

## Automatisierungstechnik nach internationaler Norm programmieren (13)

Autor: Dr. Ulrich Becker

Fachzentrum Automatisierungstechnik und vernetzte Systeme im BTZ Rohr-Kloster

Mail: Ulrich.Becker@BTZ-Rohr.de

### Grundlagen der Ablaufsprache (AS) Teil 1

In Folge 12 haben wir die integrierte Visualisierung des Programmiersystems CoDeSys vorgestellt. Damit können einfache Visualisierungen insbesondere für Inbetriebnahme, Service und Kontrolle erstellt werden. Weiter kann man Masken für andere Visualisierungsmethoden vorbereiten. In dieser Folge sollen nun die Kenntnisse über die in IEC 61131-3 zugelassenen Sprachen erweitert und die effiziente Ablaufsprache für Schrittketten vorgestellt werden. Das umfangreiche Thema wird in zwei Folgen 13 und 14 in aufeinander folgenden Ausgaben dieser Zeitschrift behandelt.

### Ablaufsteuerungen und Schrittketten

Die Beherrschung von Steuerungsabläufen mit streng vorgegebener Abfolge der Zustände ist eine klassische Aufgabe der Automatisierungstechnik. Solche Probleme treten häufig auf, so daß dafür mit der Ablaufsprache (AS) eine besondere Programmiersprache entwickelt wurde. Bereits im System Simatic S5 gab es mit Graph5 eine solche Ablaufsprache, die mit Step7/Graph7 vorteilhaft ausgebaut wurde. Die Ablaufsprache ist eine der sechs Sprachen für Automatisierungskomponenten nach IEC 61131-3.

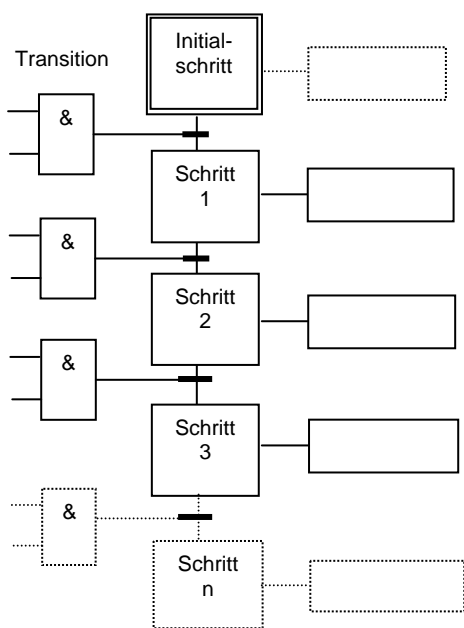
Als Voraussetzung für das Programmieren in AS ist zuerst zu untersuchen, welche unterschiedlichen Zustände eine Maschine oder Anlage bei der Ausführung ihrer bestimmungsgemäßen Funktion oder auch beim Einrichten und im Havariebetrieb annimmt und in welcher Reihenfolge dies geschieht. Häufig ist dies die eigentliche ingenieurtechnische Aufgabe, wogegen das Programmieren selbst nach festen Routinen erfolgen kann (**Bild 70**). Vor Bearbeitung einer Automatisierungsaufgabe lohnt es sich stets zu prüfen, ob diese mit den Regeln der Ablaufsteuerung zu lösen ist!

Kernstück einer Ablaufsteuerung ist eine Kette aufeinander folgender Schritte. Als Schritt werden alle sich voneinander unterscheidende Zustände der Anlage bezeichnet. So sind beispielsweise der Lauf einer Hydraulikpumpe allein oder aber zusammen mit einem Hauptantrieb einer Maschine bereits unterschiedliche Zustände und damit unterschiedliche Schritte genau so wie Vorschub eines Werkzeugs und Beharrungstellung desselben. Durch Aufeinanderfolge der Schritte entstehen Schrittketten, und es werden dadurch eindeutige (!) Abläufe erzwungen. Die Grundform sind lineare Schrittketten. Darüberhinaus sind Verzweigungen von Schrittketten und Sprünge in diesen möglich.

### Klassische Programmierung von Schrittketten

Schrittketten können auch ohne spezielle graphischen Sprachen mit klassischen Methoden programmiert werden. Ein solches Hintergrundwissen ist hilfreich bei der Anwendung der Ablaufsprache. Schrittketten bestehen aus Schrittelelementen (Schritte) und Transitionselementen (Transitionen). Die Schritte geben Aktionen (Befehle) aus. Die Grundregeln der Kette erzwingen eindeutige Abläufe auch dann, wenn unterschiedliche Maschinenzustände von gleichen Ausgangsbedingungen her erforderlich werden.

Schritte sind Boolesche Variable. In Step 7 werden dafür beispielsweise Merkerbits verwendet, die gesetzt und zurückgesetzt werden. Die Weiterschaltbedingungen sind gleichfalls vom Typ BOOL. Sie können eine beliebige Variable, aber auch Ergebnis einer logischen Verknüpfung oder ein Boolescher Ausgang von Timern oder Zählern sein.



**Grundregeln der linearen Schrittkette:**

Ein Schritt wird gesetzt, wenn der vorhergehende Schritt aktiv **und** die Weiterschaltbedingungen erfüllt sind.

LD Schritt n-1  
 AND Weiterschaltbedingungen  
**S Schritt n**

Ein Schritt wird zurückgesetzt, wenn der nachfolgende Schritt eingeschaltet wird.

LD Schritt n+1  
**R Schritt n**

Mit den Operationen S, R, ST, CALC....geben Schritte unterschiedlich wirkende Befehle aus

LD Schritt x  
 ST .....  
 S .....  
 R .....

Bild 70: Regeln der linearen Schrittkette

### Anwendung der Ablaufsprache (AS)

Vorteilhafter als eine solche selbst erstellte detaillierte Programmierung ist die Anwendung der speziellen Ablaufsprache. Diese ist eine grafisch orientierte Sprache mit Schrittelemen-ten und zugeordneten (assoziierten) Aktionen. Die Abfolge der Schritte wird durch Transitionselemente gesteuert. In CoDeSys ist die Anwendung der Ablaufsprache in POE's vom Typ PROGRAM oder FB möglich.

Bei Start der Programmierung einer POE in der Sprache AS wird eine Grundstruktur mit Initialschritt und erster Transition sowie dem Rücksprung zum Initialschritt angelegt (**Bild 71**). Hier können nun die erforderlichen Schritt-Transition-Kombinationen eingefügt werden. Bei verzweigten Ketten werden parallele und alternative Zweige eingetragen.

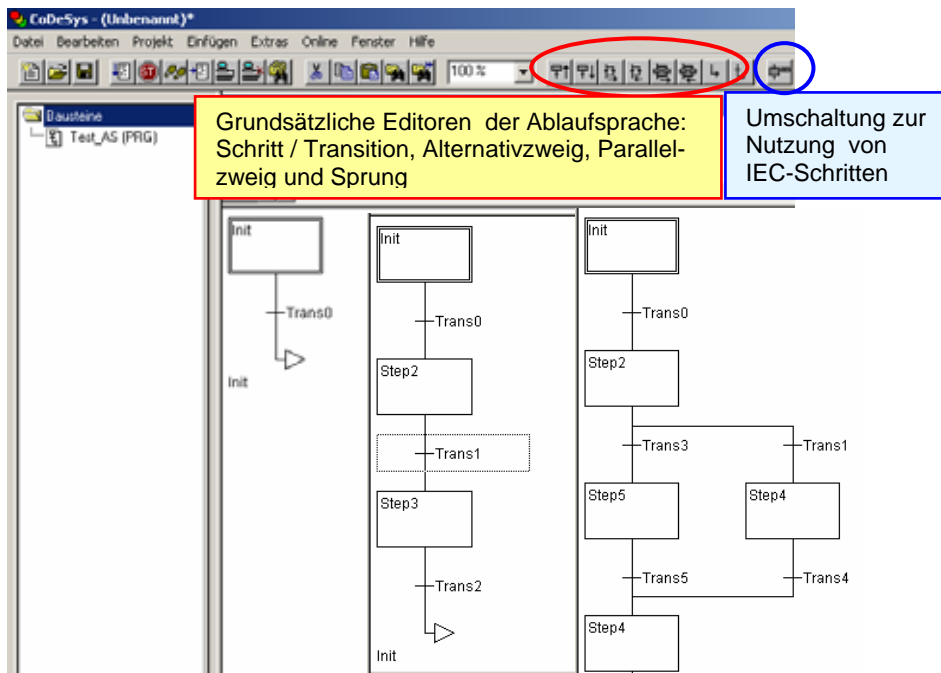


Bild 71: Schrittkette in AS der CoDeSys: Links nach Anlegen der POE, Mitte nach Einfügen zweier Schritt-Transitions-Kombinationen und rechts nach Anlegen einer alternativen Verzweigung.

## Ablaufsprache mit einfachen Schritten

**Bild 72** zeigt einige Details der Ablaufsprache. Die Programmierung von Schritten ist im System CoDeSys auf zwei Arten möglich. Sogenannte einfache Schritte bestehen aus einer Aktion und einem Flag, welches anzeigt, ob der Schritt aktiv ist. Ist zu einem Schritt eine Aktion implementiert, so erscheint ein kleines Dreieck in der rechten oberen Ecke des Schrittkästchens. Genauso erscheint dieses Zeichen bei einer Transition, wenn diese nicht allein aus einer Variablen besteht, sondern ein Programm wie z.B eine logische Verknüpfung hinterlegt ist. Dieser Fall liegt im Bild 72 für die Transition „Start“ vor. Transitionen können in beliebiger Sprache außer AS programmiert werden. Eine Transitionsbedingung muß den Wert TRUE oder FALSE haben. Deshalb sind alle Programmelemente denkbar, die zu einer Booleschen Variablen oder einer Booleschen Konstante führen. Transitionen dürfen aber keine Programme, Funktionsbausteine und auch keine Zuweisungen enthalten. Letzteres ist bei der Programmierung von Transitionen besonders zu beachten und bedeutet, daß allein die logischen Anweisungen ohne Abschluß mit dem Befehl STORE zu schreiben sind.

Aktionen werden programmiert, indem man den Schrittnamen lädt und damit Operationen wie Setzen oder Zurücksetzen oder den Aufruf von Standard FB , nicht aber Zuweisen durch STORE einleitet. Bild 72 zeigt beispielhaft das hinterlegte Programm eines Schrittes „Warten“. Wird der Schritt aktiv, so wird die Variable „Band\_Re“ zurückgesetzt und der Timer „Wartezeit“ vom Typ TON gestartet. Nach Ablauf der programmierten Zeit schaltet dessen Boolescher Ausgang in Form der Variablen „Wartezeit.Q“ auf TRUE. Diese wirkt als Transition und schaltet Aktivieren den nachfolgenden Schritt ein. Im Bild 72 wird weiter ersichtlich, daß sowohl Transitionen und selbstverständlich alle Instanzen von Standard-FB als Variable zu deklarieren sind, nicht aber die Namen der Schritte.

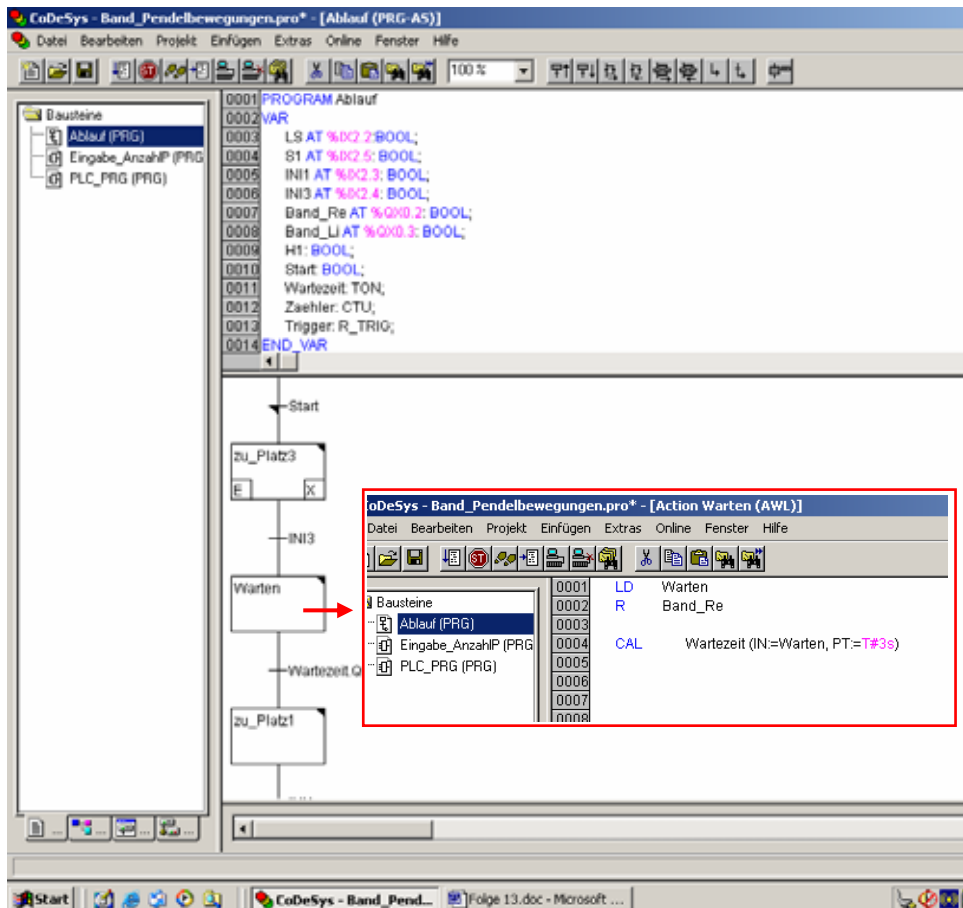


Bild 72: AS mit einfachen Schritten und hinterlegten Aktionen in AWL

Zusätzlich zur eigentlichen Schritt-Aktion kann jedem Schritt optional eine Eingangsaktion und/oder eine Ausgangsaktion hinzugefügt werden. Ein Schritt mit Eingangsaktion wird durch ein 'E' in der linken unteren Ecke gekennzeichnet, die Ausgangsaktion durch ein 'X' in der rechten unteren Ecke. Im Bild 72 trägt Schritt „zu\_Platz3“ solche zusätzlichen Aktionen. Eine Eingangsaktion wird nur einmal ausgeführt, gleich nachdem der Schritt aktiv geworden ist. Eine Ausgangsaktion wird nur einmal ausgeführt, bevor der Schritt deaktiviert wird. Ein- und Ausgangsaktionen werden also nicht zyklisch, sondern nur in einem einzigen Zyklus ausgeführt. Das erleichtert vielfältige Sonderaktionen. Ein- und Ausgangsaktionen können wiederum in einer beliebigen Sprache implementiert werden.

### Ablaufsprache mit IEC-Schritten.

Noch effektiver wird die Ablaufsprache bei Nutzung sogenannter IEC-Schritte. Um diese im System CoDeSys verwenden zu können, muß die spezielle SFC-Bibliothek „*lecsfc.lib*“ in das Projekt eingebunden werden. Weiter muß diese Darstellung im Editor unter ->Extras -> IEC Schritte benutzen aktiviert.

**Bild 73** zeigt einen Abschnitt einer solchen Schrittkette im Online-Modus. Normkonforme IEC-Schritte besteht aus einem Flag und bis zu neun assoziierten Aktionen. Diese können Programme oder boolesche Variablen sein. Über Qualifier (Bestimmungszeichen **Tabelle 13**) wird die Art der Aktivierung und Deaktivierung der Aktionen gesteuert. Verwendet man beispielsweise das Bestimmungszeichen S, ist die Aktion noch aktiv, wenn bereits der nächste Schritt abgearbeitet wird. Auch zeitliche Verzögerungen sind dabei möglich. Die Bestimmungszeichen L, D, SD, DS und SL benötigen eine Zeitangabe im TIME-Konstantenformat, z.B. L T#5s oder P T#2s300ms.

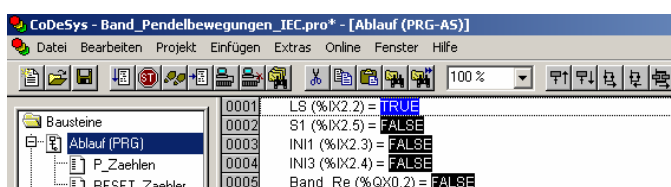
N	Non-stored	die Aktion ist solange aktiv wie der Schritt
R	overriding Reset	die Aktion wird deaktiviert
S	Set (Stored)	die Aktion wird aktiviert und bleibt bis zu einem Reset aktiv
L	time Limited	die Aktion wird für eine bestimmte Zeit aktiviert, maximal solange der Schritt aktiv ist
D	time Delayed	die Aktion wird nach einer bestimmten Zeit aktiv, sofern der Schritt noch aktiv ist und dann solange der Schritt aktiv ist
P	Pulse	die Aktion wird genau einmal ausgeführt, wenn der Schritt aktiv wird
SD	Stored and time Delayed	die Aktion wird nach einer bestimmten Zeit aktiviert und bleibt bis zu einem Reset aktiv
DS	Delayed and Stored	die Aktion wird nach einer bestimmten Zeit aktiviert, sofern der Schritt noch aktiv ist und bleibt bis zu einem Reset aktiv
SL	Stored and time Limited	die Aktion ist für eine bestimmte Zeit aktiviert

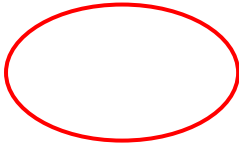
Tabelle 13: Qualifier für IEC Schritte

Ein Großteil der Aktionen sind allein durch Festlegung der Variablen und des Qualifier wirksam. Müssen darüberhinaus Bedingungen programmiert werden, so sind diese IEC-Aktionen nicht wie bei den einfachen Schritten fest einem Schritt zugeordnet. Sie liegen vielmehr getrennt von den Schritten vor und werden direkt unter dem AS-Baustein angeordnet. Innerhalb ihres Bausteins können sie mehrfach verwendet werden. Dazu müssen sie mit dem Befehl ->Extras -> Aktion assoziieren an die gewünschten Schritte angeschaltet (assoziiert) werden. Die assoziierten Aktionen an einem IEC-Schritt werden rechts vom Schritt in einem zweigeteilten Kästchen dargestellt. Das linke Feld enthält den Qualifier, evtl. mit Zeitkonstanten, das rechte Feld den Aktionsnamen bzw. booleschen Variablennamen. Ein- und Ausgangsaktionen sind wie bei einfachen Schritten möglich

Im Bild 73 wurden zwei solche Aktionen mit den Namen „P\_Zähler“ und „RESET\_Zaehler“ an den Baustein „Ablauf“ gehängt. Die Aktion „P\_Zähler“ wurde beispielsweise mit dem Schritt „Pzu\_Platz1“ verbunden. Die Aktion „Warten“ trägt als Beispiel einen Qualifier Typ D (verzögert).

Das Bild zeigt weiter auch die komfortable Art der Programmbeobachtung im Online-Modus. Alle aktiven Schritte und Aktionen werden blau dargestellt.





Bausteine und  
Aktionen

Bild 73: Schrittkette mit IEC Schritten im Online-Modus

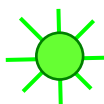
### **Ablaufsteuerung Verteilerband**

Nunmehr sollen einige der beschriebenen Möglichkeiten einfacher Schrittketten am Trainingsrack mit Verteilerband getestet werden. Dazu geben wir uns folgende Aufgabenstellung vor (**Bild 74**):

Am Platz 1 aufgelegte Teile sollen nach Betätigung des Tasters S1 zum Platz 3 transportiert werden. Sie verharren dort 3s und laufen dann zurück zum Platz 1. Von dieser Position aus sind eine wählbare Anzahl Pendelbewegungen zum Platz 3 und zurück auszuführen. Ist die Zahl erreicht, werden die Teile zur Lichtschranke abtransportiert.

Die LED Bereitschaft soll während des Initialschrittes leuchten. Die Anzahl der Pendelbewegungen soll mit dem virtuellen Taster einer Visualisierung zwischen den Werten 1..10 eingegeben werden.

LED  
Bereitschaft



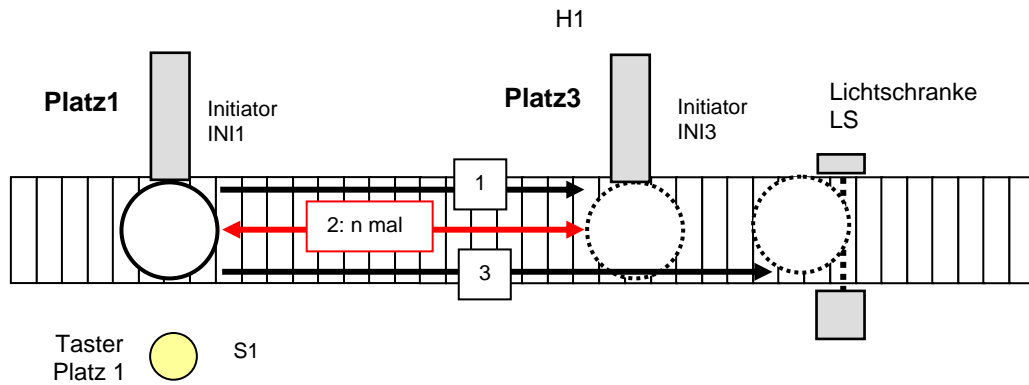


Bild 74: Technologieschema zur Aufgabenstellung Ablaufsteuerung Verteilerband

Die Lösung beginnt mit dem Entwurf der erforderlichen Schritte und Transitionen. Unter Verwendung einfacher Schritte sind diese in **Bild 75** aufgeführt. Schrittnamen können so vergeben werden, daß die Aktionen der Schritte leicht erkennbar sind. Die Pendelschritte sind zur Verdeutlichung rot gekennzeichnet.

Neben der Schrittkette sind im Bild die hinterlegten Anweisungen für Transitionen und Aktionen aufgeführt. Die Weichschaltbedingungen sind in den meisten Fällen Sensordesignale oder Boolesche Variable, in drei Fällen aber auch Ergebnis einfachster logischer Verknüpfungen.

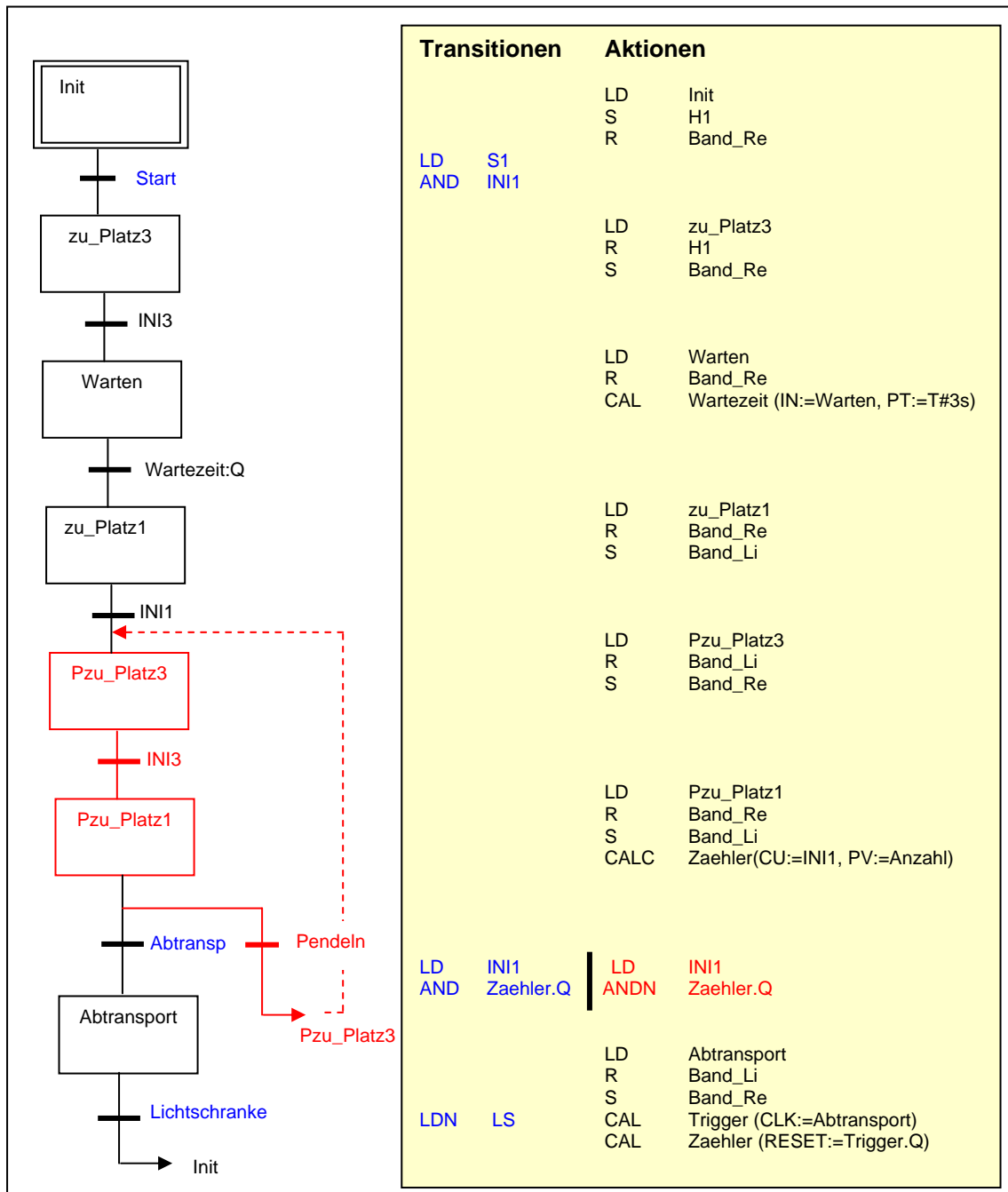


Bild 75: Ablaufsteuerung des Verteilerbandes mit zugehörigen Transitionen und Aktionen

Die Zahl der Pendelungen wird durch eine globale Variable „Anzahl:INT“ bestimmt. Um ohne zusätzliche Hardware eine Zahl zwischen 0 und 10 vorgeben zu können, kann man eine Visualisierung nach **Bild 76** erstellen (hierzu Folge 12). Im Hintergrund läuft das Programm „Eingabe\_AnzahlIP (PRG)“ mit einem zyklischen Ringzähler von 0..10. Dessen aktueller Wert zeigt das Feld „Vorgabe“. Mit einem Sprungbefehl JMPC zum Programmende wird bewirkt, daß nur bei Mausklick auf den Taster „Eingabe“ dessen Variable auf TRUE geschaltet und der aktuelle Zaehlerwert in die Variable „Anzahl“ geschrieben wird.

The screenshot shows the CoDeSys software interface. The top window is titled "CoDeSys - Band\_Pendelbewegungen\_IEC.pro\* - [Vorgabe\_AnzahlIP]". The left sidebar shows a tree view with "Visualisierungen" and "Vorgabe\_AnzahlIP". The main area displays a visualization of a pendulum count system. It features a title box "Anzahl der Pendelbewegungen", two input boxes labeled "Vorgabe" and "Aktuell", both containing the value "2". A central circular button labeled "Eingabe" is positioned between the two input boxes. Below the visualization, a yellow box contains the following ladder logic program:

```

PROGRAM Eingabe_AnzahlIP
VAR
Takt: BLINK;
Ringzaehler: CTU;
Eingabe: BOOL;
END_VAR

CAL    Takt(ENABLE := TRUE, TIMELOW := T#500ms, TIMEHIGH := T#500ms)
CAL    Ringzaehler(CU := Takt.OUT, RESET := Ringzaehler.Q, PV := 11)
LDN    Eingabe
JMPC  ENDE
LD     Ringzaehler.CV
ST     Anzahl
Ende: LD    TRUE
      RETC

```

Bild 76: Visualisierung und Hilfsprogramm zur Vorgabe einer Anzahl Pendelungen

An dieser Stelle wollen wir die Vorstellung der Ablaufsprache unterbrechen. Die Lösung der Aufgabe unter Verwendung von IEC-Schritten anstelle der vorgestellten einfachen Schritte werden wir in Folge 14 zeigen. Dort soll auch ein Blick auf die Ablaufsprache Graph 7 im System Simatic S7 geworfen werden.

**Fazit:**

Steuerungsabläufe mit streng vorgegebener Abfolge der Zustände sind eine klassische Aufgabe der Automatisierungstechnik, für die mit der Ablaufsprache (AS) eine spezifische Programmiersprache entwickelt wurde. In dieser Folge wurden deren Grundlagen vorgestellt. Die Elemente Schritt, Transition und Aktion und die Regeln für das Verschalten derselben zur Schrittkette wurden an einem übersichtlichen Beispiel erläutert. Dieses Thema wird in der nächsten Folge fortgesetzt.

## Glossar:

Ablaufsprache	Eine der 6 Programmiersprachen nach IEC 61131-3, geeignet für Ablaufsteuerungen. Grundelemente der graphischen Sprache AS sind Schritt, Transition und Aktion.
Ablaufsteuerung	Automatisierungssystem mit fest und zwangsweise vorgegebener Abfolge von Zuständen
Aktion	Anweisung zur Ausführung von Funktionen im aktiven Schritt
Assoziation	Verbindung: Aktionen werden mit Schritten assoziiert, d.h. mit diesen verbunden (angeschaltet):
Flag	Variable vom Typ BOOL, mit der bestimmte innere Zustände beim Ablauf eines Programms nach außen zur Verfügung gestellt werden, z.B. der Überlauf eines Rechnerprogramms.
Graph7	Sehr leistungsfähiges Softwaretool zur professionellen Erstellung der Programme für Ablaufsteuerungen im System Siemens Simatic S7. Graph 7 ist Bestandteil von Step 7 Professional. (siehe Online Hilfe : Graph 7 Erste Schritte und Graph 7 Handbuch)
Initialschritt	Beharrungs- und Ruhezustand der Ablaufkette nach dem Zuschalten der Spannung bzw. nach dem Initialisieren
Schritt	Element zur Bezeichnung eines Zustandes der Anlage, im Kern eine Variable vom Typ BOOL. Schritte geben Befehle (Aktionen) aus, welche die Anlage genau in diesem Zustand steuern.
Schrittfolge	auch Ablaufkette. Kernstück der Ablaufsteuerung. Die Schrittfolge enthält alle Zustände der Anlage in der richtigen Reihenfolge, linear oder verzweigt aufeinander folgend.
Transition	Weiterschaltbedingung. Die Bedingungen zum Einschalten eines neuen Schrittes und damit Verlassen des aktuellen münden programmtechnisch in die Boolesche Variable der Transition. Erhält sie den Wert TRUE, erfolgt das Weiterschalten in der Schrittfolge.