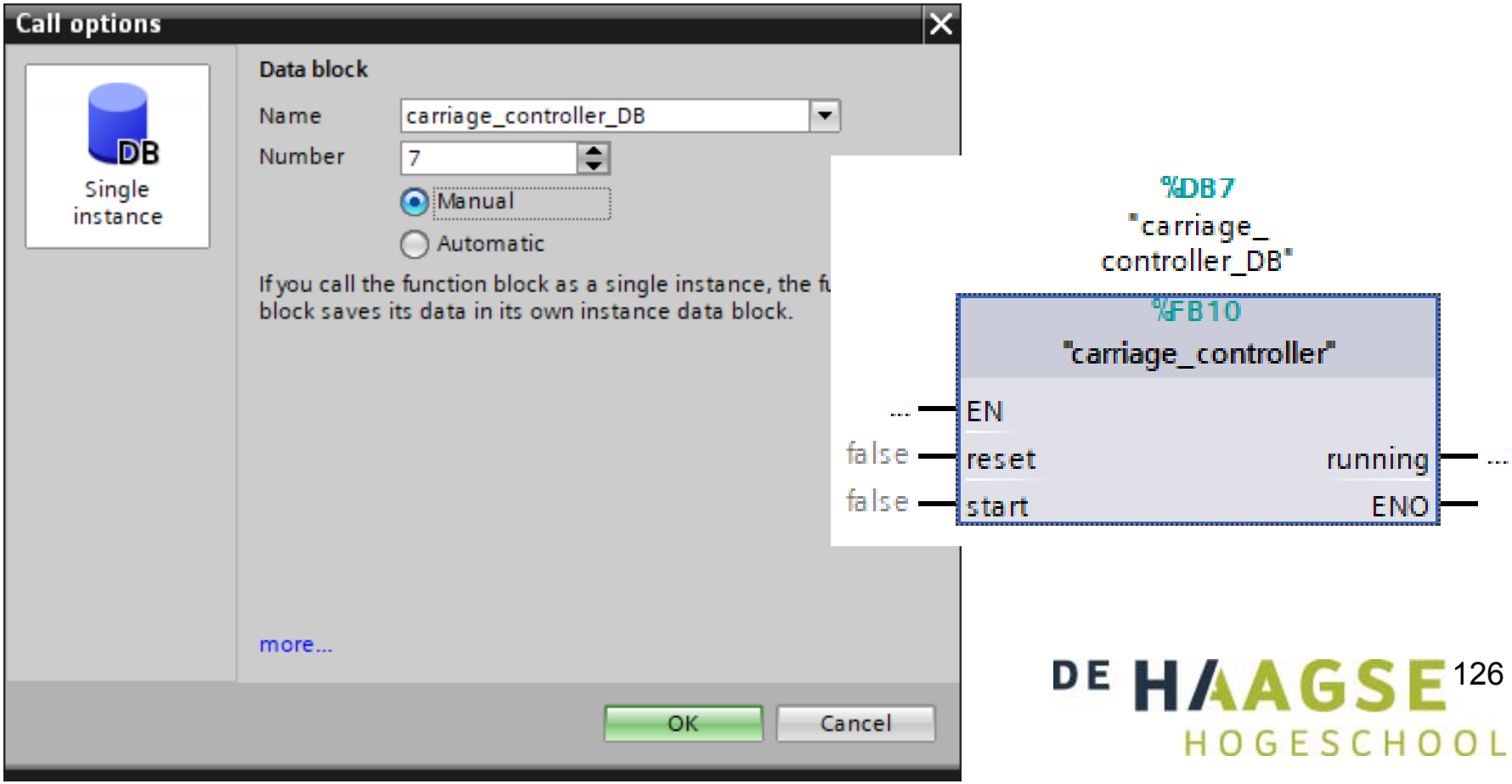
- Voorbeeld gebruik parameters in FB10 (carriage_controller):





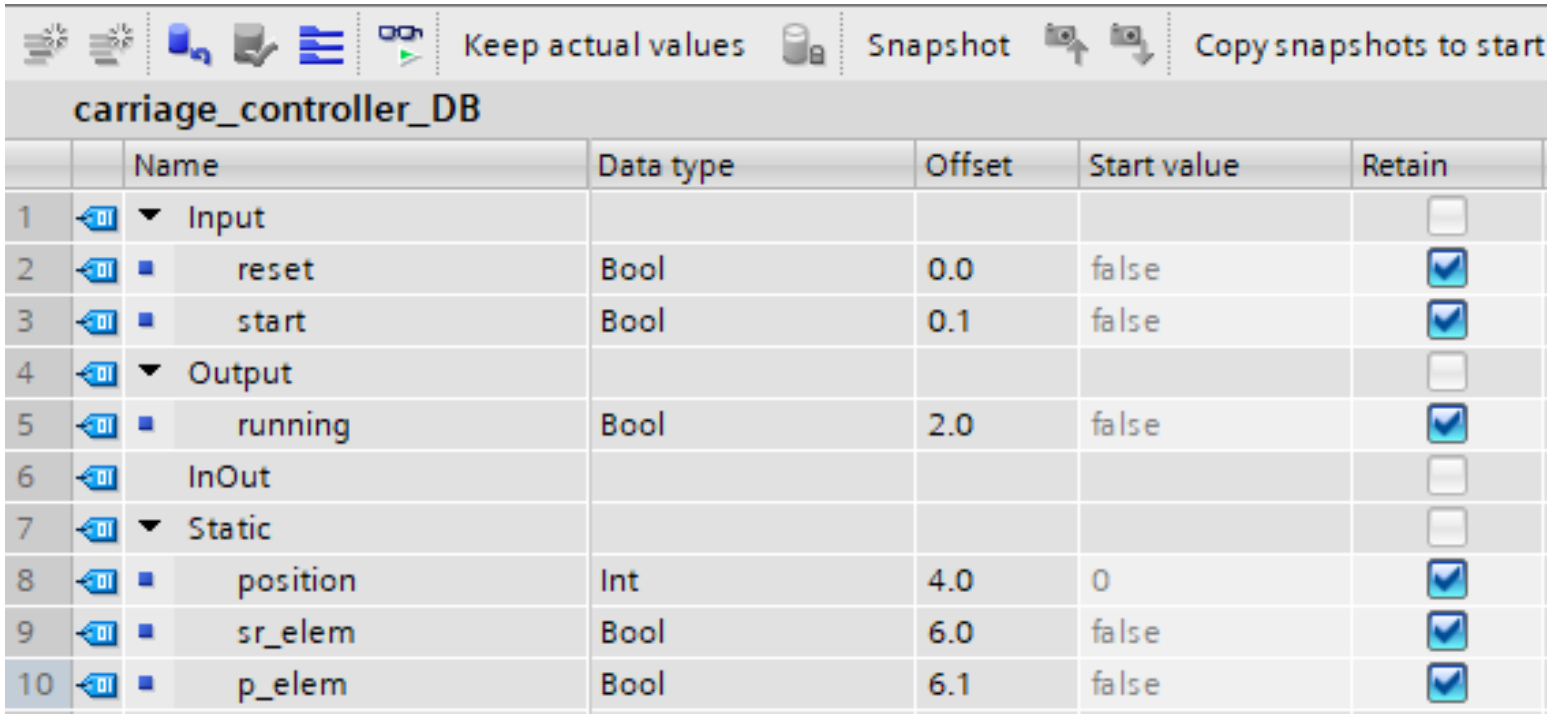
Parameteroverdracht FB (3)

- Bij het invoeren van een FB-aanroep wordt gevraagd of de bijhorende DB aangemaakt moet worden.



Parameteroverdracht FB (4)

- De DB is dan als volgt ingevuld:



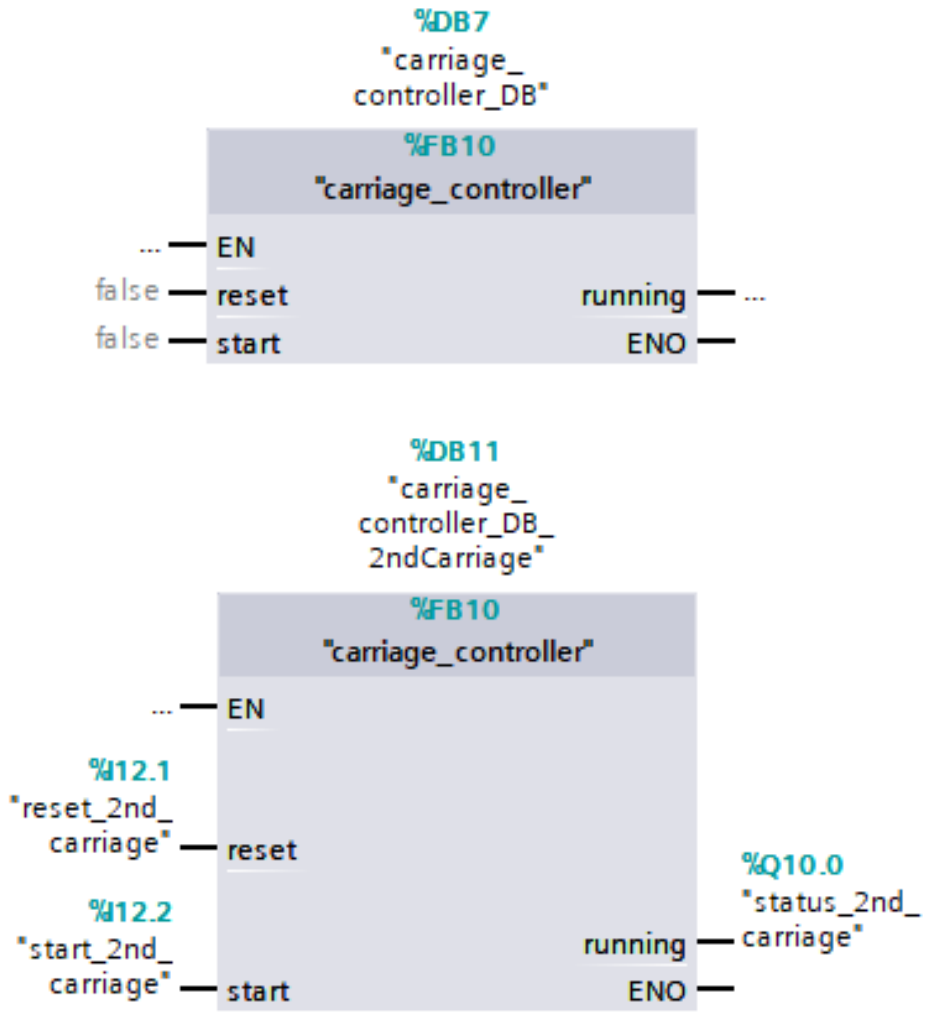
carriage_controller_DB

		Name	Data type	Offset	Start value	Retain
1		Input				
2		reset	Bool	0.0	false	☑
3		start	Bool	0.1	false	☑
4		Output				
5		running	Bool	2.0	false	☑
6		InOut				
7		Static				
8		position	Int	4.0	0	☑
9		sr_elem	Bool	6.0	false	☑
10		p_elem	Bool	6.1	false	☑

DE HAAGSE HOGESCHOOL

Parameteroverdracht FB (5)

- Let op: Een instance DB kan slechts met één FB samenwerken (b.v. DB7 met FB10).
- Een FB kan wel aan verschillende instance DB's gekoppeld worden (zie hiernaast)



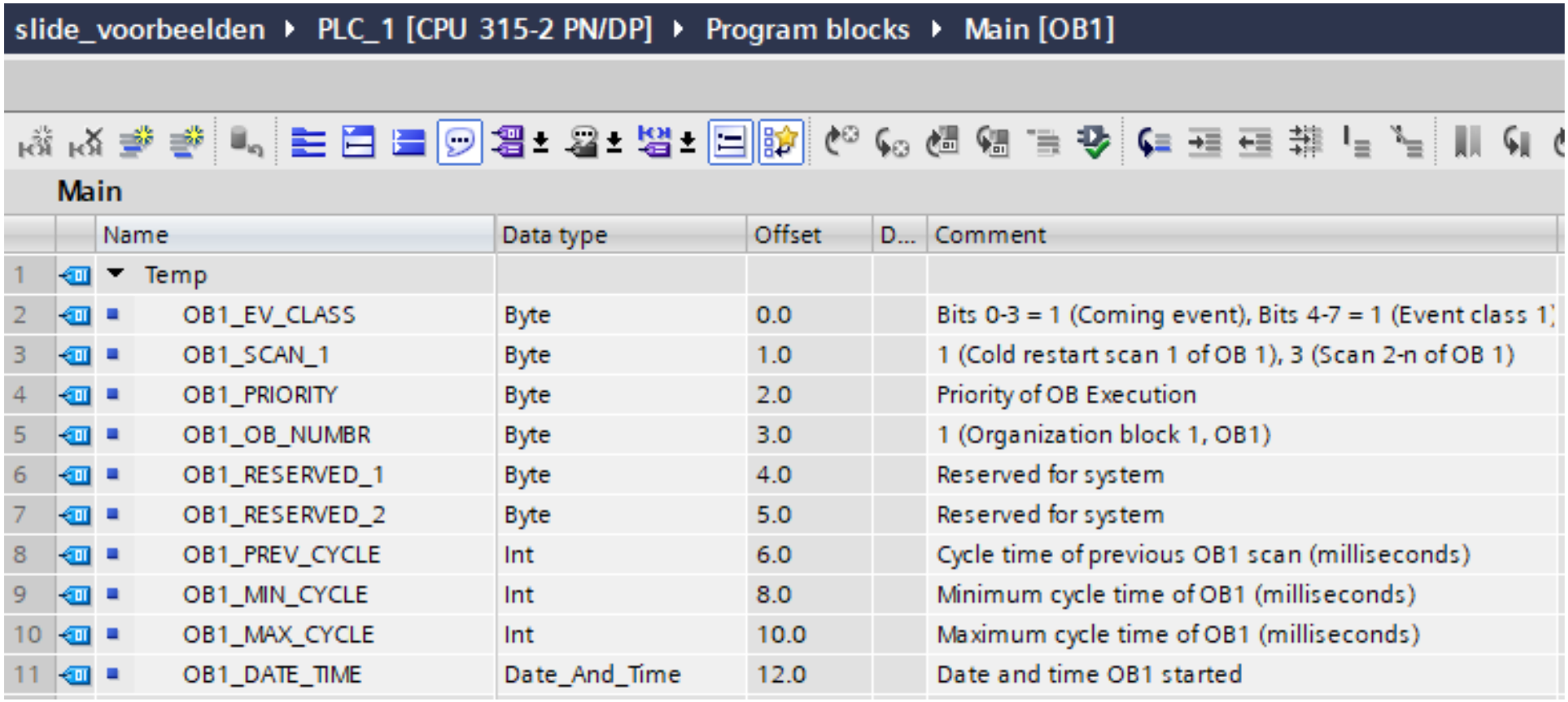
Parameteroverdracht OB (1)

- Elke OB heeft een 20-byte lokale stack waarin informatie door het O.S. wordt geschreven.
- Een deel heeft een vaste invulling, een deel is OB-specifiek.
- De informatie kan worden gebruikt om verschillende acties te ondernemen. Er wordt gebruik gemaakt van events.

```
// Get last scan cycle time
L   #OB1_PREV_CYCLE
T   MW 10
// Now read info with HMI device
```


DE HAAGSE HOGESCHOOL

Parameteroverdracht OB (2)



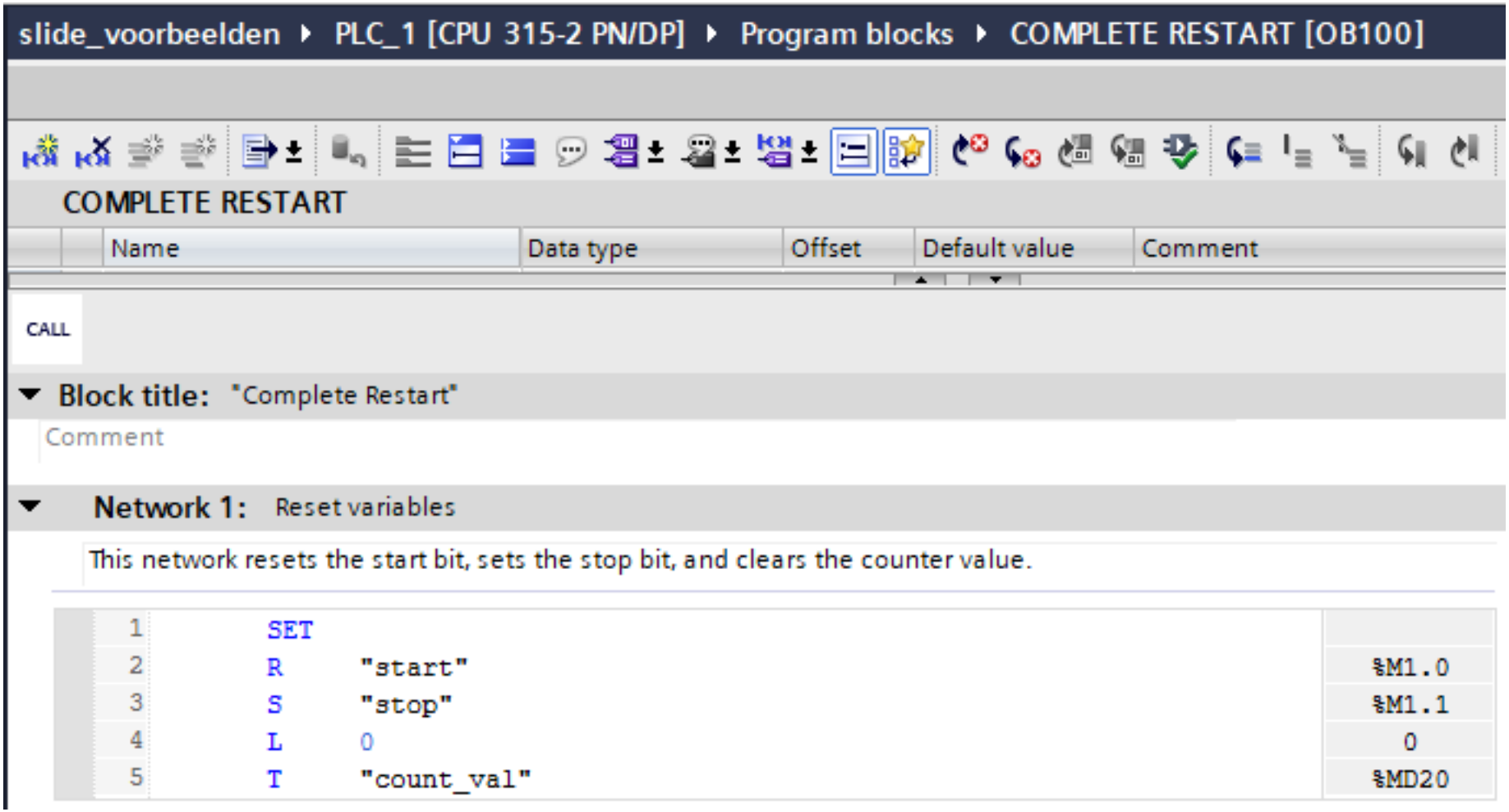
	Name	Data type	Offset	D...	Comment
1	Temp				
2	OB1_EV_CLASS	Byte	0.0		Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event class 1)
3	OB1_SCAN_1	Byte	1.0		1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n of OB 1)
4	OB1_PRIORITY	Byte	2.0		Priority of OB Execution
5	OB1_OB_NUMBR	Byte	3.0		1 (Organization block 1, OB1)
6	OB1_RESERVED_1	Byte	4.0		Reserved for system
7	OB1_RESERVED_2	Byte	5.0		Reserved for system
8	OB1_PREV_CYCLE	Int	6.0		Cycle time of previous OB1 scan (milliseconds)
9	OB1_MIN_CYCLE	Int	8.0		Minimum cycle time of OB1 (milliseconds)
10	OB1_MAX_CYCLE	Int	10.0		Maximum cycle time of OB1 (milliseconds)
11	OB1_DATE_TIME	Date_And_Time	12.0		Date and time OB1 started

Startup OB (1)

- Na het aanzetten of opstarten van de PLC wordt altijd de Warm Restart OB aangeroepen.
- Bij de S7-300 is dit OB100.
- In deze OB100 kan gebruikt worden als initialisatieroutine:
  - Merkerbits 0 of 1
  - Tellers op 0
  - Uitgangen hoog of laag

Startup OB (2)

- Hieronder een screenshot:



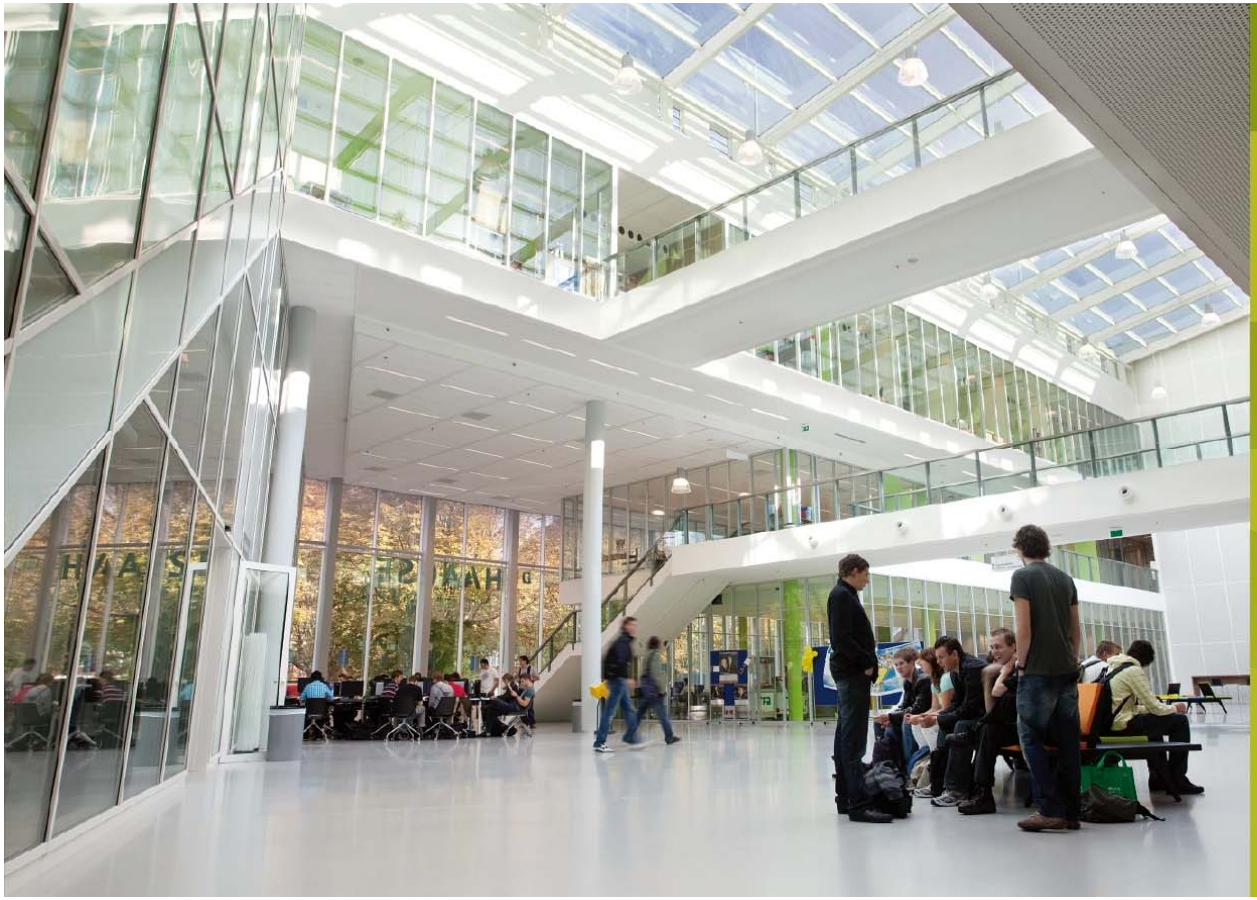
Zelfstudie

- Deze slides
- Extra uitleg en achtergrond over parameters, FC's en FB's is o.a. te vinden in: http://www.acro.be/Downloads/BasisPLC/Hoofdstuk%206_Functies%20en%20functiebouwstenen.pdf

Literatuur

- S7-300 Ladder Logic – https://support.industry.siemens.com/cs/document/45523822/simatic-step-7-v5-5-ladder-logic-%28lad%29-for-s7-300-and-s7-400-programming?dti=0&pnid=14342&lc=en-WW
- S7-300 Graph - https://support.industry.siemens.com/cs/document/1137630/s7-graph-v5-3-for-s7-300-400-programming-sequential-control-systems-?dti=0&lc=en-WW
- S7-300 Function Block Diagrams - https://support.industry.siemens.com/cs/document/45522487/simatic-step7-v5-5-function-block-diagram-%28fbd%29-for-s7-300-and-s7-400-programming?dti=0&pnid=14342&lc=en-WW
- S7300 - OB's, SFC's, SFB's etc - https://support.industry.siemens.com/cs/document/44240604/system-software-for-s7-300-400-system-and-standard-functions-volume-1-and-volume-2?dti=0&pnid=14342&lc=en-WW
- S7-300 Statement List – https://support.industry.siemens.com/cs/document/45523446/simatic-step-7-v5-5-statement-list-%28stl%29-for-s7-300-and-s7-400-programming?dti=0&pnid=14341&lc=en-WW
- Uitleg functies en bouwstenen - http://www.acro.be/Downloads/BasisPLC/Hoofdstuk%206_Functies%20en%20functiebouwstenen.pdf
- Algemene principes PLC's - http://www.patchn.com/images/articles/plc/plcbook/plcbook5_0.pdf
- *TUTORIAL STEP7 TIA PORTAL V14 MET S7-300*, J.E.J. op de Brouw op Blackboard
- TIA Portal v14 help
- Enkele kopieën uit TIA Portal v14 help over LAD op Blackboard

DE HAAGSE HOGESCHOOL



Academie voor Technology, Innovation & Society Delft

Academie voor ICT & Media

De Haagse Hogeschool, Delft
+31-15-2606311
J.E.J.opdenBrouw@hhs.nl / B.Kuiper@hhs.nl
www.dehaagsehogeschool.nl

DE HAAGSE HOGESCHOOL