9. SPS Programmiersprachen und Grundglieder: EN61131-3, Beispiele für Grundglieder

EN61131-3:

Programmiersprachen: (IEC 1131-3)

Es wurden für diese Norm keine neuen Programmsprachen definiert, sondern die am meisten genutzten Programmsprachen:

Anweisungsliste AWL
Kontaktplan KOP
Funktionsbausteine FBS-FUB
Ablaufsprache AS
Strukturierter Text

Programme:

Ein Programm ist in der IEC-1131 definiert als:
Eine logische Anordnung von allen Programmsprachelementen und Konstrukten, welche für die beabsichtigte Signalverarbeitung zur Steuerung einer Maschine oder eines Prozesses mit einem SPS-System erforderlich sind.

Variablen:

Eine Variable dient zur Identifikation von Datenobjekten. Der Inhalt der Variablen ist verbunden mit Eingängen, Ausgängen oder Speicherplatz der SPS.
Variablen können als elementare oder als abgeleitete Typen deklariert sein.

Darstellung der Einzelelement-Variable

Eine Einzelelement-Variable ist ein Datenelement von einem elementaren Datentyp z.B.: ein Eingang oder ein Ausgang. Die Darstellung erfolgt durch Symbole.

- Ein Prozentzeichen
- Ein Bezeichner für die Speicherart M….Merker
- Ein Bezeichner für die Größe: X(Bit), B (Byte), W (Word), D (double word (32 Bit)), L (long word (64Bit))
- Eine vorzeichenlose ganze Zahl

z.B.: %QX75
Q….Ausgang
X….Bit
75…Bitnummer

%IW15 (← Analogeingang)
I……Eingang
W..……Wort (2 Byte = 16 Bit)
15……..Wertnummer

Die Anweisungsliste (AWL):

Die Anweisungsliste setzt sich aus einer Abfolge von Befehlen zusammen.
Folgende Konventionen sind einzuhalten:

- Jede Anweisung muss in einer eigenen Zeile stehen
- Jede Anweisung enthält einen Operator mit zusätzlichen Modifizierungen: z.B.: IØØ Bit Ø Eingangbyte Ø
- Eine Anweisung kann eine Sprungmarke voranstellen (Verweisung erfolgen durch Sprünge)
  Der Wert des letzten Verknüpfungsergebnisses wird nach einer Anweisung durch den aktuell berechneten Wert ersetzt.

  VKE
  z.B.:
  And(%IX1
  OR %IX2
)
  bedeutet:
  VKE=VKE AND (%IX1 OR %IX2)

  AWL-Befehle:

  LD %IX3

  Entspr.: VKE= Inhalt von %IX3
  ST%QX5
  Entspr. %QX5=VKE (Verknüpfungsereignis)

  ADD ..... Addition
  GE..... Vergleich
  JMD..... Sprung
  CAL..... Aufruf Funktionsbausteine
  RET ..... Rücksprung aus einer aufgerufenen Funktion

  AWL im STEP7:

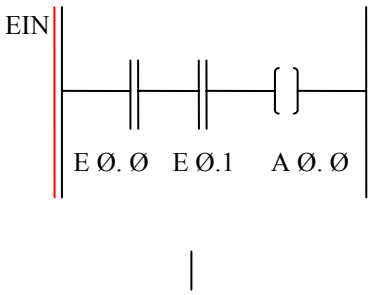
  Netzwerk 1:

  u EØ. Ø
  u EØ.1
  = AØ. Ø

  Kontaktplan KOP:

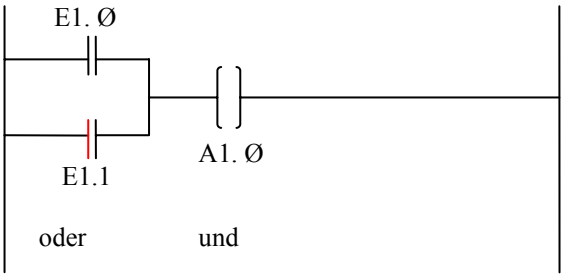
  Ein KOP-Netzwerk wird links und rechts durch jeweils eine vertikale Stromschiene begrenzt.

  - Ein Verbindungselement wird durch eine horizontale Linie gekennzeichnet.



- Ein bzw. Aus einer Verbindungselements entspricht „1“ oder „Ø“
- Zustand der linken vertikalen Linie ist immer EIN.

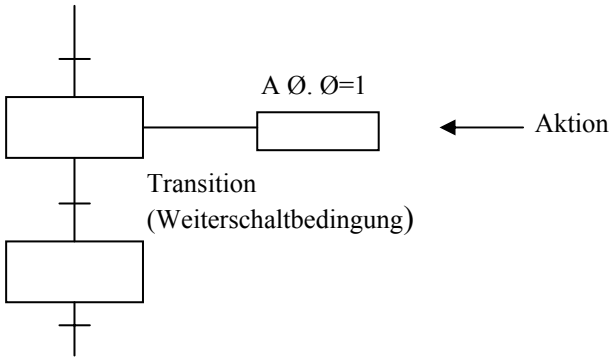
Das vertikale Verbindungselement kreuzt sich mit einem oder mehreren horizontalen Elementen. Der Zustand der rechten vertikalen Verbindung ist aus, wenn alle horizontalen Verbindungen auf der linken Seite des Elements auf AUS sind. Der Zustand muss EIN sein, wenn eine oder mehrere der horizontalen Verbindungen auf der linken Seite auf EIN sind.



Ablaufsprache AS:

Grundelemente:

- Schritt-Aktionen
- Transitionen



Jede Ablaufsteuerung muss mit einem Initial (Anfangsschritt) beginnen. Die Systeminitialisierung erfolgt über eine voreingestellte Zeitdauer für Schritte. Die Aktivierung des nächsten Schrittes muss innerhalb dieser Zeit erfolgen, sonst wechselt das System in den Zustand Störung.
Die Transitionsbedingung kann in der Sprache KOP, FUP, AWL, ST definiert werden.
Ein Schritt beinhaltet keine oder mehrere Aktionen. Eine Aktion ist eine Folge von Anweisungen in einer der normgerechten Programmiersprachen.

Zustandsbeschreibung für Verknüpfungssteuerung

Steuerungen mit kombinatorischem Charakter sowie Speicher- oder Zeitverhalten, jedoch ohne zwangsläufig schrittweisen _____ bezeichnet man allgemein als Verknüpfungssteuerung. Bei solchen Steuerungen werden die Lösungen meist empirisch gefunden. Diese Lösungen haben oft folgende Nachteile:

- schwer nachvollziehbar
- schwer erweiterbar
- schwierige Fehlersuche

Es ist daher das Ziel Verknüpfungssteuerungen mit Hilfe eines Entwurfsverfahrens zu realisieren. Eine dieser Methoden ist das so genannte Zustandsgraph. Die Beschreibung von Verknüpfungssteuerungen ohne Speicherverhalten (Schaltnetz) erfolgt oft mit Hilfe einer Funktionstabelle oder einem Karnaugh-Diagramm.
Diese Methoden sind jedoch nicht ohne weiteres auf Verknüpfungssteuerungen mit Speicherverhalten (Schaltwerk (Ampel, Lift, Bankomat)) übertragen.
Eine Methode Ablaufsteuerungen anschaulich darzustellen ist der so genannte Zustandsgraph. Jeder Zustand wird durch ein Rechteck gekennzeichnet. Um in einem Zustand zu gelangen muss man sich im vorhergehenden Zustand befinden und die zugehörige Weiterschaltbedingung erfüllt sein, d.h.: die beiden Wirkungslinien die zu einem Zustand führen werden UND - verknüpft (es müssen beiden den Zustand 1 aufweisen).
In jedem Zustand können Ausgabebefehle gegeben werden. Die Ausgabebefehle können Ausgänge, Merker, Zeiten, Zähler usw. betreffen.
Zustandsgraphen können natürliche auch Verzweigungen aufweisen, d.h.: auf einem bestimmten Zustand können je nach Erfüllung der zugehörigen Weiterschaltbedingung verschiedene Zustände folgen. Bei der Umsetzung in ein Steuerungsprogramm muss für die gegenseitige Verriegelung der Folgezustände gesorgt werden, wenn deren Weiterschaltbedingung gleichzeitig erfüllt sein können.

Wird bei der Erstellung eines Zustandsgraphen in einen Zustand übergegangen der bereits eingetragen ist, so müsste eine Wirkungslinie zu diesem führen. Besser ist es jedoch, solche Zustände nochmals mit einem runden Zustandssymbol einzuzeichnen. Ein Schaltwerk kann sich stets nur in einem Zustand befinden. Einem jeden Zustand wird ein Merker zugeordnet, der den Wert 1 aufweist, wenn der Zustand aktiv ist, d.h. man benötigt so viele Merker wie Zustände auftreten können. Die Programmierumgebung der verschiedenen SPS Hersteller bieten meistens eine eigene Sprache zur Erstellung von Zustandsgraphen an (Ablaufsprache AS bei der S7). Eine unabhängige Realisierung von Zustandsgraphen ist durch die Zuweisung von Merker für die Zustände z.B.: in FUP möglich.

Umsetzung eines Zustandgraphen in FUP

Bei der Umsetzung des Zustandgraphen werden zunächst alle Zustände in RS-Speicherglieder übertragen. Danach wird die Befehlsausgabe erstellt. Soll beispielsweise der Ausgang A1.1 in den Zuständen 2,4, und 6 ein 1-Signal haben, so wird dies mit einer ODER-Verknüpfung der Merker, die den entsprechenden Zuständen zugeordnet sind, erreicht.

Beim Einschalten des Automatisierungsgerätes muss der Grundzustand (Zustand 0) ohne Bedingung gesetzt werden. Dies erfolgt mit einem so genannten Richtimpuls. Ein solcher Richtimpuls ist in vielen SPSen vorhanden. Er kann aber auch folgendermaßen ausgelegt werden.

```
UN      M10.1   1   0       1. Durchgang
=       M10.2   1   0       2. Durchgang
S       M10.1   1   1
```

Der Merker 10.2 hat nur im 1. Durchlaufzyklus der SPS den Wert 1. Dieser Impuls kann genützt werden, um in den Zustand 0 nach dem Einschalten zugelangen. Voraussetzung allerdings ist, dass der Merker 10.1 ein nicht remanenter Merker ist. Er verliert also beim Stopp oder bei einem Spannungsabfall seinen Signalwert. Auch bei der Verwendung im Zustandmerker ist zu achten, dass diese nichtremanente Merker sind. Ansonsten hätte man beim Einschalten nicht den Zustand 0, sondern den zuletzt aktiven Zustand. Mit der Darstellung im Zustandsgraph ist die Steuerungsaufgabe bereits gelöst. Das Umsetzen des Zustandsgraphen in ein Steuerungsprogramm lässt sich einfach durchführen und kann auch mit den entsprechenden Programmen erfolgen.

FUP (Funktionsbausteinsprache (Funktionsplan))

Programmbausteine:

OB, FC, DB, FB, SFC, SFB

????????

Strukturierter Text – ST

Dabei handelt es sich um eine Programmiersprache für eine SPS, die sich stark an Pascal anlehnt.

Beispiel:

Sprachelemente:

Zuweisung:

A:= B; C:=C+1

XXX(In:=%IX5, PT:=T#300ms); //XXX ist ein beliebiger Funktionsname
A=XXX.Q; //Funktionsaufruf und Gebrauch eines Ausgangs

IF:

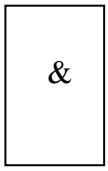
IF(D<0.0) THEN X:=0;
ELSEIF (D=0.0) THEN // = ist ein Vergleich und 0.0 ist eine Gleitkommazahl
X:=1; //:= ist eine Zuweisung
ELSE

X:=2;
END_IF;

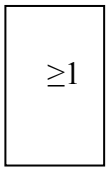
es gibt noch weitere Sprachelemente:

CASE (switch)
FOR
WHILE (WHILE-SCHLEIFE = Kopfgesteuert (brauch nicht mindestens einmal durchlaufen))
REPEAT (DO-WHILE-SCHLEIFE = Fussgesteuert (läuft mindestens einmal durch))

Grundglieder:

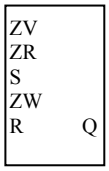


UND-Verknüpfung

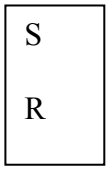


ODER-Verknüpfung

speichernde
Einschaltverzögerung

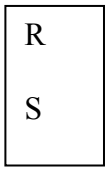


Zähler

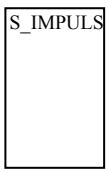


Speicherfunktion Flip-Flop (Merker)

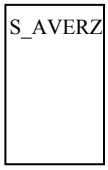
dominant rücksetzend (R gewinnt)



dominant setzend (S gewinnt)



Zeit als Impuls



Ausschaltverzögerung