# **NVS(PREL)**

# SPS (Speicher programmierbare Steuerung)

… hängt eng mit Sensor Aktor Systemen zusammen.  
SPS bekommt Information von Sensoren (Eingänge)  
und verändert aufgrund der Infos die Umwelt (Ausgänge) //schaltet Aktoren

Steuerung  
  
  
  
  
  
Regelung:

Beeinflusst das System

Regel

Istwert

Sollwert

Istwert

Steuerung

Sollwert

SPS muss echtzeitfähig sein.  
 echtzeitfähig in der Industrie: Eher garantierte Zeit in der etwas geschieht.  
  
Schutzklasse „(Wasser, Strom)“ IPS6  
Ist im Feld … Feld im Industriebetrieb dort wo es mit geht  
EMV… elektromagnetische Verträglichkeit   
  
Programmierung:  
 1. GUI oder IDE zum Erstellen des Codes  
 2. Compiler – Assembler Code (Object Code)  
 3. transportieren des Assembler Codes zur SPS (über Kabel, USB)  
 4. Start 🡺 Code rennt in einer Endlosschleife ab   
  
 KOP: Kontaktplan (Schaltungen)  
 AWL: Anweisungsliste. AWL verwendet UPN (umgekehrte polnische Notation)

# Sps

# Awl

Es gibt nur digitale Ein und Ausgänge (bei unserer)

Digital: 5V….true…ein

0V…false…aus

**Wie erfolgt Reaktion**

In C würde sowas sagen

If(E1 == true && E2 == false){

A1 = true;

}

Eingänge: E0.0 E0.1 …. E0.7

Ausgänge A0.0…A0.5

**Einfaches Programm**

Netzwerk 0 erzeuge einen Stack

LD E0.0 Lade den Zustand von Eingang E0.0 auf den Stack

U E0.1 verknüpfe den obersten Stackwert logisch mit den Zustand von E0.1

=A0.0 Der Ausgang A0.0 bekannt der logische Zustand des obersten Stackwertes

A0.0 = F0.0 && E0.1;

Netzwerk 1

LD E0.2

U A0.0

=A0.1

**Symboltabelle**

E0.0 ist es finster?

E0.1 jemand ist im Sensorbereich

A0.0 Licht

A0.1 Sirene

**Andere Schreibweise**

LD E0.0 LD E0.0

**Uld…**

Verknüpfe die 2 obersten Stackwerte mit AND , verringere den Stack um 1 und ersetze den obersten Stackwert durch das Ergebnis der Verknüpfung

U E0.1 LD E0.1

=A0.0 uld

= A0.0

(E0.0 && E0.1) || (E0.0 && E0.2)

LD E0.0

U E0.1

Ld E0.0

U E0.2

Old

LD E,A …. legt Wert auf den Stack

U E,A … verknüpft mit obersten Stak

O E,A …

ULD … verschmelzt die zwei obersten Stackwerte

OLD

UN E,A invertiere Parameter und dan verknüpf ihn oberster Stack && (!E)

OB E,A

= A …. Weise den Ausgang dem Wert des obersten Stackwertes zu

Wirken sich unmittelbar aus, werden bei jedem Schleifendurchgang der SPS am Ende aktualisiert

Statt = kann Wert auch permanent gespeichert werden 🡪 Flip Flop setzen

R S Zustand

0 0 unverändert

0 1 1

1 0 0

1 1 Kurzschluss verboten!

S R

Q2(Ausgang)

S

R

Q2

Nw x while(1){

LD E0.0 if(E0.0 && E0.1) A0.0 = true

U E0.1

S A0.0,1

Nw y if(E0.1 && E0.2) A0.0 = false

LD E0.1 }

U E0.2

R A0.0,1 hat nichts mit true oder false zu tun, sondern gibt die Anzahl der Bits an, die gesetzt oder gelöscht werden müssen

S A0.0,3 0000 0000 🡪 0000 0111

R A0.0, 2 0000 00100

2 u 3 ist eine Bitanzahl

Lampe (wenn man das erste mal draufdrückt eingehen beim zweiten mal ausgehen)

Nw x

LD E0.0

U A0.0

S A0.1,1

UO E0.0

O A0.0

S A0.1,2

**Aufgabe:**

**Schwesternrufsystem**

Patient hat Ruf-Taste

Rotes und Grünes Licht im Schwesterzimmer und beim Patientenzimmer außen

Im Zimmer bei der Tür Quittier-Taste fürs Personal

Ablauf: beide Lichter aus

Patient drückt Ruf-Taste RT

Rotes Licht geht an (vor der Tür und im Schwesterzimmer) LR on

Schwester geht zum Zimmer drückt Quittier-Taste zusätzlich grünes Licht an LG

Tätigkeit abgeschlossen Quittier-Taste nochmal gedrückt 🡪 beide Lichter aus

(Impulsdiagram im Heft)

**AWL**

**NW0**

LD RT

S LR,1

**Nw1**

LD QT

U LR

S LG,1

**NW2**

LD QT

EU

U LR

U LG

R LR,1

R LG,1

**Aufgabe:**

Wenn ein Startimpuls kommt, soll ein Balzen bis zur Endposition ausgefahren werden und dem wieder in die Ausgangslager zurückkehren.

Sensoren:

* Startimpuls E0.0 Start
* Ausgangstaster E0.1 TA
* Endtaster E0.2 TE

Akteure:

1. Motorleitungen

|  |  |  |
| --- | --- | --- |
| M1 | M2 | Was passiert |
| 0 | 0 | Stopp |
| 0 | 1 | Ausfahren |
| 1 | 0 | Einfahren |
| 1 | 1 | Nicht verwendet |

**AWL**

**NW0**

LD Start

S M1,1

**NW1**

LD TE

R M1,1

S M2,1

**NW2**

LD TA

R M2,1

# Folge Steuerungen

Eine Aktion bewirkt die nächste Zwangsfolge //z.B. Ampel

# Freifolge Steuerung

Ablauf unterbrochen, dann wieder normal weiter //z.B. Ampel mit Fußgängerknopf

# Zeitsteuerungen

z.B. Waschmaschine

## Timer

In den SPS ein Countdown-Zähler

Bestehet aus 2 Befehlen

TON T40, 300 ….Zeitdauer einstellen

…

LD T40 //schauen ob Zeit abgelaufen ist kann true oder false sein

Es gibt verschieden genaue Timer:

T30 …. T39 //auf 10ms genau

T40 … T99 //auf 100ms genau

T1 … T9 //auf 1ms genau

Zahl beim TON mit der Timergenauigkeit multiplizieren

z.B. 300 x 100 = 30000ms

🡪30 Sekunden

NW x

LD E0.0

TON T41, 50 //hängt vom obersten Stackwert ab falls true ist wird heruntergezählt sonst auf den //Startwert gesetzt

NW y

LD T41 // ist true wen herunterzählen fertig ist sonst false

= A0.0

50 40 30 20 50 40 30 20 10 50 40 30 20 10 0 50

T41

E0.0

Zählen:

MOVB 2ß VB 0 …VB0 = 20

MOV w 1000, VW0 betrift VB0 und VB1 da die Zahl größer 255 ist

VD0

VW1

VW0

usw

VB2

VB1

VB0

VB0…. 0-255

VW0…. 0-65535

VD0 … 0 – 43 Mrd.

Schleifen:

INCB VB0 …VB00++

INCW VW 0

INCD VD0

DECB

LD B = VB0, 3 if(VB0 == 3) oberster Stack = true;

Else oberster Stack = false;

LD B = VB

LD B >= VB0, 3

LD B <= VB0, 27

NW 0

LON T40

TON T40, 50

NW1

LD T40

INCB VB0

BSP:

Baustellenampel

Grün

20‘‘

Rot

Wann ist die Ampel rot?

LDB <= VB0, 39

LDB = VB0, 41

LDB = VB0, 45

LDB = VB0, 45

OLD

OLD

OLD

=lange Rot

**Mittwoch, 22. November 2017**

Parkschranken:

Sensoren:

* KeyCardSesor außen

Bei Log 1 wird Schranken geöffnet

* Induktionsschleife innen
* Lichtschranke im Schrankenbereich … falls Lichtschranke unterbrochen und Schranken nicht ganz geschlossen 🡪 Schranken öffnen
* Sensor zu … Endtaster ob Schranken ganz zu ist
* Sensor auf … Endtaster ob Schranken ganz offen ist

Motoren:

|  |  |  |
| --- | --- | --- |
| Mauf | Mzu | Aktion |
| 0 | 0 | Stopp |
| 1 | 0 | Öffnen |
| 0 | 1 | Schließen |
| 1 | 1 | Gibt’s nicht |

Falls nach kc 10‘‘ keine steigende Flanke am Lichtschranken dann Schranken schließen.